

Oracle Database 10g: Administration Workshop II

Volume I • Student Guide

D17092GC31

Edition 3.1

August 2010

D57199

ORACLE®

Authors

Tom Best
Maria Billings

Technical Contributors and Reviewers

Herbert Bradbury
Howard Bradley
Harald van Breederode
M.J. Bryksa
Donna Cooksey
Joe Fong
Andy Fortunak
Gerlinde Frenzen
Joel Goodman
Bert van Gorkom
Sushma Jagannath
Christine Jeal
Donna Keesling
Pierre Labrousse
Jerry Lee
Stefan Lindblad
Wendy Lo
Yi Lu
Claudia O'Callaghan
Andreas Reinhardt
Ira Singer
James Spiller
Janet Stern
Jean-Francois Verrier

Copyright © 2008, 2010, Oracle and/or its affiliates. All rights reserved.

Disclaimer

This document contains proprietary information and is protected by copyright and other intellectual property laws. You may copy and print this document solely for your own use in an Oracle training course. The document may not be modified or altered in any way. Except where your use constitutes "fair use" under copyright law, you may not use, share, download, upload, copy, print, display, perform, reproduce, publish, license, post, transmit, or distribute this document in whole or in part without the express authorization of Oracle.

The information contained in this document is subject to change without notice. If you find any problems in the document, please report them in writing to: Oracle University, 500 Oracle Parkway, Redwood Shores, California 94065 USA. This document is not warranted to be error-free.

Restricted Rights Notice

If this documentation is delivered to the United States Government or anyone using the documentation on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS

The U.S. Government's rights to use, modify, reproduce, release, perform, display, or disclose these training materials are restricted by the terms of the applicable Oracle license agreement and/or the applicable U.S. Government contract.

Trademark Notice

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Editors

Aju Kumar
Nita Pavitran
Arijit Ghosh

Graphic Designer

Steve Elwood

Publishers

Joseph Fernandez
Sujatha Nagendra

Contents

Preface

1 Introduction

Lesson Objectives	1-2
Course Objectives	1-3
Suggested Schedule	1-4
What Is Covered in the DBAI Course	1-5
Course Examples: The HR Schema	1-6
Oracle Database 10g: The Database for the Grid	1-7
Database Architecture: Review	1-8
Oracle Memory Structures	1-9
Oracle Processes	1-11
Reviewing Oracle Instance Management	1-12
Physical Database Structure	1-14
Oracle Managed Files (OMF)	1-16
Logical and Physical Database Structures	1-17
Database Architecture: Summary of Structural Components	1-19
Summary	1-20

2 Configuring Recovery Manager

Objectives	2-2
Backup and Recovery: Review	2-3
Features of Recovery Manager	2-4
Recovery Manager Components	2-6
Steps for Configuring RMAN	2-8
RMAN Repository Data Storage: Comparison of Options	2-9
Backup Destinations	2-11
Media Management	2-12
Using a Flash Recovery Area with RMAN	2-14
Monitoring the Flash Recovery Area with EM	2-16
Flash Recovery Area Space Usage	2-17
V\$FLASH_RECOVERY_AREA_USAGE	2-18
Backing Up the Flash Recovery Area	2-20
Benefits of Using a Flash Recovery Area	2-21
Setting Parameters That Affect RMAN	2-22

RMAN Usage Considerations	2-24
Connection Types with RMAN	2-25
Starting RMAN	2-26
Additional RMAN Command-Line Arguments	2-27
Configuring Persistent Settings for RMAN	2-28
Configuring RMAN Settings by Using EM	2-29
Control File Autobackups	2-30
Retention Policies	2-32
Managing Persistent Settings	2-34
Channel Allocation	2-35
Automatic and Manual Channel Allocation	2-36
Channel Control Options	2-37
Summary	2-39
Practice Overview: Configuring RMAN	2-40

3 Using Recovery Manager

Objectives	3-2
Issuing Recovery Manager Commands	3-3
Types of RMAN Commands	3-5
Job Commands: Example	3-6
RMAN Commands: Overview	3-7
BACKUP Command	3-9
Backup Constraints	3-10
Parallelization of Backup Sets	3-11
Compressed Backups	3-13
Image Copy	3-14
Tags for Backups and Image Copies	3-16
BACKUP Command Options	3-17
Backing Up Archived Redo Logs	3-19
Whole Database Backup	3-21
RMAN Backup Types	3-22
Differential Versus Cumulative	3-24
Block Change Tracking	3-25
Enabling Block Change Tracking	3-26
Incrementally Updating Backups	3-27
LIST Command	3-28
REPORT Command	3-29
REPORT NEED BACKUP Command	3-30
REPORT NEED BACKUP: Examples	3-31
REPORT OBSOLETE and DELETE OBSOLETE	3-32

Managing Backups with EM	3-33
RMAN Dynamic Views	3-34
Monitoring RMAN Backups	3-36
Summary	3-38
Practice Overview: Backing Up Your Database	3-39

4 Recovering from Noncritical Losses

Objectives	4-2
Causes of File Loss	4-3
Critical Versus Noncritical	4-4
Losing a TEMPFILE	4-5
Recovering from a TEMPFILE Loss	4-6
Log Group Status: Review	4-7
Losing a Redo Log Group Member	4-8
Re-creating Redo Log Files	4-9
Re-creating Indexes	4-13
Authentication Methods for Database Administrators	4-15
Re-creating a Password Authentication File	4-16
Summary	4-18
Practice Overview: Recovering from Lost TEMPFILE and Redo Log File	4-19

5 Database Recovery

Objectives	5-2
Recovery Methods	5-3
User-Managed Recovery: RECOVER Command	5-4
RMAN Recovery: RESTORE and RECOVER Commands	5-5
Recovery Using Enterprise Manager	5-6
Complete Versus Incomplete Recovery	5-7
Complete Recovery Process	5-8
Incomplete Recovery Process	5-9
Situations Requiring Incomplete Recovery	5-11
Types of Incomplete Recovery	5-12
Performing User-Managed Incomplete Recovery	5-14
User-Managed Time-Based Recovery: Example	5-16
User-Managed Cancel-Based Recovery: Example	5-18
Performing Incomplete Recovery by Using RMAN	5-20
Time-Based Recovery Using RMAN: Example	5-21
Log Sequence Recovery Using RMAN: Example	5-23
Incomplete Recovery Using Enterprise Manager	5-24
Incomplete Recovery and the Alert Log	5-25

Restore Points	5-26
Incomplete Recovery: Best Practices	5-27
Recovering a Control File Autobackup	5-29
Creating a New Control File	5-31
Recovering Read-Only Tablespaces	5-33
Read-Only Tablespace Recovery Issues	5-35
Summary	5-37
Practice Overview: Performing Incomplete Recovery	5-38

6 Flashback

Objectives	6-2
Flashback Technology: Review	6-3
Flashback Drop and the Recycle Bin	6-4
Recycle Bin	6-5
Restoring Tables from the Recycle Bin	6-7
Recycle Bin: Automatic Space Reclamation	6-8
Recycle Bin: Manual Space Reclamation	6-10
Bypassing the Recycle Bin	6-11
Querying the Recycle Bin	6-12
Querying Data from Dropped Tables	6-13
Flashback Database: Review	6-14
Flashback Database Architecture	6-15
Configuring Flashback Database	6-16
Configuring Flashback Database Using EM	6-17
Flashback Database: Examples	6-19
Performing Flashback Database Using EM	6-20
Flashback Database Considerations	6-23
Monitoring Flashback Database	6-25
Monitoring Flashback Database with EM	6-27
Guaranteed Restore Points	6-28
Summary	6-29
Practice Overview: Performing Flashback Database	6-30

7 Dealing with Database Corruption

Objectives	7-2
What Is Block Corruption?	7-3
Block Corruption Symptoms: ORA-01578	7-4
How to Handle Corruption	7-5
Corruption-Related Features	7-7
DBVERIFY Utility	7-8

Interpreting DBVERIFY Output	7-9
ANALYZE Command	7-11
Verifying Block Integrity in Real Time: DB_BLOCK_CHECKING	7-12
Verifying Block Integrity in Real Time: DB_BLOCK_CHECKSUM	7-13
Using EXP to Detect Corruption	7-14
Using Flashback for Logical Corruption	7-15
DBMS_REPAIR Package	7-16
Using DBMS_REPAIR	7-17
Block Media Recovery (BMR)	7-21
BLOCKRECOVER Command	7-22
Examples of Using BLOCKRECOVER	7-23
The RMAN BMR Interface	7-25
Alternative Actions to Take	7-26
Summary	7-27
Practice Overview: Perform Block Media Recovery	7-28

8 Monitoring and Managing Memory

Objectives	8-2
Memory Management: Overview	8-3
Oracle Memory Structures	8-4
Buffer Cache	8-6
Using Multiple Buffer Pools	8-8
Shared Pool	8-10
Large Pool	8-11
Java Pool	8-12
Redo Log Buffer	8-13
Automatic Shared Memory Management: Overview	8-14
Benefits of Automatic Shared Memory Management	8-15
How ASMM Works	8-16
Configuring ASMM by Using Database Control	8-17
Manually Configuring ASMM	8-18
Behavior of Autotuned SGA Parameters	8-21
Behavior of Manually Tuned SGA Parameters	8-22
Using the V\$PARAMETER View	8-23
Modifying the SGA_TARGET Parameter	8-24
Disabling ASMM	8-25
Manually Resizing Dynamic SGA Parameters	8-26
Program Global Area (PGA)	8-27
Automatic PGA Memory Management	8-29

PGA Management Resources	8-30
Using the Memory Advisor to Size the SGA	8-31
Using the Memory Advisor to Size the PGA	8-32
Efficient Memory Usage: Guidelines	8-33
Memory Tuning Guidelines for the Library Cache	8-35
Summary	8-37
Practice Overview: Using ASMM to Correct a Memory Allocation Problem	8-38

9 Automatic Performance Management

Objectives	9-2
Tuning Activities	9-3
Performance Planning	9-4
Instance Tuning	9-6
Performance Tuning Methodology	9-7
Statistics Collection	9-8
Oracle Wait Events	9-10
System Statistics	9-11
Displaying Session-Related Statistics	9-13
Displaying Service-Related Statistics	9-14
Troubleshooting and Tuning Views	9-15
Dictionary Views	9-16
Diagnosis of Hung or Extremely Slow Databases	9-17
Using Memory Access Mode	9-18
Using the Hang Analysis Page	9-19
Automatic Workload Repository	9-21
AWR Snapshot Baselines	9-23
Advisory Framework: Overview	9-24
Database Control and Advisors	9-26
Typical Advisor Tuning Session	9-27
Manually Invoking ADDM	9-28
Using the SQL Tuning Advisor: Review	9-29
SQL Access Advisor: Overview	9-30
Typical SQL Access Advisor Session	9-31
Workload Source	9-32
Recommendation Options	9-33
Reviewing Recommendations	9-35
Asynchronous COMMIT	9-36
Using Asynchronous COMMIT	9-37
Summary	9-38
Practice Overview: Using ADDM to Diagnose Performance Problems	9-39

10 Managing Schema Objects

- Objectives 10-2
- Table Types 10-3
- What Is a Partition and Why Use It? 10-4
- Partitions 10-5
- Creating a Partition 10-6
- Partitioning Methods 10-7
- Partition Maintenance 10-8
- Index-Organized Tables 10-9
- Index-Organized Tables and Heap Tables 10-10
- Creating Index-Organized Tables 10-12
- Clusters 10-13
- Cluster Types 10-14
- Situations Where Clusters Are Useful 10-16
- Sorted Hash Cluster: Overview 10-17
- Sorted Hash Cluster: Example 10-18
- Sorted Hash Cluster: Basic Architecture 10-19
- Schema Management Tasks 10-20
- Estimating Resource Usage 10-21
- Analyzing Growth Trends 10-22
- Managing Optimizer Statistics 10-23
- Reorganizing Schema Objects Online 10-24
- Reorganizing Objects: Impact Report 10-26
- Reorganizing Objects: Review 10-27
- Basic Steps for Manual Online Reorganization 10-28
- Summary 10-29
- Practice Overview: Managing Schema Objects 10-30

11 Managing Storage

- Objectives 11-2
- Space Management: Overview 11-3
- Free Space Management 11-4
- Types of Segments 11-5
- Allocating Extents 11-6
- Block Space Management 11-7
- Row Chaining and Migration 11-8
- Proactive Tablespace Monitoring 11-9
- Thresholds and Resolving Space Problems 11-10
- Monitoring Tablespace Space Usage 11-11
- Shrinking Segments 11-12
- Results of Shrink Operation 11-13

Space Reclamation with ASSM	11-14
Segment Advisor: Overview	11-15
Segment Advisor	11-16
Implementing Recommendations	11-18
Database Control and Segment Shrink	11-19
Shrinking Segments by Using SQL	11-20
Managing Resumable Space Allocation	11-21
Using Resumable Space Allocation	11-22
Resuming Suspended Statements	11-24
Transporting Tablespaces	11-26
Concept: Minimum Compatibility Level	11-27
Transportable Tablespace Procedure	11-28
Determining the Endian Format of a Platform	11-29
Transporting Databases	11-30
Database Transportation Procedure: Source System Conversion	11-31
Database Transportation Procedure: Target System Conversion	11-32
Database Transportation: Considerations	11-33
Summary	11-34
Practice Overview: Managing Storage	11-35

12 Automatic Storage Management

Objectives	12-2
Automatic Storage Management: Review	12-3
ASM General Architecture	12-5
ASM Instance Tasks	12-7
Creating an ASM Instance	12-8
ASM Instance Initialization Parameters	12-9
Database Instance Parameter Changes	12-10
Starting Up an ASM Instance	12-11
Accessing an ASM Instance	12-12
ASM Home Page	12-14
ASM Performance Page	12-15
ASM Configuration Page	12-16
Shutting Down an ASM Instance	12-17
DBCA and Storage Options	12-19
ASM Storage: Concepts	12-20
ASM Disk Groups	12-21
Failure Group	12-22
Disk Group Mirroring	12-23
Disk Group Dynamic Rebalancing	12-24
Managing Disk Groups	12-25

ASM Administration Page	12-26
Create Disk Group Page	12-27
Creating and Dropping Disk Groups	12-28
Adding Disks to Disk Groups	12-29
Miscellaneous ALTER Commands	12-31
ASM Files	12-33
ASMCMD Utility	12-34
Migrating Your Database to ASM Storage	12-35
Summary	12-37
Practice Overview: Using Automatic Storage Management	12-38

13 Managing Resources

Objectives	13-2
Database Resource Manager: Overview	13-3
Database Resource Manager Concepts	13-4
Why Use Resource Manager	13-5
Accessing Resource Plans	13-7
Example: SYSTEM_PLAN	13-8
Creating a New Resource Plan	13-9
Creating Consumer Groups	13-10
Assigning Users to Consumer Groups	13-11
Specifying Resource Plan Directives	13-12
Resource Allocation Methods for Resource Plans	13-13
Comparison of EMPHASIS and RATIO	13-14
Active Session Pool Mechanism	13-16
Setting the Active Session Pool	13-17
Maximum Estimated Execution Time	13-18
Consumer Group Switching	13-19
Switching Back to the Initial Consumer Group at the End of Call	13-20
Setting Idle Timeouts	13-22
Resource Consumer Group Mapping	13-23
Activating a Resource Plan for an Instance	13-25
Database Resource Manager Information	13-26
Monitoring the Resource Manager	13-27
Summary	13-30
Practice Overview: Using the Resource Manager	13-31

14 Automating Tasks with the Scheduler

Objectives	14-2
Simplifying Management Tasks	14-3

A Simple Job	14-4
Key Components and Steps	14-5
1. Creating a Program	14-6
2. Creating and Using Schedules	14-7
3. Creating and Running a Job	14-8
4. Monitoring a Job	14-9
Using a Time-Based or Event-Based Schedule	14-10
Creating a Time-Based Job	14-11
Creating an Event-Based Schedule	14-13
Creating Event-Based Schedules with Enterprise Manager	14-14
Creating an Event-Based Job	14-15
Event-Based Scheduling	14-16
Creating Complex Schedules	14-18
Creating Job Chains	14-19
Example of a Chain	14-21
1. Creating a Chain Object	14-22
2. Defining Chain Steps	14-23
3. Defining Chain Rules	14-24
4. Starting the Chain	14-25
Monitoring Job Chains	14-26
Advanced Scheduler Concepts	14-27
Creating a Job Class	14-28
Creating a Window	14-29
Prioritizing Jobs Within a Window	14-30
Summary	14-31
Practice Overview: Automating Tasks with the Scheduler	14-32

15 Database Security

Objectives	15-2
Oracle Transparent Data Encryption (TDE): Overview	15-3
TDE Process	15-5
Implementing Transparent Data Encryption	15-6
Existing Tables and TDE	15-9
Transparent Data Encryption: Considerations	15-10
Wallet Support for Usernames and Passwords	15-11
Data Pump and Transparent Data Encryption	15-12
RMAN Encrypted Backups: Overview	15-13
Transparent Mode Setup	15-14
Password Mode Setup	15-15
Dual Mode Setup	15-16
RMAN-Encrypted Backups: Considerations	15-17

Need for Data Privacy	15-18
Definition and Usage of Terms	15-19
Virtual Private Database: Overview	15-20
Virtual Private Database: Features	15-21
Column-Level VPD: Example	15-22
Creating a Column-Level Policy	15-23
Summary	15-24
Practice Overview: Using Oracle Database Security	15-25

16 Using Globalization Support

Objectives	16-2
What Every DBA Needs to Know	16-3
What Is a Character Set?	16-4
Understanding Unicode	16-6
How Are Character Sets Used?	16-8
Problems to Avoid	16-9
Another Sample Problem	16-10
Choosing Your Character Set	16-11
Database Character Sets and National Character Sets	16-12
Obtaining Character Set Information	16-13
Specifying Language-Dependent Behavior	16-14
Specifying Language-Dependent Behavior for the Session	16-15
Language- and Territory-Dependent Parameters	16-16
Specifying Language-Dependent Behavior	16-18
Linguistic Searching and Sorting	16-19
Using Linguistic Searching and Sorting	16-21
Case- and Accent-Insensitive Search and Sort	16-23
Support in SQL and Functions	16-24
Linguistic Index Support	16-25
Customizing Linguistic Searching and Sorting	16-26
Implicit Conversion Between CLOB and NCLOB	16-27
NLS Data Conversion with Oracle Utilities	16-28
NLS Data Conversion with Data Pump	16-30
Globalization Support Features	16-31
Summary	16-32
Practice Overview: Using Globalization Support Features	16-33

17 Workshop

Objectives	17-2
Workshop Methodology	17-3
Business Requirements	17-5

Database Configuration	17-6
Method for Resolving Database Issues	17-7
Summary	17-9
Practice Overview: Workshop Setup	17-10

Appendix A: Practices

Appendix B: Solutions

Appendix C: Basic Linux and vi Commands

Appendix D: Acronyms and Terms

Appendix E: Oracle Shared Servers

Objectives	E-2
Establishing a Connection	E-3
Dedicated Server Process	E-4
User Sessions	E-5
User Sessions: Dedicated Server	E-6
User Sessions: Shared Server	E-7
Processing a Request	E-8
SGA and PGA	E-9
UGA and Oracle Shared Server	E-10
Configuring Oracle Shared Server	E-11
DISPATCHERS	E-12
SHARED_SERVERS	E-14
MAX_SHARED_SERVERS	E-15
CIRCUITS	E-16
SHARED_SERVER_SESSIONS	E-17
Related Parameters	E-18
Verifying Shared Server Setup	E-19
Data Dictionary Views	E-21
Choosing a Connection Type	E-22
When Not to Use Shared Server	E-23
Summary	E-24

Appendix F: Oracle Secure Backup

Objectives	F-2
Data Protection to Tape for the Oracle Stack	F-3
The Customer Advantage: Complete Oracle Solution	F-4
Oracle Secure Backup for Centralized Tape Backup Management	F-5
Oracle Secure Backup Administrative Domain	F-6

Oracle Secure Backup: Backup Management Overview	F-7
Oracle Secure Backup Catalog	F-8
Oracle Secure Backup Users	F-9
Predefined Classes	F-11
Oracle Secure Backup Interface Options	F-12
Managing Data to Be Protected	F-13
Oracle Secure Backup Media Concepts	F-14
Volume Set Recycling	F-15
Automated Device Management	F-17
Library Management Operations	F-18
Oracle Secure Backup: Installation	F-19
Installing Oracle Secure Backup Software	F-20
Administrative Server Installation: Example	F-21
Defining Your Administrative Server in EM	F-22
The Oracle Secure Backup Device and Media Page	F-23
Adding Devices	F-24
Managing Devices by Using EM	F-25
RMAN and Oracle Secure Backup	F-26
Accessing Oracle Secure Backup from RMAN	F-27
User Preauthorization	F-28
Database Backup Storage Selector	F-29
RMAN and Oracle Secure Backup: Usage Model	F-30
Defining Database Storage Selector	F-31
Testing Your Tape Drives	F-32
Scheduling Backups by Using EM Database Control	F-33
Oracle-Suggested Strategy for Backups	F-34
Managing Tape Backups	F-35
Performing Database Recovery by Using Tape Backups	F-36
Backing Up File System Files with Oracle Secure Backup	F-37
Oracle Secure Backup Web Tool	F-38
Oracle Secure Backup Data Set Scripts	F-39
Data Set Script: Examples	F-40
Data Set Organization	F-41
Creating Data Sets Using the Web Interface	F-42
File System Files: Backup Concepts	F-43
Oracle Secure Backup Jobs	F-44
Creating On-Demand Backup Requests	F-46
Sending Backup Requests to the Scheduler	F-47
Creating Backup Schedules	F-48
Creating Backup Triggers	F-49
Viewing Job Properties and Transcripts	F-50

Restoring File System Files with Oracle Secure Backup	F-51
Creating a Catalog-Based Restore Request	F-52
Sending Catalog-Based Restore Requests to the Scheduler	F-55
Listing All Backups of a Client	F-56
Summary	F-57

Appendix G: Miscellaneous Topics

Appendix Overview	G-2
Bigfile Tablespaces: Overview	G-3
Benefits of Bigfile Tablespaces	G-4
Bigfile Tablespace: Usage Model	G-5
Creating Bigfile Tablespaces	G-7
SQL Statement Clauses	G-8
BFTs and SQL Statements: Examples	G-9
Data Dictionary Additions To Support VLDB	G-10
Extended ROWID Format and BFTs	G-11
VLDB Support: DBMS_ROWID Package	G-13
Temporary Tablespace Group (TTG): Overview	G-14
Temporary Tablespace Group: Benefits	G-15
Creating and Maintaining Temporary Tablespace Groups	G-16
Temporary Tablespace Group: SQL Examples	G-17
Summary	G-20

Appendix H: Next Steps: Continuing Your Education

Where Do You Go from Here?	H-2
Continuing Education Resources	H-3
Oracle University	H-4
Continuing Your Education	H-5
Oracle University Knowledge Center	H-6
Oracle Technology Network	H-7
Oracle Technology Training	H-8
Oracle by Example	H-9
Oracle Magazine	H-10
Oracle Applications Community	H-11
Technical Support: Oracle MetaLink	H-12
Thank You!	H-13

Index

Preface

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a
non-transferable license to use this Student Guide.

Unauthorized reproduction or distribution prohibited. Copyright© 2010, Oracle and/or its affiliates.

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a
non-transferable license to use this Student Guide.

Profile

Before You Begin This Course

Before you begin this course, you should have the following qualifications:

- Working experience with SQL and knowledge of basic Unix commands

Prerequisites

Oracle Database 10g: Administration Workshop I (D17090GC30)

How This Course Is Organized

Oracle Database 10g: Administration Workshop I is an instructor-led course featuring lecture and hands-on exercises. Online demonstrations and written practice sessions reinforce the concepts and skills introduced.

Suggested Next Courses

- *Oracle Database 10g: SQL Tuning Workshop* (D17265GC10)
- *Oracle Enterprise Manager 10g Grid Control* (D17244GC11)
- *Oracle Database 10g: Real Application Clusters* (D17276GC10)
- *Oracle Database 10g: Implement Streams* (D17333GC10)
- *Oracle Database 10g: Data Guard Administration* (D17316GC11)

Related Publications

Oracle Publications

Title	Part Number
<i>Oracle Database 2 Day DBA 10g Release 2 (10.2)</i>	B14196-02
<i>Oracle Database Administrator's Guide 10g Release 2 (10.2)</i>	B14231-01
<i>Oracle Database Application Developer's Guide - Large Objects 10g Release 2 (10.2)</i>	B14249-01
<i>Oracle Database Backup and Recovery Advanced User's Guide 10g Release 2 (10.2)</i>	B14191-02
<i>Oracle Database Backup and Recovery Basics 10g Release 2 (10.2)</i>	B14192-03
<i>Oracle Database Concepts 10g Release 2 (10.2)</i>	B14220-02
<i>Oracle Database Data Warehousing Guide 10g Release 2 (10.2)</i>	B14223-02
<i>Oracle Database Globalization Support Guide 10g Release 2 (10.2)</i>	B14225-01
<i>Oracle Database Licensing Information 10g Release 2 (10.2)</i>	B14199-02
<i>Oracle Database Net Services Administrator's Guide 10g Release 2 (10.2)</i>	B14212-02
<i>Oracle Database Net Services Reference 10g Release 2 (10.2)</i>	B14213-01
<i>Oracle Database New Features Guide 10g Release 2 (10.2)</i>	B14214-02
<i>Oracle Database Performance Tuning Guide 10g Release 2 (10.2)</i>	B14211-01
<i>Oracle Database PL/SQL Packages and Types Reference 10g Release 2 (10.2)</i>	B14258-01
<i>Oracle Database PL/SQL User's Guide and Reference 10g Release 2 (10.2)</i>	B14261-01
<i>Oracle Database Recovery Manager Quick Start Guide 10g Release 2 (10.2)</i>	B14193-03
<i>Oracle Database Recovery Manager Reference 10g Release 2 (10.2)</i>	B14194-03
<i>Oracle Database Reference 10g Release 2 (10.2)</i>	B14237-02
<i>Oracle Database Security Guide 10g Release 2 (10.2)</i>	B14266-01
<i>Oracle Database SQL Quick Reference 10g Release 2 (10.2)</i>	B14195-02
<i>Oracle Database SQL Reference 10g Release 2 (10.2)</i>	B14200-02
<i>Oracle Database Utilities 10g Release 2 (10.2)</i>	B14215-01
<i>Oracle Streams Advanced Queuing User's Guide and Reference</i>	B14257-01
<i>Oracle Streams Concepts and Administration</i>	B14229-01

Additional Publications

- System release bulletins
- Installation and user guides
- *read.me* files
- International Oracle Users Group (IOUG) articles
- *Oracle Magazine*

Typographic Conventions

Typographic Conventions in Text

Convention	Element	Example
Bold	Emphasized words and phrases in Web content only	To navigate within this application, do not click the Forward and Back buttons.
Bold italic	Glossary term (if there is a glossary)	The <i>algorithm</i> inserts the new key.
Brackets	Key names	Press [Enter].
Caps and lowercase	Buttons, check boxes, application triggers, windows	Click the Executable button. Select the Can't Delete Card check box. Assign a When-Validate-Item trigger to the ORD block. Open the Master Schedule window.
Angle brackets	Menu paths	Select File > Save.
Commas	Key sequences	Press and release the following keys one at a time: [Alt], [F], [D]
Courier new, case sensitive (default is lowercase)	Code output, directory names, file names, passwords, path names, user input, usernames	Code output: debug.set ('I', 300); Directory: bin (DOS), \$FMHOME (UNIX) File name: Locate the init.ora file. Password: Use tiger as your password. Path name: Open c:\my_docs\projects. User input: Enter 300. Username: Log in as HR.
Initial cap	Graphics labels (unless the term is a proper noun)	Customer address (<i>but</i> Oracle Payables)
Italic	Emphasized words and phrases, titles of books and courses, variables	Do <i>not</i> save changes to the database. For further information, see <i>Oracle Database SQL Reference 10g Release 2(10.2)</i> . Enter <i>user_id@us.oracle.com</i> , where <i>user_id</i> is the name of the user.

Typographic Conventions (continued)

Typographic Conventions in Text (continued)

Convention	Element	Example
Quotation marks	Interface elements with long names that have only initial caps; lesson and chapter titles in cross-references	Select “Include a reusable module component” and click Finish. This subject is covered in the lesson titled, “Working with Objects.”
Uppercase	SQL column names, commands, functions, schemas, table names, database trigger names	Use the SELECT command to view information stored in the LAST_NAME column of the EMPLOYEES table.

Typographic Conventions in Code

Convention	Element	Example
Lowercase	Column names, table names, database trigger names	SELECT last_name FROM employees; CREATE OR REPLACE TRIGGER secure_employees
	Passwords	CREATE USER scott IDENTIFIED BY tiger;
	PL/SQL objects	items.DELETE(3);
Lowercase italic	Syntax variables	CREATE ROLE role
Uppercase	SQL commands and functions	SELECT first_name FROM employees;

Typographic Conventions (continued)

Typographic Conventions in Navigation Paths

This course uses simplified navigation paths, such as the following example, to direct you through Oracle applications.

Example:

Invoice Batch Summary

(N) Invoice > Entry > Invoice Batches Summary (M) Query > Find (B) Approve

This simplified path translates to the following:

1. (N) From the Navigator window, select Invoice > Entry > Invoice Batches Summary.
2. (M) From the menu, select Query > Find.
3. (B) Click the Approve button.

Notation:

(N) = Navigator	(I) = Icon
(M) = Menu	(H) = Hyperlink
(T) = Tab	(B) = Button

Unauthorized reproduction or distribution prohibited. Copyright© 2010, Oracle and/or its affiliates.

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a
non-transferable license to use this Student Guide.

1

Introduction

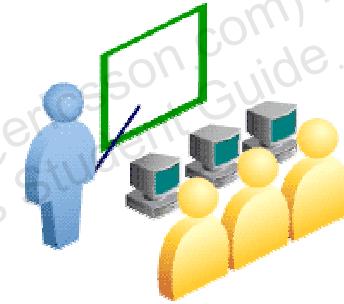
ORACLE®

Copyright © 2008, Oracle. All rights reserved.

Lesson Objectives

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

- **List the course objectives and explain the class structure**
- **Review the Oracle Database 10g architecture**



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Course Objectives

In this course, you gain experience in:

- **Using Recovery Manager (RMAN) for advanced backup and recovery**
- **Employing database monitoring practices for memory, performance, and storage**
- **Managing resources, job schedules, security, and globalization issues**



Copyright © 2008, Oracle. All rights reserved.

Suggested Schedule

1

- 1: Introduction**
- 2: Configuring RMAN**
- 3: Using RMAN**
- 4: Non-Critical Recovery**

2

- 5: Recovery**
- 6: Flashback**
- 7: Corruption**
- 8: Memory**

3

- 9: Performance**

4

- 10: Schema**
- 11: Storage**
- 12: ASM**

5

- 13: Resources**
 - 14: Scheduler**
 - 15: Security**
- 16: Globalization**
 - 17: Workshop**
 - Appendixes**

ORACLE®

Copyright © 2008, Oracle. All rights reserved.

What Is Covered in the DBAI Course

1

- 1. Introduction
- 2. Installation
- 3. DB Creation
- 4. Instance

2

- 5. Storage
- 6. Users
- 7. Schema
- 8. Data & Concurrency

3

- 9. Undo
- 10. Security
- 11. Network
- 12. Proactive Maintenance

4

- 13. Performance
- 14. Backup & Recovery Concepts
- 15. Backup

5

- 16. Recovery
- 17. Flashback
- 18. Moving Data

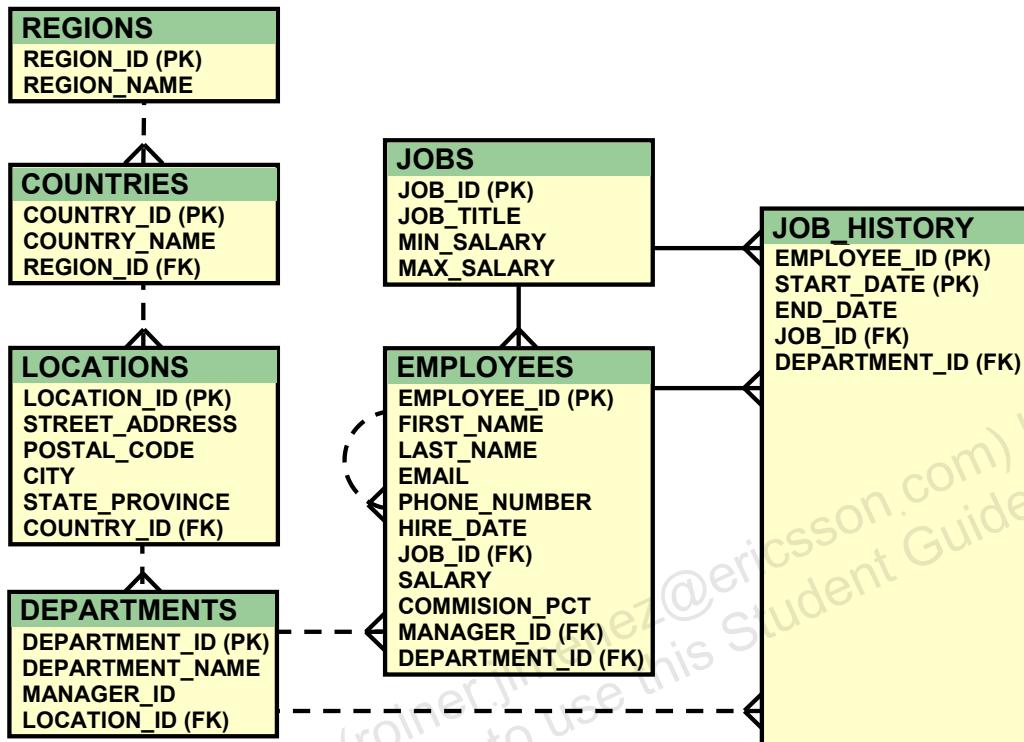
ORACLE®

Copyright © 2008, Oracle. All rights reserved.

What Is Covered in the DBAI Course

DBAI refers to the *Oracle Database 10g: Administration Workshop I* (Release 2) course.

Course Examples: The HR Schema



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Course Examples: The HR Sample Schema

The examples used in this course are from a human resources (HR) application, which can be created as part of the starter database.

The following are some principal business rules of the HR application:

- Each department may be the employer of one or more employees. Each employee may be assigned to one and only one department.
- Each job must be a job for one or more employees. Each employee must be currently assigned to one and only one job.
- When an employee changes his or her department or job, a record in the JOB_HISTORY table records the start and end dates of the past assignments.
- JOB_HISTORY records are identified by a composite primary key (PK): the EMPLOYEE_ID and the START_DATE columns.

Notation: PK = primary key, FK = foreign key

Solid lines represent mandatory foreign key (FK) constraints and dashed lines represent optional FK constraints.

The EMPLOYEES table also has an FK constraint with itself. This is an implementation of the business rule: Each employee may be reporting directly to one and only one manager. The FK is optional because the top employee does not report to another employee.

Oracle Database 10g: The Database for the Grid

- **Automatic Storage Management**
- **Portable clusterware**
- **Real Application Clusters and automatic workload management**
- **Resource Manager**
- **Oracle Streams**
- **Centralized management with Enterprise Manager Grid Control**
- **Oracle Database 10g new self-management features**

ORACLE

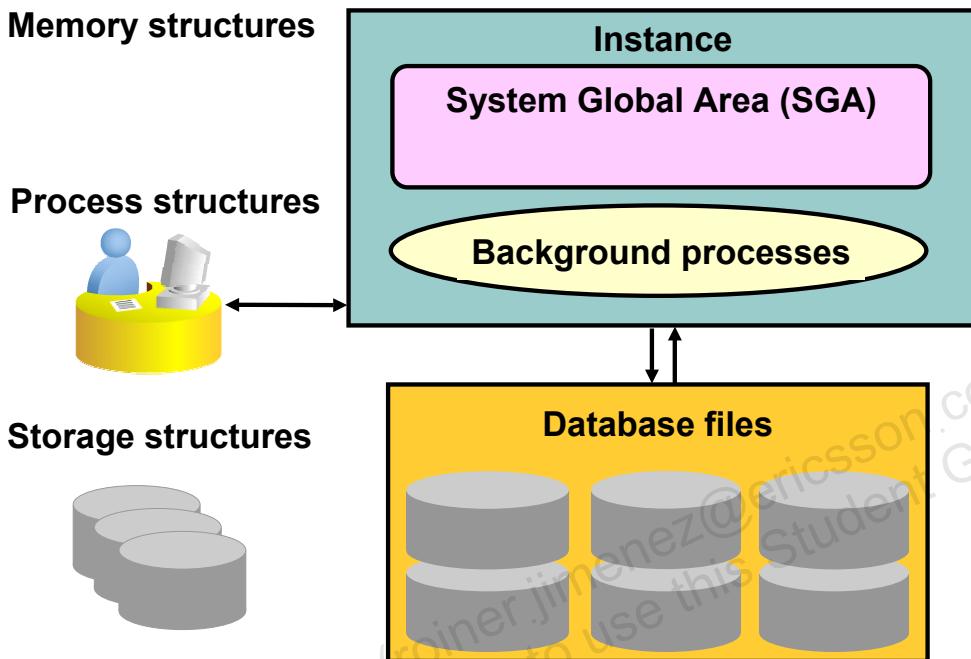
Copyright © 2008, Oracle. All rights reserved.

Oracle Database 10g: The Database for the Grid

Oracle Database 10g is the first database that is designed for grid computing. To summarize, some of the most important features are the following:

- Automatic Storage Management (ASM) virtualizes your storage and provides easy provisioning of your database storage.
- Oracle Database 10g offers portable clusterware that runs on all platforms.
- Oracle Database 10g offers automatic workload management for services within a RAC database.
- Oracle Database 10g provides additional mappings for consumer groups based on user host machine, application, OS username, or service.
- Oracle Streams can stream data between databases, nodes, or blade farms in a grid. It provides a unified framework for information sharing, combining message queuing, replication, events, and data warehouse loading into a single technology.
- Enterprise Manager Grid Control provides a single tool that can monitor and manage not only every Oracle software element (Oracle Application Server 10g and Oracle Database 10g) in your grid but also Web applications via Application Performance Management (APM), hosts, storage devices, and server load balancers.

Database Architecture: Review



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Database Architecture: Review

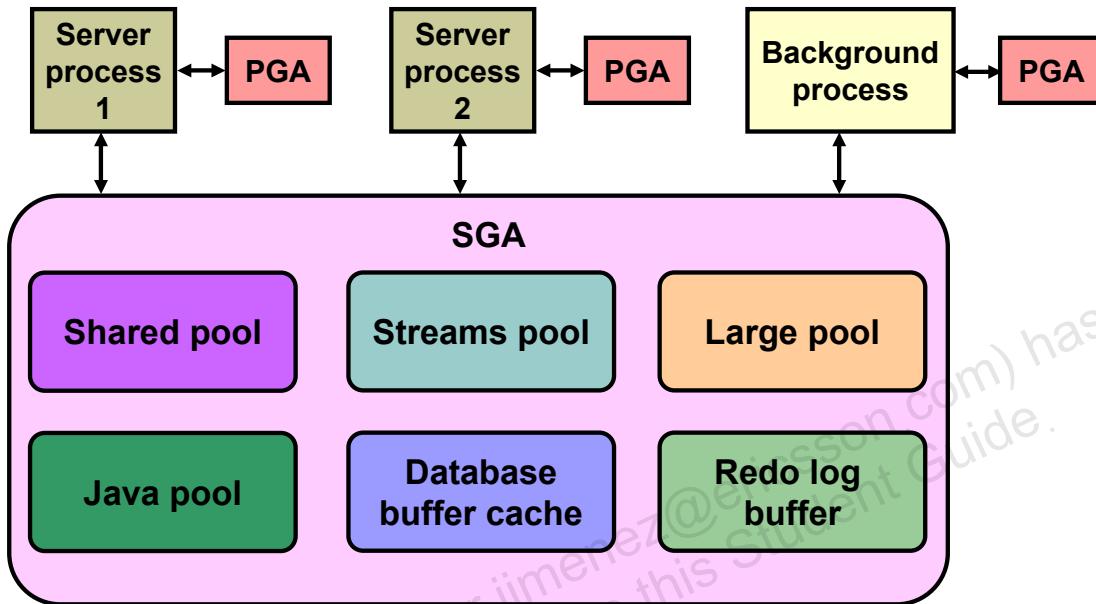
The following pages are a basic review of the Oracle database architecture. In this course, you enhance your knowledge of Oracle's database structures, processes, and utilities.

Each running Oracle database is associated with an Oracle instance. When a database is started on a database server, the Oracle software allocates a shared memory area called the System Global Area (SGA) and starts several Oracle background processes. This combination of the SGA and the Oracle processes is called an Oracle instance.

After starting an instance, the Oracle software associates the instance with a specific database. This is called mounting the database. The database is then ready to be opened, which makes it accessible to authorized users. Multiple instances can execute concurrently on the same computer, each accessing its own physical database.

You can look at the Oracle database architecture as various interrelated structural components. An Oracle database uses memory structures and processes to manage and access the database. All memory structures exist in the main memory of the computers that constitute the database server. Processes are jobs that work in the memory of these computers. A process is defined as a "thread of control" or a mechanism in an operating system that can run a series of steps.

Oracle Memory Structures



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Oracle Memory Structures

The basic memory structures associated with an Oracle instance include:

- **System Global Area (SGA):** Shared by all server and background processes
- **Program Global Area (PGA):** Private to each server and background process; there is one PGA for each process

The SGA is a memory area that contains data and control information for the instance.

The SGA includes the following data structures:

- **Database buffer cache:** Caches blocks of data retrieved from the database
- **Redo log buffer:** Caches redo information (used for instance recovery) until it can be written to the physical redo log files stored on the disk
- **Shared pool:** Caches various constructs that can be shared among users
- **Large pool:** Is an optional area that provides large memory allocations for certain large processes, such as Oracle backup and recovery operations, and I/O server processes
- **Java pool:** Is used for all session-specific Java code and data within the Java Virtual Machine (JVM)
- **Streams pool:** Is used by Oracle Streams

When you start the instance by using Enterprise Manager or SQL*Plus, the amount of memory allocated for the SGA is displayed.

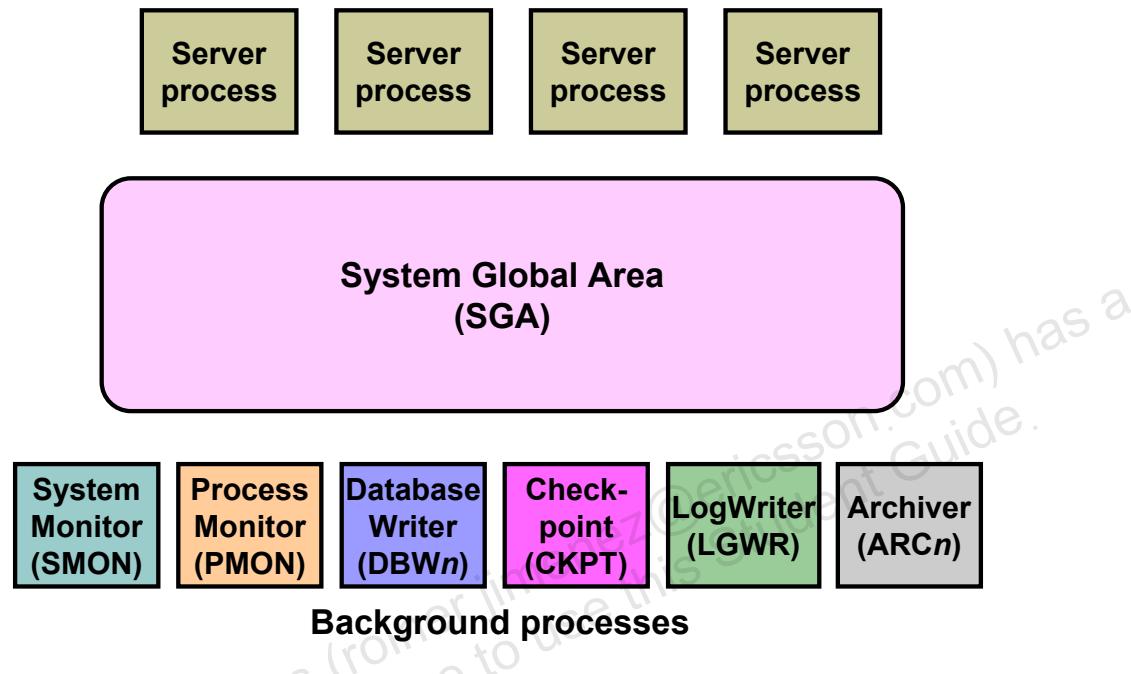
Oracle Memory Structures (continued)

A Program Global Area (PGA) is a memory region that contains data and control information for each server process. An Oracle server process services a client's requests. Each server process has its own private PGA that is created when the server process is started. Access to the PGA is exclusive to that server process, and the PGA is read and written only by the Oracle code acting on its behalf.

With the dynamic SGA infrastructure, the size of the database buffer cache, the shared pool, the large pool, the Java pool, and the Streams pool changes without shutting down the instance.

The Oracle database uses initialization parameters to create and configure memory structures. For example, the `SGA_TARGET` parameter specifies the total amount of space available to the SGA. If you set `SGA_TARGET` to 0, Automatic Shared Memory Management is disabled.

Oracle Processes



ORACLE

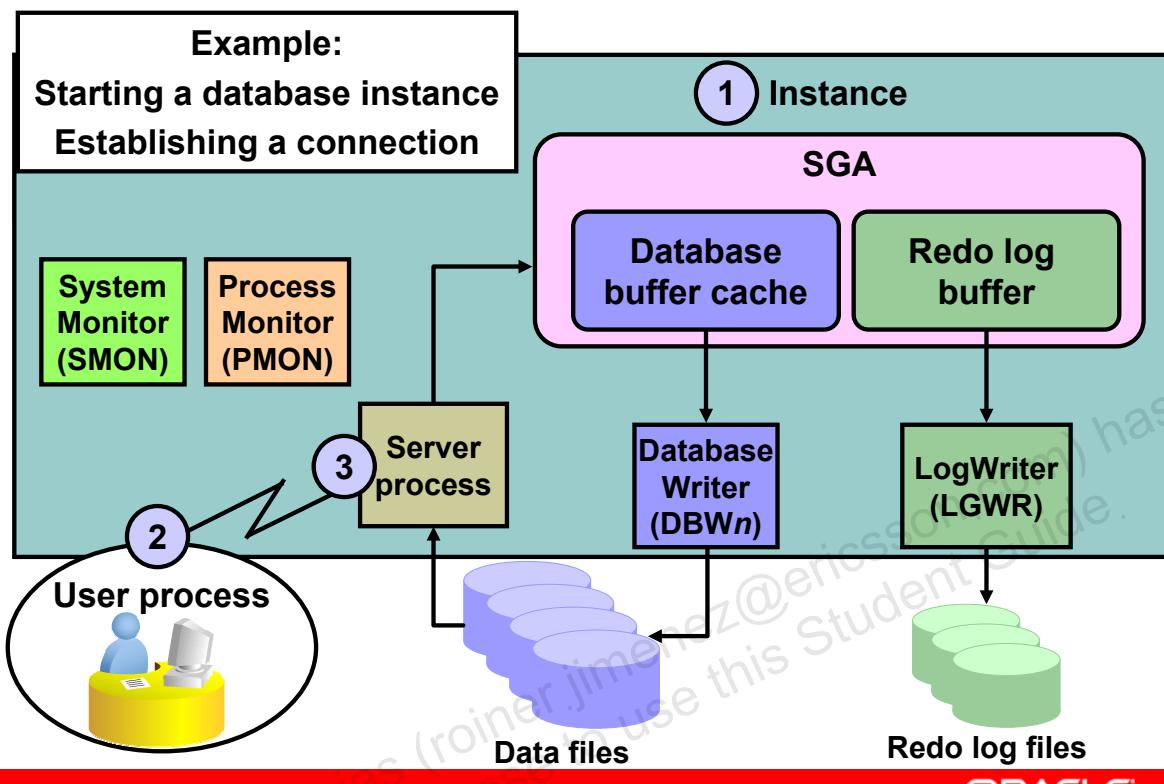
Copyright © 2008, Oracle. All rights reserved.

Oracle Processes

When you invoke an application program or an Oracle tool, such as Enterprise Manager, the Oracle server creates a server process to execute the commands issued by the application. The Oracle server also creates a set of background processes for an instance that interact with each other and with the operating system to manage the memory structures, asynchronously perform I/O to write data to disk, and perform other required tasks. Which background processes are present depends on the features that are being used in the database. The most common background processes are the following:

- **System Monitor (SMON)**: Performs crash recovery when the instance is started following a failure
- **Process Monitor (PMON)**: Performs process cleanup when a user process fails
- **Database Writer (DBWn)**: Writes modified blocks from the database buffer cache to the data files on the disk
- **Checkpoint (CKPT)**: Updates all the data files and control files of the database to indicate the most recent checkpoint
- **LogWriter (LGWR)**: Writes redo log entries to the disk
- **Archiver (ARCn)**: Copies redo log files to an archival storage when a log switch occurs

Reviewing Oracle Instance Management



Copyright © 2008, Oracle. All rights reserved.

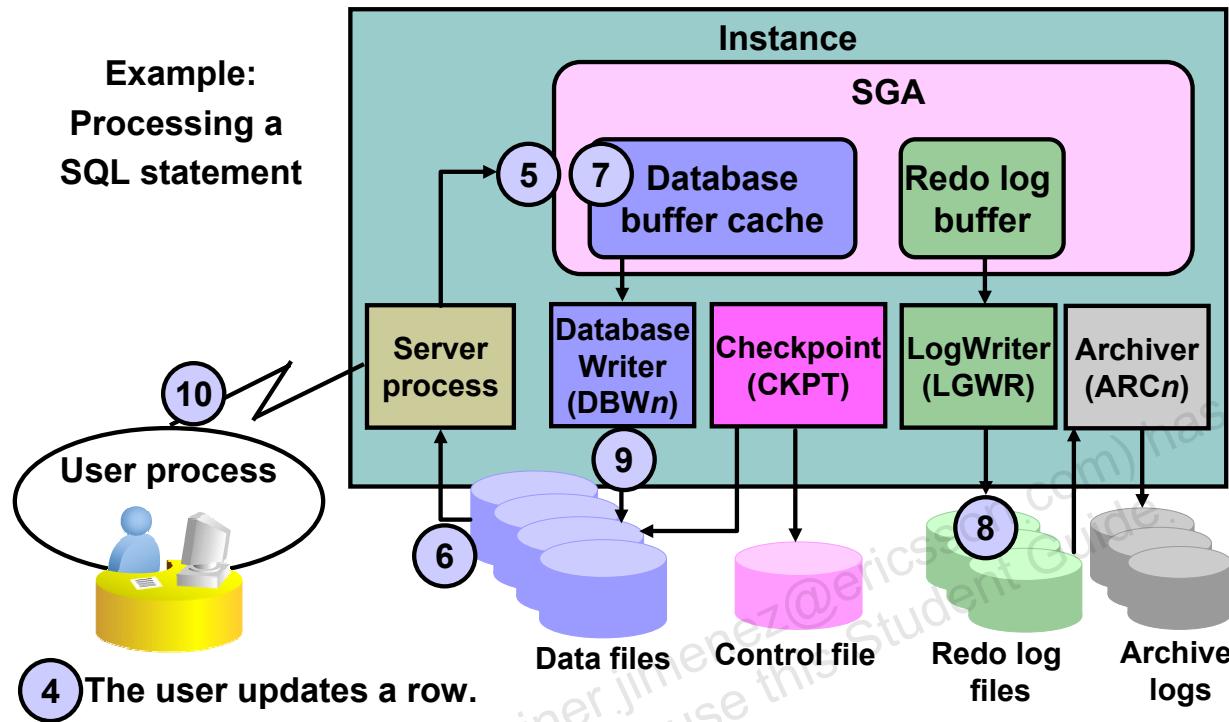
Reviewing Oracle Instance Management

The following example describes the most basic level of operations that the Oracle database performs. It illustrates an Oracle configuration where the user and associated server processes are on separate computers (connected through a network).

1. An instance has started on the computer running Oracle (often called the host or database server).
2. A computer running an application (a local computer or client workstation) runs the application in a user process. The client application attempts to establish a connection to the instance by using the Oracle Net Services driver.
3. The instance detects the connection request from the application and connects to a server process on behalf of the user process.

Reviewing Oracle Instance Management

Example:
Processing a
SQL statement



ORACLE

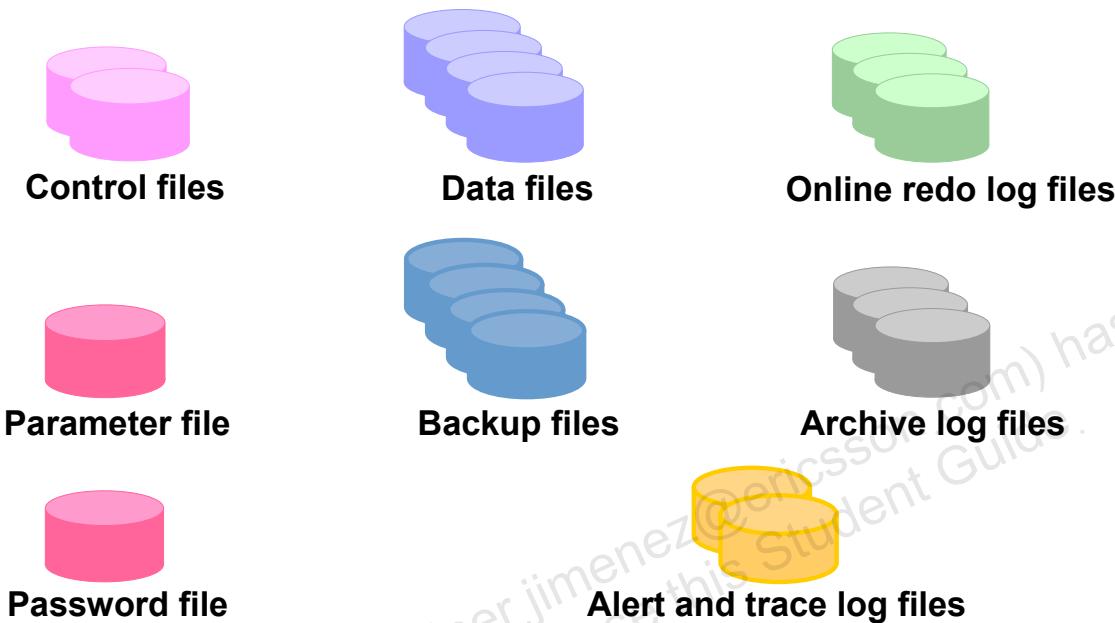
Copyright © 2008, Oracle. All rights reserved.

Reviewing Oracle Instance Management (continued)

4. The user updates a row.
5. The server process receives the statement and checks whether it is already in the shared pool of the SGA. If a shared SQL area is found, the server process checks the user's access privileges to the requested data, and the previously existing shared SQL area is used to process the statement. If the statement is not in the shared pool, then a new shared SQL area is allocated for the statement, so that it can be parsed and processed.
6. The server process retrieves any necessary data values from the actual data file (table) or from data blocks that are stored in the SGA.
7. The server process modifies the table data in the SGA.
8. When the transaction is committed, the LGWR process immediately records the transaction in the redo log file.
9. The DBWn process writes modified blocks to the disk when doing so is efficient.
10. The server process sends a success or error message across the network to the application.

Throughout this entire procedure, the other background processes run, watching for conditions that require intervention.

Physical Database Structure



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Physical Database Structure

The files that constitute an Oracle database are organized into the following:

- **Control files:** Contain data about the database itself (that is, physical database structure information). These files are critical to the database. Without them, you cannot open data files to access the data within the database.
- **Data files:** Contain the user or application data of the database
- **Online redo log files:** Allow for instance recovery of the database. If the database crashes and does not lose any data files, then the instance can recover the database with the information in these files.

The following additional files are important to the successful running of the database:

- **Parameter file:** Is used to define how the instance is configured when it starts up
- **Password file:** Allows users to connect remotely to the database and perform administrative tasks
- **Backup files:** Are used for database recovery. You typically restore a backup file when a media failure or user error has damaged or deleted the original file.
- **Archive log files:** Contain an ongoing history of the data changes (redo) that are generated by the instance. Using these files and a backup of the database, you can recover a lost data file. That is, archive logs enable the recovery of restored data files.

Physical Database Structure (continued)

- **Trace files:** Each server and background process can write to an associated trace file. When an internal error is detected by a process, the process dumps information about the error to its trace file. Some of the information written to a trace file is intended for the database administrator, whereas other information is for Oracle Support Services.
- **Alert log files:** Also known as alert logs, these are special trace files. The alert log of a database is a chronological log of messages and errors. Oracle recommends reviewing these files.

Oracle Managed Files (OMF)

Specify file operations in terms of database objects rather than file names.

Parameter	Description
<code>DB_CREATE_FILE_DEST</code>	Defines the location of the default file system directory for data files and temporary files
<code>DB_CREATE_ONLINE_LOG_DEST_n</code>	Defines the location for redo log files and control file creation
<code>DB_RECOVERY_FILE_DEST</code>	Defines the location for RMAN backups

Example:

```
SQL> ALTER SYSTEM SET DB_CREATE_FILE_DEST = '/u01/oradata';
SQL> CREATE TABLESPACE tbs_1;
```

Copyright © 2008, Oracle. All rights reserved.

Oracle Managed Files (OMF)

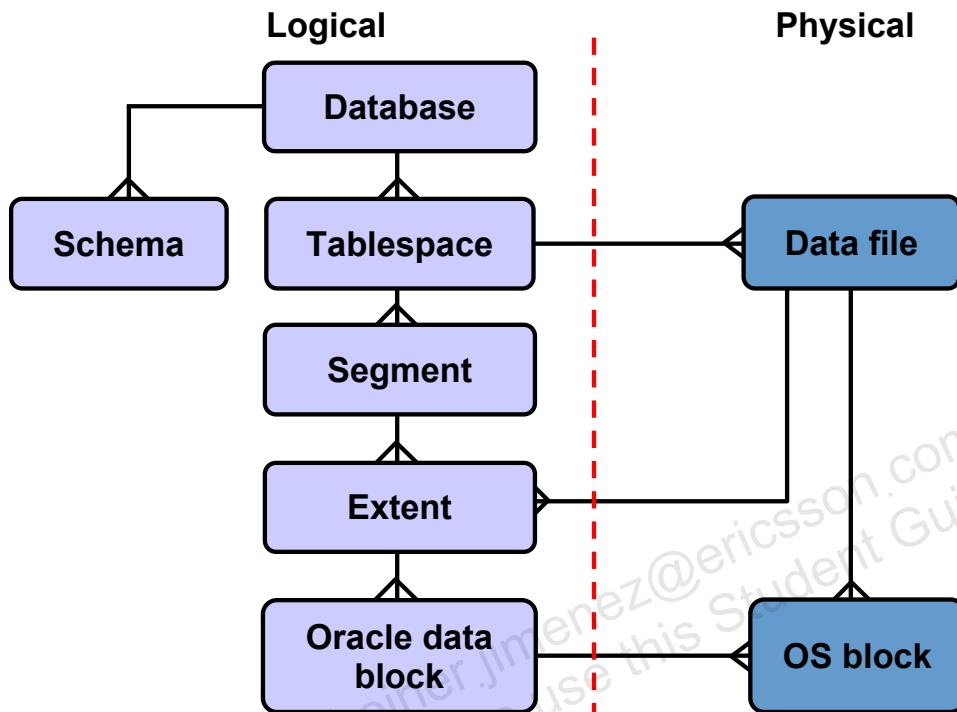
Oracle Managed Files (OMF) eliminate the need for you to directly manage the operating system files that make up an Oracle database. You specify operations in terms of database objects rather than file names. The database internally uses standard file system interfaces to create and delete files as needed for the following database structures:

- Tablespaces
- Redo log files
- Control files
- Archived logs
- Block change tracking files
- Flashback logs
- RMAN backups

A database can have a mixture of Oracle-managed and unmanaged files. The file system directory specified by either of these parameters must already exist: the database does not create it. The directory must also have permissions to allow the database to create the files in it.

The example shows that after `DB_CREATE_FILE_DEST` is set, the `DATAFILE` clause can be omitted from a `CREATE TABLESPACE` statement. The data file is created in the location specified by `DB_CREATE_FILE_DEST`.

Logical and Physical Database Structures



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Logical and Physical Database Structures

An Oracle database is a collection of data that is treated as a unit. The general purpose of a database is to store and retrieve related information. The database has logical structures and physical structures.

Tablespaces

A database is divided into logical storage units called tablespaces, which group related logical structures together. For example, tablespaces commonly group all of an application's objects to simplify some administrative operations. You may have a tablespace for application data and an additional one for application indexes.

Databases, Tablespaces, and Data Files

The relationship among databases, tablespaces, and data files is illustrated in the slide. Each database is logically divided into one or more tablespaces. One or more data files are explicitly created for each tablespace to physically store the data of all logical structures in a tablespace. If it is a TEMPORARY tablespace, instead of a data file, the tablespace has a temporary file.

Logical and Physical Database Structures (continued)

Schemas

A schema is a collection of database objects that are owned by a database user. Schema objects are the logical structures that directly refer to the database's data. Schema objects include such structures as tables, views, sequences, stored procedures, synonyms, indexes, clusters, and database links. In general, schema objects include everything that your application creates in the database.

Data Blocks

At the finest level of granularity, an Oracle database's data is stored in data blocks. One data block corresponds to a specific number of bytes of physical database space on the disk. A data block size is specified for each tablespace when it is created. A database uses and allocates free database space in Oracle data blocks.

Extents

The next level of logical database space is called an extent. An extent is a specific number of contiguous data blocks (obtained in a single allocation) that are used to store a specific type of information.

Segments

The level of logical database storage above an extent is called a segment. A segment is a set of extents allocated for a certain logical structure. For example, the different types of segments include:

- **Data segments:** Each nonclustered, non-index-organized table has a data segment. All of the table's data is stored in the extents of its data segment. For a partitioned table, each partition has a data segment. Each cluster has a data segment. The data of every table in the cluster is stored in the cluster's data segment.
- **Index segments:** Each index has an index segment that stores all of its data. For a partitioned index, each partition has an index segment.
- **Undo segments:** One UNDO tablespace is created by the database administrator to temporarily store *undo* information. The information in an undo segment is used to generate read-consistent database information and, during database recovery, to roll back uncommitted transactions for users.
- **Temporary segments:** Temporary segments are created by the Oracle database when a SQL statement needs a temporary work area to complete execution. When the statement finishes execution, the temporary segment's extents are returned to the instance for future use. Specify a default temporary tablespace for every user or a default temporary tablespace, which is used databasewide.

The Oracle database dynamically allocates space. When the existing extents of a segment are full, additional extents are added. Because extents are allocated as needed, the extents of a segment may or may not be contiguous on the disk.

Database Architecture: Summary of Structural Components

- **Memory structures**
 - System Global Area (SGA): Database buffer cache, redo buffer, and various pools
 - Program Global Area (PGA)
- **Process structures**
 - User process and server process
 - Background processes: SMON, PMON, DBW n , CKPT, LGWR, ARC n , and so on
- **Storage structures**
 - Logical: Database, schema, tablespace, segment, extent, and Oracle block
 - Physical: Files for data, parameters, redo, and OS block



Copyright © 2008, Oracle. All rights reserved.

Database Architecture: Summary of Structural Components

You reviewed at a high level the structural components of the Oracle database: memory, process structures, and storage structures. An understanding of Oracle's database architecture is a prerequisite for this course.

Summary

In this lesson, you should have learned how to:

- **List the course objectives**
- **Review the Oracle Database 10g architecture**

Configuring Recovery Manager

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Objectives

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

- **Describe the RMAN repository and recovery catalog**
- **Describe the Media Management Library interface**
- **Configure database parameters that affect RMAN operations**
- **Connect to the three different types of databases by using RMAN**
- **Configure two types of retention policies**
- **Change RMAN default settings with CONFIGURE**



Copyright © 2008, Oracle. All rights reserved.

Objectives

Recovery Manager (RMAN) is the component of the Oracle database used to perform backup and recovery operations. Enterprise Manager (EM) supplies a graphical interface to the most commonly used RMAN functionality.

The *Oracle Database 10g: Administration Workshop I* course demonstrates basic RMAN operations using the EM Database Control Console, so the graphical interface to RMAN is not covered in detail in this lesson.

Backup and Recovery: Review

The major backup and recovery-related topics covered in the Database Administration I Course are:

- **Types of failure that can happen**
 - Statement, session, instance, media, and so on
- **How to configure ARCHIVELOG mode**
- **How to automate backups**
- **How to do incremental backups**
- **How to perform and tune instance recovery**



Copyright © 2008, Oracle. All rights reserved.

Backup and Recovery: Review

The *Oracle Database 10g: Administration Workshop I* course covered the backup and recovery-related topics listed in the slide. The following lessons cover some of these topics in greater detail.

Features of Recovery Manager

RMAN provides a flexible way to:

- **Back up the database including data files, control files, and archived redo logs**
- **Manage backup and recovery tasks**
- **Perform incremental block-level backup and block-level media recovery**
- **Detect corrupted blocks during backup**
- **Use binary compression when creating backups**



Copyright © 2008, Oracle. All rights reserved.

Features of Recovery Manager

RMAN is an Oracle utility that you use to manage the backup, restore, and recovery operations on Oracle databases. RMAN has a powerful command language that is independent of the operating system.

RMAN provides several features not available when you make user-managed backups with operating system–commands.

- You can store frequently executed operations as scripts in the database.
- With block change tracking enabled in the database, RMAN can limit incremental backups to recording only those blocks that have changed since the previous backup. This improves the performance of backups and may also reduce the time it takes to perform recovery operations in ARCHIVELOG mode.
- You can use RMAN to manage the size of backup pieces and save time by parallelizing the backup operation.
- RMAN can recover an individual corrupt data block or set of data blocks within a data file rather than restoring and recovering the entire data file.

Features of Recovery Manager (continued)

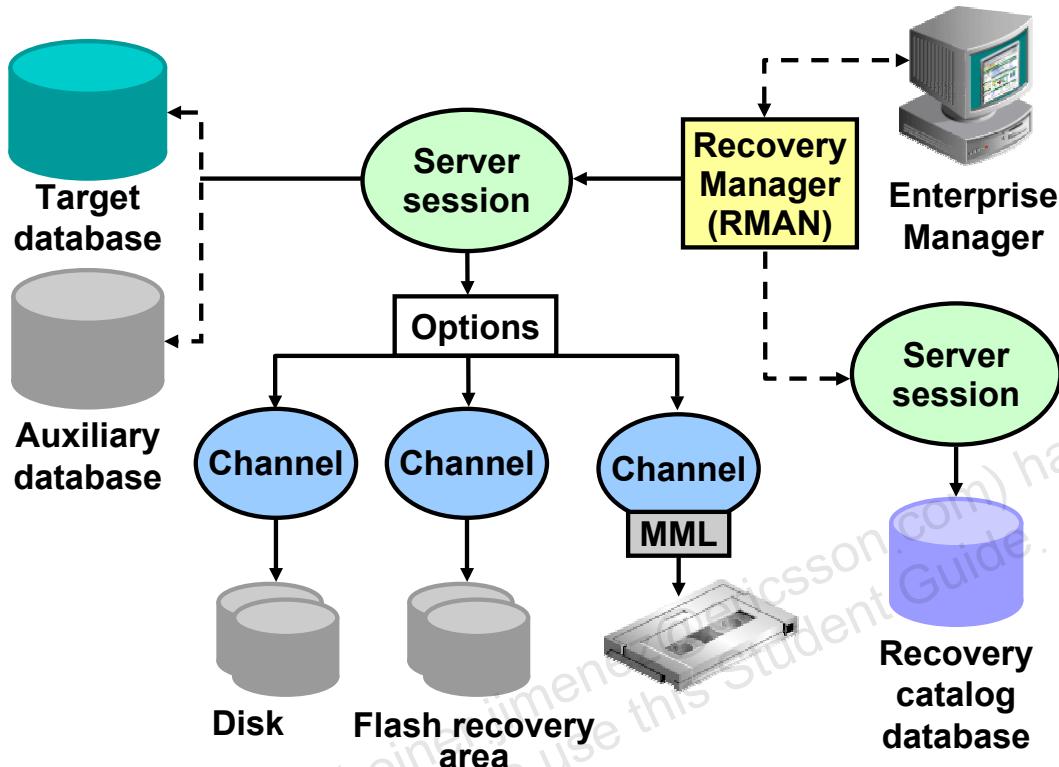
- RMAN operations can be integrated with the Scheduler to automate backup operations. The Scheduler is covered in the lesson titled “Automating Tasks with the Scheduler.”
- You can use RMAN to detect block corruption. The information relating to the block corruption that is detected during backup can be obtained by using the V\$BACKUP_CORRUPTION and V\$COPY_CORRUPTION dynamic views.
- RMAN provides performance enhancements such as:
 - Automatic parallelization of backup, restore, and recovery operations
 - No generation of extra redo during online database backups
 - Backups that are restricted to limit the number of reads per file per second to avoid interfering with OLTP work
 - The use of multiplexing, which can prevent flooding of any one file with reads and writes while still keeping a tape drive streaming
- RMAN has a media management API that works seamlessly with third-party media management tools interfacing with storage devices, which provides increased speed and reliability.
- Oracle Secure Backup allows RMAN to back up to tape without third-party media management tools.
- In the user-managed method, you need to keep track of all database files and backups. In a recovery situation, you must locate backups for each data file, copy them to the correct place using operating system—commands, and choose which redo log files to apply. RMAN manages these tasks automatically. The advantages of using RMAN are even greater if you use Oracle Managed Files or the flash recovery area.

Note

Not all of these features are covered in this course. For more information about:

- RMAN and its abilities, see the *Oracle Database Backup and Recovery Basics* and *Oracle Database Backup and Recovery Advanced User’s Guide* documentation
- The syntax of RMAN commands, refer to the *Oracle Database Recovery Manager Reference* manual

Recovery Manager Components



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Recovery Manager Components

Recovery Manager executable: The Recovery Manager command-line interface is invoked through the RMAN client application. RMAN interprets user commands and appropriately invokes server sessions to perform the desired tasks.

Enterprise Manager: The Enterprise Manager Database Control Console supplies a graphical interface to the most commonly used RMAN functionality.

Server sessions: The server processes (UNIX) or threads (Windows 2000) invoked by RMAN connect to the target database to perform the backup, restore, and recovery functions through a PL/SQL interface. These sessions read or write files from or to disk, tape, or the flash recovery area, which is a storage location specified as the default storage area for files related to database recovery.

Target database: The database for which backup and recovery operations are being performed using RMAN is called the target database. The control file of the target database contains information about its physical structure, such as the size and location of data files, online and archived redo log files, and control files. This information is used by the server sessions invoked by RMAN in backup and recovery operations.

Recovery Manager Components (continued)

Auxiliary database: An auxiliary database is used when creating a duplicate database or performing tablespace point-in-time recovery (TSPITR). For the tasks, the auxiliary database serves as the destination of the new copy of the database or the recovered tablespaces. An auxiliary database can reside on the same host as its parent or on a different host. The auxiliary database is introduced in this lesson only for completeness. For more information, refer to the *Oracle Database Backup and Recovery Advanced User's Guide*.

Channel: A channel represents one stream of data to a device type. To perform and record backup and recovery operations, RMAN requires a link to the target database. A channel establishes this link by creating a session in the target database that is able to interface with the host file system (to interface with disks) and the Media Management Library (to interface with tapes). You can allocate channels manually or preconfigure channels by using automatic channel allocation.

RMAN repository: RMAN maintains metadata about the target database and its backup and recovery operations in the RMAN repository. Among other things, RMAN stores information about its own configuration settings, the target database schema, archived redo logs, and all backup files that are on disk or tape. RMAN repository data is always stored in the control file of the target database.

Recovery catalog: The RMAN repository data can optionally be kept in a recovery catalog, which is a separate Oracle database.

Media Management Library: The Media Management Library (MML) is used by RMAN when writing to or reading from tapes. The additional media management software required for using the tape medium is provided by media and storage system vendors. Also, the Oracle Secure Backup product allows RMAN to back up to tape without requiring third-party tools.

Steps for Configuring RMAN

- 1. Determine the repository location: control file or recovery catalog.**
- 2. Define database and environment variables.**
- 3. Start RMAN and connect to the target and, optionally, the recovery catalog databases.**
- 4. Configure persistent settings.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

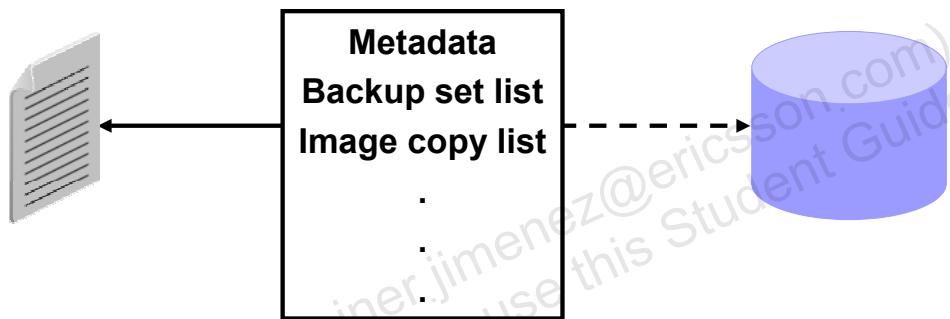
RMAN Repository Data Storage: Comparison of Options

Control file:

- Simpler administration
- Default

Recovery catalog:

- Replicates control file data
- Has room for more data
- Can service many targets
- Can store RMAN scripts



ORACLE

Copyright © 2008, Oracle. All rights reserved.

RMAN Repository Data Storage: Comparison of Options

RMAN repository data is always stored in the control file of the target database. But it can also be stored in a separate database, called a recovery catalog.

A recovery catalog preserves backup information in a separate database, which is useful in the event of a lost control file. This allows you to store a longer history of backups than what is possible with a control file-based repository. A single recovery catalog is able to store information for multiple target databases. The recovery catalog can also hold RMAN stored scripts, which are sequences of RMAN commands for common backup tasks. Centralized storage of scripts in the recovery catalog can be more convenient than working with command files.

Usage of a separate recovery catalog database is not recommended for small installations where administration of a separate recovery catalog database would be burdensome.

RMAN Repository Data Storage: Comparison of Options (continued)

How to Set Up a Recovery Catalog

1. Create the database to be used as the recovery catalog. Alternatively, identify an already existing database and use that. Because a single recovery catalog can serve more than one database, you may choose to create a single recovery catalog that services all your databases that are being backed up.
2. In the recovery catalog database, create the user that will own the recovery catalog data. For example, assume that catdb is the name of the catalog database, and rcat_ts is the name of a tablespace you have created where the catalog data is to be stored.

```
$ sqlplus sys/password@catdb as sysdba
```

```
SQL> CREATE USER rman IDENTIFIED BY cat
  2  TEMPORARY TABLESPACE temp
  3  DEFAULT TABLESPACE rcat_ts
  4  QUOTA UNLIMITED ON rcat_ts;
```

3. Grant the RECOVERY_CATALOG_OWNER role to the catalog owner. This role provides the user with all privileges required to maintain and query the recovery catalog.

```
GRANT RECOVERY_CATALOG_OWNER TO rman;
```
4. Start RMAN and log in as the catalog owner.

```
$ rman catalog rman/cat@catdb
RMAN> CREATE CATALOG TABLESPACE rcat_ts;
```
5. Connect to the target database by using RMAN, and register it in the newly created recovery catalog.

```
$ rman target sys/oracle@orcl catalog rman/cat@catdb
RMAN> register database;
```

Note: It is important to back up your recovery catalog database.

Backup Destinations

Backups can be written to:

- **Disk directory**
- **Media Management Library (tape device)**
 - Typically used for disaster recovery, when disk backups are lost
- **Flash recovery area**
 - This is a disk area set aside for backup and recovery and flashback database purposes.
 - You define the location and the size.
 - Files are automatically named by using Oracle Managed Files.
 - Files are automatically retained and deleted as necessary.

ORACLE

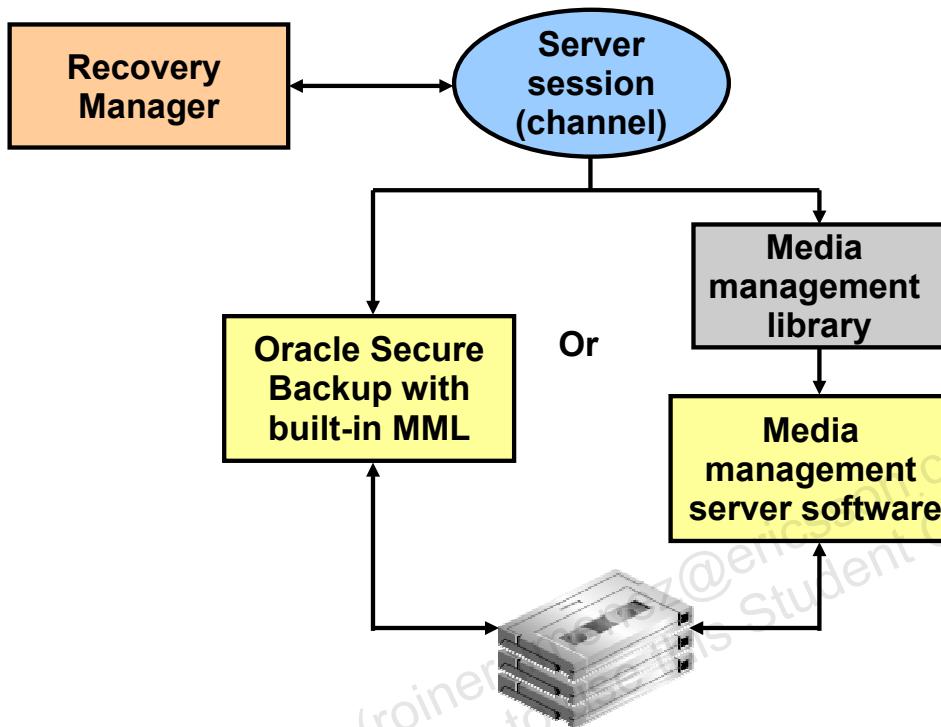
Copyright © 2008, Oracle. All rights reserved.

Backup Destinations

Backups can be written to a designated disk directory, a Media Management Library, or the flash recovery area. Specifying a disk directory or the flash recovery area means that backups go to hard-disk media. Typically, they are regularly moved offline to tape via the media management interface in order to maintain disk space availability. Any disk directory can be specified as the destination of a backup provided that it already exists.

If you set up a flash recovery area, many backup and recovery tasks are simplified for you. The Oracle database automatically names files for you, and deletes obsolete files when there is space pressure.

Media Management



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Media Management

To use tape storage for your database backups, RMAN requires Oracle Secure Backup or a media manager. Oracle Secure Backup is covered in detail in Appendix F.

A media manager is a utility that loads, labels, and unloads sequential media, such as tape drives for the purpose of backing up, restoring, and recovering data. The Oracle database calls MML software routines to back up and restore data files to and from media that is controlled by the media manager.

Some media management products can completely manage all data movement between Oracle data files and the backup devices. Some products that use high-speed connections between storage and media subsystems can reduce much of the backup load from the primary database server.

Note that the Oracle database does not need to connect to the Media Management Library (MML) software when it backs up to disk.

Media Management (continued)

The Oracle Backup Solutions Program (BSP) provides a range of media management products that are compliant with Oracle's MML specification. Software that is compliant with the MML interface enables an Oracle database session to back up data to a media manager and request the media manager to restore backups. Check with your media vendor to determine whether it is a member of the Oracle BSP.

Before you can begin using RMAN with a media manager, you must install the media manager software and make sure that RMAN can communicate with it. Instructions for this procedure should be available in the media manager vendor's software documentation.

Depending on the product that you are installing, perform the following basic steps:

1. Install and configure the media management software on the target host or production network. No RMAN integration is required at this stage.
2. Ensure that you can make non-RMAN backups of operating system files on the target database host. This step makes it easier to troubleshoot problems at a later time. Refer to your media management documentation to learn how to back up files to the media manager.
3. Obtain and install the third-party media management module for integration with the Oracle database. This module must contain the library loaded by the Oracle database when accessing the media manager.

Backup and Restore Operations Using a Media Manager

The following Recovery Manager script performs a data file backup to a tape drive controlled by a media manager:

```
run {
  # Allocating a channel of type 'sbt' for serial device
  ALLOCATE CHANNEL ch1 DEVICE TYPE sbt;
  BACKUP DATAFILE 3;
}
```

When Recovery Manager executes this command, it sends the backup request to the Oracle database session performing the backup. The Oracle database session identifies the output channel as a media management device and requests the media manager to load a tape and write the output.

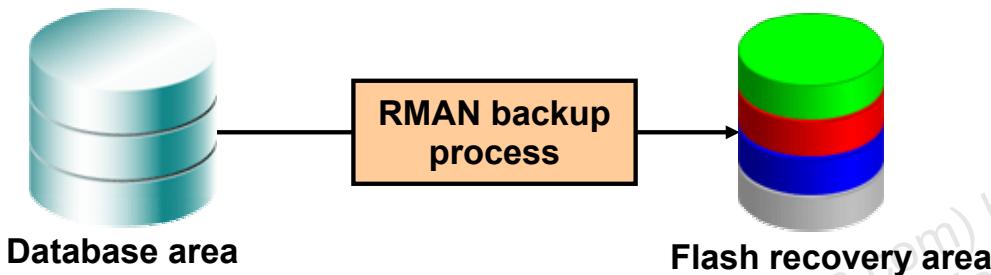
The media manager labels and keeps track of the tape and the names of the files on each tape.

The media manager also handles restore operations. When you restore a file, the following steps occur:

1. The Oracle database requests the restoration of a particular file.
2. The media manager identifies the tape containing the file and reads the tape.
3. The media manager passes the information back to the Oracle database session.
4. The Oracle database writes the file to disk.

Using a Flash Recovery Area with RMAN

You can configure RMAN to use the flash recovery area.



```
DB_RECOVERY_FILE_DEST = '/u01/oracle/fra'  
DB_RECOVERY_FILE_DEST_SIZE = 2G
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Using a Flash Recovery Area with RMAN

Although not required for RMAN operations, the flash recovery area simplifies managing disk space and files related to backup and recovery. You do not need to specify the file names for the backup files because RMAN generates the file names automatically. When the flash recovery area is used, RMAN automatically uses Oracle Managed Files (OMF) for its backup files.

Each time RMAN creates a file in the flash recovery area, the Oracle database updates the list of files that are no longer required on disk. When a file needs to be written into the flash recovery area and space is not available for that file, the Oracle database deletes a file that is on the obsolete files list and writes a notification to the alert log.

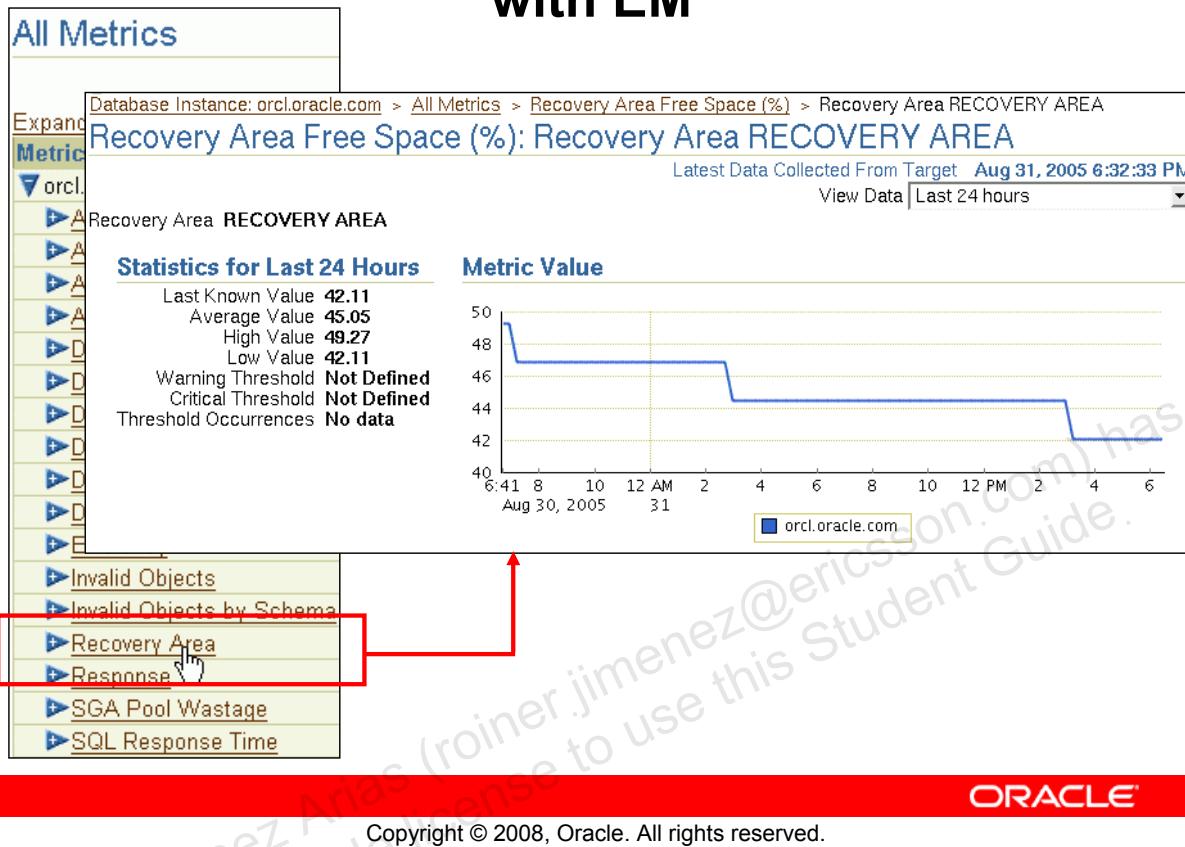
A warning is issued when the flash recovery area experiences space pressure or is low on free space because there are no files that can be deleted from the flash recovery area. To resolve the problem, you can add disk space, back up your files to a tertiary device, delete the files from the flash recovery area using RMAN, or change the RMAN retention policy. The flash recovery area is configured by setting the DB_RECOVERY_FILE_DEST initialization parameter. The DB_RECOVERY_FILE_DEST_SIZE parameter specifies its size.

Using a Flash Recovery Area with RMAN (continued)

By default, RMAN automatically places backup files into the flash recovery area, if there is one configured.

Note: A flash recovery area can be located in an Automated Storage Management (ASM) instance. A flash recovery area can also be used with Oracle Cluster File Storage (OCFS) or any local storage.

Monitoring the Flash Recovery Area with EM



Monitoring the Flash Recovery Area with EM

Real-time flash recovery area-related metrics can also be viewed through the EM Database Control Console. On the Maintenance page, scroll down to the Related Links section and select All Metrics. Scan the list and click Recovery Area.

The displayed page shows the Recovery Area Free Space (%) metric, which represents the recovery area free space as a percentage. Click the percentage number to see the graph of recovery area usage.

Flash Recovery Area Space Usage

- **Configure the retention policy to the minimum value appropriate for your database.**
- **Back up the archive log files regularly and delete the files upon completion of the backup.**
- **Use the RMAN REPORT OBSOLETE and DELETE OBSOLETE commands to remove backups and file copies that are not required.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Flash Recovery Area Space Usage

To avoid running out of space in the flash recovery area, you should never store user-managed files in this area. You should also perform the following steps as needed or appropriate:

- Delete unnecessary files from the recovery area by using RMAN.
- Take frequent backups of the recovery area by using RMAN.
- Change the RMAN retention policy to retain backups for a smaller period of time.
- Change the RMAN archived log deletion policy.
- Add disk space and increase the value of the DB_RECOVERY_FILE_DEST_SIZE database initialization parameter if you are frequently running out of space.

For example, to back up the archived log files in the recovery area and then delete the files after they have been successfully backed up, you would use the RMAN command:

```
BACKUP ARCHIVELOG ALL DELETE ALL INPUT;
```

If you use a backup solution other than RMAN, you still have to use RMAN to remove the files from the flash recovery area. After the archived log files have been backed up and removed from disk, use the RMAN CROSSCHECK and DELETE commands to reclaim the archived log space from the flash recovery area. You should do this on a regular basis, or after every backup.

V\$FLASH_RECOVERY_AREA_USAGE

You can query V\$FLASH_RECOVERY_AREA_USAGE to view the flash recovery area disk space usage.

```
SQL> SELECT file_type,
  2  percent_space_used AS used,
  3  percent_space_reclaimable AS reclaimable,
  4  number_of_files AS number
  5  FROM  v$flash_recovery_area_usage ;

FILE_TYPE          USED RECLAMABLE FILES
-----
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

V\$FLASH_RECOVERY_AREA_USAGE

Output from the V\$FLASH_RECOVERY_AREA_USAGE query

FILE_TYPE	USED	RECLAMABLE	FILES
<hr/>			
CONTROLFILE	0	0	0
ONLINELOG	0	0	0
ARCHIVELOG	69.13	0	593
BACKUPPIECE	0	0	0
IMAGECOPY	0	0	0
FLASHBACKLOG	1.77	0	37

Copyright © 2008, Oracle. All rights reserved.

V\$FLASH_RECOVERY_AREA_USAGE

The columns have the following meanings:

- **FILE_TYPE:** This is the type of the file and can have any of the following values: CONTROLFILE, ONLINELOG, ARCHIVELOG, BACKUPPIECE, IMAGECOPY, FLASHBACKLOG.
- **PERCENT_SPACE_USED:** This is the percentage of the flash recovery area that is currently being used to store files of the given type.
- **PERCENT_SPACE_RECLAMABLE:** This is the percentage of the flash recovery area that is currently being used to store files of the given type that could be deleted because they are obsolete or redundant or have been backed up to a tertiary device.
- **NUMBER_OF_FILES:** This is the number of files of the specified file type.

Backing Up the Flash Recovery Area

The screenshot shows the Oracle Database Control interface. At the top, it says "Database Instance: orcl.oracle.com > Schedule Backup". Below that is a section titled "Schedule Backup". Under "Customized Backup", there's a note: "Select the object(s) you want to back up." To the right is a button labeled "Schedule Customized Backup". There are several backup strategy options: "Whole Database" (radio button), "Tablespaces" (radio button), "Datafiles" (radio button), "Archivelogs" (radio button), and "All Recovery Files on Disk" (radio button, which is highlighted with a red box). A note below says: "These files include all archivelogs and disk backups that are not already backed up to tape". At the bottom of the interface, it says "RMAN> BACKUP RECOVERY FILES;" followed by the Oracle logo and copyright information: "Copyright © 2008, Oracle. All rights reserved."

Backing Up the Flash Recovery Area

Because of the importance of the data it contains, you should back up the files in the flash recovery area on a regular basis. To do this, navigate to the **Maintenance** tabbed page. On this page, click the **Schedule Backup** link in the **Backup/Recovery** region. Select **Customized** from the **Backup Strategy** drop-down list, and then select the **All Recovery Files on Disk** option. You can also use RMAN BACKUP commands to back up the flash recovery area.

- RMAN> BACKUP RECOVERY AREA;

This command backs up all flash recovery files created in the current flash recovery area destinations that have not previously been backed up. Files that fall into this category are full and incremental backup sets, control file autobackups, archive logs, and data file copies. Other files such as flashback logs, incremental bitmaps, current control file, and online redo log files are not backed up.

- RMAN> BACKUP RECOVERY FILES;

This command backs up all recovery files on disk that have not previously been backed up. The files that fall into this category are full and incremental backup sets, control file autobackups, archive logs, and data file copies.

When using the BACKUP RECOVERY AREA or BACKUP RECOVERY FILES commands, the destination must be to tape.

Benefits of Using a Flash Recovery Area

Using the flash recovery area for recovery-related files:

- Simplifies the location of database backups
- Automatically manages the disk space allocated for recovery files
- Does not require changes to existing scripts
- Puts database backups, archive logs, and control file backups in the flash recovery area

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Benefits of Using a Flash Recovery Area

Using a flash recovery area for all recovery-related files simplifies the ongoing administration of your database, depending on the setting of the initialization parameters DB_RECOVERY_FILE_DEST_SIZE and DB_RECOVERY_FILE_DEST. Oracle Corporation recommends the use of the flash recovery area for all recovery-related files.

Setting Parameters That Affect RMAN

- **Database initialization parameters**
 - **CONTROL_FILE_RECORD_KEEP_TIME**
 - **DB_RECOVERY_FILE_DEST** and
DB_RECOVERY_FILE_DEST_SIZE, if using the flash recovery area
- **Environment variables**
 - **NLS_DATE_FORMAT**
 - **NLS_LANG**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Setting Parameters That Affect RMAN

RMAN stores information about the target database and its backup and recovery operations in the RMAN repository. The amount of information stored can increase depending on the frequency of backups, the number of archived redo log files that are generated, and the retention period for RMAN records.

The **CONTROL_FILE_RECORD_KEEP_TIME** parameter specifies the minimum number of days RMAN information is stored in the control file before being overwritten. A low value results in information being overwritten sooner, thus minimizing control file growth. If a recovery catalog is used, a lower value should be chosen. This parameter applies only to records in the control file that are circularly reusable (such as archive log records and various backup records). It does not apply to records such as data file, tablespace, and redo thread records, which are never reused unless the corresponding object is dropped from the tablespace. The default is seven days.

Setting Parameters That Affect RMAN (continued)

If you define a flash recovery area, then, by default, that is where backup files are written. You set the flash recovery area size and location by using the DB_RECOVERY_FILE_DEST and DB_RECOVERY_FILE_DEST_SIZE initialization parameters. You also specify a retention policy that dictates when backups may be discarded. RMAN then manages your backup storage, deleting obsolete backups and backups already copied to tape when space is needed, but keeping as many backups on disk as space permits. This minimizes restores from tape during data recovery operations to shorten restore and recovery times. You can resize the flash recovery area at any time by setting the DB_RECOVERY_FILE_DEST_SIZE initialization parameters, which is dynamic.

The NLS_DATE_FORMAT and NLS_LANG environment variables determine the format used for the time parameters in RMAN commands such as RESTORE, RECOVER, and REPORT.

A database that is not mounted assumes the default character set, which is US7ASCII. If your character set is different from the default, then set NLS_LANG appropriately. For example, if the character set is WE8DEC, you can set the NLS_LANG parameter as follows:

```
NLS_LANG=american_america.we8dec
```

RMAN Usage Considerations

- **Resources: Shared memory, more processes**
- **Privileges given to users:**
 - **Database: SYSDBA**
 - **Operating system: Access to devices**
- **Remote operations:**
 - **Set up the password file**
 - **Ensure that the password file is backed up**

ORACLE

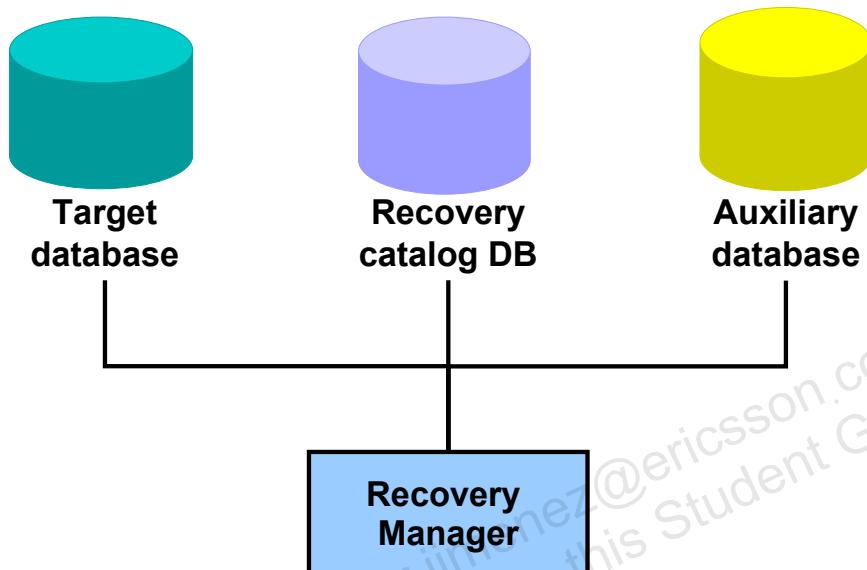
Copyright © 2008, Oracle. All rights reserved.

RMAN Usage Considerations

Before Recovery Manager is used, consider the following points:

- **Shared resources on the system:** Most of the work of RMAN is performed through Oracle database processes. The operations can also be performed in parallel to increase throughput. This implies that the PROCESSES database parameter must be sufficiently high. For the operating system–resources, this means that shared memory and semaphore settings may need to be increased.
- **Users performing privileged operations:** You must decide which users will be able to perform privileged operations. Then, grant the necessary privileges to the users' accounts at the operating system level and within the Oracle database. For example, to start up and shut down a database, a user should have the SYSDBA privilege.
- **Remote operations:** You need to use a password file to connect to the target database over Oracle Net to perform privileged operations, such as startup and shutdown, from a remote machine.
- **Use of the recovery catalog:** When you use a recovery catalog, RMAN can perform a wider variety of automated backup and recovery functions. Use of the recovery catalog involves storage space and maintenance efforts. You should decide whether to have a database dedicated to maintaining the recovery catalogs of several target databases.

Connection Types with RMAN



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Connection Types with RMAN

You can use Recovery Manager to connect to the following types of databases:

- **Target database:** RMAN connects you to the target database with the SYSDBA privilege. You must have this privilege for the connection to succeed. The target database is the instance where you want to perform typical RMAN operations.
- **Recovery catalog database:** This is an optional database that is configured for the RMAN repository. You connect to the recovery catalog database when you want to retrieve information stored within it, such as backup information or stored scripts.
- **Auxiliary database:** An auxiliary database can be a database that is:
 - Created using the RMAN DUPLICATE command
 - A temporary database that is used during tablespace point-in-time recovery (TSPITR)
 - A standby database, or a copy of your production database that can be used for disaster recovery

Starting RMAN

- **Starting RMAN locally**

```
UNIX: $ ORACLE_SID=DB01; export ORACLE_SID  
      $ rman target /
```

```
Windows NT: C:\> set ORACLE_SID=DB01  
             C:\> rman target /
```

- **Starting RMAN remotely**

```
rman target sys/password@DB01
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Starting RMAN

Local Connection

For a local RMAN connection, at an operating system prompt, enter the following:

```
UNIX:   $ ORACLE_SID=DB01; export ORACLE_SID  
         $ rman target /
```

```
Windows: C:\> SET ORACLE_SID=DB01  
          C:\> rman target /
```

If you do not specify a user ID or password when connecting to the target database, then a slash establishes a connection as user SYS by using operating system authentication.

Optionally, you can specify the NOCATALOG keyword as follows:

```
$ rman target / nocatalog
```

NOCATALOG is the default mode and is used to indicate that you are using RMAN without a recovery catalog.

Remote Connection

To connect from another server, use the Oracle Net service name for the target database:

```
$ rman target sys/target_pwd@DB01
```

Additional RMAN Command-Line Arguments

- Writing RMAN output to a log file

```
$ rman TARGET sys/oracle  
LOG=$HOME/oradata/u03/rman.log APPEND
```

- Executing a command file when RMAN is invoked

```
$ rman TARGET sys/oracle  
CMDFILE=$HOME/scripts/my_rman_script.rcv
```

- Establishing database connections on RMAN startup

```
$ rman TARGET SYS/sys_pwd@orcl CATALOG  
rman/rman@rcat
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Additional RMAN Command-Line Arguments

The LOG=*filename* argument specifies the file where RMAN output will be recorded. If not specified, then RMAN writes its message log file to standard output.

The APPEND keyword specifies that new output should be appended to the end of the message log file.

You can use CMDFILE=*filename* to run a file containing RMAN commands. RMAN terminates after running the command file.

Any file names specified on the command line that do not begin with an alphabetic character must be enclosed in single quotation marks.

You can specify connection options for RMAN, such as:

- **AUXILIARY:** Specifies a connect string to an auxiliary database
- **CATALOG:** Specifies a connect string to the database containing the recovery catalog
- **NOCATALOG:** Indicates that you are using RMAN without a recovery catalog
- **TARGET:** Specifies a connect string to the target database

When you use the SCRIPT clause, after you are connected to the target database and recovery catalog (specified using the TARGET and CATALOG clauses), RMAN runs the named stored script from the recovery catalog against the target database.

Configuring Persistent Settings for RMAN

- **RMAN is preset with default configuration settings**
- **Use the CONFIGURE command to:**
 - **Configure automatic channels**
 - **Specify the backup retention policy**
 - **Specify the number of backup copies to be created**
 - **Set the default backup type to BACKUPSET or COPY**
 - **Limit the size of backup sets**
 - **Exempt a tablespace from backup**
 - **Enable and disable backup optimization**
 - **Configure automatic backups of control files**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Configuring Persistent Settings for RMAN

To simplify ongoing use of RMAN for backup and recovery, RMAN enables you to set a number of persistent configuration settings for each target database. These settings control many aspects of RMAN's behavior when working with that database. You can save persistent configuration information such as channel parameters, parallelism, and the default device type in the RMAN repository (which is stored in either the control file or a recovery catalog database).

These settings have default values, which allow you to use RMAN immediately. However, as you develop a more advanced backup and recovery strategy, you will have to change these settings to implement that strategy. You can use the CONFIGURE command to configure persistent settings for RMAN backup, restore, duplication, and maintenance jobs. These settings are in effect for any RMAN session until the configuration is cleared or changed.

Configuring RMAN Settings by Using EM

Backup Settings

Device **Backup Set** **Policy**

Maximum Backup Piece (File) Size MB Specify a value to restrict the size of each backup piece.

Tape Settings
The following parameters require additional configuration on different media pools.

Copies of Datafile Backups Specify the number of identical copies for datafile backups.

Copies of Archivelog Backups Specify the number of identical copies for archivelog backups.

Host Credentials
To save the backup settings, supply operating system login credentials to access the target database.

* Username

* Password

Save as Preferred Credential

Backup/Recovery Settings

Backup Settings **Recovery Settings** **Recovery Catalog Settings**

ORACLE

Configuring RMAN Settings by Using EM

You can use Oracle Enterprise Manager to specify the backup settings for an instance. To specify backup settings, on the **Maintenance** page, select **Backup Settings** in the **Backup/Recovery Settings** section.

The Backup Settings property page consists of three tabs that cover Device, Backup Set, and Policy parameters. You can access the:

- **Device** page to set the disk and tape configuration settings, including the Media Management Library (MML) settings
- **Backup Set** page (shown in the slide) to specify parameters for backup sets and to enter host credentials
- **Policy** page to set various backup and retention policies before you initiate a backup, such as automatically backing up the control file and SPFILE. The Policy page also allows you to configure block change tracking support, a feature that provides faster incremental backups.

Note: Backup settings change the databasewide settings and apply to any backups that do not override settings at the backup level.

Control File Autobackups

```
RMAN> CONFIGURE CONTROLFILE AUTOBACKUP ON;
```

Backup Settings

Device Backup Set Policy

Backup Policy

Automatically backup the control file and server parameter file (SPFILE) with every backup and database structural change

Autobackup Disk Location

An existing directory or diskgroup name where the control file and server parameter file will be backed up. If you do not specify a location, the files will be backed up to the flash recovery area location.

Best practice: Oracle recommends that you enable control file autobackup.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Control File Autobackups

To avoid losing a copy of the current control file, you should configure RMAN to take automatic backups of the control file. The automatic backup of the control file occurs independently of any backup of the current control file explicitly requested as part of your backup command. If you are running RMAN in NOCATALOG mode, it is highly recommended that you activate control file autobackup. Otherwise, if you lose your control file, your database may be unrecoverable.

To configure control file autobackup, modify the backup policy for your database by using Enterprise Manager or use the following RMAN command:

```
CONFIGURE CONTROLFILE AUTOBACKUP ON;
```

By default, control file backups are turned off. If you enable control file backups, then RMAN automatically backs up the control file and the current server parameter file (if used to start up the database) in one of two circumstances:

- A successful backup is recorded in the RMAN repository.
- A structural change to the database affects the contents of the control file, which, therefore, must be backed up.

Control File Autobackups (continued)

The control file autobackup file name has a default format of %F for all device types, so that RMAN can guess the file location and restore it without a repository. This variable format translates into c-**I**IIIIIIII-YYYYMMDD-**Q**Q, where:

- **I**IIIIIIII stands for the DBID
- YYYYMMDD is a time stamp of the day the backup is generated
- **Q**Q is the hex sequence that starts with 00 and has a maximum of FF

You can change the default format by using the CONFIGURE CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE *type* TO '*string*' command. The value of string must contain the substitution variable %F and cannot contain other substitution variables. For example:

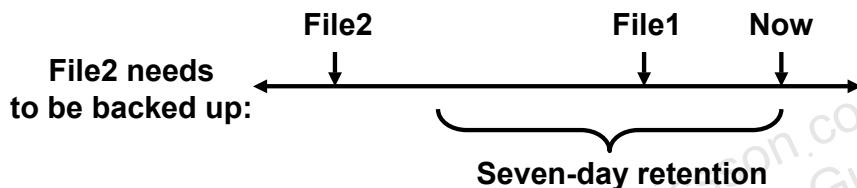
```
CONFIGURE CONTROLFILE AUTOBACKUP FORMAT  
FOR DEVICE TYPE DISK TO '/u01/oradata/cf_ORCL_auto_%F';
```

Control file backups are stored in the flash recovery area, unless otherwise specified.

With a control file backup, RMAN can recover the database even if the current control file, recovery catalog, and server parameter file are inaccessible. Because the path used to store the backup follows a well-known format, RMAN can search for and restore the server parameter file or control file from that backup.

Retention Policies

- A retention policy describes which backups will be kept and for how long.
- There are two types of retention policies:
 - **Recovery window:** Establishes a period of time within which point-in-time recovery must be possible
 - **Redundancy:** Establishes a fixed number of backups that must be kept
- These policies are mutually exclusive and can be set with the CONFIGURE command.



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Retention Policies

A retention policy describes which backups will be kept and for how long. The value of the retention policy is set by the CONFIGURE command. The best practice is to establish a period of time during which it will be possible to discover logical errors and fix the affected objects by doing a point-in-time recovery to just before the error occurred. This period of time is called the recovery window. This policy is specified in number of days. For each data file, there must always exist one backup which satisfies the following condition:

```
SYSDATE - checkpoint_time >= recovery_window
```

For example, if the policy were to be set as follows:

```
RMAN> CONFIGURE RETENTION POLICY
      2 TO RECOVERY WINDOW OF 7 DAYS;
```

Then, for each file there must be a backup that satisfies:

```
SYSDATE - (SELECT checkpoint_time
            FROM V$DATAFILE WHERE file#= ...) >= 7
```

In the example in the slide, File1 has a single backup that was made just two days ago, so it does not satisfy the retention policy. File2 has a nine-day old backup, which does satisfy the policy.

Retention Policies (continued)

You should keep the recovery window time period less than or equal to the value of the control file parameter CONTROL_FILE_RECORD_KEEP_TIME to prevent the record of older backups from being overwritten in the control file.

If you require a certain number of backups to be retained, then you can set the retention policy on the basis of the redundancy option. This option requires that a specified number of backups be cataloged before any backup is identified as obsolete. The default retention policy has a redundancy of 1, which means that only one backup of a file must exist at any given time. A backup is deemed obsolete when a more recent version of the same files has been backed up.

Managing Persistent Settings

- Use the SHOW command to list current settings:

```
RMAN> SHOW CONTROLFILE AUTOBACKUP FORMAT;  
RMAN> SHOW EXCLUDE;  
RMAN> SHOW ALL;
```

- Use the CLEAR command to reset any persistent setting to its default value:

```
RMAN> CONFIGURE BACKUP OPTIMIZATION CLEAR;  
RMAN> CONFIGURE MAXSETSIZE CLEAR;  
RMAN> CONFIGURE DEFAULT DEVICE TYPE CLEAR;
```

ORACLE

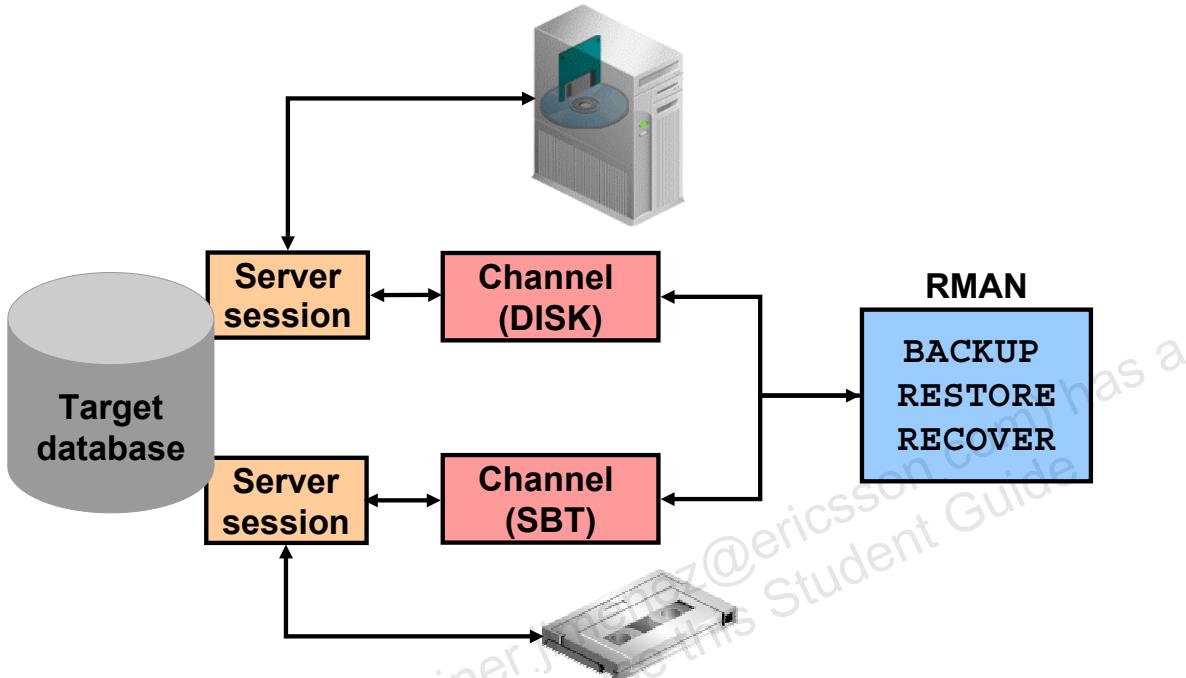
Copyright © 2008, Oracle. All rights reserved.

Managing Persistent Settings

Using the RMAN SHOW command, you can view the RMAN configuration settings. If SHOW ALL is executed when connected to a target database, only node-specific configurations and database configurations are displayed.

You can return to the default value for any CONFIGURE command by running the same command with the CLEAR option.

Channel Allocation



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Channel Allocation

A channel represents one stream of data to a device type. A channel must be allocated before you execute backup and recovery commands. Each allocated channel establishes a connection from the RMAN executable to a target database instance. An Oracle database process for the target database is created for every channel allocated.

Every BACKUP, COPY, RESTORE, or RECOVER command issued in Recovery Manager requires at least one channel. The type of media desired determines the type of channel allocated. The number of channels allocated is the maximum degree of parallelization that is used during backup, restore, or recovery.

You can query the V\$BACKUP_DEVICE view to determine supported device types.

Automatic and Manual Channel Allocation

- An automatic channel is one that is preconfigured and used for subsequent commands. To change the default device type for automatic channel allocation, use:

```
RMAN> CONFIGURE DEFAULT DEVICE TYPE TO sbt;
```

- A manually allocated channel is one that overrides the automatic channel setting. This channel overrides the automatic channel for this run block:

```
RMAN> RUN {  
2> ALLOCATE CHANNEL c1 DEVICE TYPE disk;  
3> BACKUP DATAFILE '/u01/oradata/user01.dbf';  
4> }
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Automatic and Manual Channel Allocation

With the CONFIGURE command, you can preconfigure channels for use in all RMAN sessions using automatic channel allocation. Automatic channels apply to any RMAN job in which you do *not* manually allocate channels.

By default, RMAN has preconfigured a disk channel so that you can back up to disk without doing any manual configuration. Therefore, if you are backing up to disk rather than to a media manager, you can immediately begin backing up to disk.

The ALLOCATE CHANNEL command with a RUN command and the ALLOCATE CHANNEL FOR MAINTENANCE command issued at the RMAN prompt are used to allocate a channel manually. Manual channel allocation overrides automatic allocation.

You can use manually allocated channels to issue commands (such as the CHANGE, DELETE, and CROSSCHECK commands) that delete or change the status of existing backups. Note that if you use CONFIGURE to set up automatic channels, then RMAN can use these automatic channels for maintenance operations; you do not have to manually allocate them. Manually allocated maintenance channels cannot be used for any other I/O operation, such as backup or copy.

The automatic channel feature is mutually exclusive with the manual channel feature: RMAN uses one or the other for a given job.

Channel Control Options

- **Configure parallelism:**

```
RMAN> CONFIGURE DEVICE TYPE DISK PARALLELISM 3;
```

- **Specify the maximum backup piece size:**

```
RMAN> CONFIGURE CHANNEL DEVICE TYPE DISK  
2> MAXPIECESIZE 2G;
```

- **Format the name of generated backup files:**

```
RMAN> RUN {  
2> ALLOCATE CHANNEL d1 DEVICE TYPE DISK  
3> FORMAT '/disk1/backups/%U';  
4> BACKUP DATABASE PLUS ARCHIVELOG; }
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Channel Control Options

You can specify control options for the allocated channel to change its default behavior. The configurable control options for manually and automatically allocated channels are:

- **CONNECT:** The connect string for the target instance
- **FORMAT:** The format to use for backup piece names created on this channel
- **MAXOPENFILES:** The maximum number of input files that a BACKUP command can have open at any given time (the default is 8)
- **MAXPIECESIZE:** The maximum size of each backup piece created on this channel, specified in bytes (default), kilobytes (K), megabytes (M), or gigabytes (G)

Channel Control Options (continued)

If the channel device type is SBT or SBT_TAPE, you can also specify:

- **PARMS= "ENV (...)" :** Set environment variables for the server session corresponding to this RMAN client.
- **PARMS= "SBT_LIBRARY=..." :** Specify the location of the MML to be used by the channel.

For automatic channels, you can also configure the default parallelism level and the default backup type for disk or tape backups to BACKUPSET, COMPRESSED BACKUPSET, or COPY.

Summary

In this lesson, you should have learned how to:

- **Use either the control file or a recovery catalog for the RMAN repository**
- **Change RMAN default settings with CONFIGURE**
- **Use the flash recovery area for RMAN operations**
- **Monitor the flash recovery area by using v\$ views and Enterprise Manager**
- **Implement recovery window and redundancy retention policies**
- **Implement manual and automatic channel allocation**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Practice Overview: Configuring RMAN

This practice covers the following topics:

- Using Recovery Manager to connect to a target database in default NOCATALOG mode
- Displaying the default RMAN configuration settings
- Configuring control file autobackups
- Altering the backup retention policy for a database



Copyright © 2008, Oracle. All rights reserved.

Using Recovery Manager

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Objectives

Commands
BACKUP
Block C.T.
Inc. Update
LIST
REPORT
DELETE

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

- **Use the RMAN BACKUP command to create backup sets and image copies**
- **Manage the backups and image copies taken with RMAN**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Objectives

Recovery Manager (RMAN) is the component of the Oracle database used to perform backup and recovery operations. Enterprise Manager (EM) supplies a graphical interface to the most commonly used RMAN functionality.

The *Oracle Database 10g: Administration Workshop I* class demonstrates basic RMAN operations using the EM Database Control Console; therefore, the graphical interface to RMAN is not covered in detail in this lesson.

Issuing Recovery Manager Commands

- **Interactive client**
 - Enter commands at **RMAN prompt**.
 - Use when performing analysis, or when running reports or stored scripts.
- **Batch mode**
 - Use with automated jobs.
 - Specify a command file when starting RMAN.
 - Specify the log file name to capture session log.
- **Pipe interface**
 - Specify the **PIPE** command-line argument.
 - Use to communicate data between sessions or between RMAN and an external application.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Issuing Recovery Manager Commands

Recovery Manager has its own command language. There are multiple ways of entering commands for RMAN by using the command-line interface (CLI).

To run RMAN commands interactively, start RMAN and then enter commands into the command-line interface. For example:

```
$ rman TARGET sys/sys_pwd@db1  
RMAN> BACKUP DATABASE;
```

You can enter RMAN commands into a file, and then run the command file by specifying its name on the command line. This is referred to as batch mode processing. The contents of the command file should be identical to commands that would be entered at the command line. For example:

```
$ rman TARGET SYS/sys_pwd@prod1 @'/oracle/backup_all_10.rcv'
```

When running in batch mode, RMAN reads input from a command file and writes output messages to a log file (if specified). RMAN parses the command file in its entirety before compiling it or executing any commands. There is no need to place an exit command in the file because RMAN terminates when the end of the file is reached.

The RMAN pipe interface is an alternative method for issuing commands to RMAN and receiving the output from those commands. RMAN obtains commands and sends output by using the DBMS_PIPE PL/SQL package.

Issuing Recovery Manager Commands (continued)

The pipe interface is invoked by using the PIPE command-line parameter. RMAN uses two private pipes: one for receiving commands and the other for sending output. The names of the pipes are derived from the value of the PIPE parameter.

For example:

```
% rman PIPE abc TARGET SYS/pwd@trgt
```

RMAN opens the two pipes in the target database: ORA\$RMAN_ABC_IN, which RMAN uses to receive user commands, and ORA\$RMAN_ABC_OUT, which RMAN uses to send all output back to RMAN. All messages on both the input and output pipes are of the VARCHAR2 type.

When the pipe interface is used, RMAN does not read or write any data by using the operating system shell. You can use pipes to communicate with another session in the same instance or a waiting application such as a UNIX program. By using this interface, it is possible to write a portable programmatic interface to RMAN.

See the *Oracle Database Backup and Recovery Advanced User's Guide* for more information about using pipes with RMAN.

Refer to the *PL/SQL Packages and Types Reference* for more information about the DBMS_PIPE package and creating pipes within the Oracle database.

Types of RMAN Commands

RMAN commands are of the following types:

- **Stand-alone command:**
 - Is executed individually at the RMAN prompt
 - Cannot appear as subcommands within RUN
- **Job command:**
 - Must be within the braces of a RUN command
 - Is executed as a group

Some commands can be executed as either a stand-alone or a job command.



Copyright © 2008, Oracle. All rights reserved.

Types of RMAN Commands

You can issue two basic types of RMAN commands: stand-alone and job commands.

Stand-alone commands are executed at the RMAN prompt and are generally self-contained.

Some of the stand-alone commands are:

- CHANGE
- CONNECT
- CREATE CATALOG, RESYNC CATALOG
- CREATE SCRIPT, DELETE SCRIPT, REPLACE SCRIPT

Job commands are usually grouped and executed sequentially inside a command block. If any command within the block fails, RMAN ceases processing; no further commands within the block are executed. The effects of any already executed commands still remain, though; they are not undone in any way.

An example of a command that can be run only as a job command is ALLOCATE CHANNEL.

The channel is allocated only for the execution of the job, so it cannot be issued as a stand-alone command. There are some commands that can be issued either at the prompt or within a RUN command block, such as BACKUP DATABASE. If you issue stand-alone commands, RMAN allocates any needed channels by using the automatic channel allocation feature.

You can execute stand-alone and job commands in interactive mode or batch mode.

Job Commands: Example

Job commands appear inside a RUN command block:

```
RMAN> RUN {  
 2> BACKUP AS BACKUPSET  
 3> FORMAT '/u01/db01/backup/%d_%s_%p'  
 4> DURATION 10:00 MINIMIZE LOAD  
 5> (DATABASE);  
 6> SQL 'alter system archive log current';  
 7> }
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

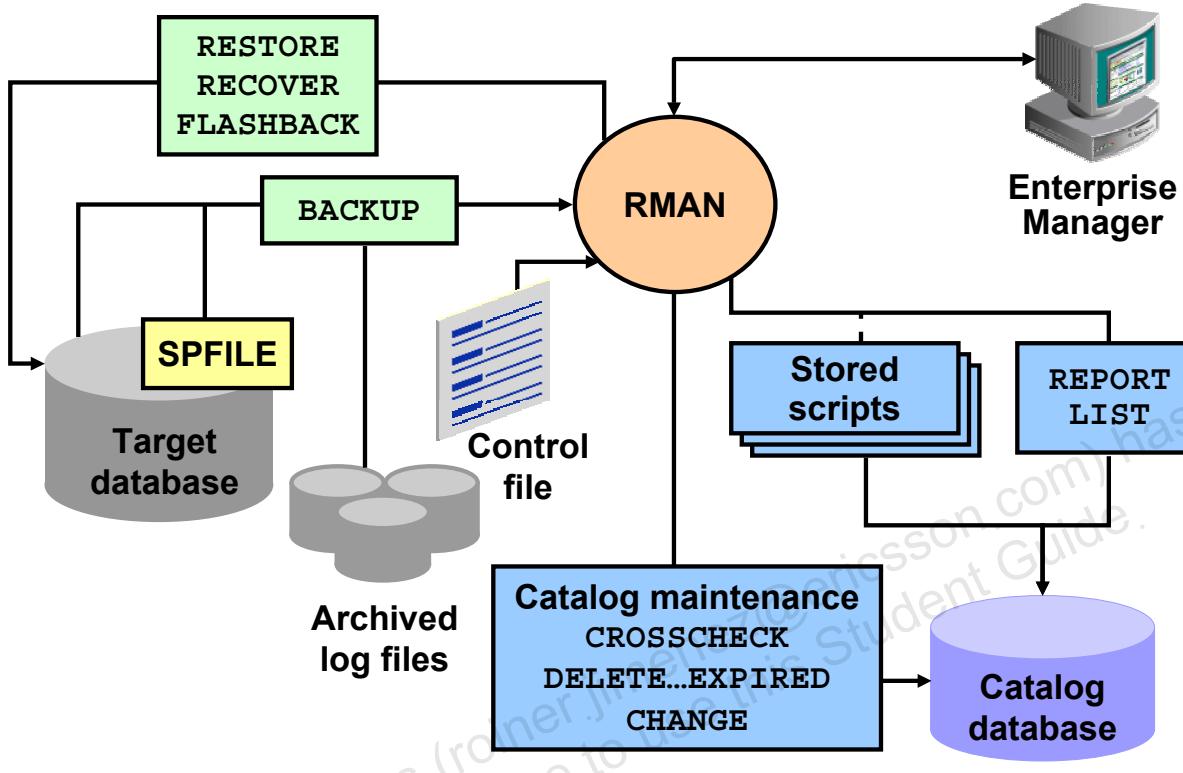
Job Commands: Example

Unlike stand-alone commands, job commands must appear within the braces of a RUN command. The following are examples of job commands:

- ALLOCATE CHANNEL
- SWITCH

RMAN executes the job commands inside a RUN command block sequentially. If any command within the block fails, then RMAN ceases processing; no further commands within the block are executed. In effect, the RUN command defines a unit of command execution. When the last command within a RUN block completes, the Oracle database releases any server-side resources such as I/O buffers or I/O slave processes allocated within the block.

RMAN Commands: Overview



Copyright © 2008, Oracle. All rights reserved.

ORACLE

RMAN Commands: Overview

The typical RMAN commands that you run against the target database include:

- BACKUP to back up a database, tablespace, data file (current or copy), control file (current or copy), SPFILE, archived log, or backup set for a target or standby database. Backing up a backup set is an easy way to move a backup from disk to tape.
- DUPLICATE to create a clone database or a standby database from backups (backup sets or image copies) of the target database
- FLASHBACK to perform a Flashback Database operation, returning the database to (or to just before) a target time, as specified by time, SCN, or log sequence number
- RECOVER to recover, and RESTORE to restore files from backups or image copies

The RMAN reporting commands include:

- LIST for querying the recovery catalog or control file and producing a list of the backups, copies, archived redo logs, and database incarnations recorded there
- REPORT for performing detailed analysis of the recovery catalog or control file

RMAN Commands: Overview (continued)

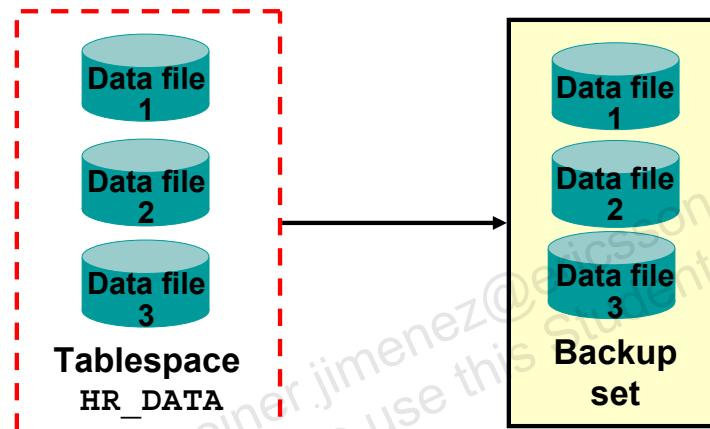
RMAN provides the following command sets for catalog maintenance:

- CROSSCHECK checks the status of a backup or a copy on disk or tape.
- DELETE lists specified backup objects and prompts for confirmation to remove them.
- CHANGE alters the status of backup objects in the repository.
- LIST shows what CROSSCHECK/DELETE EXPIRED would process if run.

BACKUP Command

Commands
 > **BACKUP**
 Block C.T.
 Inc. Update
 LIST
 REPORT
 DELETE

```
RMAN> BACKUP AS BACKUPSET
2>   FORMAT '/BACKUP/df_%d_%s_%p.bus'
3> TABLESPACE hr_data;
```



ORACLE

Copyright © 2008, Oracle. All rights reserved.

BACKUP Command

A backup is a copy of data from your database that can be used to reconstruct that data. The results of a backup created through RMAN can be either image copies or backup sets. An image copy is a bit-for-bit identical copy of a database file. RMAN can also store its backups in an RMAN-exclusive format called a backup set. A backup set is a collection of files called backup pieces, each of which may contain one or more database file backups.

When performing a backup using RMAN, you can specify:

- The type of backup to be performed. You can perform backups of the entire database to include every used data block in the files (a **FULL** backup) or incremental backups (**INCREMENTAL**). If **CONFIGURE CONTROLFILE AUTOBACKUP** is enabled, RMAN automatically backs up the control file and the current server parameter file after **BACKUP** commands.
- What to backup. Valid values are **DATABASE**, **DATAFILE**, **TABLESPACE**, **ARCHIVELOG**, **CURRENT CONTROLFILE**, or **SPFILE**.
- Whether an image copy (**AS COPY**) or backup set (**AS BACKUPSET**) is created
- The file name format and location for backup pieces (**FORMAT**)
- Which data files or archived redo logs should be excluded from the backup set (**SKIP**)
- A maximum size for a backup set (**MAXSETSIZE**)
- That the input files should be deleted upon the successful creation of the backup set (**DELETE INPUT**)

Backup Constraints

- **The database must be mounted or open.**
- **You cannot back up online redo logs.**
- **Only “clean” backups are usable in NOARCHIVELOG mode.**
- **Only “current” data file backups are usable in ARCHIVELOG mode.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

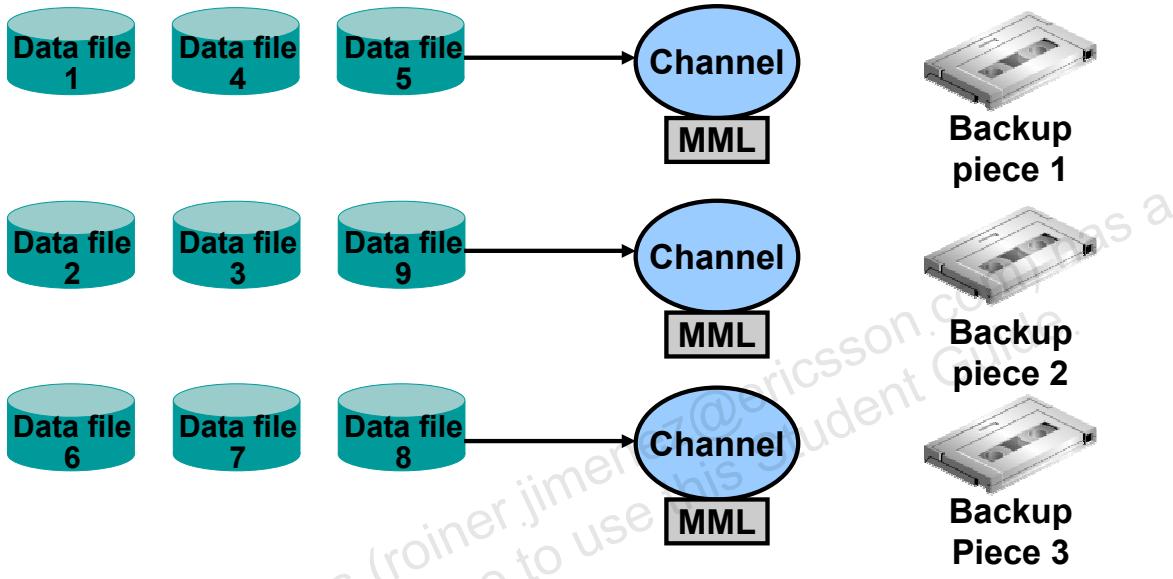
Backup Constraints

When performing a backup by using Recovery Manager, you must be aware of the following:

- The target database must be mounted for Recovery Manager to connect.
- Backups of online redo logs are not supported.
- If the target database is in NOARCHIVELOG mode, only “clean” tablespace and data file backups can be taken (that is, backups of “offline normal” or “read only” tablespaces). Database backups can be taken only if the database has first been shut down cleanly and restarted in MOUNT mode.
- If the target database is in ARCHIVELOG mode, only “current” data files can be backed up (restored data files are made current by recovery).
- If a recovery catalog is used, the recovery catalog database must be open.

Parallelization of Backup Sets

For performance, allocate multiple channels and assign files to specific channels.



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Parallelization of Backup Sets

You can configure parallel backups by setting the PARALLELISM option of the CONFIGURE command to greater than 1 or by manually allocating multiple channels. RMAN parallelizes its operation and writes multiple backup sets in parallel. The server sessions divide the work of backing up the specified files.

Example

```

RMAN> RUN {
  2>   ALLOCATE CHANNEL c1 DEVICE TYPE sbt;
  3>   ALLOCATE CHANNEL c2 DEVICE TYPE sbt;
  4>   ALLOCATE CHANNEL c3 DEVICE TYPE sbt;
  5>   BACKUP
  6>     INCREMENTAL LEVEL = 0
  7>       (DATAFILE 1,4,5 CHANNEL c1)
  8>       (DATAFILE 2,3,9 CHANNEL c2)
  9>       (DATAFILE 6,7,8 CHANNEL c3);
10>   sql 'ALTER SYSTEM ARCHIVE LOG CURRENT';
11> }
  
```

Parallelization of Backup Sets (continued)

When backing up data files, you can specify the files to be backed up by either their path name or their file number. For example, the following two commands perform the same action:

```
BACKUP DEVICE TYPE sbt DATAFILE '/home/oracle/system01.dbf';
BACKUP DEVICE TYPE sbt DATAFILE 1;
```

When you create multiple backup sets and allocate multiple channels, RMAN automatically parallelizes its operation and writes multiple backup sets in parallel. The allocated server sessions share the work of backing up the specified data files, control files, and archived redo logs. You cannot stripe a single backup set across multiple channels.

Parallelization of backup sets is achieved by:

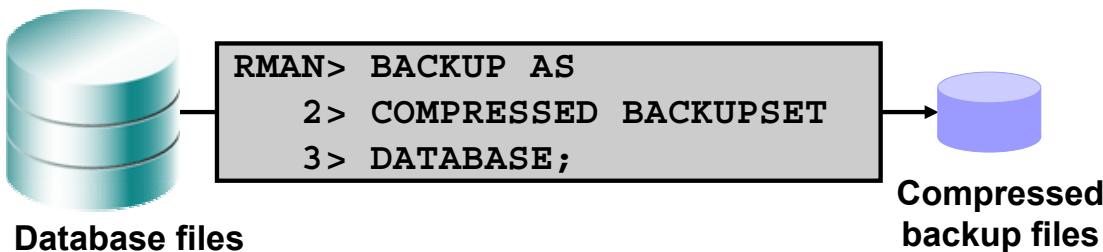
- Configuring PARALLELISM to greater than 1 or allocating multiple channels
- Specifying many files to back up

Example

- There are nine files that need to be backed up (data files 1 through 9).
- Assign the data files to a backup set so that each set has approximately the same number of data blocks to back up (for efficiency).
 - Data files 1, 4, and 5 are assigned to backup set 1.
 - Data files 2, 3, and 9 are assigned to backup set 2.
 - Data files 6, 7, and 8 are assigned to backup set 3.

Note: You can also use the FILESPERSET parameter to limit the number of data files that are included in a backup set.

Compressed Backups



```
RMAN> CONFIGURE DEVICE TYPE
2> DISK PARALLELISM 2
3> BACKUP TYPE TO
4> COMPRESSED BACKUPSET;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Compressed Backups

Compressed backups reduce the amount of space required for storing backup sets.

You create a compressed backup of a database by using the following command:

```
RMAN> BACKUP AS COMPRESSED BACKUPSET DATABASE;
```

The compressed backup set feature cannot be used with pre-Oracle Database 10g databases. The database initialization parameter `COMPATIBILITY` must be set to at least 10.0.0.0. The compression applies only to backup sets, and not to image copies.

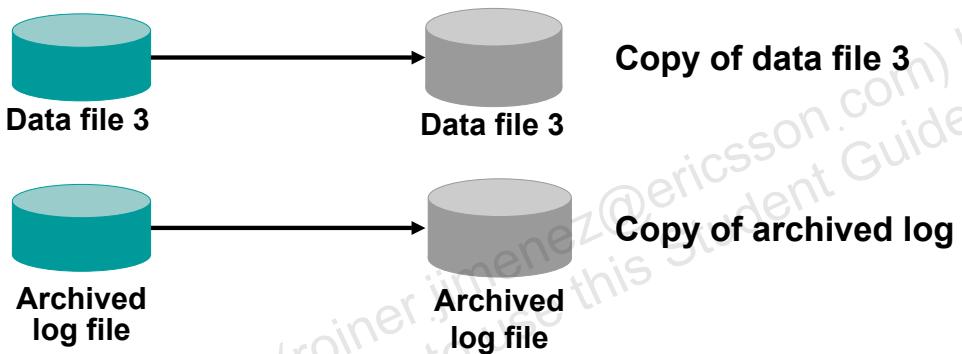
You can configure RMAN to automatically make compressed backup sets with the `CONFIGURE DEVICE TYPE` command:

```
RMAN> CONFIGURE DEVICE TYPE DISK PARALLELISM 2
2> BACKUP TYPE TO COMPRESSED BACKUPSET;
```

By default, compression is disabled.

Image Copy

```
RMAN> BACKUP AS COPY  
2> DATAFILE '/ORADATA/users_01_db01.dbf'  
3> FORMAT '/BACKUP/users01.dbf';  
  
RMAN> BACKUP AS COPY  
4> ARCHIVELOG LIKE 'arch_1060.arc'  
5> FORMAT 'arch_1060.bak';
```



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Image Copy

An image copy is a clone of a single data file, archived redo log, or control file. An image copy can be created with the BACKUP AS COPY command or with an OS command.

When you create the image copy with the RMAN BACKUP AS COPY command, the server session validates the blocks in the file and records the copy in the control file.

An image copy has the following characteristics:

- An image copy can be written only to disk. When large files are being considered, copying may take a long time, but restoration time is reduced considerably because the copy is available on the disk.
- If files are stored on disk, they can be used immediately by using the SWITCH command in RMAN, which is equivalent to the ALTER DATABASE RENAME FILE SQL statement.
- In an image copy, all blocks are copied, whether they contain data or not, because an Oracle database process copies the file and performs additional actions such as checking for corrupt blocks and registering the copy in the control file. To speed up the process of copying, you can use the NOCHECKSUM parameter.
- Image copy can be part of a full or incremental level 0 backup because a file copy always includes all blocks. You must use the level 0 option if the copy will be used in conjunction with an incremental backup set.

Image Copy (continued)

The example in the slide creates two image copies:

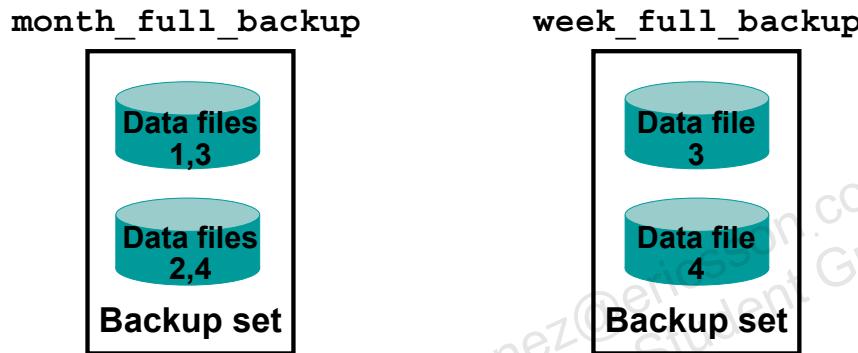
- A copy of the `users01_db01.dbf` data file, renamed as `users01.dbf`, and stored in the BACKUP directory
- A copy of the archived log with sequence number 1060

The example assumes that you are using automatic channel allocation. If you manually allocate channels, you must include the COPY command within the RUN statement as follows:

```
RMAN> RUN {
  2> ALLOCATE CHANNEL c1 type disk;
  3> COPY
  4> DATAFILE '/ORADATA/users_01_db01.dbf' to
      '/BACKUP/users01.dbf',
  6> ARCHIVELOG 'arch_1060.arc' to
      'arch_1060.bak';
  8> }
```

Tags for Backups and Image Copies

A tag is a logical name assigned to a backup set or image copy.



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Tags for Backups and Image Copies

A tag is a meaningful name that you can assign to a backup set or image copy. The advantages of tags are as follows:

- Tags provide a useful reference to a collection of file copies or a backup set.
- Tags can be used in the LIST command to locate backed-up files easily.
- Tags can be used in the RESTORE and SWITCH commands.
- The same tag can be used for multiple backup sets or file copies.

If a nonunique tag references more than one data file, then RMAN chooses the most current available file.

Example

- Each month, a full backup of data files 1, 2, 3, and 4 is performed. The tag in the control file for this backup is `month_full_backup`, even though the physical file name generated is `df_DB00_863_1.dbf`:

```
RMAN> BACKUP TAG 'month_full_backup' DATAFILE 1,2,3,4;
```
- Each week, a full backup of data files 3 and 4 is performed. The tag name for this backup is `week_full_backup`.

```
RMAN> BACKUP TAG 'week_full_backup' DATAFILE 3,4;
```

BACKUP Command Options

- **Check for physical block corruptions.**
- **Scan for logical corruptions and physical corruptions.**
- **Set a threshold on the number of detected corruptions allowed before aborting.**
- **Validate the target input files before performing a backup operation.**
- **Duplex the backup set.**
- **Overwrite an existing backup set or image copy.**
- **Pass control of the data transfer between storage devices and the data files on disk to the media management layer.**
- **Encrypt the backup files.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

BACKUP Command Options

During the backup operation, an Oracle database process computes a checksum for each block to detect corruption. RMAN verifies the checksum when restoring the copy. This is referred to as physical corruption detection. You can use the NOCHECKSUM option to suppress the checksum operation and speed up the backup process. If the database is already maintaining block checksums, then this option has no effect.

You can use the CHECK LOGICAL option to test data and index blocks that pass physical corruption checks for logical corruption—for example, corruption of a row piece or index entry. If logical corruption is detected, the block is logged in the alert log and trace file of the server process. You can set a threshold for the allowed number of logical and physical corruptions with the MAXCORRUPT parameter. As long as the sum of physical and logical corruptions that is detected for a file remains below this value, the RMAN backup operation completes and the Oracle database populates the V\$DATABASE_BLOCK_CORRUPTION view with the corrupt block ranges. If MAXCORRUPT is exceeded, then the operation terminates without populating the view.

You can use the VALIDATE option to check for physical and logical errors in database files. When using the BACKUP command with the VALIDATE option, RMAN scans the specified files and verifies their contents, testing whether this file can be backed up. This command does not actually back up the specified files.

Encryption of backup files is covered in the lesson titled “Database Security.”

BACKUP Command Options (continued)

You can create up to four identical copies of each backup piece by duplexing the backup set. Use any of the following commands to produce a duplexed backup set:

- BACKUP COPIES
- SET BACKUP COPIES
- CONFIGURE . . . BACKUP COPIES

RMAN does not produce multiple backup sets, but produces identical copies of each backup piece in the set. You cannot use this option with the BACKUP AS COPY command to create multiple image copies.

If you specify REUSE, you enable RMAN to overwrite an already existing backup set or image copy with the same file name as the file that BACKUP is currently creating.

The PROXY copy functionality backs up the specified files by giving the media management software control over the data transfer between storage devices and the data files on disk. The media manager—not RMAN—decides how and when to move data. When you run BACKUP with the PROXY option, RMAN performs these steps:

1. It searches for a channel of the specified device type that is proxy-capable. If no such channel is found, then RMAN issues a warning and attempts a conventional (that is, nonproxy) backup of the specified files.
2. If RMAN locates a proxy-capable channel, it calls the media manager to check whether it can proxy copy the files. If the media manager cannot proxy copy, then RMAN uses conventional backup sets to back up the files.

If you do not want RMAN to try a conventional copy when a proxy copy fails, use the ONLY option.

Because image copies are written only to disk, you cannot use the PROXY option with the BACKUP AS COPY command.

Note: If you specify PROXY, then the %p variable must be included in the FORMAT string either explicitly or implicitly within %U.

Backing Up Archived Redo Logs

- **Online redo log file switch is automatic.**
- **Archived log failover is performed.**
- **You can specify a range of archived redo logs to back up.**
- **Backup sets of archived redo log files cannot contain any other type of file.**

```
RMAN> BACKUP
2>   FORMAT '/disk1/backup/ar_%t_%s_%p'
3>   ARCHIVELOG FROM SEQUENCE=234
4>   DELETE INPUT;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Backing Up Archived Redo Logs

A common problem experienced by DBAs is not knowing whether an archived log has been completely copied out to the archive log destination before attempting to back it up. RMAN has access to control file or recovery catalog information, so it knows which logs have been archived and can be restored during recovery.

You can back up archived redo log files with the BACKUP ARCHIVELOG command or back them up while backing up data files and control files with the BACKUP ... PLUS ARCHIVELOG command. Archived logs always end up in a separate backup set even if PLUS ARCHIVELOG is used; in that case, a single backup command generates more than one backup set. If you want to back up only certain archived log files, you can specify the range of archived logs to back up. Archived logs are always put into a separate backup set from other files.

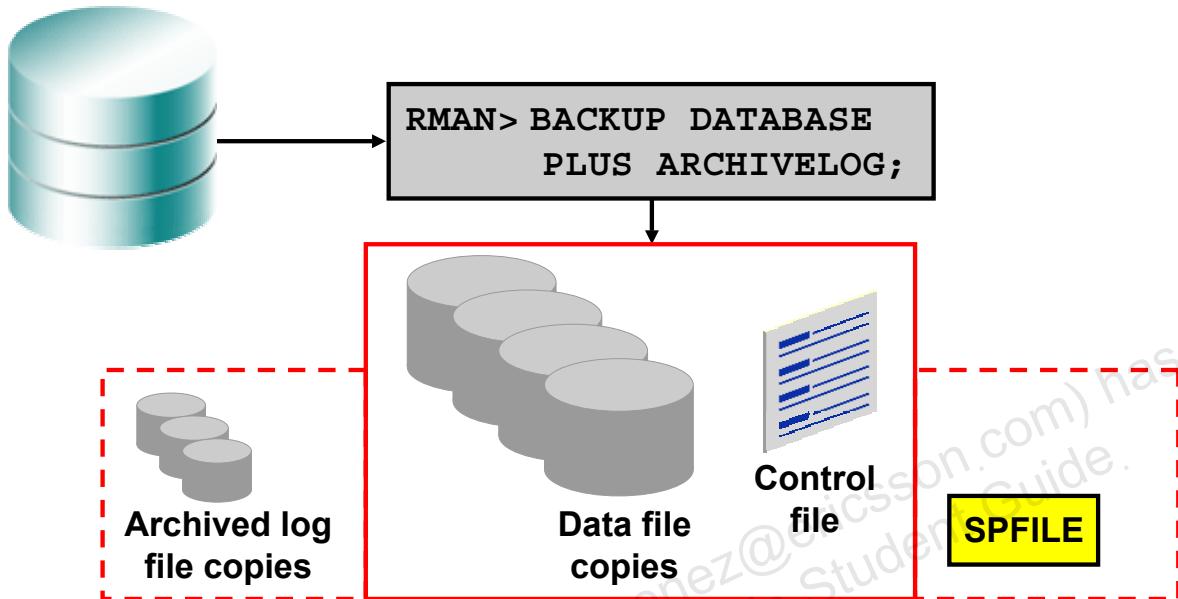
You can also use the NOT BACKED UP *integer* TIMES clause of the BACKUP ARCHIVELOG command to back up only those logs that have not been backed up at least the number of times specified. This is a convenient way to back up archived logs on specified media (for example, you want to keep at least three copies of each log on tape).

Backing Up Archived Redo Logs (continued)

RMAN automatically performs archived log failover. If any corrupt blocks are detected in an archived redo log file, RMAN searches other archiving destinations for a file without corrupt blocks.

The example shown in the slide backs up all archived redo logs with a sequence number of 234 or higher to a backup set. After the archived logs are copied, they are deleted from disk and marked as deleted in the V\$ARCHIVED_LOG view.

Whole Database Backup



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Whole Database Backup

A whole database backup is a copy of all data files and the control file. You can optionally include the SPFILE and archived log files. Using Recovery Manager to make an image copy of all the database files is as easy as mounting the database, starting RMAN, and entering the BACKUP command shown in the slide.

This is possible if you have already issued the following CONFIGURE commands:

- **CONFIGURE DEFAULT DEVICE TYPE TO disk;**
- **CONFIGURE DEVICE TYPE DISK BACKUP TYPE TO COPY;**
- **CONFIGURE CONTROLFILE AUTOBACKUP ON;**

You can also create a backup (either a backupset or image copies) of previous image copies of all data files and control files in the database by using the following command:

```
RMAN> BACKUP COPY OF DATABASE;
```

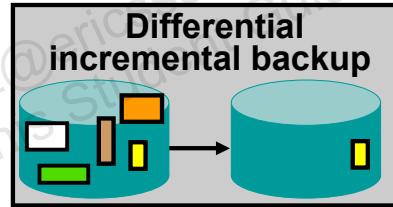
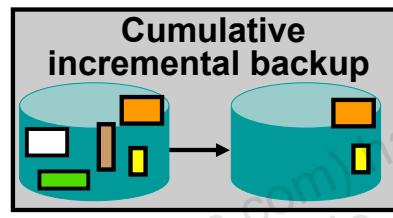
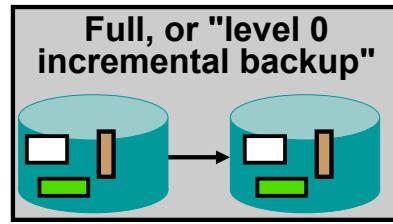
By default, RMAN executes each BACKUP command serially. However, you can parallelize the copy operation by:

- Using the **CONFIGURE DEVICE TYPE DISK PARALLELISM *n***, where *n* is the desired degree of parallelism
- Allocating multiple channels
- Specifying one **BACKUP AS COPY** command and listing multiple files

Note: A high degree of parallelism requires more machine resources, but can save time.

RMAN Backup Types

- A **full backup** contains all used data file blocks.
- A **level 0 incremental backup** is equivalent to a full backup that has been marked as level 0.
- A **cumulative level 1 incremental backup** contains only blocks modified since the last level 0 incremental backup.
- A **differential level 1 incremental backup** contains only blocks modified since the last incremental backup.



ORACLE

Copyright © 2008, Oracle. All rights reserved.

RMAN Backup Types

Full Backups

A full backup is different from a whole database backup. A full data file backup is a backup that includes every used data block in the file. RMAN copies all blocks into the backup set or image copy, skipping only data file blocks that have never been used. For a full image copy, the entire file contents are reproduced exactly. A full backup cannot be part of an incremental backup strategy; it cannot be the parent for a subsequent incremental backup.

Incremental Backups

An incremental backup is either a level 0 backup, which includes every block in the data files except blocks that have never been used, or a level 1 backup, which includes only those blocks that have been changed since a previous backup was taken. A level 0 incremental backup is physically identical to a full backup. The only difference is that the level 0 backup can be used as the base for a level 1 backup, but a full backup can never be used as the base for a level 1 backup.

Incremental backups are specified through the `INCREMENTAL` keyword of the `BACKUP` command. You specify `INCREMENTAL LEVEL = [0 | 1]`.

RMAN Backup Types (continued)

Incremental Backups (continued)

RMAN can create multilevel incremental backups as follows:

- **Differential:** The default type of incremental backup that backs up all blocks changed after the most recent incremental backup at either level 1 or level 0
- **Cumulative:** Backs up all blocks changed after the most recent backup at level 0

Examples

- To perform an incremental backup at level 0, use the following command:

```
    RMAN> BACKUP INCREMENTAL LEVEL 0 DATABASE;
```

- To perform a differential incremental backup, use the following command:

```
    RMAN> BACKUP INCREMENTAL LEVEL 1 DATABASE;
```

- To perform a cumulative incremental backup, use the following command:

```
    RMAN> BACKUP INCREMENTAL LEVEL 1 CUMULATIVE DATABASE;
```

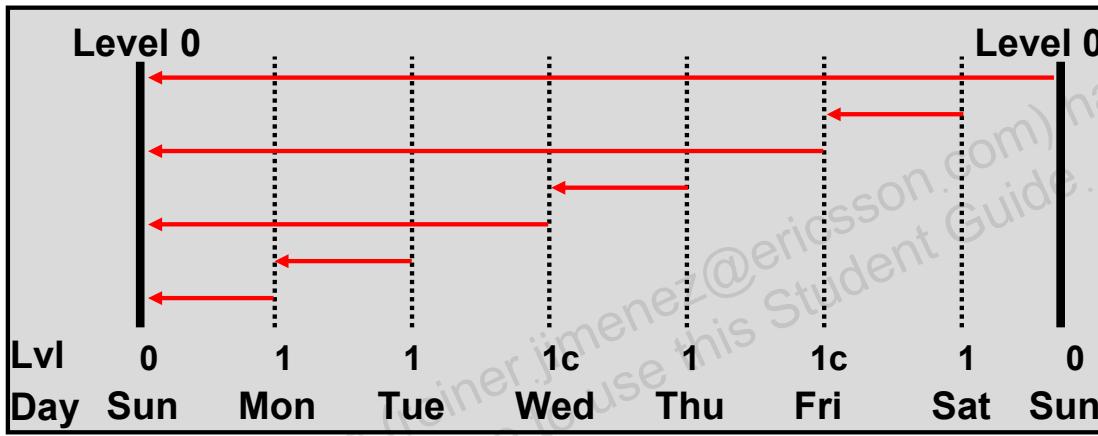
RMAN makes full backups by default if neither FULL nor INCREMENTAL is specified. Unused block compression causes never-written blocks to be skipped when backing up data files to backup sets, even for full backups.

A full backup has no effect on subsequent incremental backups, and is not considered part of any incremental backup strategy, although a full image copy backup can be incrementally updated by applying incremental backups with the RECOVER command. This is covered later in this lesson.

Note: It is possible to perform incremental backups of a database that is in NOARCHIVELOG mode, provided, of course, that it has been shut down.

Differential Versus Cumulative

- A differential incremental backup contains all blocks changed since the last incremental backup.
- A cumulative incremental backup contains all blocks changed since the last level 0 incremental backup.



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Differential Versus Cumulative

Cumulative incremental backups duplicate changes already copied by the previous noncumulative incremental backups at the same level. Therefore, if an incremental level 1 backup is taken, then the following cumulative level 1 backs up all newly modified blocks plus those already backed up by the incremental level 1. This means that only one incremental backup of the same level is needed to completely recover.

Comparison of differential and cumulative backups:

- Differential incremental backups are faster, write out fewer blocks, and produce smaller backup files. This results in a faster backup process, but during recovery, RMAN has to retrieve each incremental backup and apply it.
- Cumulative incremental backups may take longer, write out more blocks, and produce larger backup files. Cumulative backups are provided for recovery speed because fewer backups must be applied when recovering.

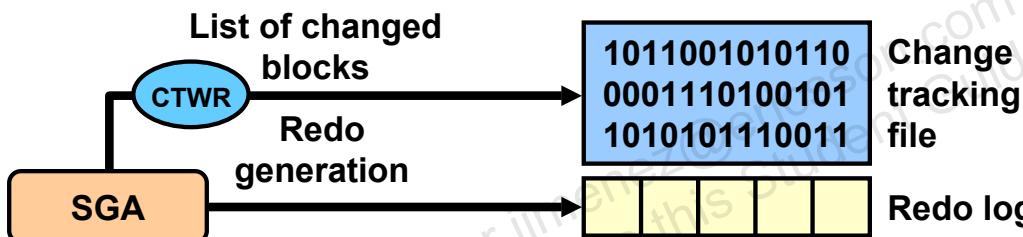
In the graphic shown in the slide, a company has decided on a backup strategy that uses both differential and cumulative backups. Every Sunday, a level 0 incremental backup is taken. Twice a week, on Wednesday and Friday, a cumulative backup is taken to reduce the recovery time of the database. On other days, a differential backup is taken to reduce the backup time and storage requirements.

Block Change Tracking

Commands
BACKUP
> Block C.T.
Inc. Update
LIST
REPORT
DELETE

The backup process can be streamlined by enabling block change tracking, which:

- Records changed blocks in a change tracking file
- Is used automatically by RMAN, if enabled
- Optimizes incremental backups by avoiding full data file scans during backup



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Block Change Tracking

The goal of an incremental backup is to back up only those data blocks that have changed since a previous backup. The entire data file is read during each incremental backup, even if just a very small part of that file has changed since the last incremental backup.

The block change tracking feature uses the change tracking writer (CTWR) background process to record the physical location of all database changes in a new type of file called the *change tracking file*. After enabling change tracking, the first level 0 incremental backup still has to scan the entire data file because the change tracking file does not yet reflect the status of the blocks. For subsequent incremental backups, RMAN uses the change tracking data to determine which blocks to read during an incremental backup, improving performance by eliminating the need to read the entire data file.

You use the same commands to perform incremental backups if block change tracking is enabled, and the change tracking files themselves generally require little maintenance after initial configuration. The size of the block change tracking file is proportional to:

- Database size, in bytes, and the number of enabled threads in a RAC environment
- The number of old backups maintained by the block change tracking file

Enabling Block Change Tracking

```
SQL> ALTER DATABASE ENABLE
2> BLOCK CHANGE TRACKING
3> USING FILE '/mydir/rman_change_track.f'
4> REUSE;
```

Database Instance: orcl.oracle.com > Backup Settings

Backup Settings

[Device](#) [Backup Set](#) **Policy**

Backup Policy

Automatically backup the control file and server parameter file (SPFILE) with every backup and database structural change

Autobackup Disk Location

An existing directory or diskgroup name where the control file and server parameter file will be backed up. If you do not specify a location, the files will be backed up to the flash recovery area location.

Optimize the whole database backup by skipping unchanged files such as read-only and offline datafiles that have been backed up

Enable block change tracking for faster incremental backups

Block Change Tracking File

Specify a location and file, otherwise an Oracle managed file will be created in the database area.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Enabling Block Change Tracking

You enable block change tracking using the Maintenance page of Enterprise Manager (EM). Click the Backup Settings link and then click the Policy tab. You must supply the operating system login credentials to save this backup setting.

The minimum size for the block change tracking file is 10 MB, and new space is allocated in 10 MB increments. You do not have to specify a name for the block change tracking file if the db_create_file_dest parameter is set, in which case an Oracle-managed file is created.

You can use the following commands to configure fast incremental backup manually:

```
SQL> ALTER DATABASE ENABLE BLOCK CHANGE TRACKING;
SQL> ALTER DATABASE DISABLE BLOCK CHANGE TRACKING;
```

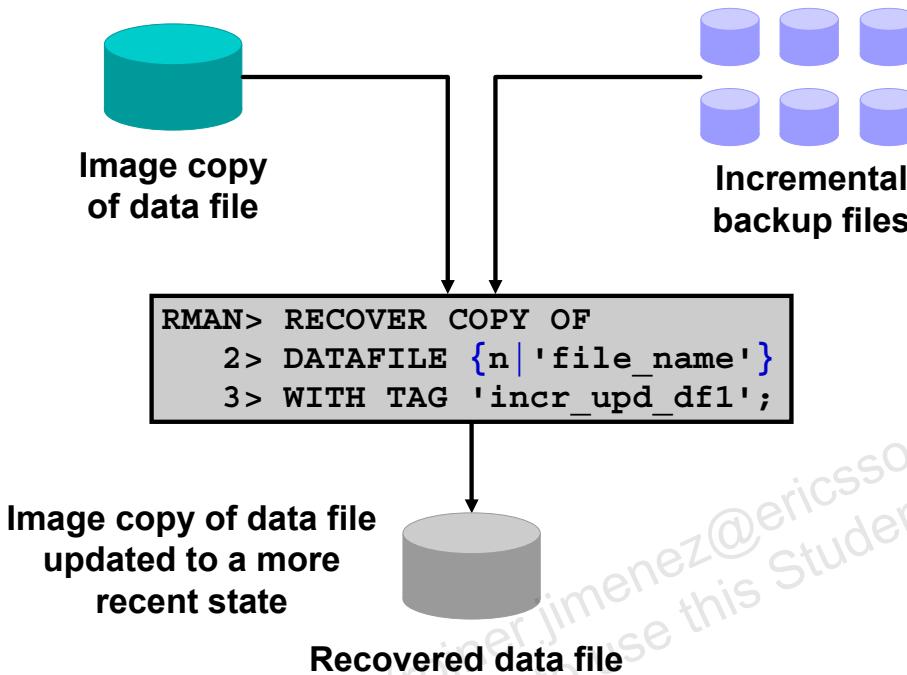
If the db_create_file_dest parameter is not configured, use the USING FILE clause to specify a user-defined directory location and file name for the tracking file.

You can view details for the current block change tracking configuration by querying the V\$BLOCK_CHANGE_TRACKING view.

Change tracking is disabled by default because it does introduce some minimal performance overhead on your database during normal operations.

Incrementally Updating Backups

Commands
BACKUP
Block C.T.
> **Inc. Update**
LIST
REPORT
DELETE



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Incrementally Updating Backups

With the Incrementally Updated Backups feature, you can use RMAN to apply incremental backups to data file image copies. With this recovery method, RMAN can recover a copy of a data file, or roll forward the image copy, to the specified point in time by applying the incremental backups. The image copy is updated with all changes up through the SCN at which the incremental backup was taken.

RMAN can use the resulting updated image copy in media recovery just as it would use a full image copy taken at that SCN, without the overhead of performing a full image copy of the database or data file every day.

Incrementally updating backups requires two different commands:

- Use the BACKUP INCREMENTAL LEVEL 1... FOR RECOVER OF COPY WITH TAG ... form of the BACKUP command to create incremental backups that can be incrementally updated. If an incremental level 0 backup does not already exist, then executing this command creates a level 0 backup with the specified tag.
- Apply any incremental backups to a set of data file copies with the same tag using the RECOVER COPY ... WITH TAG ... form of the BACKUP command. If there is no incremental backup or no data file copy, the command generates a message but does not generate an error.

Tags must be used to identify the incremental backups and data file copies created for use in this strategy, so that they do not interfere with other backup strategies that you implement.

LIST Command

Commands
BACKUP
Block C.T.
Inc. Update
> LIST
REPORT
DELETE

- **List backup sets and copies of data files:**

```
RMAN> LIST BACKUP OF DATABASE;
```

```
RMAN> LIST BACKUP OF DATAFILE
2>   "/db01/ORADATA/u03/users01.dbf";
```

- **List backup sets and copies of any data file for a specified tablespace:**

```
RMAN> LIST COPY OF TABLESPACE "SYSTEM";
```

- **List backup sets and copies containing archive logs for a specified range:**

```
RMAN> LIST COPY OF DATABASE ARCHIVELOG
2> FROM TIME= 'SYSDATE - 7 ';
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

LIST Command

Use the RMAN LIST command to display information about backup sets, proxy copies, and image copies recorded in the repository. Use this command to list:

- Backups and copies that do not have the AVAILABLE status in the RMAN repository
- Backups and copies of data files that are available and can possibly be used in a restore operation
- Backup sets and copies that contain a backup of a specified list of data files or specified tablespaces
- Backup sets and copies that contain a backup of any archived logs with a specified name or range
- Backup sets and copies restricted by tag, completion time, recoverability, or device
- Incarnations of a specified database or of all databases known to the repository
- Stored scripts in the recovery catalog

To use the LIST command, you must be connected to the target database. If you are connected in NOCATALOG mode, then the target database must be mounted. If you connect using a recovery catalog, then the target instance must be started, but does not need to be mounted.

You can control how the output is organized (BY BACKUP or BY FILE) as well as the level of detail in the output (VERBOSE or SUMMARY).

REPORT Command

- **Produces a detailed analysis of the repository**
- **Produces reports to answer:**
 - **What are the data files in the database?**
 - **Which files need a backup?**
 - **Which backups can be deleted?**
 - **Which files are unrecoverable?**



Commands
BACKUP
Block C.T.
Inc. Update
LIST
> REPORT
DELETE

ORACLE

Copyright © 2008, Oracle. All rights reserved.

REPORT Command

With this command, you can analyze information in the RMAN repository in more detail.

Reports can be produced for a variety of questions, such as:

- What is the structure of the database?
RMAN> REPORT SCHEMA;
- Which files need to be backed up?
RMAN> REPORT NEED BACKUP ...;
- Which backups can be deleted (that is, are obsolete)?
RMAN> REPORT OBSOLETE;
- Which files are not recoverable because of unrecoverable operations?
RMAN> REPORT UNRECOVERABLE ...;

REPORT NEED BACKUP Command

- **Lists all data files that require a backup**
- **Assumes that the most recent backup is used during a restore**
- **Provides four options:**
 - **Incremental**
 - **Days**
 - **Redundancy**
 - **Recovery window**
- **Uses the current retention policy configuration if no options are specified**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

REPORT NEED BACKUP Command

The REPORT NEED BACKUP command is used to identify all data files that need a backup. The report assumes that the most recent backup would be used in the event of a restore.

There are four options:

- **Incremental:** An integer that specifies the maximum number of incremental backups that should be restored during recovery. If complete recovery of a data file requires more than the specified number of incremental backups, then the data file requires a new full backup. For example, to report files that need three or more incremental backups for recovery:

```
RMAN > REPORT NEED BACKUP incremental 3 database;
```

- **Days:** An integer that specifies the maximum number of days since the last full or incremental backup of a file. The file needs a backup if the most recent backup is equal to or greater than this number.

For example, to report what system files have not been backed up for three days:

```
RMAN > REPORT NEED BACKUP days 3 tablespace system;
```

- **Redundancy:** An integer that specifies the minimum level of redundancy considered necessary. For example, redundancy level two requires a backup if there are not two or more backups.
- **Recovery window:** A time window in which RMAN should be able to recover the database

REPORT NEED BACKUP: Examples

- **Files that need more than three incremental backups for recovery:**

```
RMAN> REPORT NEED BACKUP incremental 3;
```

- **Files that have not been backed up for three days:**

```
RMAN> REPORT NEED BACKUP days 3;
```

- **Backup needed if there are not two or more backups already:**

```
RMAN> REPORT NEED BACKUP redundancy 2;
```

- **Backup needed to recover to three days ago:**

```
RMAN> REPORT NEED BACKUP  
2>      recovery window of 3 days;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

REPORT NEED BACKUP: Examples

You can evaluate recovery time by reporting on the number of backups that need to be processed in order to recover files. If you do not want to ever have to process more than three incremental backups during a recovery, you can use this command that reports which files violate that requirement:

```
RMAN> REPORT NEED BACKUP incremental 3;
```

If you have recently performed some significant changes in the database, you need to consider which files have not been backed up since that activity. If the changes happened three days ago, the following command reports which files have not been backed up while those changes were implemented:

```
RMAN> REPORT NEED BACKUP days 3;
```

You may want to see which files do not have a certain number of backups already in existence, as a what-if scenario that is different from your retention policy. To do that for redundancy of two, you can enter this command:

```
RMAN> REPORT NEED BACKUP redundancy 2;
```

You can report on which files need to be backed up in order to be able to recover to as recently as three days ago by entering this command:

```
RMAN> REPORT NEED BACKUP recovery window of 3 days;
```

REPORT OBSOLETE and DELETE OBSOLETE

Commands
BACKUP
Block C.T.
Inc. Update
LIST
REPORT
> DELETE

- **Find all recovery files that are deemed obsolete according to the current retention policy settings:**

```
RMAN> REPORT OBSOLETE;
```

- **List the obsolete recovery files, if no more than two backup copies are needed:**

```
RMAN> REPORT OBSOLETE REDUNDANCY 2;
```

- **Delete the backup set with a backup set key of 4:**

```
RMAN> DELETE BACKUPSET 4;
```

- **Delete the recovery files considered obsolete, because they have more than two backups:**

```
RMAN> DELETE OBSOLETE REDUNDANCY 2;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

REPORT OBSOLETE and DELETE OBSOLETE

An obsolete backup is not the same as an expired backup. An obsolete backup is no longer needed according to the user's retention policy. An expired backup is a backup that the CROSSCHECK command fails to find on the specified media device; the repository indicates it is there, but the backup files have been deleted or moved.

Using the REPORT OBSOLETE command, you can identify files that are no longer needed to satisfy backup retention policies. By default, the REPORT OBSOLETE command reports which files are obsolete under the currently configured retention policy. You can generate reports of files that are obsolete according to different retention policies by using REDUNDANCY or RECOVERY WINDOW retention policy options with the REPORT OBSOLETE command.

If you run REPORT OBSOLETE with no options and no retention policy is configured, then RMAN issues an error message.

The DELETE command can remove any file that the LIST and CROSSCHECK commands can operate on. For example, you can delete backup sets, archived redo logs, and data file copies. The DELETE command removes both the physical file and the catalog record for the file. The DELETE OBSOLETE command deletes backups that are no longer needed. It uses the same REDUNDANCY and RECOVERY WINDOW options as REPORT OBSOLETE.

If you delete backups without using RMAN, you can use the CROSSCHECK or UNCATALOG commands to remove the files from the recovery catalog.

Managing Backups with EM

Manage Current Backups

Catalog Additional Files Crosscheck All Delete All Obsolete Delete All Expired

This backup data was retrieved from the database control file.

Backup Sets **Image Copies**

Search

Status Available ▾
 Contents Datafile Archived Redo Log SPFILE Control File
 Completion Time Within a month ▾ **Go**

Results

Crosscheck Change to Unavailable Delete Validate

Select All | Select None

Select	Key	Tag	Completion Time ▾	Contents	Device Type	Status	Keep	Pieces
<input type="checkbox"/>	4	EXAMPLE	Aug 18, 2005 6:46:31 AM	DATAFILE	DISK	AVAILABLE	NO	1
<input type="checkbox"/>	3	USERS	Aug 18, 2005 6:46:08 AM	DATAFILE	DISK	AVAILABLE	NO	1

Navigation aid:

Database home page > Maintenance > Manage Current Backups

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Managing Backups with EM

You can manage your backup records through Enterprise Manager. Backup maintenance functions provided in Enterprise Manager include the following:

- Viewing lists of backups (backup sets and image copies) recorded in the RMAN repository
- Cross-checking your repository:
 - Verifying that the backups listed in the repository exist and are accessible
 - Marking as expired any backups not accessible at the time of the cross-check
- Deleting the record of expired backups from your RMAN repository
- Deleting obsolete backups from the repository and from disk

You can access the Manage Current Backups page in Enterprise Manager by clicking Manage Current Backups in the Backup/Recovery region of the Maintenance page. The Manage Current Backups page has two property pages: Backup Set (the initial view) and Image Copy. Each serves a similar purpose, listing the backups as recorded in the Recovery Manager repository.

Note: If you use a flash recovery area for your backup storage, many maintenance activities are reduced or eliminated because of the flash recovery area's automatic management of disk space and file retention based on the retention policy.

RMAN Dynamic Views

- **V\$ARCHIVED_LOG**
- **V\$BACKUP_CORRUPTION**
- **V\$BACKUP_DEVICE**
- **V\$BACKUP_FILES**
- **V\$BACKUP_PIECE**
- **V\$BACKUP_REDOLOG**
- **V\$BACKUP_SET**
- **V\$BACKUP_SPFILE**
- **V\$COPY_CORRUPTION**
- **V\$RMAN_CONFIGURATION**



ORACLE

Copyright © 2008, Oracle. All rights reserved.

RMAN Dynamic Views

You can use the following views to obtain RMAN information stored in the control file:

- V\$ARCHIVED_LOG shows which archives have been created, backed up, and cleared in the database.
- V\$BACKUP_CORRUPTION shows which blocks have been found to be corrupt during a backup of a backup set.
- V\$BACKUP_DATAFILE is useful for creating equal-sized backup sets by determining the number of blocks in each data file. It can also help you find the number of corrupt blocks in the data file.
- V\$BACKUP_DEVICE displays information about supported backup devices. The special device type DISK is not returned by this view because it is always available.
- V\$BACKUP_FILES displays information about all RMAN backups (both image copies and backup sets) and archived logs. This view simulates the LIST BACKUP and LIST COPY RMAN commands.
- V\$BACKUP_PIECE shows backup pieces created for backup sets.
- V\$BACKUP_REDOLOG shows archived logs stored in backup sets.
- V\$BACKUP_SET shows backup sets that have been created.
- V\$BACKUP_SPFILE displays information about server parameter files in backup sets.
- V\$COPY_CORRUPTION shows which blocks have been found to be corrupt during an image copy.

RMAN Dynamic Views (continued)

- V\$DATABASE_BLOCK_CORRUPTION lists database blocks marked as corrupt during the most recent RMAN backup.
- V\$RMAN_CONFIGURATION lists information about RMAN persistent configuration settings.

If you use proxy settings for RMAN, you can query:

- V\$PROXY_ARCHIVEDLOG
- V\$PROXY_DATAFILE

For backup performance statistics, you can query:

- V\$BACKUP_ASYNC_IO
- V\$BACKUP_SYNC_IO

Monitoring RMAN Backups

- Correlate server sessions with channels by using the SET COMMAND ID command.
- Query V\$PROCESS and V\$SESSION to determine which sessions correspond to which RMAN channels.
- Query V\$SESSION_LONGOPS to monitor the progress of backups and copies.
- Use an operating system utility to monitor the process or threads.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Monitoring RMAN Backups

To correlate a process with a channel during a backup, perform the following steps:

1. Start Recovery Manager and connect to the target database and, optionally, the recovery catalog.
2. Set the COMMAND_ID parameter after allocating the channels and then copy the desired object. The string specified by the SET COMMAND_ID command is entered into the V\$SESSION.CLIENT_INFO column of all allocated channels.

```
run {
    allocate channel t1 type disk;
    set command id to 'rman';
    backup datafile 1;
    release channel t1;}
```

3. Query the V\$PROCESS and V\$SESSION views to get the session identifier (SID) and the operating system process identifier (SPID) for the channels using the previously specified COMMAND_ID string.

```
SELECT sid, spid, client_info
FROM v$process p, v$session s
WHERE p.addr = s.paddr
AND client_info LIKE '%id=rman%';
```

Monitoring RMAN Backups (continued)

4. Query the V\$SESSION_LONGOPS view to get the status of the copy.

```
SELECT sid, serial#, context, sofar, totalwork,  
       round(sofar/totalwork*100,2) "% Complete"  
  FROM V$SESSION_LONGOPS  
 WHERE opname LIKE 'RMAN:%'  
   AND opname NOT LIKE 'RMAN: aggregate%'  
   AND totalwork != 0;
```

5. If you use a channel of type sbt and the copy process appears to hang, query V\$SESSION_WAIT by using the SID obtained in step 3 to determine whether RMAN is waiting for a media manager function call to complete.

```
SELECT * FROM V$SESSION_WAIT WHERE event LIKE '%sbt%';
```

Rainer Jimenez Arias (rainer.jimenez@ericsson.com) has a
non-transferable license to use this Student Guide.

Summary

In this lesson, you should have learned how to:

- **Use the RMAN BACKUP command to create backup sets and image copies**
- **List backups and image copies taken with RMAN**
- **Report and delete obsolete RMAN backups**



Copyright © 2008, Oracle. All rights reserved.

Practice Overview: Backing Up Your Database

This practice covers the following topics:

- **Enabling archival of redo logs for a database**
- **Using RMAN to display the database structure**
- **Using Recovery Manager to back up data files and the control file**
- **Using Recovery Manager to make image copies of data files**
- **Creating a compressed backup of a database**
- **Scheduling a backup job**



Copyright © 2008, Oracle. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2010, Oracle and/or its affiliates.

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a
non-transferable license to use this Student Guide.

Recovering from Noncritical Losses



ORACLE®

Copyright © 2008, Oracle. All rights reserved.

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a
non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- **Recover temporary tablespaces**
- **Recover a redo log group member**
- **Recover from a lost index**
- **Re-create the password file**



Copyright © 2008, Oracle. All rights reserved.

Causes of File Loss

File loss can be caused by:

- **User error**
- **Application error**
- **Media failure**



ORACLE

Copyright © 2008, Oracle. All rights reserved.

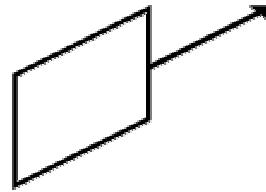
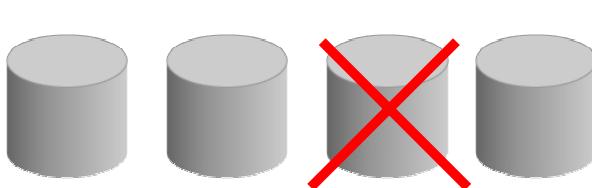
Causes of File Loss

Files can be lost or damaged due to:

- **User error:** An administrator may inadvertently delete or copy over a necessary operating system file.
- **Application error:** An application or script can also have a logic error in it, as it processes database files, resulting in a lost or damaged file.
- **Media failure:** A disk drive or controller may fail fully or partially, and introduce corruption into files, or even cause a total loss of files.

Critical Versus Noncritical

A noncritical file loss is one where the database can continue to function.



You fix the problem by taking one of these actions:

- **Create a new file.**
- **Rebuild the file.**
- **Recover the lost or damaged file.**



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Critical Versus Noncritical

A noncritical file is one that the database and most applications can operate without. For example, if the database loses one multiplexed redo log file, there are still other redo log files that can be used to keep the database operating.

Although the loss of a noncritical file does not cause the database to crash, it can impair the functioning of the database. For example:

- The loss of an index tablespace can cause applications and queries to run much slower, or even make the application unusable, if the indexes were used to enforce constraints.
- The loss of an online redo log group, as long as it is not the current online log group, can cause database operations to be suspended until new log files are generated.
- The loss of a temporary tablespace can prevent users from running queries or creating indexes until they have been assigned to a new temporary tablespace.

Losing a TEMPFILE

SQL statements that require TEMP space to execute fail if one of the tempfiles is missing.

```
SQL> select * from big_table order by
1,2,3,4,5,6,7,8,9,10,11,12,13;
select * from big_table order by
1,2,3,4,5,6,7,8,9,10,11,12,13
*
ERROR at line 1:
ORA-01565: error in identifying file
'/u01/app/oracle/oradata/orcl/temp01.dbf'
ORA-27037: unable to obtain file status
Linux Error: 2: No such file or directory
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Losing a TEMPFILE

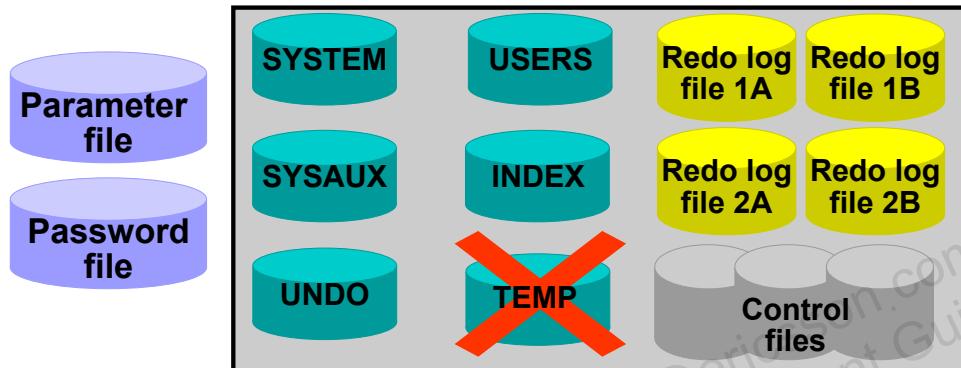
If a tempfile belonging to the TEMP tablespace is lost or damaged, the TEMP tablespace will not be available. This problem manifests itself as errors during the execution of SQL statements that require TEMP space for sorting.

The SQL statement shown in the slide has a long list of columns to order by, which results in the need for TEMP space. This is when the missing file error is encountered.

The Oracle database can start up with a missing temporary file. If any of the temporary files do not exist when the database is started, they are created automatically, and the database opens normally. When this happens, a message like the following appears in the alert log during startup:

```
Recreating tempfile
/u01/app/oracle/oradata/orcl/temp01.dbf
```

Recovering from a TEMPFILE Loss



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Recovering from a TEMPFILE Loss

You can recover from a lost TEMPFILE without restarting the database.

For example, to recover the database when the temp01.dbf data file belonging to the default temporary tablespace TEMP has been deleted at the OS level, add a new data file, and then drop the one that was deleted:

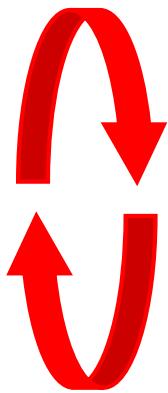
```
SQL> ALTER TABLESPACE temp ADD TEMPFILE  
'/u01/app/oracle/oradata/orcl/temp02.dbf' SIZE 20M;
```

Tablespace altered.

```
SQL> ALTER TABLESPACE temp DROP TEMPFILE  
'/u01/app/oracle/oradata/orcl/temp01.dbf' ;
```

Tablespace altered.

Log Group Status: Review



A redo log group has a status of one of the following values at any given time:

- **CURRENT:** The LGWR process is currently writing redo data to it.
- **ACTIVE:** It is no longer being written to, but it is still required for instance recovery.
- **INACTIVE:** It is no longer being written to, and it is no longer required for instance recovery.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Log Group Status: Review

To deal with the loss of a redo log file, it is important to understand the possible states of redo log groups. Redo log groups cycle through three different states as part of the normal running of the Oracle database. They are, in order of the cycle:

- **CURRENT:** This state means that the redo log group is being written to by LGWR to record redo data for any transactions going on in the database. The log group remains in this state until there is a switch to another log group.
- **ACTIVE:** The redo log group still contains redo data that is required for instance recovery. This is the status during the time when a checkpoint has not yet executed that would write out to the data files all data changes that are represented in the redo log group.
- **INACTIVE:** The checkpoint discussed above has indeed executed, meaning that the redo log group is no longer needed for instance recovery, and is free to become the next CURRENT log group.

Losing a Redo Log Group Member

The alert log and the archiver process (ARCn) trace file record an error when a redo member file is missing.

```
Errors in file
/u01/app/oracle/admin/orcl/bdump/orcl_arcl_25739.trc:
ORA-00313: open failed for members of log group 2 of
thread 1
ORA-00312: online log 2 thread 1:
'./u01/app/oracle/oradata/orcl/redo02b.log'
ORA-27037: unable to obtain file status
Linux Error: 2: No such file or directory
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Losing a Redo Log Group Member

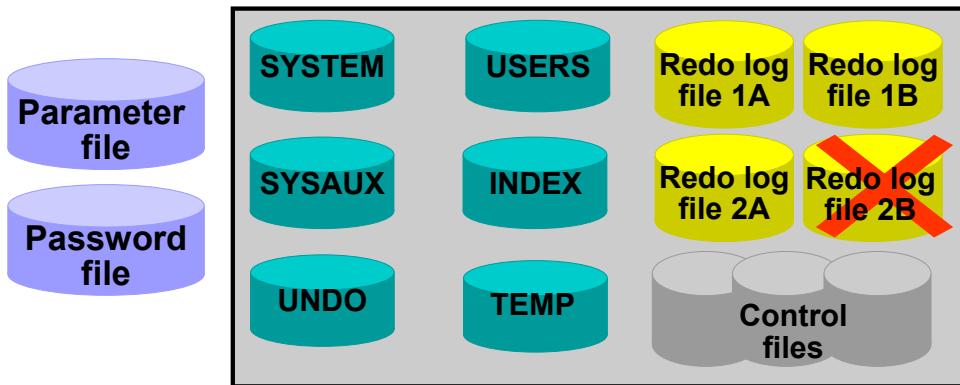
Redo data is crucial for recovery because it contains the record of all modifications to the database, allowing you to roll forward after restoring from a backup.

A redo log group can contain multiple members, and each one is identical to the other within the group. This is for redundancy purposes. At least two redo log groups must be available for the database to continue running, and at least one member must be available in each group.

If a member of a redo log group is inaccessible, there is an error written to the alert log and to the trace file of the archiver process.

If you lose a noncurrent redo log group, then you can use the ALTER DATABASE CLEAR LOGFILE statement to re-create all members in the group. No transactions are lost. If the lost redo log group was archived before it was lost, then nothing further is required. Otherwise, you should immediately take a new full backup of your database. Backups from before the log was lost are not recoverable because of the lost log.

Re-creating Redo Log Files



```
SQL> ALTER DATABASE DROP LOGFILE MEMBER
      > '/u01/app/oracle/oradata/orcl/redo02b.log';
SQL> !rm /u01/app/oracle/oradata/orcl/redo02b.log
SQL> ALTER DATABASE ADD LOGFILE MEMBER
      > '/u01/app/oracle/oradata/orcl/redo02b.log'
      > TO GROUP 2;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Re-creating Redo Log Files

In some cases, you may want to drop an entire group of redo log members, or you may want to drop one or more specific redo log members. For example, if a disk failure occurs, you may need to drop all the redo log files on the failed disk so that the database does not try to write to the inaccessible files.

To drop a redo log group, you must have the ALTER DATABASE system privilege. Before dropping a redo log group, consider the following restrictions and precautions:

- An instance requires at least two groups of redo log files, regardless of the number of members in the groups.
- You can drop a redo log group or group member only if it is inactive.
- A redo log group should be archived (if archiving is enabled) before dropping it. To see whether this has happened, use the V\$LOG view.

```
SELECT GROUP#, ARCHIVED, STATUS FROM V$LOG;
GROUP#    ARC STATUS
-----
1  YES ACTIVE
2  NO  CURRENT
3  YES INACTIVE
```

Re-creating Redo Log Files (continued)

Drop a redo log group by using the ALTER DATABASE SQL statement with the DROP LOGFILE clause. The following statement drops redo log group number 3:

```
ALTER DATABASE DROP LOGFILE GROUP 3;
```

To drop a redo log member, you must have the ALTER DATABASE system privilege.

Consider these restrictions and precautions before dropping individual redo log members:

- You can drop redo log files so that a multiplexed redo log becomes temporarily asymmetric. For example, if you use duplexed groups of redo log files, you can drop one member of one group, even though all other groups have two members each. However, you should rectify this situation immediately so that all groups have at least two members, thus eliminating the single point of failure possible for the redo log.
- An instance always requires at least two valid groups of redo log files, regardless of the number of members in the groups.
- Make sure that the group to which a redo log member belongs is archived (if archiving is enabled) before dropping the member. To see whether this has happened, query the V\$LOG view.
- You can drop a redo log member only if it is not part of an active or current group. If the group you want to drop is active, then force a log switch to occur. The group is then either active or inactive. If it is inactive, you can proceed to drop it. If it is active, then you first need to force a checkpoint in order to make it inactive. The following example shows the progression of log group #1 from CURRENT to ACTIVE to INACTIVE:

```
SQL> SELECT GROUP#, ARCHIVED, STATUS FROM V$LOG;
GROUP# ARC STATUS
```

```
-----  
1 NO CURRENT  
2 YES INACTIVE  
3 YES INACTIVE
```

```
SQL> alter system switch logfile;
```

```
SQL> SELECT GROUP#, ARCHIVED, STATUS FROM V$LOG;
GROUP# ARC STATUS
```

```
-----  
1 YES ACTIVE ←  
2 NO CURRENT  
3 YES INACTIVE
```

```
SQL> alter system checkpoint;
```

```
System altered.
```

```
SQL> SELECT GROUP#, ARCHIVED, STATUS FROM V$LOG;
GROUP# ARC STATUS
```

```
-----  
1 YES INACTIVE ←  
2 NO CURRENT  
3 YES INACTIVE
```

Re-creating Redo Log Files (continued)

To drop specific inactive redo log members, use the ALTER DATABASE statement with the DROP LOGFILE MEMBER clause. The following statement drops the redo log file /u01/app/oracle/oradata/orcl/redo02b.log:

```
ALTER DATABASE DROP LOGFILE MEMBER  
  '/u01/app/oracle/oradata/orcl/redo02b.log';
```

When a redo log group or redo log member is dropped from the database, and you are not using the Oracle Managed Files (OMF) feature, the operating system files are not deleted from disk; only the control file is updated. After dropping a redo log group or redo log member, make sure that the drop completed successfully, and then use the appropriate operating system-command to delete the dropped redo log files, if they are not already gone.

When using Oracle Managed Files, the cleanup of operating systems files is done automatically for you.

Re-creating Redo Log Files

The screenshot shows two Oracle Enterprise Manager pages. The top page is titled 'Redo Log Groups' and displays a table of redo log groups. The bottom page is titled 'Edit Redo Log Group: 2' and shows the details for group #2, which is inactive and has a file size of 51200 KB. It also lists the 'Redo Log Members' for this group, which include two files: redo02.log and redo02b.log. Both files are located in the directory /u01/app/oracle/oradata/orcl/. The 'Edit' button in the main toolbar of the 'Redo Log Groups' table is highlighted with a red box and a red arrow points to the 'Edit' button in the 'Redo Log Members' table.

Select	Group	Status	# of Members	Archived	Size (KB)	Sequence	First Change#
<input type="radio"/>	1	Current	1	No	51200	50	1847791
<input checked="" type="radio"/>	2	Inactive	2	Yes	51200	48	1792269
<input type="radio"/>	3	Inactive	1	Yes	51200	49	1821085

Select	File Name	File Directory	Action Buttons
<input type="radio"/>	redo02.log	/u01/app/oracle/oradata/orcl/	<input type="button" value="Edit"/> <input type="button" value="Remove"/>
<input checked="" type="radio"/>	redo02b.log	/u01/app/oracle/oradata/orcl/	<input type="button" value="Edit"/> <input type="button" value="Remove"/>

ORACLE

Copyright © 2008, Oracle. All rights reserved.

Re-creating Redo Log Files (continued)

With Enterprise Manager, you can create or edit information about the redo log groups associated with the current database. From the Administration page, select Redo Log Groups in the Storage region.

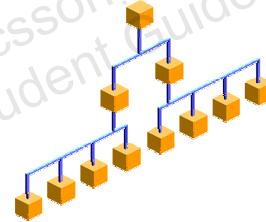
The Redo Log Groups page displays information about each redo log group, enabling you to view or edit a group. Select an individual redo log group and click View. The Redo Log Members table lists the files and directories that comprise the members of the redo log group. You can add or delete members from the group by clicking Edit.

Re-creating Indexes

Use options to reduce the time it takes to create the index:

- **PARALLEL**
- **NOLOGGING**

```
SQL> CREATE INDEX rname_idx
  2  ON hr.regions (region_name)
  3  PARALLEL 4;
```



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Re-creating Indexes

When creating or re-creating an index, you can use the following keywords to reduce the creation time:

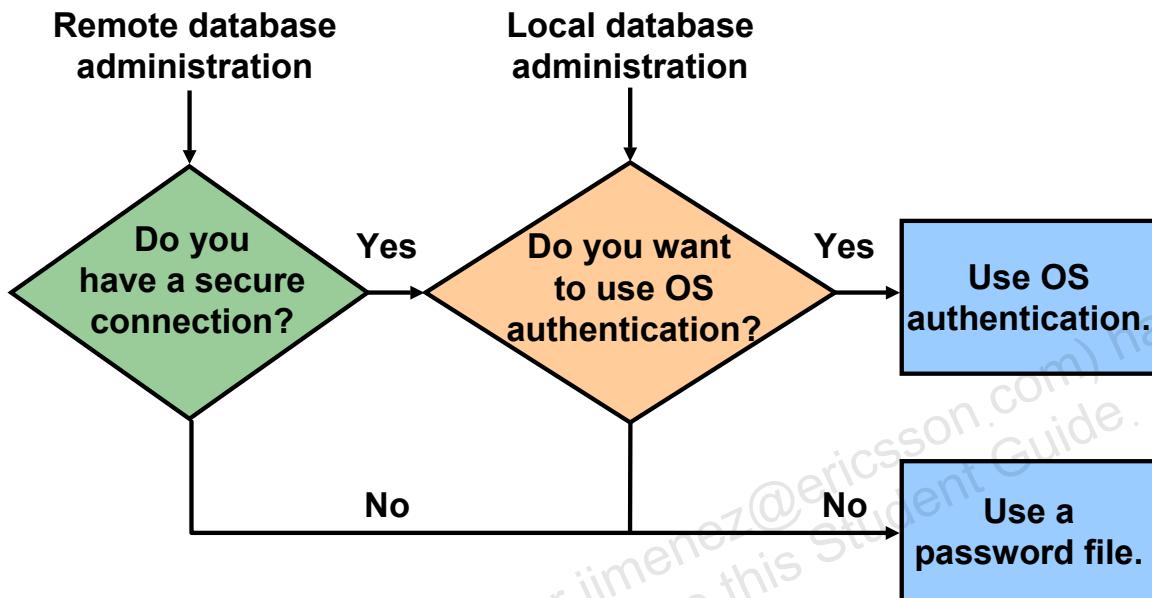
- **PARALLEL (NOPARALLEL is the default):** Multiple processes can work together simultaneously to create an index. By dividing the work necessary to create an index among multiple server processes, the Oracle server can create the index more quickly than if a single server process created the index sequentially. The table is randomly sampled and a set of index keys is found that equally divides the index into the same number of pieces as the specified degree of parallelism. A first set of query processes scans the table, extracts key, row ID pairs, and sends each pair to a process in a second set of query processes based on key. Each process in the second set sorts the keys and builds an index in the usual fashion. After all index pieces are built, the parallel coordinator concatenates the pieces (which are ordered) to form the final index.

Re-creating Indexes (continued)

- **NOLOGGING:** Using this keyword makes index creation faster because it creates a very minimal amount of redo log entries as a result of the creation process. This greatly minimized redo generation also applies to direct path inserts and Direct Loader (SQL*Loader) inserts. This is a permanent attribute and thus appears in the data dictionary. It can be updated with the ALTER INDEX NOLOGGING/LOGGING command at any time.

When an index is lost, it may be faster and simpler just to re-create it rather than attempt to recover it. One way to determine the SQL for creating the index is by using the `impdp` `SQLFILE=<filename>` command on a previously generated `expdp` output file. This generates the SQL statements needed to create the objects in the dump file. The `expdp` and `impdp` utilities are covered in detail in the *Oracle Database 10g: Administration Workshop I* course.

Authentication Methods for Database Administrators



ORACLE

Copyright © 2008, Oracle. All rights reserved.

Authentication Methods for Database Administrators

Depending on whether you want to administer your database locally on the same machine on which the database resides or to administer many different database servers from a single remote client, you can choose either operating system or password file authentication to authenticate database administrators:

- If the database has a password file and you have been granted the SYSDBA or SYSOPER system privilege, then you can be authenticated by a password file.
- If the server is not using a password file, or if you have not been granted SYSDBA or SYSOPER privileges and are, therefore, not in the password file, you can use operating system authentication. On most operating systems, authentication for database administrators involves placing the operating system username of the database administrator in a special group, generically referred to as OSDBA. Users in that group are granted SYSDBA privileges. A similar group, OSOPER, is used to grant SYSOPER privileges to users.

Operating system authentication takes precedence over password file authentication. Specifically, if you are a member of the OSDBA or OSOPER group for the operating system, and you connect as SYSDBA or SYSOPER, you will be connected with associated administrative privileges *regardless of the username/password that you specify*.

Re-creating a Password Authentication File

- 1. Log in to the database by using OS authentication.**
 - 2. Set the REMOTE_LOGIN_PASSWORDFILE parameter to NONE and restart the database.**
 - 3. Re-create the password file by using orapwd.**
- ```
$ orapwd file=$ORACLE_HOME/dbs/orapwORCL
password=admin entries=5
```
- 4. Set REMOTE\_LOGIN\_PASSWORDFILE to EXCLUSIVE.**
  - 5. Add users to the password file and assign appropriate privileges to each user.**
  - 6. Restart the instance.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Re-creating a Password Authentication File

The Oracle database provides a password utility, orapwd, to create a password file. When you connect using the SYSDBA privilege, you are connecting as SYS schema and not the schema associated with your username. For SYSOPER, you are connected to the PUBLIC schema. Access to the database using the password file is provided by special GRANT commands issued by privileged users.

Typically, the password file is not included in backups because, in almost all situations, the password file can be re-created as a last resort. If you lose the password file, to re-create it requires that you shut down and restart the database at least once. To avoid unnecessary down time, you should include the password file in your backups.

It is critically important to the security of your system that you protect your password file and the environment variables that identify the location of the password file. Any user with access to these could potentially compromise the security of the connection.

**Note:** Do not remove or modify the password file if you have a database or instance mounted using REMOTE\_LOGIN\_PASSWORDFILE=EXCLUSIVE (or SHARED). If you do, you will be unable to reconnect remotely using the password file. Even if you replace it, you cannot use the new password file because the time stamps and checksums will be wrong.

## Re-creating a Password Authentication File (continued)

### Using a Password File

1. Log in to the database by using OS authentication.
2. Set the REMOTE\_LOGIN\_PASSWORDFILE parameter to NONE and restart the database.
3. Create the password file by using the password utility orapwd.

```
orapwd file=filename password=password entries=max_users
```

Here:

- **filename** is the name of the password file (mandatory).
- **password** is the password for SYSOPER and SYSDBA (mandatory).
- **Entries** is the maximum number of distinct users allowed to connect as SYSDBA or SYSOPER. If you exceed this number, you must create a new password file. It is safer to have a larger number. There are no spaces around the equal-to (=) character.

**Example:** orapwd file=\$ORACLE\_HOME/dbs/orapwU15  
                  password=admin entries=5

4. Set the REMOTE\_LOGIN\_PASSWORDFILE parameter to EXCLUSIVE, where:
  - **EXCLUSIVE** indicates that only one instance can use the password file and that the password file contains names other than SYS. By using an EXCLUSIVE password file, you can grant SYSDBA or SYSOPER privileges to individual users.
5. Connect to the database by using the password file created in step 3.  
      CONNECT sys/admin AS SYSDBA
6. Restart the instance.

### Password File Locations

UNIX: \$ORACLE\_HOME/dbs

Windows: %ORACLE\_HOME%\database

### Maintaining the Password File

Delete the existing password file by using operating system–commands, and create a new password file by using the password utility.

## Summary

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

- **Recover temporary tablespaces**
- **Recover a redo log group member**
- **Recover from a lost index**
- **Re-create the password file**



Copyright © 2008, Oracle. All rights reserved.

## Practice Overview: Recovering from Lost TEMPFILE and Redo Log File

**This practice covers the following topics:**

- Starting the database with a missing tempfile
- Creating a new temp file
- Altering the default temporary tablespace for a database
- Recovering from a lost online redo log member



Copyright © 2008, Oracle. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2010, Oracle and/or its affiliates.

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a  
non-transferable license to use this Student Guide.

# Database Recovery

ORACLE®

Copyright © 2008, Oracle. All rights reserved.

# Objectives

User man.  
RMAN CLI  
EM  
Complete  
Incomplete  
Control file  
RO TBS

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

- **Perform complete or incomplete user-managed recovery**
- **Identify situations where incomplete recovery is necessary**
- **Perform complete or incomplete recovery by using RMAN**
- **Perform incomplete recovery based on time, SCN, log sequence, restore points, or the cancel method**
- **Recover an automatically backed up control file**
- **Use Enterprise Manager to perform recovery**
- **Recover read-only tablespaces**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

# Recovery Methods

**There are two methods for performing recovery:**

- **User-managed recovery**
  - Files must be maintained and moved into place manually.
  - Use SQL\*Plus commands.
- **RMAN recovery**
  - Files are managed automatically.
  - Use RMAN functionality including all repository maintenance and reporting capabilities.
  - This can be done by using Enterprise Manager.
  - Oracle Corporation recommends using this method.

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Recovery Methods

User-managed recovery requires more manual involvement than RMAN recovery, which is the recommended method and the one covered in more detail in this lesson.

## User-Managed Recovery: RECOVER Command

- **Restore all database files from a backup and then recover the database:**

```
SQL> RECOVER DATABASE
```

- **Restore the damaged data files from a backup and then recover the data files:**

```
SQL> RECOVER TABLESPACE index_tbs
```

Or:

```
SQL> RECOVER DATAFILE
2> '/oradata/indx01.dbf'
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### User-Managed Recovery: RECOVER Command

The first step in performing user-managed media recovery is to manually restore the data files by copying them from a backup. If you do not restore a data file to its original location, you must update the control file with the new location by using an ALTER DATABASE RENAME FILE command. You must also restore any archived logs files needed to recover the restored data files. For RMAN restorations, you would use the SET NEWNAME command to specify the new location for that file.

You can use one of the following commands to recover the database or data file:

- RECOVER [AUTOMATIC] DATABASE  
This command can be used only for a closed database recovery.
- RECOVER [AUTOMATIC] TABLESPACE <NUMBER> | <NAME>  
This command can be used only for an open database recovery.
- RECOVER [AUTOMATIC] DATAFILE <'filename'> | <NAME>  
This command can be used for both an open and a closed database recovery.

The AUTOMATIC keyword instructs the Oracle database to automatically generate the name of the next archived redo log file needed to continue the recovery operation. Otherwise, you are prompted for these names.

# RMAN Recovery: RESTORE and RECOVER Commands

User man.  
> **RMAN CLI**  
EM  
Complete  
Incomplete  
Control file  
RO TBS

```
run{
 sql "ALTER TABLESPACE inv_tbs OFFLINE IMMEDIATE";
 RESTORE TABLESPACE inv_tbs;
 RECOVER TABLESPACE inv_tbs DELETE ARCHIVELOG;
 sql "ALTER TABLESPACE inv_tbs ONLINE";
}
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## RMAN Recovery: RESTORE and RECOVER Commands

If you use RMAN to perform media recovery, it restores from backup any archived redo logs required during the recovery operation. If backups are stored on a media manager, channels must be configured or allocated for use in accessing backups stored there.

Reconstructing the contents of all or part of a database from a backup typically involves two phases: retrieving a copy of the data file from a backup, and reapplying changes to the file since the backup from the archived and online redo logs, to bring the database to the desired SCN (usually the most recent one). The RESTORE command retrieves the data file onto disk from a backup location on tape, disk, or other media, and makes it available to the database server. The RECOVER command takes the restored copy of the data file and applies to it the changes recorded in the database's redo logs.

A very useful option in managing disk space associated with these restored files is the DELETE ARCHIVELOG option, which causes the deletion of restored archived redo logs from disk when they are no longer needed for the RECOVER operation.

For restores to the flash recovery area, the DELETE ARCHIVELOG option is automatically in effect.

# Recovery Using Enterprise Manager

User man.  
RMAN CLI  
> EM  
Complete  
Incomplete  
Control file  
RO TBS

With the Enterprise Manager Recovery Wizard, you can create and run an RMAN script to perform the recovery.

Database Instance: orcl.oracle.com

Home Performance Administration Maintenance

The Administration tab displays links that allow you to administer database objects and initiate database operations inside an Oracle database. The Maintenance tab displays links that provide functions that control the flow of data between or outside Oracle databases.

**High Availability**

|                                        |                                           |                                                |
|----------------------------------------|-------------------------------------------|------------------------------------------------|
| <a href="#">Backup/Recovery</a>        | <a href="#">Backup/Recovery Settings</a>  | <a href="#">Oracle Secure Backup</a>           |
| <a href="#">Schedule Backup</a>        | <a href="#">Backup Settings</a>           | <a href="#">Oracle Secure Backup</a>           |
| <a href="#">Perform Recovery</a>       | <a href="#">Recovery Settings</a>         | <a href="#">Device and Media</a>               |
| <a href="#">Manage Current Backups</a> | <a href="#">Recovery Catalog Settings</a> | <a href="#">File System Backup and Restore</a> |
| <a href="#">Manage Restore Points</a>  |                                           |                                                |
| <a href="#">Backup Reports</a>         |                                           |                                                |

RMAN> RECOVER DATABASE ...

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Recovery Using Enterprise Manager

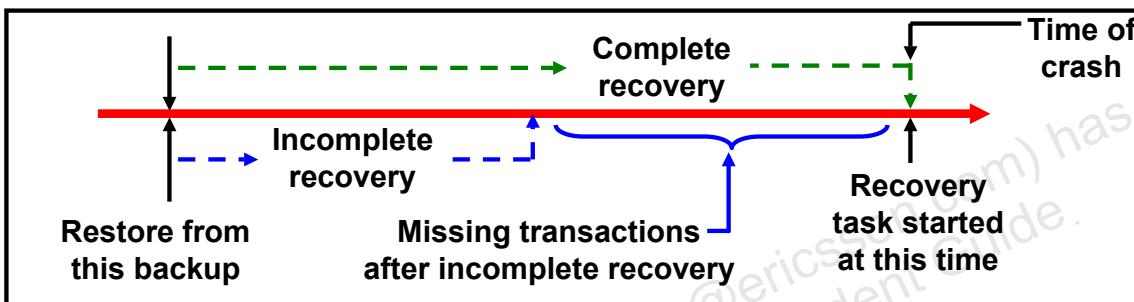
You can also perform complete or incomplete recovery by using the Recovery Wizard available through Enterprise Manager. On the Login page, log in as a user with the SYSDBA privilege. After clicking Login, you see the Database Summary page.

On the Database Summary page, click the Maintenance tab. The Maintenance page provides the user with various backup, restore, and recovery options. On the Maintenance page, click Perform Recovery.

# Complete Versus Incomplete Recovery

There are two types of recovery:

- Complete recovery brings the database up to the present, including all committed data changes made to the point in time when the recovery was requested.



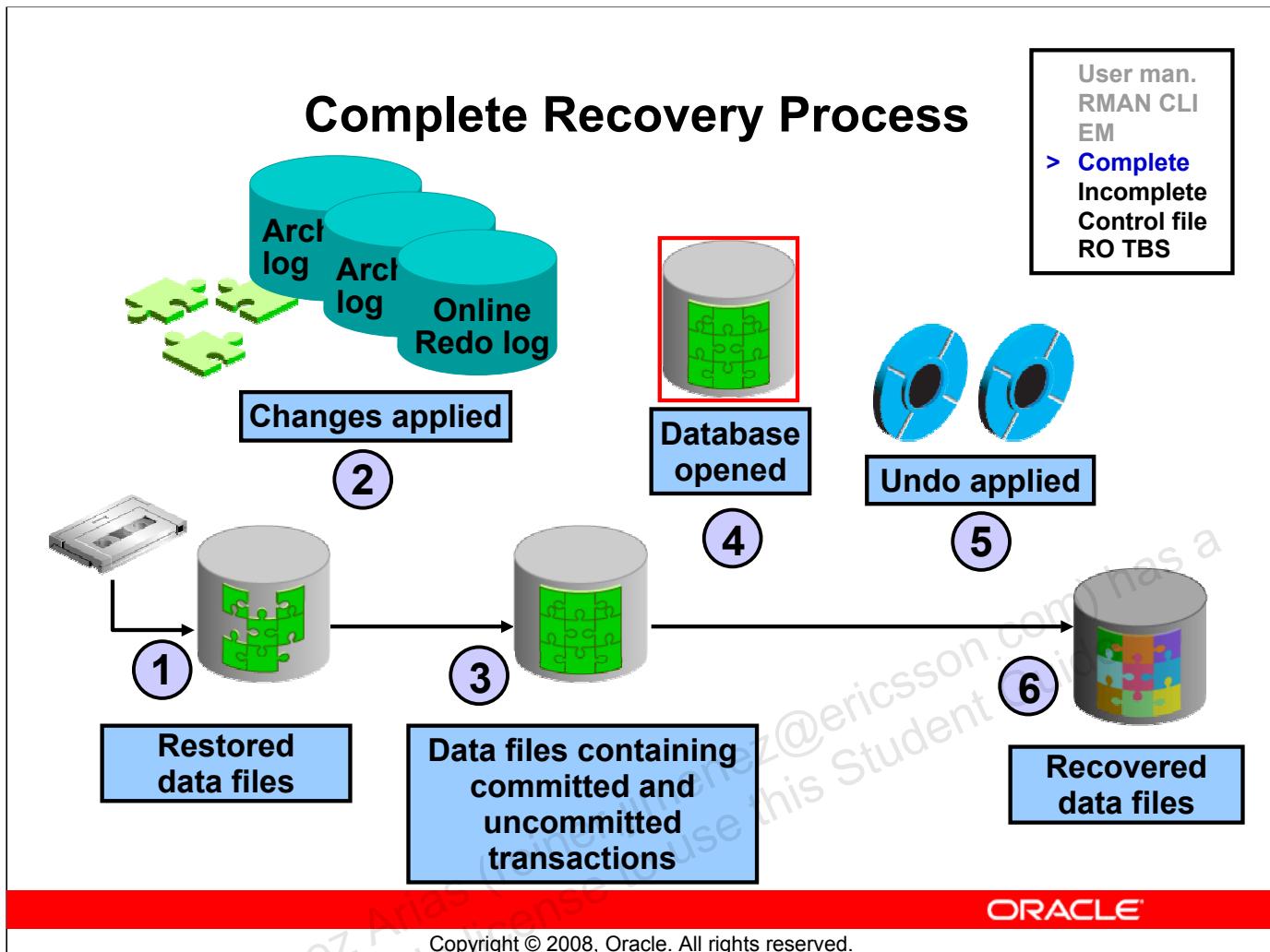
- Incomplete recovery brings the database up to a specified point in time in the past, before the recovery operation was requested.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Complete Versus Incomplete Recovery

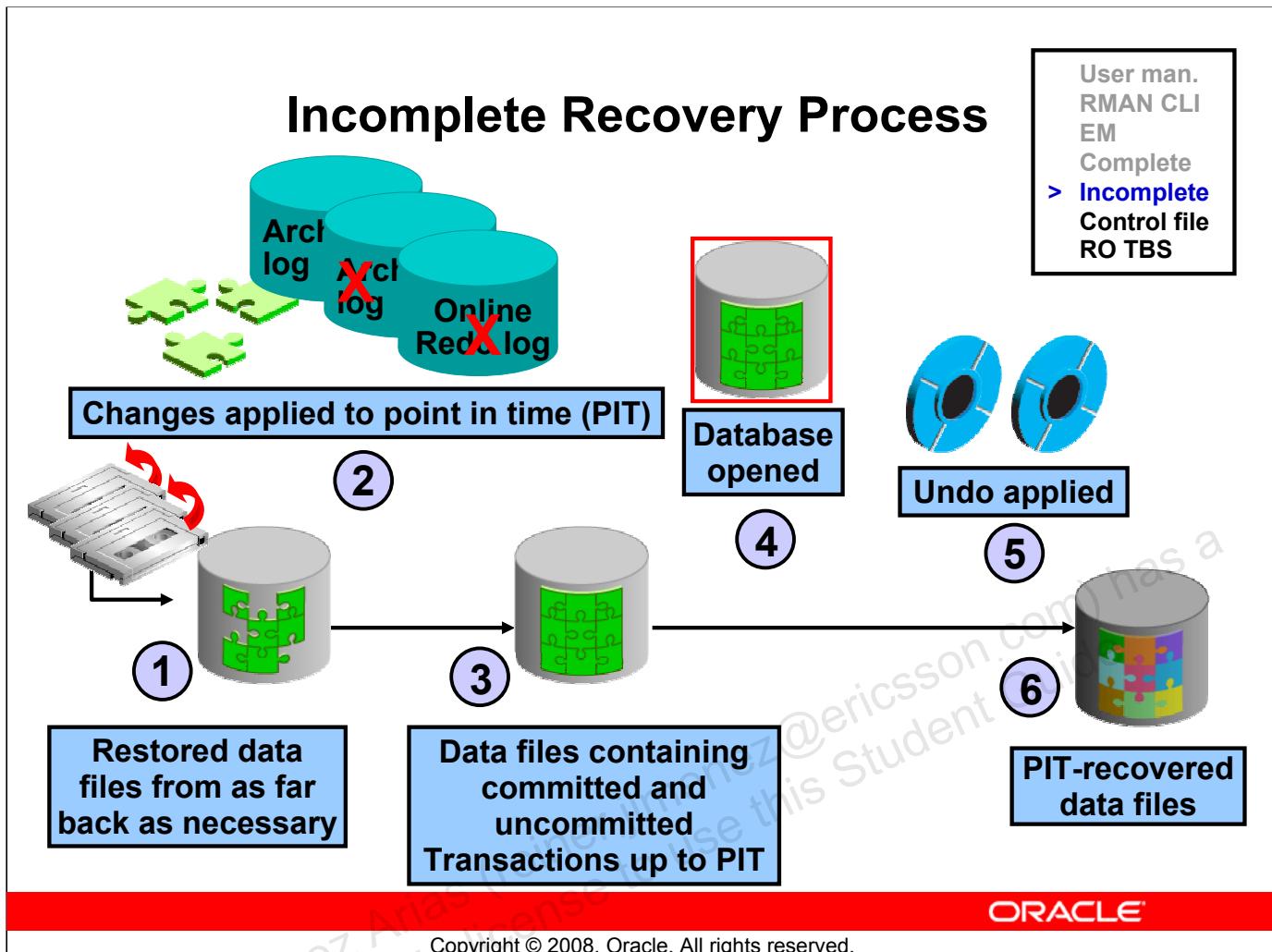
When you perform complete recovery, you bring the database to the state where it is fully up-to-date, including all committed data modifications done to the present time. Incomplete recovery, however, brings the database to some point in the past. This means there are missing transactions; any data modifications done between the recovery destination time and the present are lost. In many cases, this is the desirable goal because there may have been some things done to the database that should be undone. Recovering to a point in the past is a way to remove those unwanted transactions.



## Complete Recovery Process

These are the steps that take place during complete recovery:

1. Damaged or missing files are restored from a backup.
2. Changes from incremental backups, archived redo log files, and online redo log files are applied as necessary. The redo log changes are applied to the data files until the current online log is reached and the most recent transactions have been reentered. Undo blocks are generated during this entire process. This is referred to as rolling forward or cache recovery.
3. The restored data files may now contain committed and uncommitted changes.
4. The database is opened before undo is applied. This is to provide higher availability.
5. The undo blocks are used to roll back any uncommitted changes. This is sometimes referred to as transaction recovery.
6. The data files are now in a recovered state and are consistent with the other data files in the database.



### Incomplete Recovery Process

Incomplete recovery, or database point-in-time recovery, uses a backup to produce a noncurrent version of the database. That is, you do not apply all of the redo records generated after the most recent backup. Perform this type of recovery only when absolutely necessary. To perform incomplete recovery, you need:

- A valid offline or online backup of all the data files made before the recovery point
- All archived logs from the time of the backup until the specified time of recovery

The progression taken to perform an incomplete recovery is listed below:

1. Restore data files from backup. The backup that is used may not be the most recent one, if your restore point destination is to be not very recent.
2. Apply redo from the archived redo log files, including as many as necessary to reach the restore point destination.
3. Now the data files contain some committed and some uncommitted transactions because the redo can contain uncommitted data.
4. The database is opened before undo is applied. This is to provide higher availability.
5. While the redo was being applied, redo supporting the undo data files was also applied. So, the undo is available to be applied to the data files in order to undo any uncommitted transactions. That is done next.
6. The data files are now recovered to the point in time that you chose.

## Incomplete Recovery (continued)

Point-in-time recovery is the only option if you must perform a recovery and discover that you are missing an archived log containing transactions that occurred sometime between the time of the backup you are restoring from and the target recovery SCN. Without the missing log, you have no record of the updates to your data files during that period. Your only choice is to recover the database from the point in time of the restored backup, as far as the unbroken series of archived logs permits, and then open the database with the RESETLOGS option. All changes in or after the missing redo log file are lost.

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a  
non-transferable license to use this Student Guide.

## Situations Requiring Incomplete Recovery

- **Complete recovery fails because of a missing archived log file.**
- **One or more unarchived redo log files and a data file are lost.**
- **Non-multiplexed online redo log files are lost.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Situations Requiring Incomplete Recovery

You usually perform incomplete recovery of the whole database in the following situations:

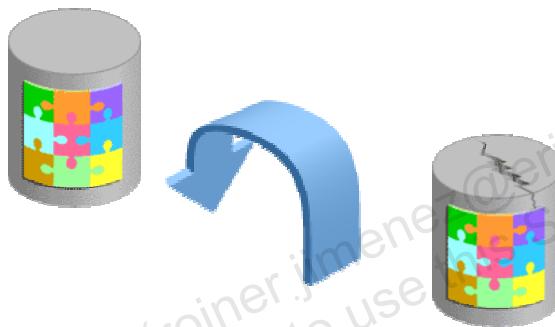
- **Loss of redo logs:** Redo logs were not mirrored and you lost a redo log before it was archived, along with a data file. Recovery cannot continue past the lost redo log.
- **Missing archive:** A complete recovery operation fails because of a bad or missing archived log. Recovery can only be completed to a time in the past, before applying the archived log.

You must specify the USING BACKUP CONTROLFILE clause in the RECOVER DATABASE command when using an old copy of the control file for recovery or to open the database.

# Types of Incomplete Recovery

**There are four types of incomplete recovery:**

- **Time-based recovery**
- **Cancel-based recovery**
- **Change-based recovery**
- **Log sequence recovery**



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Types of Incomplete Recovery

### Time-Based Recovery

Using the UNTIL TIME clause, you specify the previous point in time to which the database should be recovered. Recovery terminates after all changes up to the specified time are committed. Use this approach when a user makes unwanted changes to data or drops important tables, and the approximate time of the error is known. Recovery time and data loss are minimized if you are notified immediately. Well-tested programs, security, and procedures should prevent the need for this type of recovery.

### Cancel-Based Recovery

During the recovery process, you enter CANCEL at the recovery prompt (instead of a log file name) to terminate recovery. Use this approach when:

- A current redo log file or group is damaged and is not available for recovery. Mirroring should prevent the need for this type of recovery.
- An archived redo log file needed for recovery is lost. Frequent backups and multiple archive destinations should prevent the need for this type of recovery.

## Types of Incomplete Recovery (continued)

### Change-Based Recovery

Using the UNTIL CHANGE clause for user-managed backups and the UNTIL SCN clause for RMAN-managed backups, you specify the system change number (SCN) of the last committed change to be recovered. Recovery terminates after all changes up to the specified SCN are committed. Use this approach when recovering databases in a distributed environment. You can optionally use the UNTIL RESTORE POINT syntax and specify an alias for the SCN, called a restore point. Restore points are covered later in this lesson.

### Log Sequence Recovery

With RMAN-managed backups, you can specify the last log sequence number to be used for database recovery with the UNTIL SEQUENCE clause. After all log files up to but not including the specified log file have been applied, recovery terminates.

# Performing User-Managed Incomplete Recovery

- Recover a database until time:

```
SQL> RECOVER DATABASE UNTIL
2 TIME '2005-12-14:12:10:03';
```

- Recover a database until cancel:

```
SQL> RECOVER DATABASE UNTIL CANCEL;
```

- Recover using the backup control file:

```
SQL> RECOVER DATABASE
2 UNTIL TIME '2005-12-14:12:10:03'
3 USING BACKUP CONTROLFILE;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Performing User-Managed Incomplete Recovery

The following command is used to perform incomplete recovery:

```
RECOVER [AUTOMATIC] DATABASE option
```

Here:

- **AUTOMATIC:** Automatically applies archived and redo log files
- **option:** UNTIL TIME 'YYYY-MM-DD:HH24:MI:SS'  
                  UNTIL CANCEL  
                  UNTIL CHANGE <integer>  
                  USING BACKUP CONTROLFILE

**Note:** To apply redo log files automatically during recovery, you can use the SQL\*Plus SET AUTORECOVERY ON command, enter AUTO at the recovery prompt, or use the RECOVER AUTOMATIC command.

## Performing User-Managed Incomplete Recovery

To perform user-managed incomplete recovery, follow these steps:

1. Shut down the database.
2. Restore data files.
3. Mount the database.
4. Recover the database.
5. Open the database with the RESETLOGS option.



Copyright © 2008, Oracle. All rights reserved.

### Performing User-Managed Incomplete Recovery (continued)

1. If the database is open, shut it down by using the NORMAL, IMMEDIATE, or TRANSACTIONAL option.
2. Restore all data files from backup (the most recent, if possible). You may also need to restore archived logs. If there is enough space available, restore to the LOG\_ARCHIVE\_DEST location or use the ALTER SYSTEM ARCHIVE LOG START TO <LOCATION> command or the SET LOGSOURCE <LOCATION> command to change the location.
3. Mount the database.
4. Recover the database by using the RECOVER DATABASE command.
5. To synchronize data files with control files and redo logs, open the database by using the RESETLOGS option.

## User-Managed Time-Based Recovery: Example

### This is the scenario:

- A job ran in error, and its effects have to be undone.
- This happened 15 minutes ago, and there has been little database activity since then.
- You decide to perform incomplete recovery to restore the database back to its state as of 15 minutes ago.

```
SQL> SHUTDOWN IMMEDIATE
$ cp /BACKUP/*.dbf/u01/db01/ORADATA
SQL> STARTUP MOUNT
SQL> RECOVER DATABASE UNTIL TIME '2005-11-28:11:44:00';
SQL> ALTER DATABASE OPEN RESETLOGS;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### User-Managed Time-Based Recovery: Example

The following is a typical scenario employing UNTIL TIME recovery. Assume the following facts:

- The current time is 12:00 p.m. on November 28, 2005.
- A job was run incorrectly, and many tables in several schemas were affected.
- This happened at approximately 11:45 a.m.
- Database activity is minimal because most staff are currently in a meeting. The state of the database before the job ran must be restored.

Because the approximate time of the error is known and the database structure has not changed since 11:45 a.m., you can use the UNTIL TIME method:

1. If the database is open, shut it down by using the NORMAL, IMMEDIATE, or TRANSACTIONAL option.
2. Restore all data files from backup (the most recent if possible). You may also need to restore archived logs. If there is enough space available, restore to the LOG\_ARCHIVE\_DEST location or use the ALTER SYSTEM ARCHIVE LOG START TO <LOCATION> command or the SET LOGSOURCE <LOCATION> command to change the location.
3. Mount the database.

## User-Managed Time-Based Recovery: Example (continued)

4. Recover the database:

```
SQL> recover database until time '2005-11-28:11:44:00'
ORA-00279: change 148448 ... 11/27/05 17:04:20 needed for
thread ...
Media recovery complete.
```

5. To synchronize data files with control files and redo logs, open the database by using the RESETLOGS option:

```
SQL> alter database open resetlogs;
SQL> archive log list
...
Oldest online log sequence 0
Next log sequence to archive 1
Current log sequence 1
```

When recovery is successful, notify users that the database is available for use, and any data entered after the recovery time (11:44 a.m.) will need to be reentered.

## User-Managed Cancel-Based Recovery: Example

The scenario is the same as the one for the time-based example, except for these findings:

- Redo logs are not multiplexed.
- One of the online redo logs is missing.
- The missing redo log is not archived.
- The redo log contained information from 11:34 a.m.
- Twenty-six minutes of data are lost.
- Users can reenter their data manually.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### User-Managed Cancel-Based Recovery: Example

After searching through the directory for the redo log files, you notice that redo log log2a.rdo cannot be located and has not been archived. Therefore, you cannot recover past this point.

Querying V\$ARCHIVED\_LOG confirms the absence of archived log sequence 48 (log2a.rdo):

```
SQL> SELECT * FROM v$archived_log;
RECID STAMP ... FIRST_CHANGE# FIRST_TIME
----- ----- ... ----- -----
 1 318531466 ... 88330 05-11-15:12:43
 47 319512880 ... 309067 05-11-28:11:26
```

## User-Managed Cancel-Based Recovery: Example

**Recover the database as follows:**

- 1. Shut down the database.**
- 2. Restore all data files from the most recent backup.**
- 3. You already have a valid backup, so mount the database.**
- 4. Execute RECOVER DATABASE UNTIL CANCEL.**
- 5. Execute ALTER DATABASE OPEN RESETLOGS to open the database.**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

### User-Managed Cancel-Based Recovery: Example (continued)

The steps for cancel-based recovery are the same as for time-based recovery, except for the RECOVER DATABASE step. When the RECOVER DATABASE UNTIL CANCEL command is executed, it recovers the database until it cannot find a log file. When you are prompted for the file name, enter CANCEL, and the recovery stops at that point in time.

# Performing Incomplete Recovery by Using RMAN

- 1. Mount the database.**
- 2. Allocate multiple channels for parallelization.**
- 3. Restore all data files.**
- 4. Recover the database by using UNTIL TIME, UNTIL SEQUENCE, or UNTIL SCN.**
- 5. Open the database by using RESETLOGS.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Performing Incomplete Recovery by Using RMAN

RMAN can perform recovery of the database to a past time, SCN, or log sequence number.

Incomplete recovery of the database requires you to open the database with the RESETLOGS option. This option gives the online redo logs a new time stamp and SCN, thus eliminating the possibility of corrupting data files by the application of obsolete archived redo logs. You cannot recover some data files to a time before the RESETLOGS and others to a time after the RESETLOGS. You must recover all data files to the same SCN. The only exception is if the data file is offline normal or read-only. You can bring files in read-only or offline normal tablespaces online after the RESETLOGS because there are no transactional changes for these files stored in the redo logs.

**Note:** You can use RMAN to restore the data files only if the backups were taken or registered with RMAN.

## Time-Based Recovery Using RMAN: Example

```
RMAN> RUN {
 2> SET UNTIL TIME = '2005-11-28:11:44:00';
 3> RESTORE DATABASE;
 4> RECOVER DATABASE;
 5> ALTER DATABASE OPEN RESETLOGS; }
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Time-Based Recovery Using RMAN: Example

At 12:00 p.m. on Tuesday, November 28, 2005, you determine that the OE.ORDERS table was dropped in error. The approximate time of failure is known and the database structure has not changed since 11:44 a.m. You can use the UNTIL TIME method:

1. If the target database is open, perform a clean shutdown.
2. Mount the target database. Do not back up the database during the recovery.
3. Ensure that NLS\_LANG and NLS\_DATE\_FORMAT environment variables are set appropriately:

```
$ NLS_LANG=american_america.we8iso8859p15
$ NLS_DATE_FORMAT='YYYY-MM-DD:HH24:MI:SS'
```

4. Start Recovery Manager and connect to the target database.  

```
$ rman target rman/rman@ORCL
```
5. You can allocate multiple channels to improve the performance:  

```
RMAN> run {allocate channel c1 type DISK;
2> allocate channel c2 type DISK;
```

## Time-Based Recovery Using RMAN: Example (continued)

6. Specify the time for recovery and restore all data files from a backup with RMAN commands. RMAN chooses the correct files based on the SET UNTIL command:

```
RMAN> ... set until time = '2005-11-28:11:44:00';
RMAN> ... restore database;
```

**Note:** If you need to restore archived redo log files to a new location, use the RMAN SET ARCHIVELOG DESTINATION TO <location> command.

7. Recover the database to the time specified in the SET UNTIL command:

```
RMAN> ... recover database;
```

8. Open the database by using the RESETLOGS option:

```
RMAN> ... alter database open resetlogs; }
```

9. Check whether the table exists.

10. Notify users that the database is available for use, and that they should reenter any data that was not committed before the system failure occurred.

## Log Sequence Recovery Using RMAN: Example

```
RMAN> RUN {
 2> SET UNTIL SEQUENCE 120 THREAD 1;
 3> ALTER DATABASE MOUNT;
 4> RESTORE DATABASE;
 5> RECOVER DATABASE; # recovers through log 119
 6> ALTER DATABASE OPEN RESETLOGS;
 7> }
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Log Sequence Recovery Using RMAN: Example

The UNTIL SEQUENCE clause specifies a redo log sequence number and thread as an upper limit. RMAN selects only files that can be used to recover up to but not including the specified log sequence number. The example in the slide assumes that log sequence 120 was lost due to a disk crash and the database needs to be recovered using the available archived redo logs.

# Incomplete Recovery Using Enterprise Manager

Database Instance: orcl.oracle.com > Perform Recovery

## Perform Recovery

### Whole Database Recovery

- Recover to the current time or a previous point-in-time Perform Whole Database Recovery  
Datafiles will be restored from the latest usable backup as required.
- Restore all datafiles  
Specify Time, SCN or log sequence. The backup taken at or prior to that time will be used. No recovery will be performed in this operation.
- Recover from previously restored datafiles

### Object Level Recovery

Object Type

Operation Type  Recover to current time

Datafile will be restored as required.

Restore datafiles

Specify Time, SCN or log sequence. The backup taken at or prior to that time will be used. No recovery will be performed in this operation.

Recover from previously restored datafiles

Block Recovery

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Incomplete Recovery Using Enterprise Manager

To recover the entire database to a point in time, on the Perform Recovery page, select the “Recover to the current time or a previous point-in-time” option. Then, click Perform Whole Database Recovery.

Choose “Restore all datafiles” to restore all the data files in the database to a previous backup state. You can also recover from previously restored data files. This uses redo data to roll the database forward to some point.

## Incomplete Recovery and the Alert Log

The following are some best practices regarding the alert log in incomplete recovery scenarios:

- Check the alert log before and after recovery.
- Look for error information, hints, and SCNs.
- Confirm that steps in the recovery process were successful.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Incomplete Recovery and the Alert Log

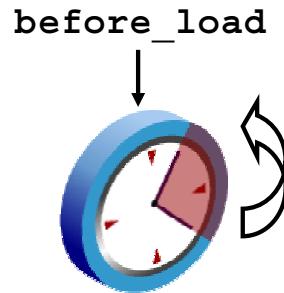
During recovery, progress information is stored in the alert log. This file should always be checked before and after recovery. The following is a sample entry in the alert.log file:

```
$ vi $ORACLE_BASE/admin/orcl/bdump/alert_orcl.log
...
ALTER DATABASE RECOVER database until cancel
Media Recovery Start
ORA-279 signalled during: ALTER DATABASE RECOVER database
until cancel ...
Fri Aug 26 15:22:46 2005
ALTER DATABASE RECOVER CONTINUE DEFAULT
...
Fri Aug 26 15:28:27 2005
ALTER DATABASE RECOVER CANCEL
Fri Aug 26 15:28:27 2005
Media Recovery Canceled
Completed: ALTER DATABASE RECOVER CANCEL
```

# Restore Points

## A restore point:

- **Serves as an alias for an SCN or a specific point in time**
- **Is stored in the control file**
- **Can be used with:**
  - RECOVER DATABASE
  - FLASHBACK DATABASE
  - FLASHBACK TABLE



1

```
SQL> CREATE RESTORE POINT before_load;
```

2

```
RMAN> RECOVER DATABASE UNTIL RESTORE POINT before_load;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Restore Points

Creating a normal restore point assigns the restore point name to a specific point in time or SCN. This name functions as a kind of bookmark or alias that you can use with commands that recognize a RESTORE POINT clause as shorthand for specifying a point in time.

Before performing any operation that you may have to reverse, you can create a normal restore point. The name of the restore point and the SCN are recorded in the control file. Then, if you later need to refer to that point in time in a RECOVER DATABASE command, you can do so without figuring out a time stamp or SCN value. Using Flashback Database, Flashback Table, or point-in-time recovery, you can refer to the target time using the name of the restore point instead of a time expression or SCN. Defining a normal restore point before an operation to be reversed later eliminates the need to manually record an SCN in advance, or to investigate the correct SCN after the fact by using features such as Flashback Query.

Normal restore points are very lightweight. The control file can maintain a record of thousands of normal restore points with no significant impact upon database performance. Normal restore points eventually age out of the control file if not manually deleted, so they require no ongoing maintenance.

**Note:** The use of restore points with FLASHBACK commands is covered in the lesson titled “Flashback.”

## Incomplete Recovery: Best Practices

- **Plan for and practice scenarios ahead of time.**
- **Investigate and verify that incomplete recovery is necessary.**
- **Follow all steps carefully.**
- **Take whole database backups before and after recovery.**
- **Always verify that the recovery was successful.**
- **Take advantage of restore points.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Incomplete Recovery: Best Practices

- It is important to follow all recovery steps carefully because most incomplete recovery problems are caused by a DBA error during the recovery process.
- During database recovery, transaction activity can be only rolled forward to the desired time, not back to the desired time. This is the reason why all data files must be restored for the database to be taken back in time. Failure to restore all data files prevents the (unsynchronized) database from opening.
- Before starting incomplete recovery, perform a whole closed database backup (including control files and redo logs). This is helpful in the following ways:
  - It allows you to recover from error. If your recovery fails (for example, you recover past the desired point of recovery), redo logs and control files cannot be used for the next recovery unless there is a backup of these files.
  - It saves time if the recovery fails. In this situation, you can restore the data files from the new backup, rather than from a previous backup, which needs archives applied.

**Note:** If a whole backup is not performed, at least archive the current redo log:

```
SQL> ALTER SYSTEM ARCHIVE LOG CURRENT
```

and back up the control file:

```
SQL> ALTER DATABASE BACKUP CONTROLFILE TO
'/u01/data/backup.ctl';
```

## Incomplete Recovery: Best Practices (continued)

- Perform a whole closed backup after a successful recovery. This can save many hours if another recovery is required before completion of the next scheduled backup. If you are using Oracle Database 10g, this step is now optional.
- Always verify that the problem has been corrected before allowing users to access the system, in case the recovery failed and needs to be performed again.

Consider the following example:

- A database at log sequence 14 has archived logs from sequence 2 (`arch_2.rdo`) to sequence 13 (`arch_13.rdo`).
- After performing incomplete recovery, a new database incarnation is created, setting the database log sequence to 1.
- Archived logs `arch_2.rdo` to `arch_13.rdo` are now part of the old database incarnation.
- After a few log switches, the archived log `arch_2.rdo` is overwritten and is backed up with all other archives (including the old archived logs `arch_3.rdo` to `arch_13.rdo`).
- At a later stage, if recovery requires `arch_6.rdo`, you need to make sure that the archived log restored from the backup is for the correct database incarnation; otherwise, an error will result.

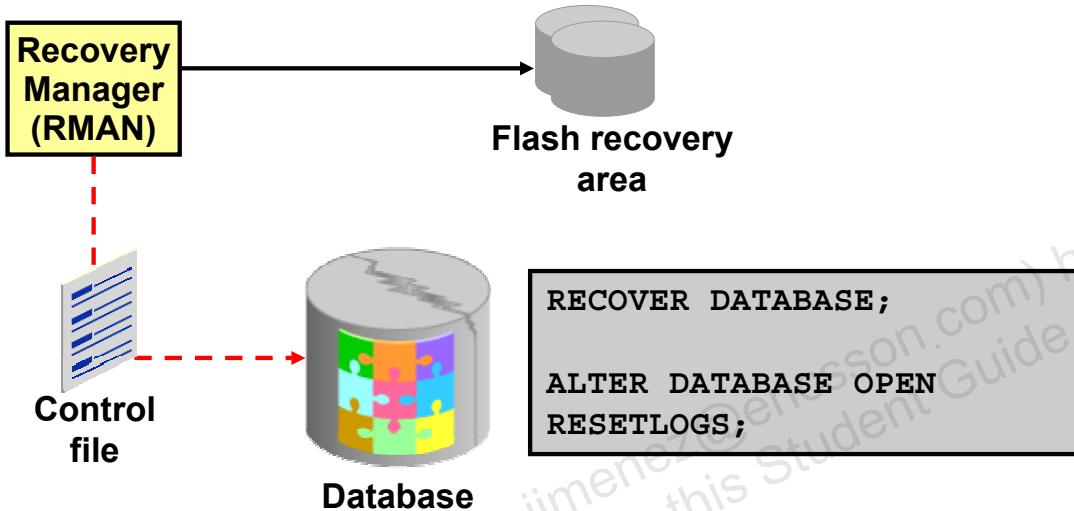
To prevent confusion, you can use the `%r` format option of the `log_archive_format` database initialization parameter to:

- Automatically incorporate the database resetlogs ID into the archived log file names
- Ensure that unique names are constructed for the archived log files across multiple incarnations of the database

# Recovering a Control File Autobackup

```
RMAN> RESTORE CONTROLFILE TO
2> '/oradata/ctlfile.bak' FROM AUTOBACKUP;
```

User man.  
RMAN CLI  
EM  
Complete  
Incomplete  
> **Control file**  
RO TBS



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Recovering a Control File Autobackup

If you are not using a recovery catalog, you should have autobackup of the control file configured, so that you are able to quickly restore the control file if needed. The commands used for restoring your control file are the same, whether or not you are using a flash recovery area. However, if you are using a flash recovery area, RMAN implicitly cross-checks backups and image copies listed in the control file, and catalogs any files in the flash recovery area not recorded in the restored control file, improving the usefulness of the restored control file in the restoration of the rest of your database.

**Note:** Tape backups are not automatically cross-checked after the restoration of a control file. If you are using tape backups, then after restoring the control file and mounting the database, you must cross-check the backups on tape.

## Recovering a Control File Autobackup (continued)

To restore the control file from an autobackup, the database must be in a NOMOUNT state. Then, you use the RESTORE CONTROLFILE FROM AUTOBACKUP command:

```
RMAN> SHUTDOWN IMMEDIATE;
RMAN> STARTUP NOMOUNT;
RMAN> set dbid = 123456789;
RMAN> RESTORE CONTROLFILE FROM AUTOBACKUP;
```

RMAN searches for a control file autobackup. If one is found, RMAN restores the control file from that backup to all the control file locations listed in the CONTROL\_FILES initialization parameter.

If you have a recovery catalog, you do not have to set the DBID or use the control file autobackup to restore the control file. You can use the RESTORE CONTROLFILE command with no arguments:

```
RMAN> RESTORE CONTROLFILE;
```

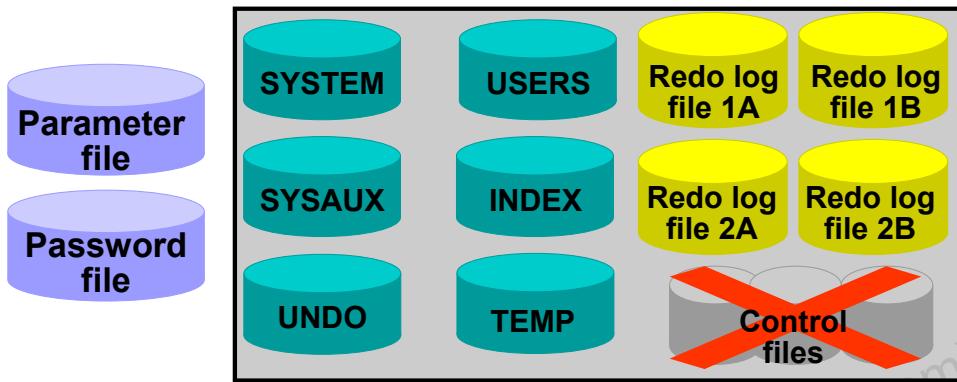
The instance must be in the NOMOUNT state when you perform this operation, and RMAN must be connected to the recovery catalog. The restored control file is written to all locations listed in the CONTROL\_FILES initialization parameter.

If you have also lost the SPFILE for the database and need to restore it from the autobackup, the procedure is similar to restoring the control file from autobackup. You must first set the DBID for your database, and then use the RESTORE SPFILE FROM AUTOBACKUP command.

After you have started the instance with the restored server parameter file, RMAN can restore the control file from the autobackup. After you restore and mount the control file, you have the backup information necessary to restore and recover the database.

After restoring the control files of your database from backup, you must perform complete media recovery and then open your database with the RESETLOGS option.

## Creating a New Control File



```
SQL> ALTER DATABASE BACKUP CONTROLFILE
TO TRACE;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Creating a New Control File

If you have autobackup of the control file configured, you should rarely, if ever, need to re-create your control file. But, if needed, you can generate and save a script that does the re-creation.

The `ALTER DATABASE BACKUP CONTROLFILE TO TRACE` command generates a user trace file that contains the SQL command to re-create the control file. Copy the trace file to a script file, such as `new_control.sql`; delete the trace header information before the words `STARTUP NOMOUNT`; and make any other desired changes, such as increasing `MAXDATAFILES`, `MAXLOGFILES`, and so on. Run the script to create a new control file.

You must use this command while the database is mounted or open and while you are connected as a user with DBA privileges.

# Creating a New Control File

Database Instance: orcl.oracle.com > Control Files      Logged in As SYS

## Control Files

**General** [Advanced](#) [Record Section](#)

[Backup To Trace](#)

### Control File Mirror Images

Oracle strongly recommends that your database has a minimum of two control files and that they are located on separate disks. If a control file is damaged due to a disk failure, it could be restored using the intact copy of the control file from the other disk. You can specify their location in the database's initialization parameter file.

| Valid | File Name     | File Directory                |
|-------|---------------|-------------------------------|
| VALID | control01.ctl | /u01/app/oracle/oradata/orcl/ |
| VALID | control02.ctl | /u01/app/oracle/oradata/orcl/ |
| VALID | control03.ctl | /u01/app/oracle/oradata/orcl/ |

**General** [Advanced](#) [Record Section](#)

ORACLE

Copyright © 2008, Oracle. All rights reserved.

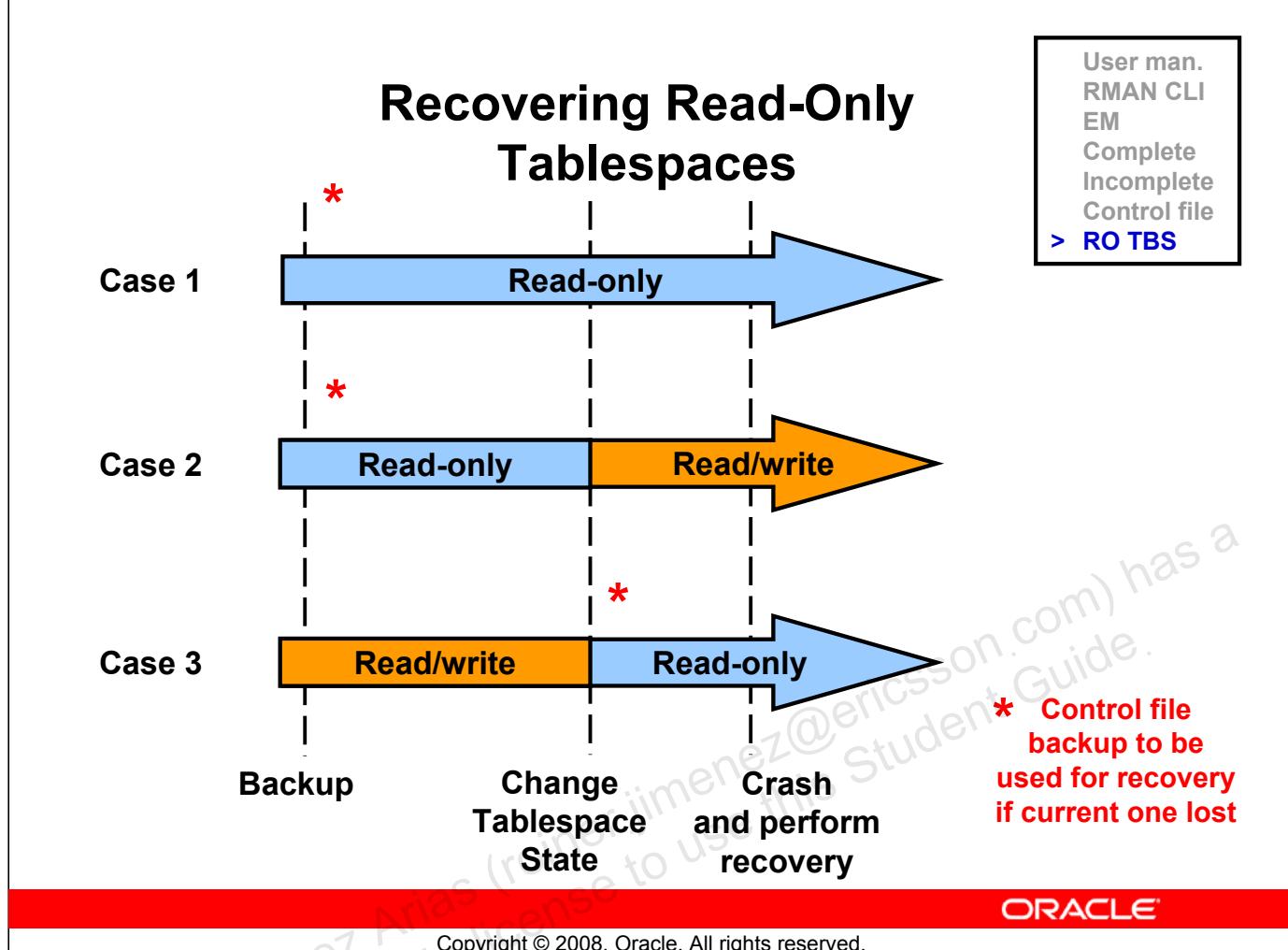
## Creating a New Control File (continued)

The Database Control Console allows you to manage the control files used by your database. On the Administration page, select Control Files in the Storage section.

You can use the Control Files General page to view the control file mirror images and their locations. Click Backup To Trace in order to create a trace file for the control file.

You can also write your own CREATE CONTROLFILE command, but you need to supply the full path names and sizes for:

- The redo log files
- All the data files associated with the database, including the data files for SYSTEM and SYSAUX



### Recovering Read-Only Tablespaces

Making a tablespace read-only prevents write operations on the data files in the tablespace, regardless of a user's update privilege level. The primary purpose of read-only tablespaces is to eliminate the need to perform backup and recovery of large, static portions of a database. Read-only tablespaces also provide a way to protecting historical data so that users cannot modify it. Because read-only tablespaces can never be updated, they can reside on CD-ROM or WORM (write once, read many) devices.

The method of recovering a read-only tablespace depends on the backups that are available and whether the tablespace was altered to read/write or read-only within the recovery period.

**Case 1:** The tablespace being recovered is read-only, and was read-only when the last backup occurred. In this case, you can simply restore the tablespace from the backup. There is no need to apply any redo information.

**Case 2:** The tablespace being recovered is read/write, but was read-only when the last backup occurred. In this case, you need to restore the tablespace from the backup and apply the redo information from the point when the tablespace was made read/write.

**Case 3:** The tablespace being recovered is read-only, but was read/write when the last backup occurred. You should always back up a tablespace after making it read-only to avoid this situation. However, if this does occur, you must restore the tablespace from the backup and recover up to the time that the tablespace was made read-only.

## **Recovering Read-Only Tablespaces (continued)**

In all three cases, if the current control file is not available, the asterisk denotes which control file backup should be used for recovery. This is necessary because when a backup control file is used, the recovery process will require you to perform an OPEN RESETLOGS. This updates the datafile headers, and datafiles cannot be written to if they are read-only.

# Read-Only Tablespace Recovery Issues

**Special considerations must be taken for read-only tablespaces when:**

- **Re-creating a control file**
- **Renaming data files**
- **Using a backup control file**



Copyright © 2008, Oracle. All rights reserved.

## Read-Only Tablespace Recovery Issues

### Re-creating a Control File

If you need to re-create a control file with the CREATE CONTROL FILE command and your database has read-only tablespaces, you must follow special procedures. The steps are listed in the trace file that is generated by the ALTER DATABASE BACKUP CONTROLFILE TO TRACE command. Because the tablespace is read-only, there are no changes being made to it, and thus it is assumed that no recovery is needed. The tablespace data files are not included in the control file; therefore, as the database is started up with the new control file, the control file contents are cross-checked against the files in the data dictionary. Any files found in the data dictionary that are not in the control file are added to the control file with a name like MISSINGnnnnn. After the database is open, you must rename the files in the control file by using the ALTER DATABASE RENAME FILE command.

```
ALTER DATABASE RENAME FILE 'MISSING00005'
 TO '/u01/app/oracle/oradata/orcl/example01.dbf';
ALTER TABLESPACE "EXAMPLE" ONLINE;
```

Oracle Corporation recommends enabling control file autobackup in RMAN so that you do not have to depend on this workaround that is required for a control file that has been backed up using the BACKUP CONTROLFILE TO TRACE command.

## **Read-Only Tablespace Recovery Issues (continued)**

### **Changing Data File Location**

If you cannot restore a copy of the data files in a read-only tablespace to the correct destination, you can use the ALTER DATABASE RENAME command to place the files in a new location.

### **Backup Control File**

If you have a read-only tablespace on read-only media, then you may encounter errors or poor performance when recovering with the USING BACKUP CONTROLFILE option. This situation occurs when the backup control file indicates that a tablespace was read/write when the control file was backed up. In this case, media recovery may attempt to write to the files. For read-only media, the database issues an error saying that it cannot write to the files.

Following are alternatives that you can use to recover read-only media when using a backup control file:

- Take data files from read-only tablespaces offline before performing recovery with a backup control file, and then bring the files online at the end of media recovery.
- Use the correct version of the control file for the recovery. If the tablespace will be read-only when recovery completes, then the control file backup must be from a time when the tablespace was read-only. Similarly, if the tablespace will be read/write at the end of recovery, then the control file must be from a time when the tablespace was read/write.

### **Recovering the Database**

If a datafile is read-only at the point in time to which the database is being recovered, then RMAN does not recover it. To change this behavior and have RMAN verify the datafile headers are current for the point in time, specify the CHECK READONLY option with the RECOVER DATABASE command.

# Summary

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

- **Perform complete or incomplete user-managed recovery**
- **Perform complete or incomplete recovery by using RMAN**
- **Identify situations where incomplete recovery is necessary**
- **Perform incomplete recovery based on time, SCN, log sequence, restore points, or the cancel method**
- **Recover a control file autobackup**
- **Use Enterprise Manager to perform recovery**
- **Recover read-only tablespaces**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Practice Overview: Performing Incomplete Recovery

**This practice covers the following topics:**

- **Point-in-time recovery using RMAN**
- **Recovery from the loss of control files**



Copyright © 2008, Oracle. All rights reserved.

# 0 Flashback

ORACLE®

Copyright © 2008, Oracle. All rights reserved.

# Objectives

Recycle bin  
Flashback DB  
Config. FB DB  
Monitor FB DB  
Guar. Res. Pt.

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

- **Query the recycle bin**
- **Configure Flashback Database**
- **Perform Flashback Database to a point in time**
- **Monitor flashback log statistics**
- **Enable and disable the Flashback Database feature**
- **Use the Enterprise Manager Recovery Wizard to perform Flashback Database**
- **Use guaranteed restore points with Flashback Database**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

# Flashback Technology: Review

| Object Level | Scenario Examples                                    | Flashback Technology | Uses           | Affects Data |
|--------------|------------------------------------------------------|----------------------|----------------|--------------|
| Database     | Truncate table;<br>Undesired multitable changes made | Database             | Flashback logs | TRUE         |
| Table        | Drop table                                           | Drop                 | Recycle bin    | TRUE         |
|              | Update with the wrong WHERE clause                   | Table                | Undo data      | TRUE         |
|              | Compare current data with data from the past         | Query                | Undo data      | FALSE        |
|              | Compare versions of a row                            | Version              | Undo data      | FALSE        |
| Tx           | Investigate several historical states of data        | Transaction          | Undo data      | FALSE        |

Copyright © 2008, Oracle. All rights reserved.

## Flashback Technology: Review

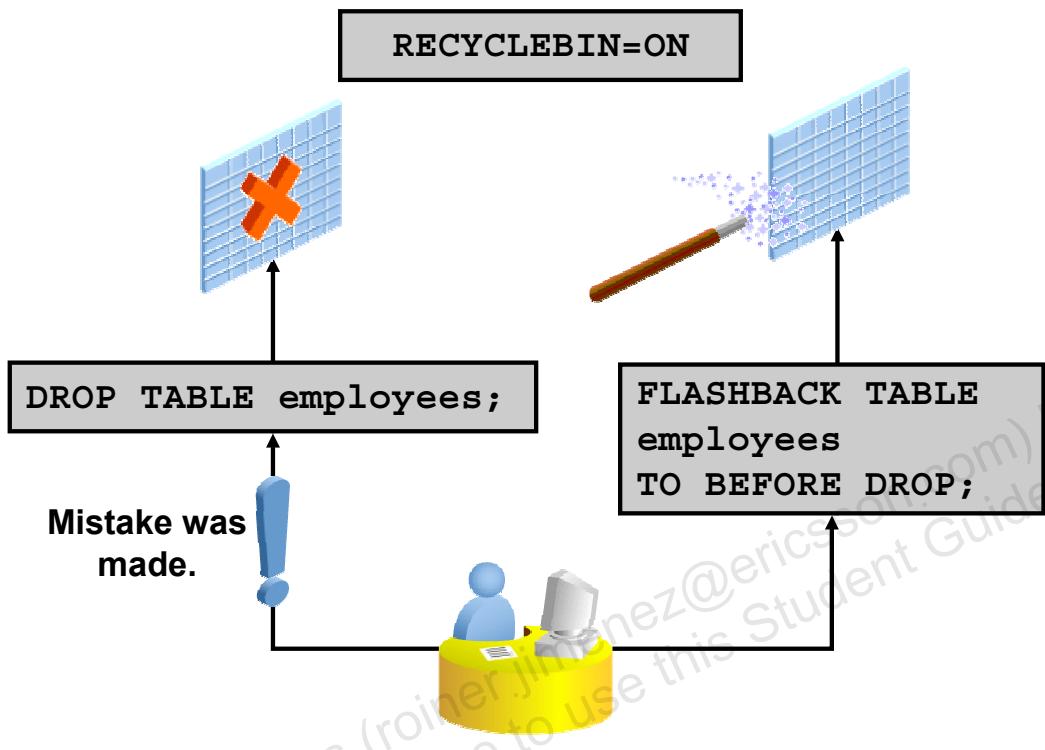
Flashback technology must be used when a logical corruption occurs in the Oracle database, and you need to recover data quickly and easily. As with human errors, it is difficult to identify the objects and rows that are affected by an erroneous transaction. With flashback technology, you can diagnose how errors are introduced into the database, and then you can repair the damage. You can view the transactions that have contributed to specific row modifications, view the entire set of versions of a given row during some time period, or just view data as it appeared at a specific time in the past. The table in the slide shows typical uses of flashback technology.

Flashback Database uses the flashback logs to perform flashback. Flashback Drop uses the recycle bin. All other techniques use undo data.

Not all flashback features modify the database. Some are simply methods to query other versions of data. Those are tools for you to use to investigate a problem and aid in recovery. The results of those flashback queries can help you do one of these two things:

- Determine which type of database-modifying flashback operation to perform to fix the problem.
- Feed the result set of these queries into an INSERT, UPDATE, or DELETE statement that enables you to easily repair the erroneous data.

## Flashback Drop and the Recycle Bin



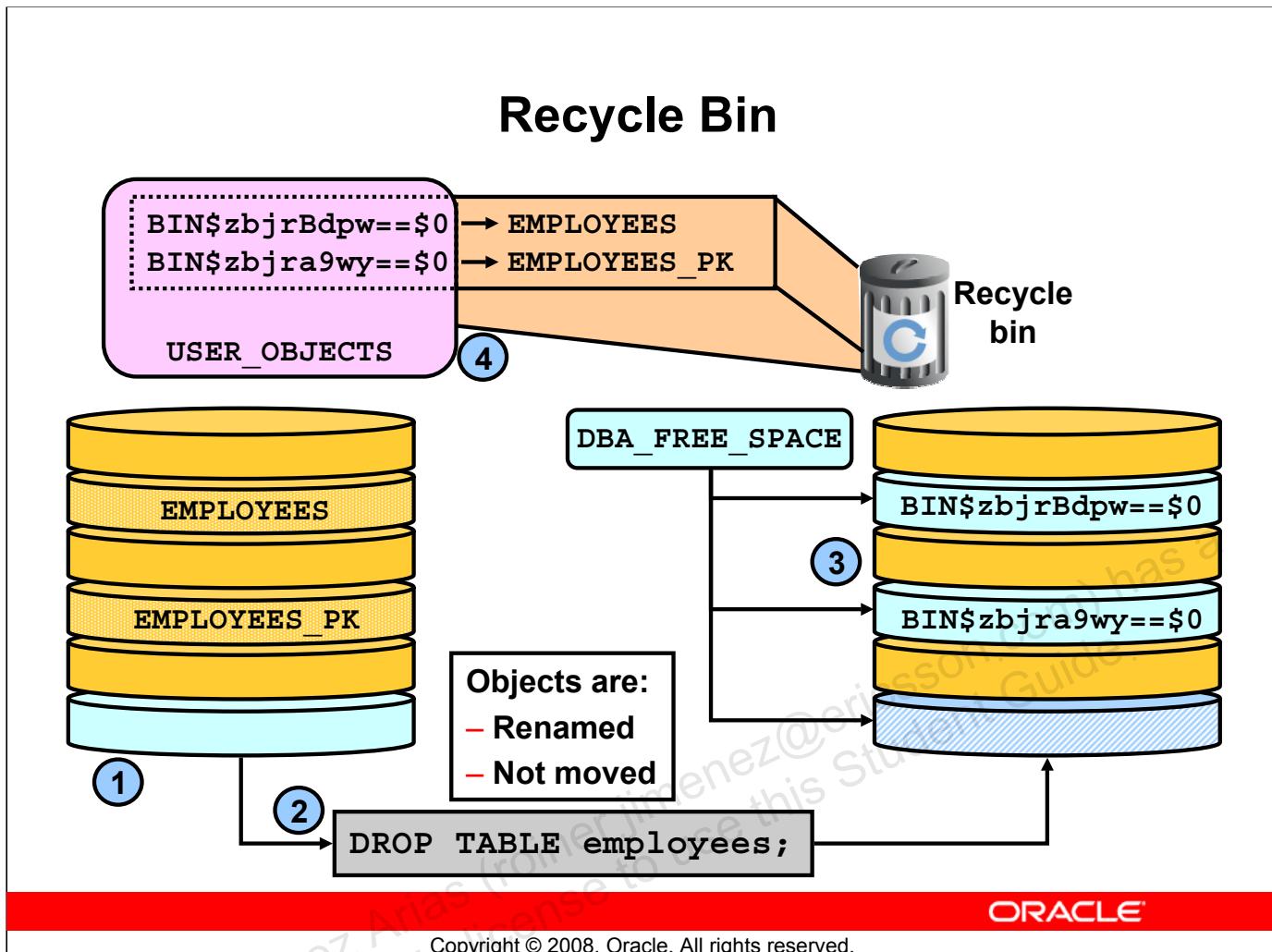
Copyright © 2008, Oracle. All rights reserved.

### Flashback Drop and the Recycle Bin

In previous releases of the Oracle database, if you dropped a table by mistake, you had to recover the database to a prior time to recover the dropped table. This procedure was often time consuming and resulted in loss of work of other transactions.

Oracle Database 10g introduces the Flashback Drop feature, which you can use to undo the effects of a **DROP TABLE** statement without having to use point-in-time recovery.

**Note:** The **RECYCLEBIN** initialization parameter is used to control whether the Flashback Drop capability is turned ON or OFF. If the parameter is set to OFF, then dropped tables do not go into the recycle bin. If this parameter is set to ON, the dropped tables go into the recycle bin and can be recovered. By default, **RECYCLEBIN** is set to ON.



### Recycle Bin

Without the recycle bin enabled, when you drop a table, the space associated with the table and its dependent objects is immediately reclaimable (that is, it can be used for other objects).

If the recycle bin is enabled, when you drop a table, then the space associated with the table and its dependent objects is not immediately reclaimable, even though it does appear in DBA\_FREE\_SPACE. Instead, the dropped objects are temporarily placed in the recycle bin and still belong to their owner. The space used by recycle bin objects is never automatically reclaimed unless there is space pressure. This enables you to recover recycle bin objects for the maximum possible duration.

When a dropped table is moved to the recycle bin, the table and its associated objects and constraints are renamed using system-generated names. This is necessary to avoid name conflicts that may arise if you later create a new object with the same name.

The recycle bin itself is a data dictionary table that maintains the relationships between the original names of dropped objects and their system-generated names. You can query the content of the recycle bin by using the DBA\_RECYCLEBIN view.

## Recycle Bin (continued)

The diagram in the slide illustrates this new behavior:

1. You have created a table called EMPLOYEES in your tablespace.
2. You drop the EMPLOYEES table.
3. The extents occupied by EMPLOYEES are now considered as free space.
4. EMPLOYEES is renamed and inserted into the recycle bin.

# Restoring Tables from the Recycle Bin

- **Restore dropped tables and dependent objects.**
- **If multiple recycle bin entries have the same original name:**
  - Use unique, system-generated names to restore a particular version
  - When using original names, the restored table is last in, first out (LIFO)
- **Rename the original name if that name is currently used.**

```
FLASHBACK TABLE <table_name>
TO BEFORE DROP [RENAME TO <new_name>];
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Restoring Tables from the Recycle Bin

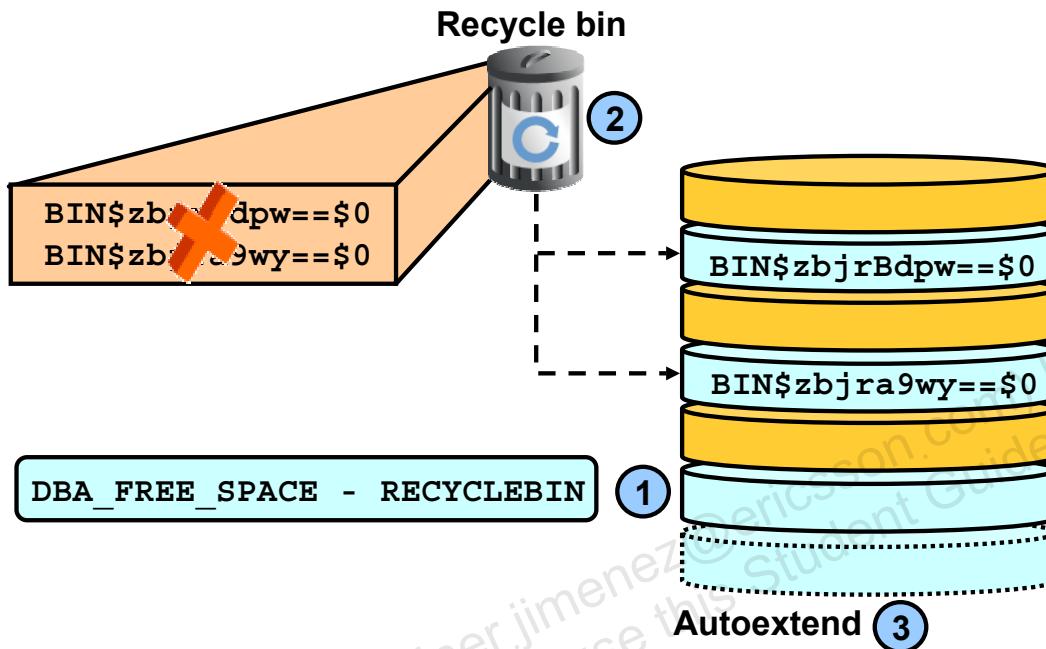
Use the FLASHBACK TABLE . . . TO BEFORE DROP command to recover a table and all of its possible dependent objects from the recycle bin. You can specify either the original name of the table or the system-generated name assigned to the object when it was dropped.

If you specify the original name, and if the recycle bin contains more than one object of that name, then the object that was moved to the recycle bin most recently is recovered first (LIFO: last in, first out). If you want to retrieve an older version of the table, you can specify the system-generated name of the table that you want to retrieve, or issue additional FLASHBACK TABLE . . . TO BEFORE DROP statements until you retrieve the table you want.

If a new table of the same name has been created in the same schema since the original table was dropped, then an error is returned unless you also specify the RENAME TO clause.

**Note:** When you flash back a dropped table, the recovered indexes, triggers, and constraints keep their recycle bin names. Therefore, it is advisable to query the recycle bin and DBA\_CONSTRAINTS before flashing back a dropped table. In this way, you can rename the recovered indexes, triggers, and constraints to more usable names.

# Recycle Bin: Automatic Space Reclamation



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Recycle Bin: Automatic Space Reclamation

As long as the space used by recycle bin objects is not reclaimed, you can recover those objects by using Flashback Drop. The following are the recycle bin object reclamation policies:

- Manual cleanup when you explicitly issue a PURGE command
- Automatic cleanup under space pressure: While objects are in the recycle bin, their corresponding space is also reported in DBA\_FREE\_SPACE because their space is automatically reclaimable. The free space in a particular tablespace is then consumed in the following order:
  1. Free space not corresponding to recycle bin objects
  2. Free space corresponding to recycle bin objects. In this case, recycle bin objects are automatically purged from the recycle bin using a first in, first out (FIFO) algorithm.
  3. Free space automatically allocated if the tablespace is autoextensible

## Recycle Bin: Automatic Space Reclamation (continued)

Suppose you create a new table inside the TBS1 tablespace. If there is free space allocated to this tablespace that does not correspond to a recycle bin object, then this free space is used as a first step. If this is not enough, free space is used that corresponds to recycle bin objects that reside inside TBS1. If the free space of some recycle bin objects is used, then these objects get purged automatically from the recycle bin. At this time, you can no longer recover those objects by using the Flashback Drop feature. As a last resort, if space requirement is not yet satisfied, the TBS1 tablespace is extended if possible.

# Recycle Bin: Manual Space Reclamation

```
PURGE { TABLE <table_name> | INDEX <index_name> }
```

```
PURGE TABLESPACE <ts_name> [USER <user_name>]
```

```
PURGE [USER_ | DBA_] RECYCLEBIN
```

Database Instance: orcl.oracle.com > Tables > Recycle Bin

## Recycle Bin

### Results

| Select                              | Object Name   | Schema | Object Type | Tablespace | Drop Time           | Create Time         | Size | Operation                                   |
|-------------------------------------|---------------|--------|-------------|------------|---------------------|---------------------|------|---------------------------------------------|
| <input type="checkbox"/>            | ▼ Recycle Bin |        |             |            |                     |                     |      | <input type="button" value="View Content"/> |
| <input type="checkbox"/>            | ► EMPLOYEES   | HR2    | TABLE       | USERS      | 2005-08-31:17:44:17 | 2005-08-31:17:40:45 | 8    | <input type="button" value="View Content"/> |
| <input checked="" type="checkbox"/> | JOB\$         | HR2    | TABLE       | USERS      | 2005-08-31:17:44:30 | 2005-08-31:17:41:29 | 8    | <input type="button" value="View Content"/> |

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Recycle Bin: Manual Space Reclamation

Use the PURGE command to permanently remove objects from the recycle bin. When an object is purged from the recycle bin, the object and its dependent objects are permanently removed from the database. As a consequence, objects purged from the recycle bin are no longer recoverable by using the Flashback Drop feature. The following are the possible uses of PURGE:

- PURGE TABLE purges the specified table.
- PURGE INDEX purges the specified index.
- PURGE TABLESPACE purges all the objects residing in the specified tablespace. In addition, objects residing in other tablespaces may get purged if they are dependent. Optionally, you can also specify the USER clause to purge only those objects that belong to the specified user running low on disk quota for the specified tablespace.
- PURGE RECYCLEBIN purges all the objects that belong to the current user. RECYCLEBIN and USER\_RECYCLEBIN are synonymous.
- PURGE DBA\_RECYCLEBIN purges all the objects. You must have enough system privileges or the SYSDBA system privilege to issue this command.

**Note:** For PURGE TABLE and PURGE INDEX commands, if you specify an original name and if the recycle bin contains more than one object of that name, then the object that has been in the recycle bin the longest is purged first (FIFO).

## Bypassing the Recycle Bin

```
DROP TABLE <table_name> [PURGE] ;
```

```
DROP TABLESPACE <ts_name>
[INCLUDING CONTENTS] ;
```

```
DROP USER <user_name> [CASCADE] ;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Bypassing the Recycle Bin

You can use the `DROP TABLE PURGE` command to permanently drop a table and its dependent objects from the database. When you use this command, the corresponding objects are not moved to the recycle bin. This command provides the same functionality that the `DROP TABLE` command provided in previous releases.

When you issue the `DROP TABLESPACE . . . INCLUDING CONTENTS` command, the objects in the tablespace are not placed in the recycle bin. Moreover, objects in the recycle bin belonging to the tablespace are purged. When you issue the same command without the `INCLUDING CONTENTS` clause, the tablespace must be empty for the command to succeed. However, there can be objects belonging to the tablespace in the recycle bin. In this case, these objects are purged.

When you issue the `DROP USER . . . CASCADE` command, the user and all the objects owned by the user are permanently dropped from the database. Any objects in the recycle bin belonging to the dropped user are purged.

## Querying the Recycle Bin

```
SELECT owner, original_name, object_name,
 type, ts_name, droptime, related, space
 FROM dba_recyclebin
 WHERE can_undrop = 'YES';
```

```
SELECT original_name, object_name,
 type, ts_name, droptime, related, space
 FROM user_recyclebin
 WHERE can_undrop = 'YES';
```

```
SQL> SHOW RECYCLEBIN
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Querying the Recycle Bin

You can view all the objects that you have dropped by querying `user_recyclebin` or `RECYCLEBIN`.

`dba_recyclebin` shows you all the objects that have been dropped by all users and that are still in the recycle bin.

You can also use the SQL\*Plus `SHOW RECYCLEBIN` command. This command shows you only those objects that can be “undropped.”

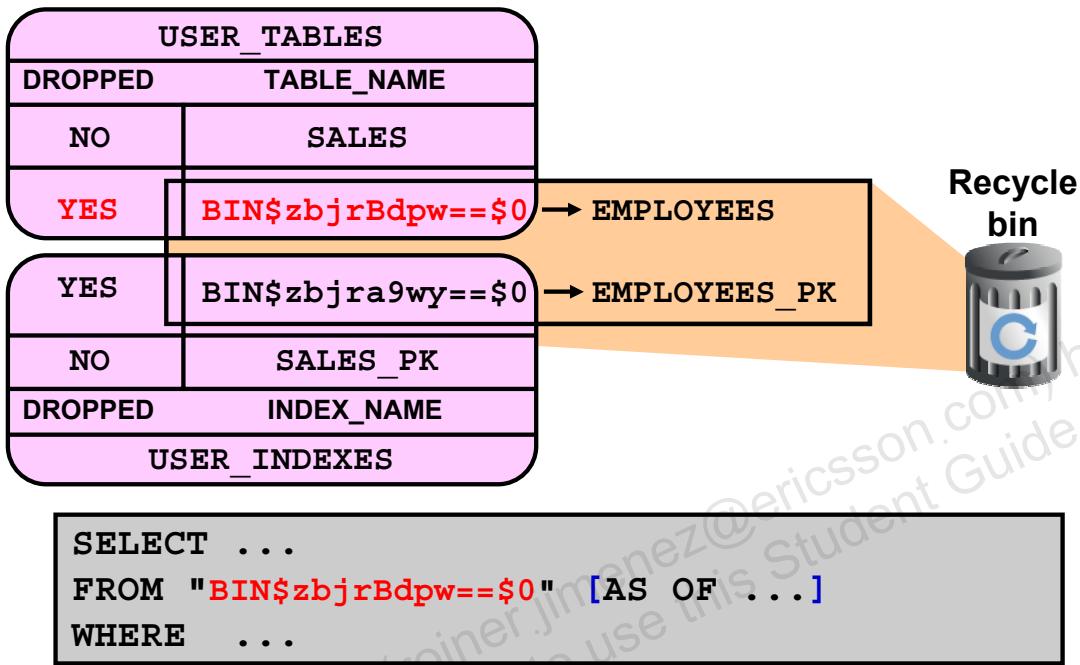
The examples show how to extract important information from the recycle bin:

- `original_name` is the name of the object before it is dropped.
- `object_name` is the system-generated name of the object after it is dropped.
- `type` is the object’s type.
- `ts_name` is the name of the tablespace to which the object belongs.
- `droptime` is the date at which the object was dropped.
- `related` is the object identifier of the dropped object.
- `space` is the number of blocks currently used by the object.

You can also see the content of the recycle bin by using Database Control.

**Note:** For detailed information about the `DBA_RECYCLEBIN` view, see the *Oracle Database Reference* guide.

# Querying Data from Dropped Tables



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Querying Data from Dropped Tables

When you drop a table, the table is moved to the recycle bin, and its original name is changed to a unique, system-generated name. Because you still own the dropped table, you can still see its characteristics from various dictionary views such as DBA\_TABLES, DBA\_OBJECTS, DBA\_SEGMENTS, and so on. To make a distinction between tables that are in the recycle bin and tables that are not, the DBA\_TABLES view has a new column called DROPPED that is set to YES for tables that were dropped but are still in the recycle bin.

So, as long as a system-generated table name is in the recycle bin, you can use it in a SELECT statement and also in flashback queries.

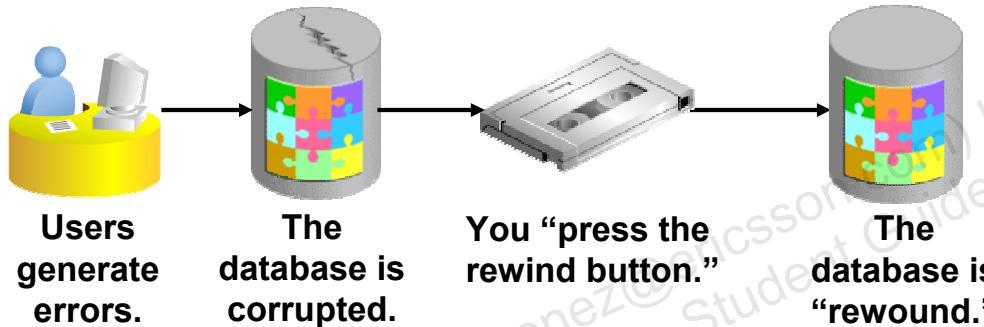
However, you cannot issue any sort of DML or DDL statements on objects that reside in the recycle bin.

# Flashback Database: Review

Recycle bin  
> Flashback DB  
Config. FB DB  
Monitor FB DB  
Guar. Res. Pt.

## The Flashback Database operation:

- Works like a rewind button for the database
- Can be used in cases of logical data corruptions made by users



ORACLE

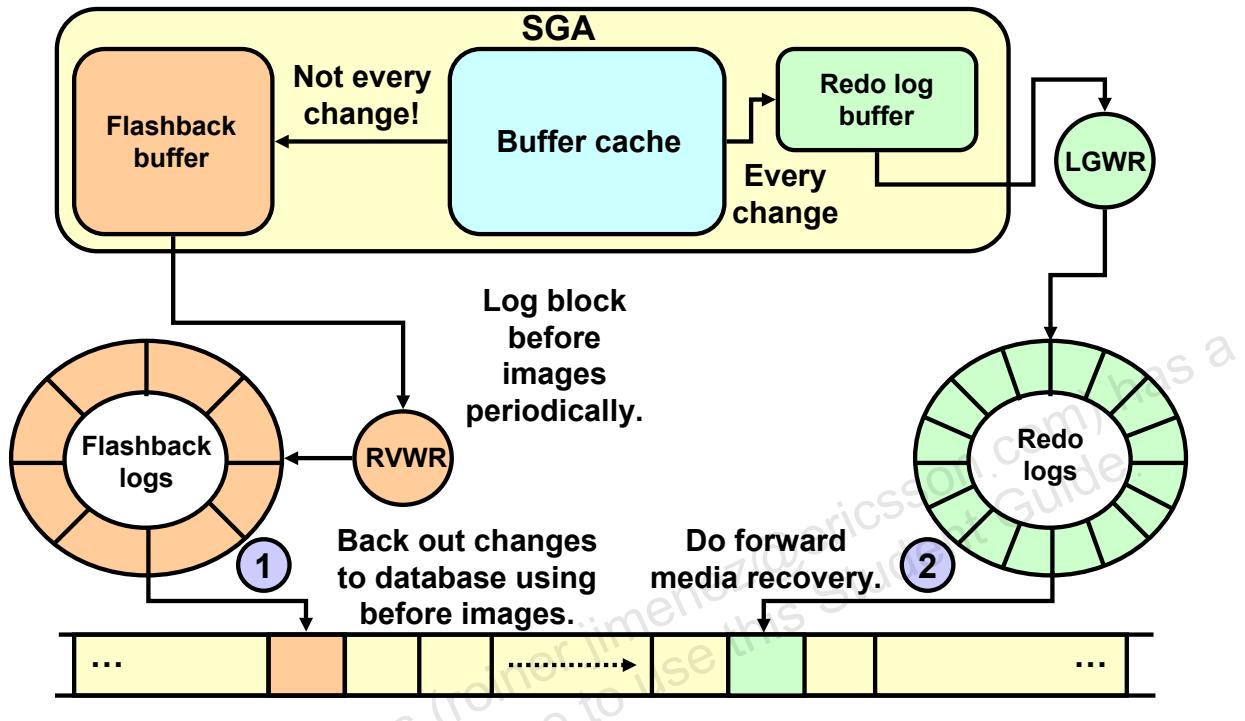
Copyright © 2008, Oracle. All rights reserved.

## Flashback Database: Review

With Flashback Database, you can quickly bring your database to an earlier point in time by undoing all the changes that have taken place since that time. This operation is fast because you do not need to restore backups. You can use this feature to undo changes that have resulted in logical data corruptions.

If you have experienced a loss of media or physical corruption in your database, then you must use traditional recovery methods.

# Flashback Database Architecture



Copyright © 2008, Oracle. All rights reserved.

**ORACLE**

## Flashback Database Architecture

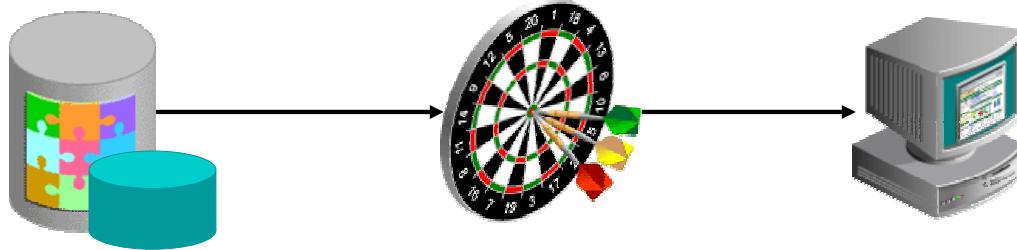
When you enable Flashback Database, the new RVWR background process is started. This background process sequentially writes Flashback Database data from the flashback buffer to the Flashback Database logs, which are circularly reused. Subsequently, when a FLASHBACK DATABASE command is issued, the flashback logs are used to restore to the blocks' before images, and then redo data is used to roll forward to the desired flashback time.

The overhead of enabling Flashback Database depends on the read-write mix of the database workload. Because queries do not need to log any flashback data, the more write-intensive the workload, the higher the overhead of turning on Flashback Database.

**Note:** Flashback Database logs are not archived.

# Configuring Flashback Database

Recycle bin  
Flashback DB  
> Config. FB DB  
Monitor FB DB  
Guar. Res. Pt.



1. Configure the flash recovery area.
2. Set the retention target.
3. Enable Flashback Database.

```
SQL> SHUTDOWN IMMEDIATE;
SQL> STARTUP MOUNT EXCLUSIVE;
SQL> ALTER SYSTEM SET
 2 DB_FLASHBACK_RETENTION_TARGET=2880 SCOPE=BOTH;
SQL> ALTER DATABASE FLASHBACK ON;
SQL> ALTER DATABASE OPEN;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Configuring Flashback Database

You can configure Flashback Database as follows:

1. Configure the flash recovery area.
2. Set the retention target with the DB\_FLASHBACK\_RETENTION\_TARGET initialization parameter. You can specify an upper limit, in minutes, on how far back you want to be able to flashback the database. The example uses 2880 minutes, which is equivalent to two days. This parameter is only a target and does not provide any guarantee. Your flashback time interval depends on how much flashback data has been kept in the flash recovery area.
3. Enable Flashback Database with the following command:

```
ALTER DATABASE FLASHBACK ON;
```

Before you can issue the command to enable Flashback Database, the database must be configured for archiving and started in MOUNT EXCLUSIVE mode.

You can determine whether Flashback Database is enabled with the following query:

```
SELECT flashback_on FROM v$database;
```

You can disable Flashback Database with the ALTER DATABASE FLASHBACK OFF command. As a result, all existing Flashback Database logs are deleted automatically.

**Note:** You can enable Flashback Database only when the database is mounted in exclusive mode, not open.

# Configuring Flashback Database Using EM

**Make sure that the database is in ARCHIVELOG mode.**

## Media Recovery

The database is currently in ARCHIVELOG mode. In ARCHIVELOG mode, hot space for logs. If you change the database to ARCHIVELOG mode, you should cold backups and data may be lost in the event of database corruption.

ARCHIVELOG Mode\*

Log Archive Filename Format\* %t\_%s\_%r.dbf

The naming convention for the archived log files. %s: log sequence number; %t: thread number; %r: redo log member number.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Configuring Flashback Database Using EM

Log in to Enterprise Manager Database Console. On the Maintenance page, select Backup/Recovery Settings, and then choose Recovery Settings. Make sure that your database is in ARCHIVELOG mode. If not, select ARCHIVELOG Mode and then click Continue. You will need to shut down and restart the instance for your changes to take effect.

# Configuring Flashback Database Using EM

## Enable flashback logging and specify the flash recovery area.

|    |                           |     |       |
|----|---------------------------|-----|-------|
| 10 | USE_DB_RECOVERY_FILE_DEST | n/a | VALID |
|----|---------------------------|-----|-------|

**TIP** It is recommended that archive log files be written to multiple locations spread across the different disks.  
**TIP** You can specify up to 10 archive log destinations.

### Flash Recovery Area

It is highly recommended that you use flash recovery area to automate your disk backup management.

Flash Recovery Area Location /u01/app/oracle/flash\_recovery\_area

Flash Recovery Area Size 2 GB

Flash Recovery Area Size must be set when the location is set

Used Flash Recovery Area Size (MB) 124.195

Enable flashback logging for fast database point-in-time recovery\*

The flash recovery area must be set to enable flashback logging. When using flashback logs, you may recover your entire database to a prior point-in-time without restoring files. Flashback is the preferred point-in-time recovery method in the recovery wizard when appropriate.

Specify how far back you wish to flash the database in the future

Flashback Retention Time 24 Hours

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Configuring Flashback Database Using EM (continued)

When you are certain that the database is in ARCHIVELOG mode, return to the Recovery Settings page and scroll down to the Media Recovery and Flash Recovery Area regions to observe the new settings. When the flash recovery area and archiving are configured, USE\_DB\_RECOVERY\_FILE\_DEST is configured for archive log destination 10. Enable flashback logging by selecting Enable Flashback Logging. You also have the ability to set the flashback retention time, and you can view important information regarding your flashback database window.

Review the flash recovery area location. The flash recovery area is a unified storage location for all recovery-related files and activities in an Oracle database. All files that are needed to completely recover a database from a media failure are part of the flash recovery area. The recovery-related files that can be created in the flash recovery area include: archived redo log files, control files, backups created by Recovery Manager (RMAN), and flashback logs. By allocating a storage location and unifying recovery-related files within a specific area, the Oracle database server relieves the database administrator from having to manage the disk files created by these components. The default location for the flash recovery area is \$ORACLE\_BASE. If you would like it in a different location, change it now. Scroll down to the bottom of the Recovery Settings page and click **Apply**.

## Flashback Database: Examples

```
RMAN> FLASHBACK DATABASE TO TIME =
2> "TO_DATE('2004-05-27 16:00:00',
3> 'YYYY-MM-DD HH24:MI:SS')";

RMAN> FLASHBACK DATABASE TO SCN=23565;

RMAN> FLASHBACK DATABASE
2> TO SEQUENCE=223 THREAD=1;
```

```
SQL> FLASHBACK DATABASE
2 TO TIMESTAMP(SYSDATE-1/24);

SQL> FLASHBACK DATABASE TO SCN 53943;

SQL> FLASHBACK DATABASE TO RESTORE POINT b4_load;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Flashback Database: Examples

You can use the RMAN FLASHBACK DATABASE command to execute the Flashback Database operation. You can use SEQUENCE and THREAD to specify a redo log sequence number and thread as a lower limit. RMAN selects only files that can be used to flash back to, but not including, the specified sequence number.

Alternatively, you can use the FLASHBACK DATABASE SQL command to return the database to a past time or SCN. If you use the TO SCN clause, you must provide a number. If you specify TO TIMESTAMP, you must provide a time stamp value. You can also specify a restore point name.

**Note:** The database must be mounted in exclusive mode to issue the FLASHBACK DATABASE command and must be opened with the RESETLOGS option when finished.

# Performing Flashback Database Using EM

Select object and operation type.

Perform Recovery: Type

**Warning**

The database will be shut down to perform this operation.

**Operation** - You cannot restore or recover the whole database  
database will be shut down and brought to the MOUNTED state

**Type**

Object Type

Operation  Recover to the current time or a previous point-in-time  
Type Datafiles will be restored from the latest usable backup as required.  
 Restore all datafiles  
Need to specify Time, SCN or log sequence. The backup taken at or prior to that time will be used.  
 Recover from previously restored datafiles

**Host Credentials**

To perform recovery, supply operating system login credentials.

\* Username

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Performing Flashback Database Using EM

On the Maintenance page, choose Perform Recovery. From the Object Type drop-down list, select Whole Database. Next, select “Recover to the current time or a previous point-in-time” as Operation Type. Finally, provide the operating system credentials for a database user (that is, one who belongs to the dba group). When these steps are completed, click Continue to proceed to the next step in the Flashback Database operation.

# Performing Flashback Database Using EM

## Launching the Recovery Wizard:

The screenshot shows the Oracle Enterprise Manager 10g Database Control interface. At the top, there's a menu bar with links for Setup, Preferences, Help, and Logout. A blue header bar displays the text "Database Control". Below it, a message says "Database: orcl.us.oracle.com". The main content area is titled "Recovery Wizard". It contains a message: "The database will be shutdown and started/mounted. Please wait some time for this operation to complete and click 'Refresh'. You will be redirected to a page to go through the recovery wizard." To the right of this message is a "Refresh" button. At the bottom of the page, there's a footer with links for Database, Setup, Preferences, Help, and Logout, followed by copyright information: "Copyright © 1996, 2004, Oracle. All rights reserved." and "About Oracle Enterprise Manager 10g Database Control".

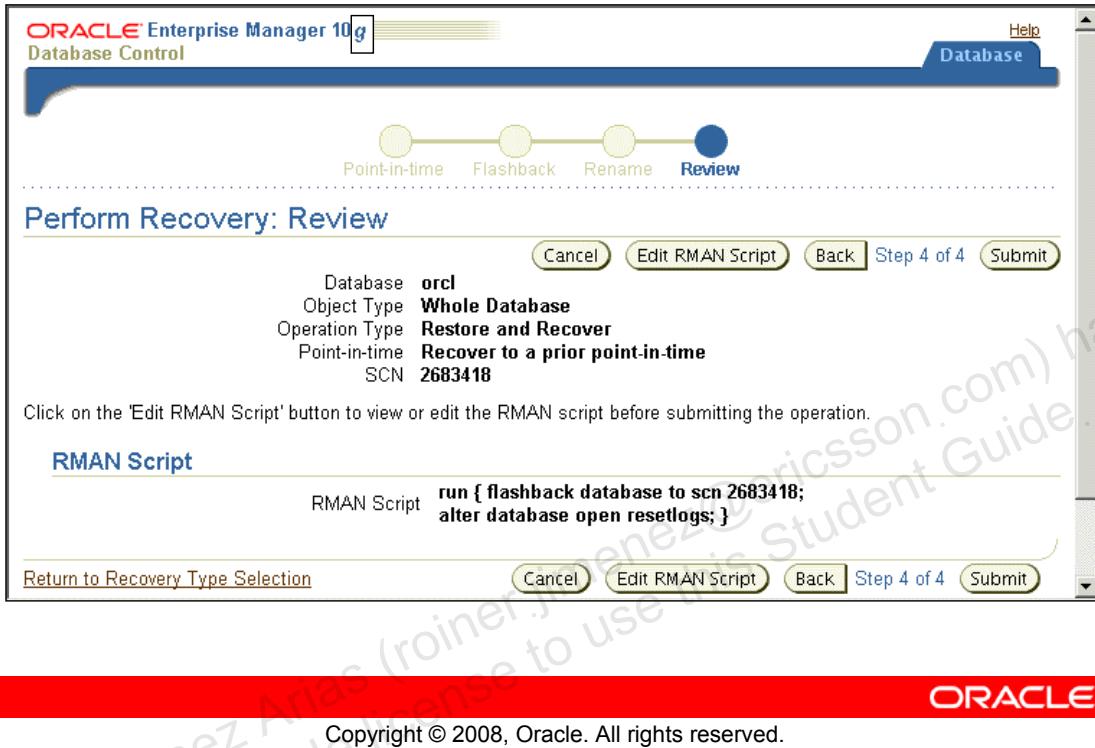
ORACLE®

Copyright © 2008, Oracle. All rights reserved.

## Performing Flashback Database Using EM (continued)

After the recovery operation type has been chosen, the Recovery Wizard is launched. You are informed that the database will be shut down and restarted in MOUNT mode. This operation takes several minutes and you will be informed of the delay. After waiting an appropriate amount of time, you must click Refresh to continue with the operation.

# Performing Flashback Database Using EM



## Performing Flashback Database Using EM (continued)

The Recovery Wizard is now started. At this stage, the database is shut down and started in MOUNT mode. Click Refresh.

This displays the Perform Recovery: Point-in-time page. On this page, select the “Recover to a prior point-in-time” option, and then specify either a date or an SCN. Then, click Next.

The Perform Recovery: Flashback page appears next, and you can choose to perform either recovery using flashback or regular recovery. Choose the corresponding option, and then click Next.

This brings you directly to the Perform Recovery: Review page that is shown in the slide. Click Submit to flash back the database.

# Flashback Database Considerations

- **When the Flashback Database operation completes, open the database:**
  - In read-only mode to verify that the correct target time or SCN was used
  - With a `RESETLOGS` operation to allow for DML
- **The opposite of “flash back” is “recover.”**
- **You cannot use Flashback Database in the following situations:**
  - The control file has been restored or re-created.
  - A tablespace has been dropped.
  - A data file has been shrunk.
- **Use the `TO BEFORE RESETLOGS` to flash back to before the last `RESETLOGS` operation.**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Flashback Database Considerations

In situations where you cannot use the Flashback Database feature, you should use an incomplete recovery operation to return the database to a specific time. After the Flashback Database operation is complete, you can open the database in read-only mode to verify that the correct target time or SCN was used. If not, you can flash back the database again, or perform a recovery to roll forward the database. So, to undo a Flashback Database operation, you should recover the database forward.

You cannot use Flashback Database to recover a data file that was dropped during the span of time you are flashing back. The dropped data file is added to the control file and marked offline, but it is not flashed back. Flashback Database cannot flash back a data file to a time after its creation and before the resize operation. If a file was resized during the span of time to which you are going to flash back the database, then you should take the file offline before beginning the Flashback Database operation. This is applicable for files that are shrunk rather than expanded. You can use Flashback Database with data files that you have configured for automatic extension.

## **Flashback Database Considerations (continued)**

You can flash back to just before the last RESETLOGS operation by supplying the TO BEFORE RESETLOGS clause in the FLASHBACK DATABASE command.

**Note:** The flashback retention target is not an absolute guarantee that flashback will be available. If space is needed for required files in the flash recovery area, flashback logs may be deleted automatically.

# Monitoring Flashback Database

Recycle bin  
Flashback DB  
Config. FB DB  
> **Monitor FB DB**  
Guar. Res. Pt.

To monitor the ability to meet your retention target:

- View the flash recovery area disk quota:

```
SQL> SELECT estimated_flashback_size,
 2 flashback_size
 3 FROM V$FLASHBACK_DATABASE_LOG;
```

- Determine the current flashback window:

```
SQL> SELECT oldest_flashback_scn,
 2 oldest_flashback_time
 3 FROM V$FLASHBACK_DATABASE_LOG;
```

- Monitor logging in the Flashback Database logs:

```
SQL> SELECT *
 2 FROM V$FLASHBACK_DATABASE_STAT;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Monitoring Flashback Database

It is important for you to monitor space usage of the flash recovery area so that you know how well you are meeting your retention target. Use the V\$FLASHBACK\_DATABASE\_LOG view to monitor the Flashback Database retention target:

- ESTIMATED\_FLASHBACK\_SIZE uses previously logged flashback data to provide an estimate of how much disk space is needed in the flash recovery area for flashback logs to meet the current flashback retention target. The estimate is based on the workload since the instance was started, or during the most recent time interval equal to the flashback retention target, whichever is shorter.
- FLASHBACK\_SIZE gives you the current size in bytes of the flashback data.
- OLDEST\_FLASHBACK\_SCN and OLDEST\_FLASHBACK\_TIME display the approximate lowest SCN and time to which you can flash back your database. CURRENT\_SCN in V\$DATABASE gives you the current database SCN.

Use the V\$FLASHBACK\_DATABASE\_STAT view to monitor the overhead of logging flashback data in the Flashback Database logs. This view contains 24 hours of information, with each row representing a one-hour time interval. You can use this view to determine rate changes in the flashback data generation.

```
SQL> SELECT begin_time, end_time, flashback_data, db_data,
 2 redo_data, estimated_flashback_size AS EST_FB_SZE
 3 FROM V$FLASHBACK_DATABASE_STAT;
```

## Monitoring Flashback Database (continued)

| BEGIN_TIME | END_TIME  | FLASHBACK_DATA | DB_DATA  | REDO_DATA | EST_FB_SZE |
|------------|-----------|----------------|----------|-----------|------------|
| 12-FEB-04  | 12-FEB-04 | 16384          | 0        | 24576     | 0          |
| 12-FEB-04  | 12-FEB-04 | 6594560        | 7471104  | 1533440   | 815923200  |
| 12-FEB-04  | 12-FEB-04 | 17235968       | 12361728 | 5150720   | 839467008  |
| 12-FEB-04  | 12-FEB-04 | 311648256      | 37249024 | 10272768  | 855195648  |

Based on this information, you may need to adjust the retention time or the flash recovery area size.

FLASHBACK\_DATA and REDO\_DATA represent the number of bytes of flashback data and redo data written, respectively, during the time interval, and DB\_DATA gives the number of bytes of data blocks read and written. This view also contains the estimated flashback space needed for the interval.

You can query V\$RECOVERY\_FILE\_DEST to view information regarding the flash recovery area. The column descriptions are:

- **NAME:** Flash recovery area name, indicating location string
- **SPACE\_LIMIT:** Disk limit specified in the DB\_RECOVERY\_FILE\_DEST\_SIZE parameter
- **SPACE\_USED:** Space used by flash recovery area files (in bytes)
- **SPACE\_RECLAMABLE:** Amount of space that can be reclaimed by deleting obsolete, redundant, and other low-priority files through the space management algorithm
- **NUMBER\_OF\_FILES:** Number of files

```
SQL> SELECT name, space_limit AS quota,
 2 space_used AS used,
 3 space_reclaimable AS reclaimable,
 4 number_of_files AS files
 5 FROM v$recovery_file_dest ;
```

| NAME                     | QUOTA      | USED       | RECLAMABLE | FILES |
|--------------------------|------------|------------|------------|-------|
| /u01/flash_recovery_area | 5368709120 | 2509807104 | 203386880  | 226   |

# Monitoring Flashback Database with EM

## Flash Recovery

Flash Recovery Area is enabled for this database. The chart shows space used by each file type that is not reclaimable by Oracle. Performing backups to a tertiary storage is one way to make space reclaimable. Usable Flash Recovery Area includes free and reclaimable space.

Flash Recovery Area Location /u01/app/oracle/flash\_recovery\_area

Flash Recovery Area Size 2 GB

Flash Recovery Area Size must be set when th

Reclaimable Flash Recovery Area (MB) 132.2

Free Flash Recovery Area (MB) 900.6

Enable Flashback Database - flashback logging can be used database point-in-time recovery\*

The flash recovery area must be set to enable flashback logging. When using flashback logs to recover your entire database to a prior point-in-time without restoring files. Flashback is the point-in-time recovery method in the recovery wizard when appropriate.

Specify how far back you wish to flashback the database in the future

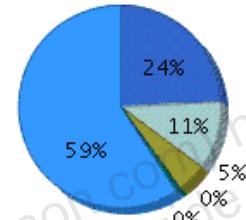
Flashback Retention Time 24 Hour

Current size of the flashback logs(MB) 196.383

Lowest SCN in the flashback data 1252302

Flashback Time Oct 4, 2005 10:31:50 AM

Flash Recovery Area Usage



ORACLE

Copyright © 2008, Oracle. All rights reserved.

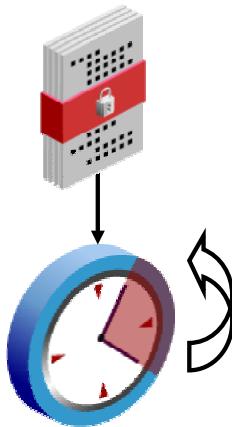
## Monitoring Flashback Database with EM

Most of the Flashback Database statistics mentioned on the preceding pages can be viewed from the Recovery Settings page. These metrics include the current space used by all flashback logs, the lowest SCN, and the time of the lowest SCN in the flashback data.

# Guaranteed Restore Points

Recycle bin  
Flashback DB  
Config. FB DB  
Monitor FB DB  
> Guar. Res. Pt.

A guaranteed restore point ensures that you can perform a FLASHBACK DATABASE command to that SCN at any time.



```
SQL> CREATE RESTORE POINT before_load
2 GUARANTEE FLASHBACK DATABASE;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Guaranteed Restore Points

Like normal restore points, which are covered in the lesson titled “Database Recovery,” guaranteed restore points can be used as aliases for SCNs in recovery operations. However, they also provide specific functionality related to the use of the Flashback Database feature.

Creating a guaranteed restore point at a particular SCN enforces the requirement that you can perform a Flashback Database operation to return your database to its state at that SCN, even if flashback logging is not enabled for your database. If flashback logging is enabled, creating a guaranteed restore point enforces the retention of flashback logs required for Flashback Database back to any point in time after the creation of the earliest guaranteed restore point.

A guaranteed restore point can be used to revert a whole database to a known good state days or weeks ago, as long as there is enough disk space in flash recovery area to store the needed logs.

**Note:** As with normal restore points, guaranteed restore points can be used to specify a point in time for RECOVER DATABASE operations. See the lesson titled “Database Recovery” for more information.

# **Summary**

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

- **Query the recycle bin**
- **Configure Flashback Database**
- **Perform Flashback Database to a point in time using Enterprise Manager or RMAN**
- **Monitor flashback log statistics**
- **Enable and disable the Flashback Database feature**
- **Use guaranteed restore points with Flashback Database**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Practice Overview: Performing Flashback Database

This practice covers the following topics:

- **Performing Flashback Database to undo unwanted transactions**
- **Monitoring the Flashback Database retention**
- **Determine the size of the flashback logs**



Copyright © 2008, Oracle. All rights reserved.

# Dealing with Database Corruption



ORACLE

Copyright © 2008, Oracle. All rights reserved.

# Objectives

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

- **Identify the causes of database corruption:**
  - Hardware
  - Software
- **Detect database corruption by using:**
  - ANALYZE
  - DBVERIFY
  - DB\_BLOCK\_CHECKING
  - DBMS\_REPAIR
- **Repair corruptions by using RMAN**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

# What Is Block Corruption?

- Whenever a block is read or written, a consistency check is performed.
  - Block version
  - DBA (data block address) value in cache as compared to the DBA value in the block buffer
  - Block-checksum, if enabled
- A corrupt block is identified as being one of the following:
  - Media corrupt
  - Logically (or software) corrupt

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## What Is Block Corruption?

A corrupted data block is a block that is not in a recognized Oracle format, or whose contents are not internally consistent. Typically, corruptions are caused by faulty hardware or operating system problems. The Oracle database identifies corrupt blocks as either “logically corrupt” or “media corrupt.” If it is logically corrupt, then there is an Oracle internal error. Logically corrupt blocks are marked corrupt by the Oracle database after it detects the inconsistency. If it is media corrupt, then the block format is not correct; the information in the block does not make any sense after being read from disk.

You can repair a media corrupt block by recovering the block or dropping the database object that contains the corrupt block, or both. If media corruption is due to faulty hardware, the problem will not be completely resolved until the hardware fault is corrected.

## Block Corruption Symptoms: ORA-01578

**The error ORA-01578: "ORACLE data block corrupted (file # %s, block # %s)":**

- **Is generated when a corrupted data block is found**
- **Always returns the relative file number and block number**
- **Is returned to the session that issued the query being performed when the corruption was discovered**
- **Appears in the alert.log file**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

### Block Corruption Symptoms: ORA-01578

Usually, the ORA-01578 error is the result of a hardware problem. If the ORA-01578 error is always returned with the same arguments, it is most likely a media corrupt block.

If the arguments change each time, there may be a hardware problem, and the customer should have the memory and page space checked, and the I/O subsystem checked for bad controllers.

**Note:** ORA-01578 returns the relative file number and the accompanying ORA-01110 error displays the absolute file number.

## How to Handle Corruption

- **Check the alert log and operating system log file.**
- **Use available diagnostic tools to find out the type of corruption.**
- **Determine whether the error persists by running checks multiple times.**
- **Recover data from the corrupted object if necessary.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### How to Handle Corruption

Always try to find out whether the error is permanent. Run the ANALYZE command multiple times or, if possible, perform a shutdown and a startup and try again to perform the operation that failed earlier.

Find out whether there are more corruptions. If you encounter one, there may be other corrupted blocks, as well. Use tools such as DBVERIFY to handle such a situation.

# How to Handle Corruption

- **Resolve any hardware issues:**
  - Memory boards
  - Disk controllers
  - Disks
- **Recover or restore data from the corrupt object if necessary.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## How to Handle Corruption (continued)

There is no point in continuing to work if there are hardware failures. When you encounter hardware problems, the vendor should be contacted and the machine should be checked and fixed before continuing. A full hardware diagnostics session should be run.

Many types of hardware failures are possible:

- Faulty I/O hardware or firmware
- Operating system I/O or caching problem
- Memory or paging problems
- Disk repair utilities

## Corruption-Related Features

| Feature              | Corruption Detected | Repairs Corruption |
|----------------------|---------------------|--------------------|
| DBVERIFY             | Physical            | FALSE              |
| ANALYZE              | Logical             | FALSE              |
| DB_BLOCK_CHECKING    | Logical             | FALSE              |
| DB_BLOCK_CHECKSUM    | Physical            | FALSE              |
| exp                  | Physical            | FALSE              |
| Flashback            | Logical             | TRUE               |
| DBMS_REPAIR          | Logical             | TRUE               |
| Block media recovery | None                | TRUE               |

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Corruption-Related Features

There are many tools available for detecting, diagnosing, and repairing corruption in the Oracle database. The slide shows a summary of the ones covered in this lesson.

## DBVERIFY Utility

- **Works only on data files; redo log files cannot be checked**
- **Checks block consistency**
- **Can be used while the database is open**
- **Name of the utility program: dbv**

```
$ dbv file=/u01/oradata/users01.dbf \
blocksize=8192
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### DBVERIFY Utility

DBVERIFY is an external command-line utility that performs a physical data structure integrity check on a database that is offline or online. It can be used against backup files and online files (or pieces of files). Use DBVERIFY primarily when you need to ensure that a backup of a database or data file is valid before it is restored or as a diagnostic aid when you have encountered data corruption problems. Because DBVERIFY can be run against an offline database, integrity checks are significantly faster.

Limitations of DBVERIFY include the following:

- DBVERIFY cannot detect problems such as INDEX versus TABLE mismatches, which can be detected by the ANALYZE TABLE . . . VALIDATE STRUCTURE CASCADE command.
- DBVERIFY does *not* verify redo log files or control files.
- DBVERIFY only checks a block in isolation; it does not know whether the block is part of an existing object or not.
- For raw devices, you should use the END parameter to avoid scanning blocks past the end of the data file space:

```
dbv FILE=/dev/rdsk/r1.dbf END=last_data_block#
```

# Interpreting DBVERIFY Output

- A “page” is a block.
- If the head and tail do not match, DBVERIFY rereads the block. If they match, an influx block is reported; otherwise, a corruption is signaled.

```
Total Pages Examined : 12800
Total Pages Processed (Data) : 4408
Total Pages Failing (Data) : 0
Total Pages Processed (Index): 1264
.
.
.
Total Pages Marked Corrupt : 4
Total Pages Influx : 0
Highest block SCN : 654836 (0.654836)
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Interpreting DBVERIFY Output

Influx blocks are split blocks. If DBVERIFY reports influx blocks but does not report corruption, it means that when it read the block the first time, DBW $n$  was writing a new version and it caught part of the old and part of the new version of this block. Also, DBVERIFY checks only for logical corruption. Therefore, it is possible for corruption to occur above the high-water mark. A sample DBVERIFY check is shown below:

```
$ dbv file=example01.dbf blocksize=8192
DBVERIFY: Release 10.2.0.1.0 - Production on Fri Sep 9
13:17:45 2005
```

Copyright (c) 1982, 2005, Oracle. All rights reserved.

DBVERIFY - Verification starting : FILE = example01.dbf

DBVERIFY - Verification complete

## Interpreting DBVERIFY Output (continued)

```
Total Pages Examined : 12800
Total Pages Processed (Data) : 4409
Total Pages Failing (Data) : 0
Total Pages Processed (Index): 1264
Total Pages Failing (Index) : 0
Total Pages Processed (Other): 1539
Total Pages Processed (Seg) : 0
Total Pages Failing (Seg) : 0
Total Pages Empty : 5588
Total Pages Marked Corrupt : 0
Total Pages Influx : 0
Highest block SCN : 654836 (0.654836)
```

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a  
non-transferable license to use this Student Guide.

## ANALYZE Command

- **Performs a logical block check**
- **Does not mark blocks as soft corrupt; only reports them**
- **Validates index and table entries**

```
SQL> ANALYZE TABLE table_name VALIDATE
 2 STRUCTURE CASCADE;

SQL> ANALYZE INDEX index_name VALIDATE
 2 STRUCTURE;
```

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

### ANALYZE Command

Use the ANALYZE command to validate the structure of a table or table partitions, and index or index partitions. The object to be analyzed must be local and it must be in your own schema or you must have the ANALYZE ANY system privilege. The CASCADE option validates an object, including all related objects. You can run this in a SQL\*Plus session on a specific object, to perform an integrity check and to determine whether an error is persistent, by running the same ANALYZE command several times.

For partitioned tables, ANALYZE also verifies that the row belongs to the correct partition. If the row does not collate correctly, the row ID is inserted in the INVALID\_ROWS table.

```
SQL> ANALYZE TABLE partitioned_table PARTITION (p1)
 2 VALIDATE STRUCTURE INTO invalid_rows;
```

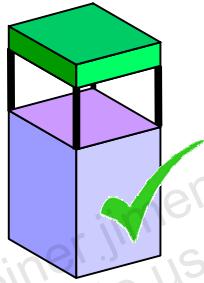
A simple select statement (SELECT \* FROM *table*) performs a full table scan, which means that it reads all the data blocks up to the high-water mark of the table. You could use this to perform a quick check for corruptions in your current table data. You can also use Data Pump to export objects; this also fully scans each table.

**Note:** ANALYZE validates bitmap information for automatic segment space management (ASSM) segments but does not account for unformatted blocks under the high-water mark in ASSM segments.

## Verifying Block Integrity in Real Time: **DB\_BLOCK\_CHECKING**

**The DB\_BLOCK\_CHECKING initialization parameter:**

- **Controls the degree of self-consistency checks performed on each block as it is processed**
- **Can prevent memory and data corruption**
- **Can be set by using the ALTER SESSION or ALTER SYSTEM command**



**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

### Verifying Block Integrity in Real Time: DB\_BLOCK\_CHECKING

When DB\_BLOCK\_CHECKING is set to TRUE, the Oracle database performs block checking for all data blocks. The Oracle database checks a block by reading the data on the block and making sure that it is self-consistent. Block checking can often prevent memory and data corruption.

Block checking typically causes 1% to 10% overhead, depending on workload. The more updates or inserts being executed, the more expensive it is to turn on block checking. The four possible values for DB\_BLOCK\_CHECKING are:

- **OFF:** No block checking is performed in any tablespaces except for SYSTEM.
- **LOW:** Basic block header checks are performed after block contents change in memory (for example, after UPDATE or INSERT statements, and on-disk reads).
- **MEDIUM:** All LOW checks, as well as block checking for all non-index-organized table blocks, are performed.
- **FULL:** All LOW and MEDIUM checks, as well as checks on index blocks, are performed.

You should set DB\_BLOCK\_CHECKING to FULL if the performance overhead is acceptable. The default value is FALSE, which, for backward compatibility, is equivalent to OFF. Even if this is turned off, block checking for the SYSTEM tablespace is always turned on.

## Verifying Block Integrity in Real Time: DB\_BLOCK\_CHECKSUM

**The DB\_BLOCK\_CHECKSUM initialization parameter:**

- **Determines whether a checksum is maintained and verified on each block**
- **Can prevent corruption caused by underlying I/O systems**



**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

### Verifying Block Integrity in Real Time: DB\_BLOCK\_CHECKSUM

If DB\_BLOCK\_CHECKSUM is set to TRUE, the DBW $n$  process and the direct loader calculate a checksum and store it in the cache header of every data block when writing it to disk. A checksum is a number calculated from all the bytes stored in the block. When the block is subsequently read, the checksum is recomputed and the stored value is checked with this computed value. Because checksums allow the database to detect corruption caused by underlying disks, storage systems, or I/O systems with only a 1% to 2% overhead, Oracle recommends that you set DB\_BLOCK\_CHECKSUM to TRUE, which is the default.

# Using EXP to Detect Corruption

**Conventional export can be used to detect corruption.**

```
$ exp hr/hr tables=departments

About to export specified tables via Conventional Path
...
. . . exporting table DEPARTMENTS
EXP-00056: ORACLE error 1578 encountered
ORA-01578: ORACLE data block corrupted (file # 5, block #
51)
ORA-01110: data file 5:
'/u01/app/oracle/oradata/orcl/example01.dbf'
```

ORACLE

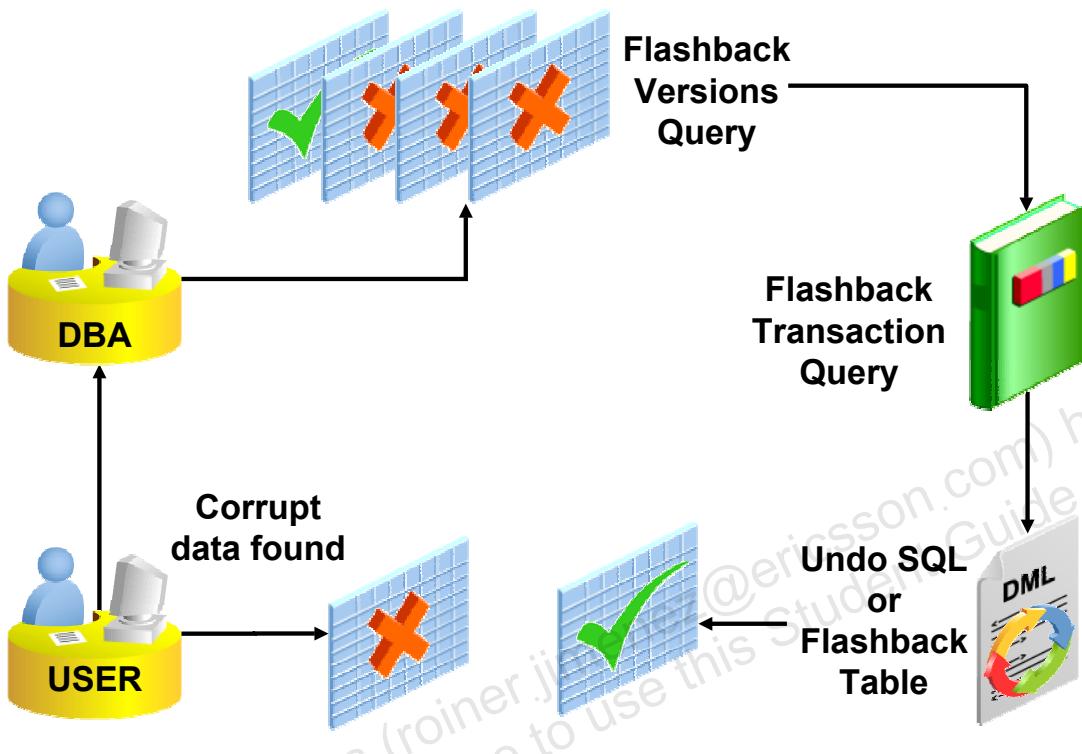
Copyright © 2008, Oracle. All rights reserved.

## Using EXP to Detect Corruption

Another option for detecting block corruption is the export utility, or EXP. Because EXP fully scans every block of the object it is exporting, it reports an errors when it encounters corruption.

**Note:** The Data Pump Export utility can also be used to detect corruption, because it performs the same kind of activity on the database as Export does.

# Using Flashback for Logical Corruption



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Using Flashback for Logical Corruption

After a physical corruption has been ruled out, you can use a combination of the flashback features to determine when a logical corruption occurred. For example, you can first use Flashback Versions Query to view the values of a row over a period of time. Flashback Versions Query can also return the transaction ID. When you have the transaction ID, you can view all objects affected by the same transaction.

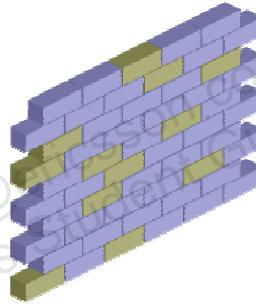
To undo the data corruption, you can use the information returned in the UNDO\_SQL column to fix the error, or you can flashback the table to the time(SCN) before the transaction was issued.

**Note:** Details about performing a Flashback Versions Query are covered in the *Oracle Database 10g: Administration Workshop I* course.

## DBMS\_REPAIR Package

### Available procedures

- **CHECK\_OBJECT**
- **FIX\_CORRUPT\_BLOCKS**
- **DUMP\_ORPHAN\_KEYS**
- **REBUILD\_FREELISTS**
- **SEGMENT\_FIX\_STATUS**
- **SKIP\_CORRUPT\_BLOCKS**
- **ADMIN\_TABLES**



ORACLE

Copyright © 2008, Oracle. All rights reserved.

### DBMS\_REPAIR Package

Another way to manage data block corruption is to use the DBMS\_REPAIR package. You can use DBMS\_REPAIR to detect and repair corrupt blocks in tables and indexes. Using this approach, you can address corruptions where possible, and also continue to use objects while you attempt to rebuild or repair them. The procedures contained in the package include:

- **CHECK\_OBJECT**: Detects and reports corruptions in a table or index
- **FIX\_CORRUPT\_BLOCKS**: Marks blocks that were previously identified by the CHECK\_OBJECT procedure as software (or logically) corrupt
- **DUMP\_ORPHAN\_KEYS**: Reports index entries into an orphan key table that point to rows in corrupt data blocks
- **REBUILD\_FREELISTS**: Rebuilds an object's free lists
- **SEGMENT\_FIX\_STATUS**: Provides the capability to fix the corrupted state of a bitmap entry when segment space management is AUTO
- **SKIP\_CORRUPT\_BLOCKS**: Ignores blocks marked corrupt during table and index scans. If not used, you get the ORA-01578 error when encountering blocks marked corrupt.
- **ADMIN\_TABLES**: Provides administrative functions (create, drop, or purge) for repair of orphan key tables. These tables are always created in the SYS schema.

**Note:** Orphan keys are index entries that point to table rows that no longer exist.

# Using DBMS\_REPAIR

## 1. Detect and report corruptions.

```
SET SERVEROUTPUT ON
DECLARE num_corrupt INT;
BEGIN
 num_corrupt := 0;
 DBMS_REPAIR.CHECK_OBJECT (
 schema_name => 'HR',
 object_name => 'DEPARTMENTS',
 repair_table_name => 'REPAIR_TABLE',
 corrupt_count => num_corrupt);
END;
```

## 2. Evaluate the costs and benefits of DBMS\_REPAIR.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Using DBMS\_REPAIR

Your first task, before using DBMS\_REPAIR, should be the detection and reporting of corruptions. Reporting not only indicates what is wrong with a block, but also identifies the associated repair directive. After using the options presented earlier in this lesson, you can consider using the DBMS\_REPAIR package to correct the corruption. The CHECK\_OBJECT procedure checks the specified objects, and populates a repair table with information about corruptions and repair directives. Before executing any of the DBMS\_REPAIR procedures, you must build the repair table by using the ADMIN\_TABLES procedure:

```
BEGIN
 DBMS_REPAIR.ADMIN_TABLES (
 table_name => 'REPAIR_TABLE',
 table_type => DBMS_REPAIR.REPAIR_TABLE,
 action => DBMS_REPAIR.CREATE_ACTION,
 tablespace => 'USERS');
END;
```

Before using DBMS\_REPAIR, you must weigh the benefits of its use in relation to the liabilities. You should also examine other options available for addressing corrupt objects.

## Using DBMS\_REPAIR (continued)

A first step is to answer the following questions:

- What is the extent of the corruption? To determine whether there are corruptions and repair actions, execute the CHECK\_OBJECT procedure, and query the repair table.
- What are the other options that are available for addressing block corruptions?
- What logical corruptions or side effects are introduced when using DBMS\_REPAIR to make an object usable? Can these be addressed satisfactorily?
- If repair involves loss of data, can this data be retrieved?

You can retrieve data from the index when a data block is marked corrupt. The DUMP\_ORPHAN\_KEYS procedure can help you retrieve this information. Of course, retrieving data in this manner depends on the amount of redundancy between the indexes and the table. The following example illustrates the creation of an orphan key table for the USERS tablespace.

```
BEGIN
 DBMS_REPAIR.ADMIN_TABLES (
 table_name => 'ORPHAN_KEY_TABLE',
 table_type => DBMS_REPAIR.ORPHAN_TABLE,
 action => DBMS_REPAIR.CREATE_ACTION,
 tablespace => 'USERS');
END;
```

## Using DBMS\_REPAIR

### 3. Make objects usable.

```
SET SERVEROUTPUT ON
DECLARE num_fix INT;
BEGIN
 num_fix := 0;
 DBMS_REPAIR.FIX_CORRUPT_BLOCKS (
 schema_name => 'HR',
 object_name => 'DEPARTMENTS',
 object_type => DBMS_REPAIR.TABLE_OBJECT,
 repair_table_name => 'REPAIR_TABLE',
 fix_count => num_fix);
END;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Using DBMS\_REPAIR (continued)

DBMS\_REPAIR makes the object usable by ignoring corruptions during table and index scans. By using the FIX\_CORRUPT\_BLOCKS and SKIP\_CORRUPT\_BLOCKS procedures, you can make a corrupt object usable by establishing an environment that skips corruptions that remain outside the scope of repair capabilities of DBMS\_REPAIR.

If corruptions involve a loss of data, such as a bad row in a data block, all such blocks are marked corrupt by the FIX\_CORRUPT\_BLOCKS procedure. Then, you can run the SKIP\_CORRUPT\_BLOCKS procedure, which skips blocks marked corrupt for the object. When skip is set, table and index scans skip all blocks marked corrupt. This applies to both media and software corrupt blocks. You can see whether this is currently in effect for a table by viewing the SKIP\_CORRUPT column of the DBA\_TABLES data dictionary view.

If an index and table are out of sync, then a SET TRANSACTION READ ONLY transaction can be inconsistent in situations where one query probes only the index, and then a subsequent query probes both the index and the table. If the table block is marked corrupt, then the two queries return different results, thereby breaking the rules of a read-only transaction. One way to approach this is to not skip corruptions when in a SET TRANSACTION READ ONLY transaction.

## Using DBMS\_REPAIR

### 4. Repair corruptions and rebuild lost data.

```
SET SERVEROUTPUT ON
DECLARE num_orphans INT;
BEGIN
 num_orphans := 0;
 DBMS_REPAIR.DUMP_ORPHAN_KEYS (
 schema_name => 'SCOTT',
 object_name => 'PK_DEPT',
 object_type => DBMS_REPAIR.INDEX_OBJECT,
 repair_table_name => 'REPAIR_TABLE',
 orphan_table_name => 'ORPHAN_KEY_TABLE',
 key_count => num_orphans);
 DBMS_OUTPUT.PUT_LINE('orphan key count: ' ||
 TO_CHAR(num_orphans));
END;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Using DBMS\_REPAIR (continued)

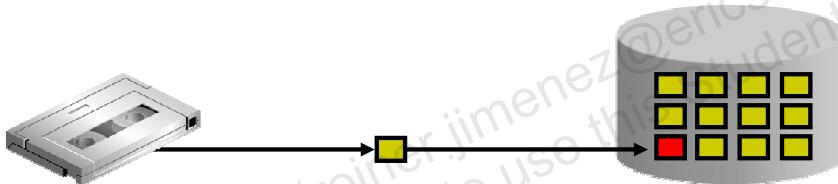
After making an object usable, you can recover data by using the DUMP\_ORPHAN\_KEYS procedure. This procedure reports on index entries that point to rows in corrupt data blocks. All such index entries are inserted into an orphan key table that stores the key and row ID of the corruption. The orphan key table must have been previously created.

After the index entry information has been retrieved, you can rebuild the index by using the ALTER INDEX ... REBUILD ONLINE statement.

# Block Media Recovery (BMR)

## Block media recovery:

- **Lowers the mean time to recover (MTTR)**
- **Increases availability during media recovery**
  - The data file remains online during recovery.
  - Only blocks being recovered are inaccessible.
- **Is invoked through RMAN via the BLOCKRECOVER command.**
  - Restores individual blocks from available backups
  - Coordinates with the server to have them recovered



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Block Media Recovery (BMR)

BMR reduces the smallest recoverable unit of media recovery from a data file to a block. When a small number of blocks in the database are known to require media recovery, it is more efficient to selectively restore and recover just those blocks. Only blocks that are being recovered need to be unavailable, allowing continuous availability of the rest of the database during recovery.

BMR provides two main benefits over file-level recovery:

- Lowers the mean time to recover (MTTR)
- Allows increased availability of data during media recovery because the data file that is being recovered remains online

BMR uses existing recovery mechanisms to apply changes from the redo stream to versions of blocks that are restored from suitable backups. RMAN must be used to perform BMR. RMAN restores individual data blocks from available backups and coordinates with the Oracle server process to have them recovered. Without block-level recovery, if even a single block is corrupt, the entire file must be restored and all redo data must be applied to that file. The reduction in MTTR that is realized by using block-level recovery includes both restore and recovery time. Note that only complete recovery is possible. Incomplete recovery would render the database logically inconsistent.

## BLOCKRECOVER Command

### The RMAN BLOCKRECOVER command:

- **Identifies the backups containing the blocks to recover**
- **Reads the backups and accumulates requested blocks into in-memory buffers**
- **Manages the block media recovery session by reading the archive logs from backup if necessary**
- **Cannot be used for incomplete recovery**

```
RMAN> BLOCKRECOVER DATAFILE 6 BLOCK 3;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### BLOCKRECOVER Command

RMAN supports BMR by means of the BLOCKRECOVER command. When the user encounters block corruption, the error message, alert log, and trace files indicate which block is causing problems. The DBA can then invoke this command to restore only the block in question, thus saving an enormous amount of down time and data unavailability.

The BLOCKRECOVER command does the following:

- Identifies the backups from which to obtain the blocks to recover
- Reads the backups and accumulates requested blocks into in-memory buffers. If any of the desired blocks are corrupt (either media or logical corruption), RMAN reads the next oldest backup of that file, looking for a good copy of the block. The UNTIL option limits selection to backup sets or file copies that are taken at or before the specified time, SCN, or log sequence; this forces BLOCKRECOVER to use an older backup instead of the most recent one.
- Starts and manages the block media recovery session, reading the archive logs from backup if necessary
- Always does a complete recovery. No point-in-time recovery is possible by using the BLOCKRECOVER command.

## Examples of Using BLOCKRECOVER

- **Recovering a group of corrupt blocks**
- **Limiting block media recovery by type of restore**
- **Limiting block media recovery by backup tag**
- **Limiting block media recovery by time, SCN, or log sequence**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Examples of Using BLOCKRECOVER

- Recovering a group of corrupt blocks

```
BLOCKRECOVER DATAFILE 2 BLOCK 12, 13
DATAFILE 7 BLOCK 5, 98, 99 DATAFILE 9 BLOCK 19;
```
  - This example recovers a series of blocks and restores only from data file copies:

```
{
 BLOCKRECOVER DATAFILE 3 BLOCK 1,2,3,4,5
 TABLESPACE sales DBA 4194405, 4194409, 4194412
 FROM DATAFILE COPY;
}
```
- Note:** DBA is data block address.
- Limiting BMR by backup tag:

```
BLOCKRECOVER TABLESPACE SYSTEM DBA 4194404, 4194405
FROM TAG "weekly_backup";
```
  - The following example recovers two blocks in the SYSTEM tablespace and forces the blocks to be restored from backups that were created at least two days ago:

```
BLOCKRECOVER TABLESPACE SYSTEM DBA 4194404, 4194405 RESTORE
UNTIL TIME 'SYSDATE-2';
```

## Examples of Using BLOCKRECOVER (continued)

- The following example recovers two blocks and forces the blocks to be restored by using backups that were made before SCN 100:

```
BLOCKRECOVER DATAFILE 9 BLOCK 13 DATAFILE 2 BLOCK 19 RESTORE
UNTIL SCN 100;
```

- The following example recovers two blocks and forces the blocks to be restored by using backups that were made before log sequence 7024:

```
BLOCKRECOVER DATAFILE 9 BLOCK 13 DATAFILE 2 BLOCK 19 RESTORE
UNTIL SEQUENCE 7024;
```

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a  
non-transferable license to use this Student Guide.

## The RMAN BMR Interface

**Dynamic views show the current state of corruption.**

- **The V\$DATABASE\_BLOCK\_CORRUPTION view shows the list of currently corrupted database blocks.**

```
RMAN> BLOCKRECOVER CORRUPTION LIST
2> RESTORE UNTIL TIME 'sysdate - 10';
```

- **The V\$BACKUP\_CORRUPTION view shows the list of corrupted blocks in data file backups.**
- **The V\$COPY\_CORRUPTION view shows the list of corrupted blocks in image file copies.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### The RMAN BMR Interface

By using the CORRUPTION LIST clause, you can recover blocks that are listed in V\$DATABASE\_BLOCK\_CORRUPTION. This view is populated during a backup operation, as corrupt blocks are encountered. If the backup encounters more than the tolerated number of corrupt blocks, then the view is not populated at all. You should then run the BACKUP...VALIDATE command to fully populate this view with a record of all corrupt blocks.

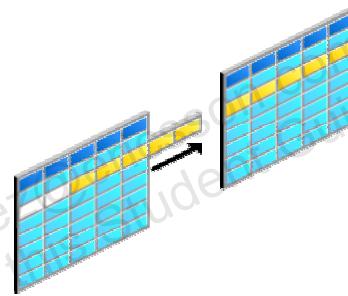
After a block has been repaired through block media recovery (or normal media recovery), V\$DATABASE\_BLOCK\_CORRUPTION is not updated until you take a new backup. The UNTIL clause specifies that only backups and copies that are created before the specified time, SCN, or log sequence number should be restored and used for the recovery. RMAN will still recover fully to the present time.

Two types of corruption result in rows that are being added to V\$BACKUP\_CORRUPTION and V\$COPY\_CORRUPTION by the BACKUP and COPY commands, respectively:

- **Physical corruption (sometimes called media corruption):** The Oracle server does not recognize the block at all: the checksum is invalid, the block contains all zeros, or the header and footer of the block do not match. Physical corruption checking is ON by default, and can be turned off with the NOCHECKSUM option.
- **Logical corruption:** The block has a valid checksum, the header and footer match, and so forth, but the contents are logically inconsistent. Logical checking is OFF by default, and can be turned on with the CHECK LOGICAL option.

# Alternative Actions to Take

- **Table: The data in the corrupted block is lost.**
  - Drop the table and re-create it, and import data from an export dump.
  - Use SQL or PL/SQL to pull data out of the table into a newly created table.
- **Index: Drop and re-create the index.**



**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Alternative Actions to Take

If you do not plan to restore data files and recover, use the following statement to determine which object has corrupted blocks.

The absolute file number (for example, 5) and block number (for example, 2) can be found in the error message; for example:

ORA-01578: ORACLE data block corrupted (file #5, block # 2)

Run the following command:

```
SQL> SELECT segment_name, segment_type, relative_fno
 2 FROM dba_extents
 3 WHERE file_id = 5
 4 AND 2 BETWEEN block_id AND block_id + blocks - 1;
SEGMENT_NAME SEGMENT_TYPE RELATIVE_FNO
----- -----
EXAMPLE TABLE PARTITION 5
```

After you determine which object is corrupted, there are other options available to you for recovering from it. For a table, you can use an export dump file. You can also use SQL to generate a good table from the corrupted one. This may require knowledge of the business data. If you have a corrupted index, it may be most efficient to simply drop and re-create the index.

# Summary

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

- **Identify the causes of database corruption:**
  - Hardware
  - Software
- **Detect database corruption by using:**
  - ANALYZE
  - dbverify
  - DB\_BLOCK\_CHECKING
  - DBMS\_REPAIR
- **Repair corruptions by using RMAN**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Practice Overview: Perform Block Media Recovery

**This practice covers the following topics:**

- **Discovering corruption**
- **Identifying the location of the corruption**
- **Recovering from the corruption by using block media recovery**

**ORACLE®**

Copyright © 2008, Oracle. All rights reserved.

# Monitoring and Managing Memory

8

ORACLE®

Copyright © 2008, Oracle. All rights reserved.

# Objectives

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

- **Describe the memory components in the SGA**
- **Implement Automatic Shared Memory Management**
- **Manually configure SGA parameters**
- **Configure automatic PGA memory management**

**ORACLE®**

Copyright © 2008, Oracle. All rights reserved.

# Memory Management: Overview

**DBAs must consider memory management to be a crucial part of their job because:**

- **There is a finite amount of memory available**
- **Allocating more memory to serve certain types of functions can improve overall performance**
- **Automatically tuned memory allocation is often the appropriate configuration, but specific environments or even short-term conditions may require further attention**



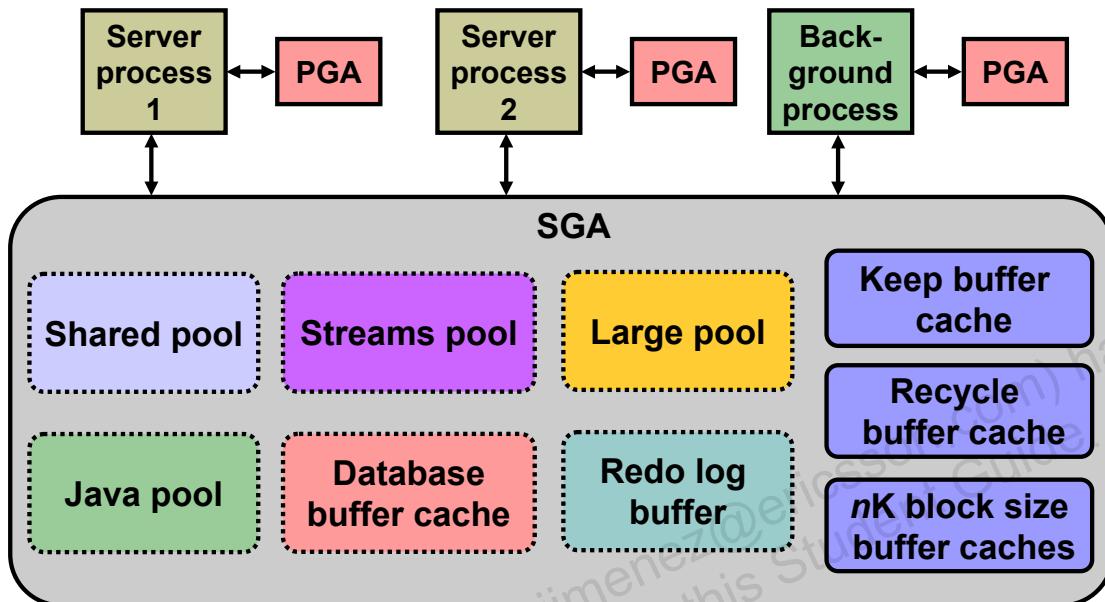
Copyright © 2008, Oracle. All rights reserved.

## Memory Management: Overview

Because there is a finite amount of memory available on a database server and thus, on an Oracle database instance, you must pay attention to how memory is allocated. If too much memory is allowed to be used by a particular area that does not need it, then there is the possibility that there are other functional areas unnecessarily doing without enough memory to perform optimally. With the ability to have memory allocation automatically determined and maintained for you, the task is simplified greatly. But even automatically tuned memory needs to be monitored for optimization and may need to be manually configured to some extent.

Beyond the introduction to memory tuning presented in the *Oracle Database 10g: Administration Workshop I* course, this lesson describes how automatic memory tuning works, when to do manual tuning, what the Program Global Area (PGA) is, and details of each of the memory structures in an Oracle instance.

# Oracle Memory Structures



**ORACLE®**

Copyright © 2008, Oracle. All rights reserved.

## Oracle Memory Structures

The basic memory structures associated with an Oracle instance include:

- **System Global Area (SGA):** Shared by all server and background processes
- **Program Global Area (PGA):** Private to each server and background process; there is one PGA for each process

The System Global Area (SGA) is a shared memory area that contains data and control information for the instance, including the following:

- **Database buffer cache:** Caches blocks of data retrieved from disk
- **Redo log buffer:** Caches redo information until it can be written to disk
- **Shared pool:** Caches various constructs that can be shared among users
- **Large pool:** Optional area used for buffering large I/O requests in support of parallel query, shared server, Oracle XA, and certain types of backup operations
- **Java pool:** Holds session-specific Java code and data within the Java Virtual Machine (JVM)
- **Streams pool:** Used by Oracle Streams
- **Keep buffer cache:** Holds data that is kept in the buffer cache as long as possible
- **Recycle buffer cache:** Holds data that is quickly aged out of the buffer cache
- **nK block size buffer caches:** Caches data blocks that are of a different size than the default database block size; used to support transportable tablespaces

## Oracle Memory Structures (continued)

The size of the database buffer cache, shared pool, large pool, streams pool, and Java pool can be adjusted automatically to meet present needs. Also, those memory buffers, along with the keep buffer cache, the recycle buffer cache, and the  $nK$  block size buffer caches, can be changed without shutting down the instance.

The preconfigured database has been pretuned with adequate settings for the memory parameters. However, as your database usage expands, you may find it necessary to alter the settings of the memory parameters.

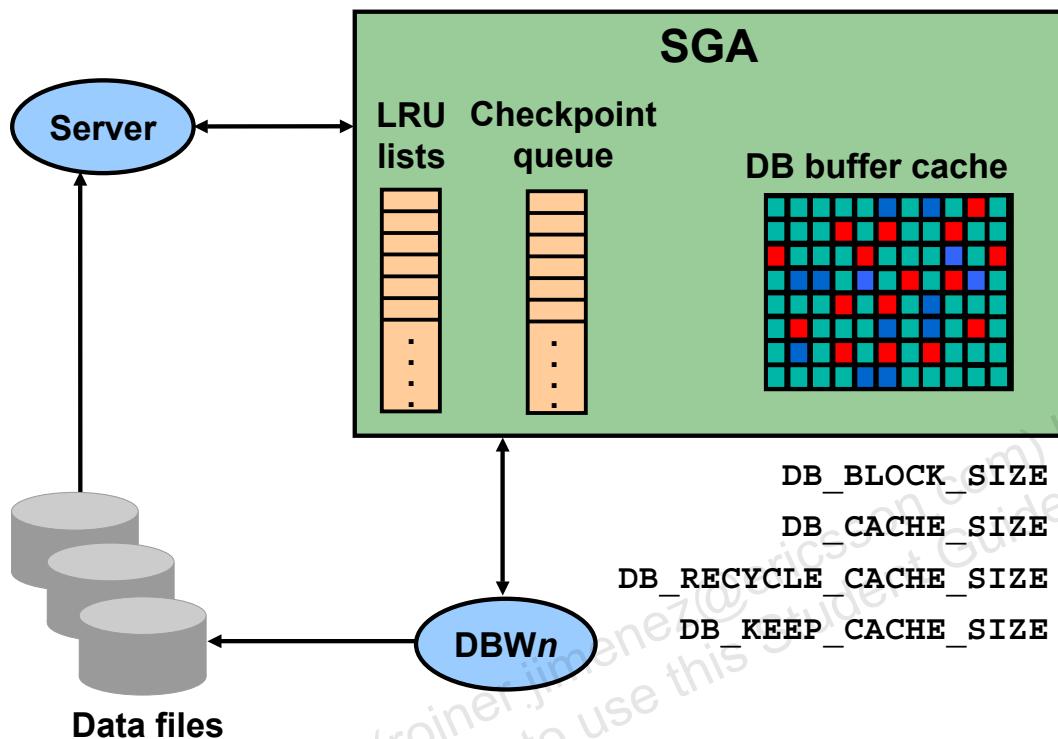
The Oracle database provides alerts and advisors to identify memory-sizing problems and to help you determine appropriate values for memory parameters.

A Program Global Area (PGA) is a memory region that contains data and control information for each server process. A server process is a process that services a client's requests. Each server process has its own private PGA that is created when the server process is started. Access to it is exclusive to that server process.

The amount of PGA memory used and the contents of the PGA depend on whether the instance is configured in shared server mode. Generally, the PGA contains the following:

- **Private SQL area:** Contains data such as bind information and run-time memory structures. Each session that issues a SQL statement has a private SQL area.
- **Session memory:** Memory allocated to hold session variables and other information related to the session

## Buffer Cache



**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

### Buffer Cache

You can configure the buffer cache by specifying a value for the DB\_CACHE\_SIZE parameter. The buffer cache holds copies of the data blocks from the data files having block size of DB\_BLOCK\_SIZE. The buffer cache is a part of the SGA; so all users can share these blocks. The server processes read data from the data files into the buffer cache. To improve performance, the server process sometimes reads multiple blocks in a single read operation. The DBWn process writes data from the buffer cache into the data files. To improve performance, DBWn writes multiple blocks in a single write operation.

At any given time, the buffer cache may hold multiple copies of a single database block. Only one current copy of the block exists, but to satisfy queries, server processes may need to construct read-consistent copies from past image information. This is called a consistent read (CR) block.

The least recently used (LRU) list monitors the usage of buffers. The buffers are sorted on the basis of a combination of how recently and how often they have been referenced. Thus, buffers that are most frequently and recently used are found at the most recently used end. Incoming blocks are copied to a buffer from the least recently used end, which is then assigned to the middle of the list, as a starting point. From here, the buffer works its way up or down the list, depending on usage.

## Buffer Cache (continued)

Buffers in the buffer cache can be in one of four states:

- **Pinned:** The block is either currently being read into the cache or being written to. Other sessions wait to access the block.
- **Clean:** The buffer is now unpinned and is a candidate for immediate aging out if the current contents (data block) are not referenced again. Either the contents are in sync with disk or the buffer contains a CR snapshot of a block.
- **Free/unused:** The buffer is empty because the instance just started. This state is very similar to the clean state, except that the buffer has not been used.
- **Dirty:** The buffer is no longer pinned but the contents (data block) have changed and must be flushed to disk by DBW $n$  before it can be aged out.

Server processes use the buffers in the buffer cache, but the DBW $n$  process makes buffers in the cache available by writing changed buffers back to the data files. The checkpoint queue lists the buffers that are to be written out to disk.

Then Oracle database supports multiple block sizes in the same database. The standard block size is used for the SYSTEM tablespace. You specify the standard block size by setting the initialization parameter DB\_BLOCK\_SIZE. Legitimate values are from 2K to 32K, and the default is 8K. The cache sizes of nonstandard block size buffers are specified by the following parameters:

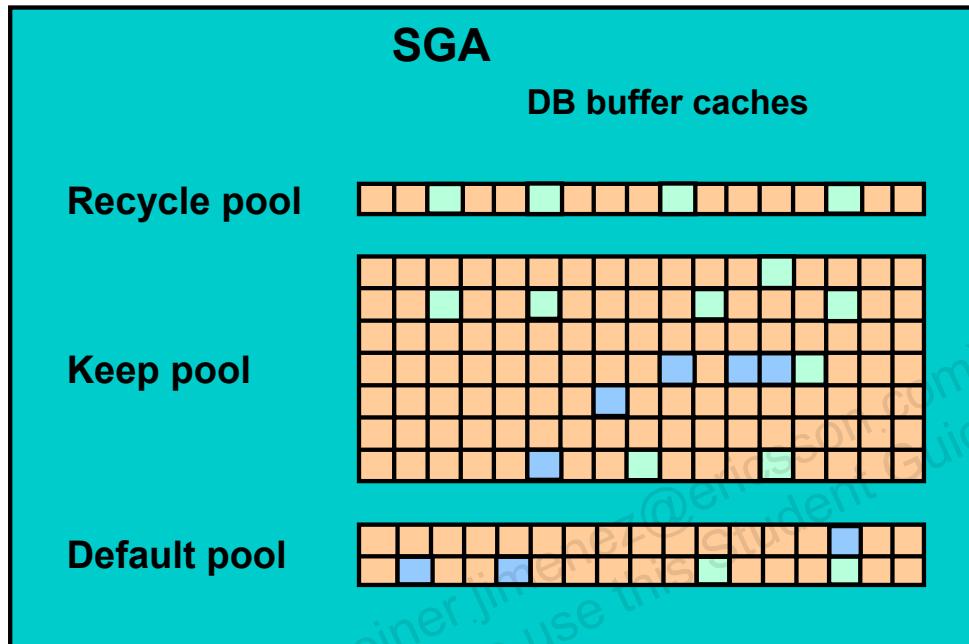
- DB\_2K\_CACHE\_SIZE
- DB\_4K\_CACHE\_SIZE
- DB\_8K\_CACHE\_SIZE
- DB\_16K\_CACHE\_SIZE
- DB\_32K\_CACHE\_SIZE

The DB\_nK\_CACHE\_SIZE parameters cannot be used to size the cache for the standard block size. If the value of DB\_BLOCK\_SIZE is nK, it is illegal to set DB\_nK\_CACHE\_SIZE. The size of the cache for the standard block size is always determined from the value of DB\_CACHE\_SIZE.

Each buffer cache has a limited size, so typically not all the data on disk can fit in the cache. When the cache is full, subsequent cache misses cause the Oracle database to write dirty data already in the cache to disk to make room for the new data. (If a buffer is not dirty, it does not need to be written to disk before a new block can be read into the buffer.) Subsequent access to any data that was written to disk results in additional cache misses.

The size of the cache affects the likelihood that a request for data will result in a cache hit. If the cache is large, it is more likely to contain the data that is requested. Increasing the size of a cache increases the percentage of data requests that result in cache hits.

# Using Multiple Buffer Pools



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Using Multiple Buffer Pools

The database administrator (DBA) may be able to improve the performance of the database buffer cache by creating multiple buffer pools. You assign objects to a buffer pool depending on how the objects are accessed. There are three buffer pools:

- **Keep:** This pool is used to retain objects in memory that are likely to be reused. Keeping these objects in memory reduces I/O operations. Buffers are kept in this pool by ensuring that the pool is sized larger than the total size of the segments assigned to the pool. This means that buffers do not have to be aged out. The keep pool is configured by specifying a value for the DB\_KEEP\_CACHE\_SIZE parameter.
- **Recycle:** This pool is used for blocks in memory that have little chance of being reused. The recycle pool is sized smaller than the total size of the segments assigned to the pool. This means that blocks read into the pool will often have to age out a buffer. The recycle pool is configured by specifying a value for the DB\_RECYCLE\_CACHE\_SIZE parameter.
- **Default:** This pool always exists. It is equivalent to the buffer cache of an instance without a keep or a recycle pool and is configured with the DB\_CACHE\_SIZE parameter.

**Note:** The memory in the keep or recycle pool is not a subset of the default buffer pool.

# Using Multiple Buffer Pools

```
CREATE INDEX cust_idx ...
 STORAGE (BUFFER_POOL KEEP ...);

ALTER TABLE oe.customers
 STORAGE (BUFFER_POOL RECYCLE);

ALTER INDEX oe.cust_lname_ix
 STORAGE (BUFFER_POOL KEEP);
```



Copyright © 2008, Oracle. All rights reserved.

## Using Multiple Buffer Pools (continued)

The BUFFER\_POOL clause is used to define the default buffer pool for an object. It is part of the STORAGE clause and is valid for CREATE and ALTER table, cluster, and index statements. The blocks from an object without an explicitly set buffer pool go into the default buffer pool.

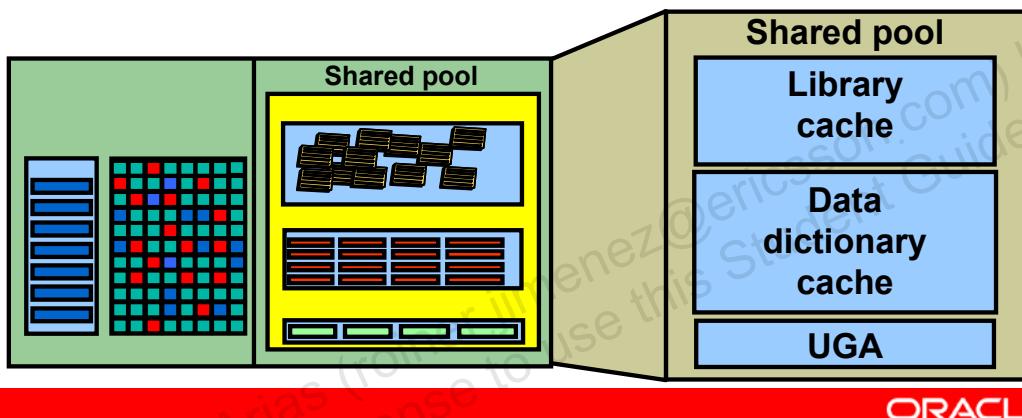
The syntax is BUFFER\_POOL [KEEP | RECYCLE | DEFAULT].

When the default buffer pool of an object is changed using the ALTER statement, blocks that are already cached remain in their current buffers until they are flushed out by the normal cache management activity. Blocks read from disk are placed into the newly specified buffer pool for the segment.

Because buffer pools are assigned to a segment, objects with multiple segments can have blocks in multiple buffer pools. For example, an index-organized table can have different pools defined on both the index and the overflow segment.

## Shared Pool

- **Size is specified by using SHARED\_POOL\_SIZE.**
- **Library cache contains statement text, parsed code, and execution plan.**
- **Data dictionary cache contains definitions for tables, columns, and privileges from the data dictionary tables.**
- **The User Global Area (UGA) contains session information if using Oracle shared server.**



Copyright © 2008, Oracle. All rights reserved.

### Shared Pool

You can specify the size of the shared pool with the SHARED\_POOL\_SIZE initialization parameter. The shared pool is a memory area that stores information shared by multiple sessions. It contains different types of data, as shown in the graphic in the slide.

#### Library Cache

The library cache contains shared SQL and PL/SQL areas—the fully parsed or compiled representations of PL/SQL blocks and SQL statements.

PL/SQL blocks include:

- Procedures and functions
- Packages
- Triggers
- Anonymous PL/SQL blocks

#### Data Dictionary Cache

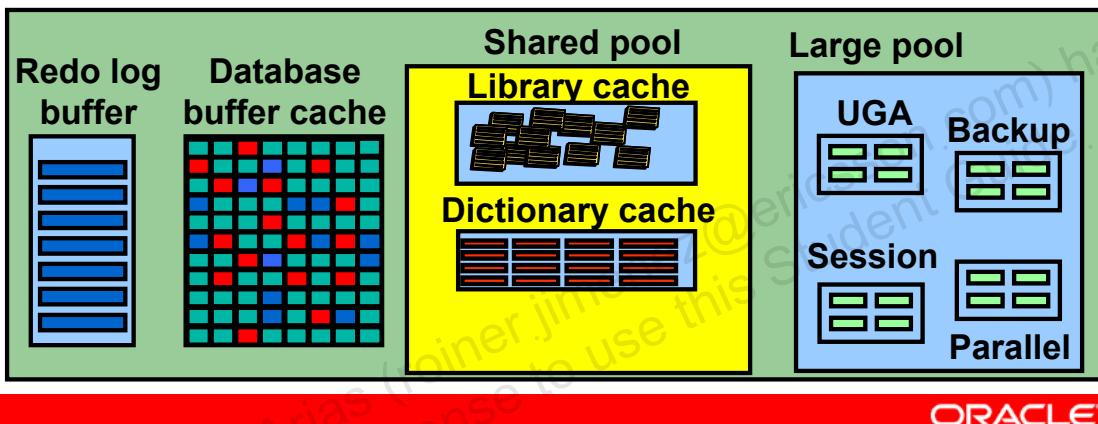
The data dictionary cache holds definitions of dictionary objects in memory.

#### User Global Area

The UGA contains the session information for the Oracle shared server. The UGA is located in the shared pool when using a shared server session and if the large pool is not configured.

# Large Pool

- Can be configured as a separate memory area in the SGA
- Is sized by the `LARGE_POOL_SIZE` parameter
- Is used to store data in memory for:
  - UGA
  - Backup and restore operations
  - Parallel query messaging



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Large Pool

### Existence of the Large Pool

The large pool must be explicitly configured. The memory of the large pool does not come out of the shared pool, but directly out of the SGA, thus adding to the amount of shared memory that the Oracle server needs for an instance at startup.

### Advantages of the Large Pool

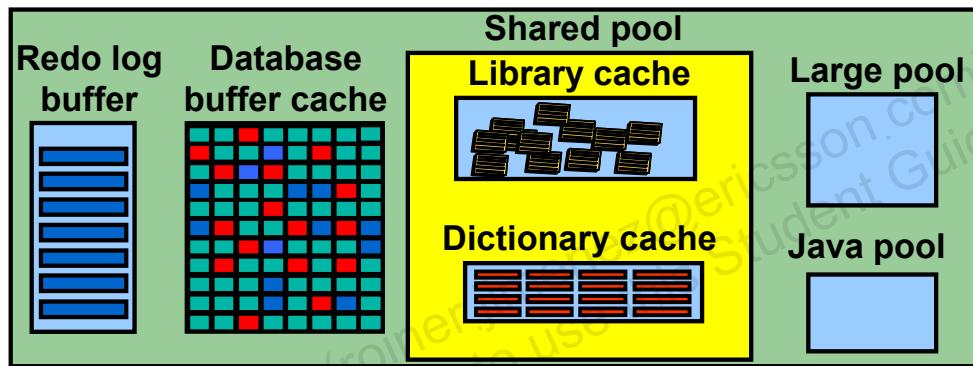
The large pool is used to provide large allocations of session memory for:

- I/O server processes
- Backup and restore operations
- Oracle shared server processes and the Oracle XA interface (used where transactions interact with more than one database)

By allocating session memory from the large pool for the Oracle shared server, the shared pool has less fragmentation that would come from having large objects frequently allocated and deallocated in it. Segregating large objects out of the shared pool results in more efficient shared pool usage, which means more of its memory is available to service new requests, and to retain existing data if needed.

# Java Pool

- Can be configured as a separate memory area in the SGA
- Is sized by the JAVA\_POOL\_SIZE parameter
- Is used to store data in memory for all session-specific Java code and data within the JVM



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Java Pool

The Java pool is a structure in the SGA that is used for all session-specific Java code and data within the Java engine.

### Shared Pool

Shared pool memory is used by the class loader within the JVM. The class loader uses about 8 KB of memory per loaded class. The shared pool is also used when compiling Java source code in the database or when using Java resource objects in the database. Shared pool memory is also consumed when you create call specifications and as the system tracks dynamically loaded Java classes at run time.

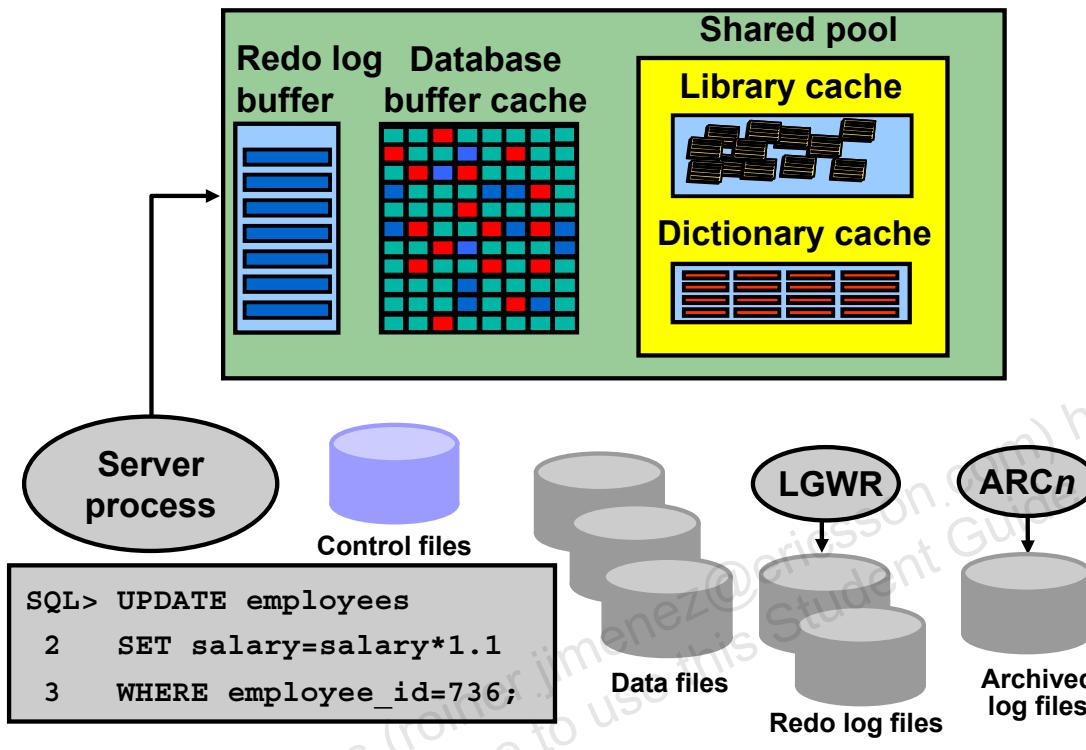
### Java Pool

The OracleJVM memory manager allocates all other Java states during run-time execution from the Java pool, including the shared in-memory representation of Java method and class definitions, as well as the Java objects migrated to session space at end-of-call.

Java pool memory is used in different ways, depending on whether the Oracle database server is using shared servers or not.

For more information about Java pool memory usage, refer to the *Oracle Database Java Developer's Guide*.

# Redo Log Buffer



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Redo Log Buffer

The Oracle server processes copy redo entries from the user's memory space to the redo log buffer for each DML or DDL statement. The redo entries contain the information necessary to reconstruct or redo changes made to the database by DML and DDL operations. They are used for database recovery and take up continuous sequential space in the buffer.

The redo log buffer is a circular buffer; the server processes can copy new entries over the entries in the redo log buffer that have already been written to disk. The LGWR process normally writes fast enough to ensure that space is always available in the buffer for new entries. The LGWR process writes the redo log buffer to the active online redo log file (or members of the active group) on disk. The LGWR process copies to disk all redo entries that have been entered into the buffer since the last time LGWR wrote to disk.

### What Causes LGWR to Write?

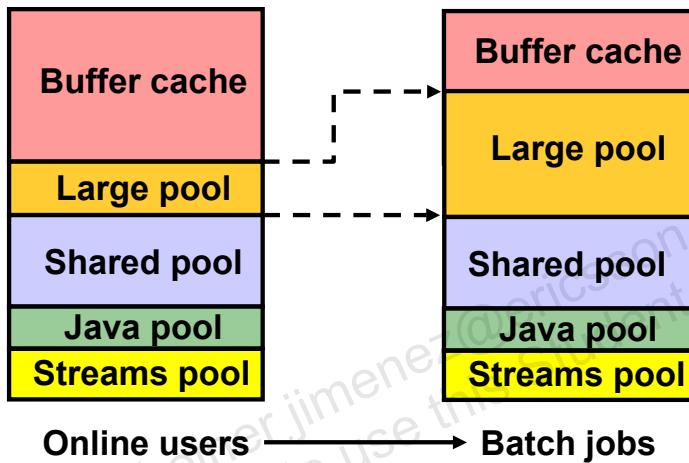
LGWR writes out the redo data from the redo log buffer:

- When a user process commits a transaction
- Every three seconds or when the redo log buffer is one-third full or contains at least 1 MB of data
- When a DBW $n$  process writes modified buffers to disk, if the corresponding redo log data has not already been written to disk

# Automatic Shared Memory Management: Overview

- Automatically adapts to workload changes
- Maximizes memory utilization
- Helps eliminate out-of-memory errors

Example:



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Automatic Shared Memory Management: Overview

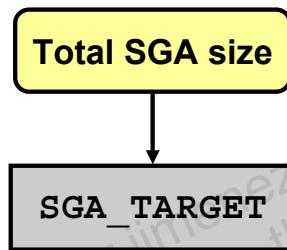
Automatic Shared Memory Management (ASMM) is another key self-management enhancement in the Oracle database. This functionality automates the management of the most important shared memory structures used by an Oracle database instance, and relieves you of having to configure these components manually. Besides making more effective use of available memory and thereby reducing the cost incurred for acquiring additional hardware memory resources, the ASMM feature significantly simplifies Oracle database administration by introducing a more dynamic, flexible, and adaptive memory management scheme.

For example, in a system that runs large online transactional processing (OLTP) jobs during the day (requiring a large buffer cache) and runs parallel batch jobs at night (requiring a large value for the large pool), you would have to simultaneously configure both the buffer cache and the large pool to accommodate your peak requirements.

With ASMM, when the OLTP job runs, the buffer cache grabs most of the memory to allow for good I/O performance. When the data analysis and reporting batch job starts up later, the memory is automatically migrated to the large pool so that it can be used by parallel query operations without producing memory overflow errors.

# Benefits of Automatic Shared Memory Management

`DB_CACHE_SIZE`  
`SHARED_POOL_SIZE`  
`LARGE_POOL_SIZE`  
`JAVA_POOL_SIZE`  
`STREAMS_POOL_SIZE`



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Benefits of Automatic Shared Memory Management

Automatic Shared Memory Management simplifies the configuration of the System Global Area (SGA). In the past, you needed to manually specify the amount of memory to be allocated for the database buffer cache, shared pool, Java pool, large pool, and Streams pool. It is often a challenge to size these components optimally. Undersizing can lead to poor performance and out-of-memory errors (ORA-4031), whereas oversizing can waste memory.

This feature enables you to specify a total memory amount to be used for all SGA components. The Oracle database periodically redistributes memory between the components in the slide according to workload requirements.

In earlier releases, you did not have exact control over the total size of the SGA because memory was allocated for the fixed SGA and for other internal metadata allocations over and above the total size of the user-specified SGA parameters. This additional memory was usually between 10 and 20 MB.

The SGA size initialization parameter (`SGA_TARGET`) includes all memory in the SGA, including the automatically sized components, the manually sized components, and any internal allocations during startup.

## How ASMM Works

- **ASMM is based on workload information that MMAN captures in the background.**
- **MMAN uses memory advisors.**
- **Memory is moved to where it is needed the most.**
- **If an SPFILE is used (which is recommended):**
  - Component sizes are saved across shutdowns.
  - Saved values are used to bootstrap component sizes.
  - There is no need to relearn optimal values.

ORACLE®

Copyright © 2008, Oracle. All rights reserved.

### How ASMM Works

The Automatic Shared Memory Management feature uses a background process named Memory Manager (MMAN). MMAN serves as the SGA memory broker and coordinates the sizing of the memory components. The SGA memory broker keeps track of the sizes of the components and pending resize operations.

The SGA memory broker observes the system and workload in order to determine the ideal distribution of memory. It is never complacent and performs this check every few minutes so that memory can always be present where needed. In the absence of Automatic Shared Memory Management, components had to be sized to anticipate their individual worst-case memory requirements.

On the basis of workload information, Automatic Shared Memory Management:

- Captures statistics periodically in the background
- Uses memory advisors
- Performs what-if analysis to determine the best distribution of the memory
- Moves memory to where it is most needed
- Saves component sizes across shutdown if an SPFILE is used (the sizes can be resurrected from before the last shutdown)

# Configuring ASMM by Using Database Control

The screenshot shows the Oracle Enterprise Manager 10g Database Control interface. The title bar reads "ORACLE Enterprise Manager 10g Database Control". The main menu includes "Setup", "Preferences", "Help", "Logout", and "Database". The current page is "Memory Parameters". A sub-menu "SGA" is selected, indicated by a red box around the "Enable" button. The page content includes:

- Automatic Shared Memory Management:** Disabled (button highlighted with a red box).
- SGA Components (MB):**
  - Shared Pool: 60
  - Buffer Cache: 24
  - Large Pool: 8
  - Java Pool: 48
  - Other (MB): 1
- Total SGA (MB):** 161
- SGA Distribution Pie Chart:**

| Pool         | Percentage |
|--------------|------------|
| Shared Pool  | 49.6%      |
| Buffer Cache | 14.9%      |
| Large Pool   | 5%         |
| Java Pool    | 29.8%      |
| Other        | 0.8%       |
- Maximum SGA Size:** The Maximum SGA Size specifies how much memory is allocated when the database starts up. It can be dynamically changed. The current value is 164 MB.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Configuring ASMM by Using Database Control

To configure Automatic Shared Memory Management by using Database Control, perform the following steps:

1. Click the Administration tab.
2. Select Memory Parameters under the Instance heading.
3. Click the SGA tab.
4. Click the Enable button for Automatic Shared Memory Management, and then enter the total SGA size (in MB).

When you use Database Control to enable ASMM, statements to change the autotuned parameters are automatically issued. All of these parameters, except for DB\_CACHE\_SIZE, are set to zero to indicate that they have no minimum size. DB\_CACHE\_SIZE is set to 4 MB in order to define a minimum such that enough of the SYSTEM tablespace can be resident in memory. Without these changes, the previously set values would be interpreted as (relatively high) minimum sizes, most likely putting an unnecessary burden on the memory-sizing algorithm.

**Note:** When you click Enable, you can enter a value for SGA\_TARGET.

## Manually Configuring ASMM

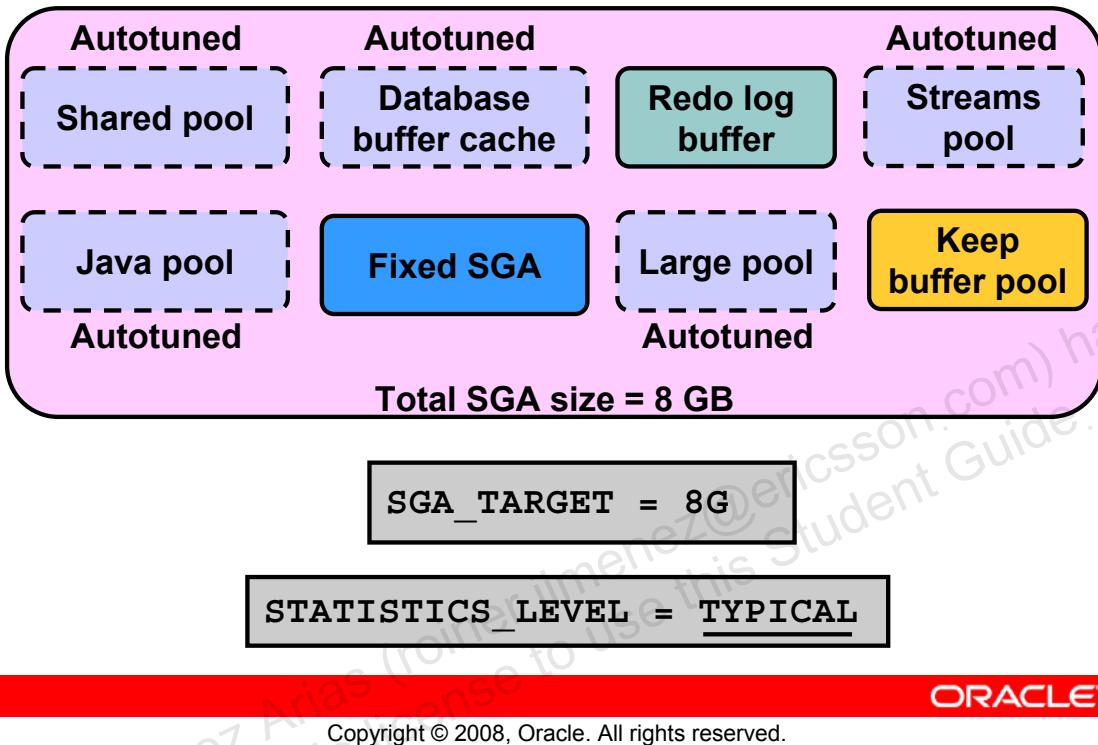
**To have minimum sizes for some of the memory components, you should manually configure those component sizes:**

- If the automatically computed sizes do not serve your purpose for some reason
- If there are short peaks or valleys in required memory that are not detected by ASMM
- If you simply want a guaranteed minimum amount of memory for a specific component



Copyright © 2008, Oracle. All rights reserved.

# Manually Configuring ASMM



## Manually Configuring ASMM

You can manually configure Automatic Shared Memory Management by using the **SGA\_TARGET** initialization parameter. By default, **SGA\_TARGET** is set to 0, which means that Automatic Shared Memory Management is disabled and you must manually configure the database memory.

## Manually Configuring ASMM (continued)

If you specify a nonzero value for SGA\_TARGET, the following five memory pools are automatically sized: database buffer cache (default pool), shared pool, large pool, Streams pool, and the Java pool.

The parameters used to configure these memory pools (for example, SHARED\_POOL\_SIZE) are referred to as autotuned parameters.

The following buffers are referred to as manually sized components:

- Log buffer
- Other buffer caches (KEEP/RECYCLE, nondefault block size)
- Fixed SGA and other internal allocations

**Note:** Automatic Shared Memory Management requires that STATISTICS\_LEVEL be set to TYPICAL or ALL.

## Behavior of Autotuned SGA Parameters

- When **SGA\_TARGET** is not set or is set to zero:
  - Autotuned parameters behave as normal
  - **SHARED\_POOL\_SIZE** may need to be increased from settings used in earlier database versions

```
SELECT SUM(bytes)/1024/1024 size_mb
FROM v$sgastat WHERE pool = 'shared pool';
```

- When **SGA\_TARGET** is set to a nonzero value:
  - Default value of autotuned parameters is zero
  - The specified value is used as a minimum size

```
SELECT component, current_size/1024/1024 size_mb
FROM v$sga_dynamic_components;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Behavior of Autotuned SGA Parameters

When **SGA\_TARGET** is not set or is equal to zero, autotuned SGA parameters behave as normal: They specify the actual size of those components.

When Automatic Shared Memory Management is enabled, you can still specify values for the autotuned SGA parameters, but the values carry a different meaning. If you set any of the autotuned parameters to a nonzero value, the specified value is used as a lower bound by the autotuning algorithm. For example, if **SGA\_TARGET** is set to 8 GB and **SHARED\_POOL\_SIZE** is set to 1 GB, the Automatic Shared Memory Management algorithm does not shrink the shared pool to below 1 GB. Use the second query given in the slide to determine the actual size of the autotuned components in the SGA.

# Behavior of Manually Tuned SGA Parameters

- Some components are not autotuned.
  - KEEP and RECYCLE buffer caches
  - Multiple block size caches
- These components must be manually configured using database parameters.
- The memory used by these components reduces the amount of memory available for autotuning the SGA.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Behavior of Manually Tuned SGA Parameters

The manually tuned SGA parameters are:

- DB\_KEEP\_CACHE\_SIZE
- DB\_RECYCLE\_CACHE\_SIZE
- DB\_nK\_CACHE\_SIZE ( $n = 2, 4, 8, 16, 32$ )

If you want to use memory components that are not autotuned, you must configure the appropriate parameters. The values specified for these parameters precisely control the sizes of the memory components.

When SGA\_TARGET is set, the total size of manual SGA size parameters is subtracted from the SGA\_TARGET value, and the balance is given to the autotuned SGA components.

For example, if SGA\_TARGET is set to 8 GB and DB\_RECYCLE\_CACHE\_SIZE is set to 1 GB, then the total size of the five autotuned components (shared pool, Java pool, default buffer cache, Streams pool, and large pool) is limited to 7 GB. The 7 GB size includes the fixed SGA and log buffer, and only after memory has been allocated to them is the rest of the memory divided between the autotuned components. The size of the recycle cache is 1 GB, as specified by the parameter.

## Using the V\$PARAMETER View

```
SGA_TARGET = 8G
```

```
DB_CACHE_SIZE = 0
JAVA_POOL_SIZE = 0
LARGE_POOL_SIZE = 0
SHARED_POOL_SIZE = 0
STREAMS_POOL_SIZE = 0
```

```
SELECT name, value, isdefault
FROM v$parameter
WHERE name LIKE '%size';
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Using the V\$PARAMETER View

When you specify a nonzero value for SGA\_TARGET and do not specify a value for an autotuned SGA parameter, the value of the autotuned SGA parameters in the V\$PARAMETER view is 0, and the value of the ISDEFAULT column is TRUE.

If you have specified a value for any of the autotuned SGA parameters, the value that is displayed when you query V\$PARAMETER is the value that you specified for the parameter.

# Modifying the SGA\_TARGET Parameter

- **The SGA\_TARGET initialization parameter:**
  - Is dynamic
  - Can be increased up to SGA\_MAX\_SIZE
  - Can be reduced until all components reach their minimum size
- **A change in the value of SGA\_TARGET affects only automatically sized components.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Modifying the SGA\_TARGET Parameter

SGA\_TARGET is a dynamic parameter and can be changed through Database Control or with the ALTER SYSTEM command.

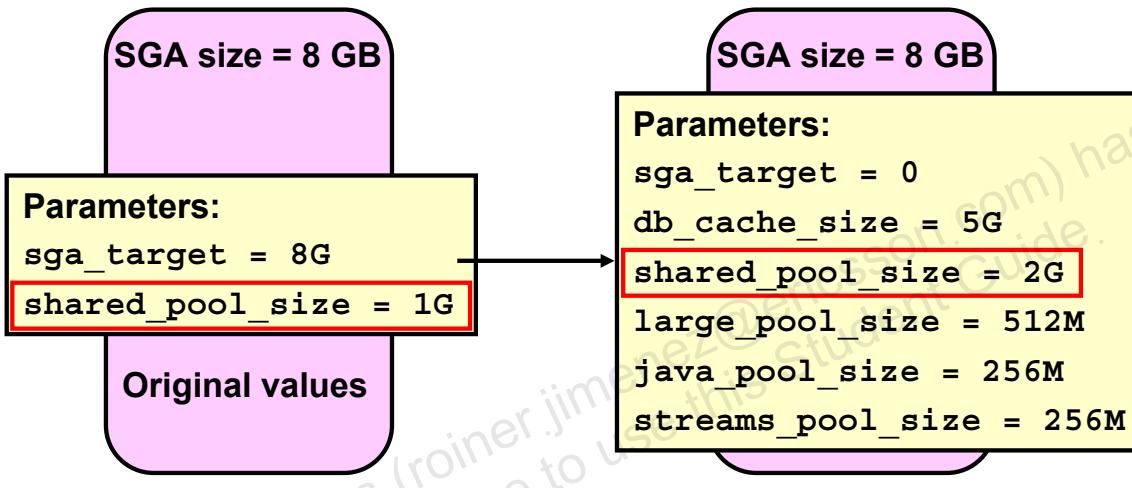
SGA\_MAX\_SIZE is the maximum amount of memory that can be allocated to the SGA. It cannot be changed without restarting the database. SGA\_TARGET can be increased up to the value of SGA\_MAX\_SIZE. It can be reduced until any one of the autotuned components reaches its minimum size: either a user-specified minimum or an internally determined minimum.

- If you increase the value of SGA\_TARGET, the additional memory is distributed according to the autotuning policy across the autotuned components.
- If you reduce the value of SGA\_TARGET, the memory is taken away by the autotuning policy from one or more of the autotuned components.

Suppose that SGA\_MAX\_SIZE is set to 10 GB and SGA\_TARGET is set to 8 GB. If DB\_KEEP\_CACHE\_SIZE is set to 1 GB and you increase SGA\_TARGET to 9 GB, then the additional 1 GB is distributed only among the components controlled by SGA\_TARGET. The value of DB\_KEEP\_CACHE\_SIZE is not affected. Likewise, if SGA\_TARGET is reduced to 7 GB, then the 1 GB is taken from only those components that are controlled by SGA\_TARGET. This decrease does not affect the settings of manually controlled parameters such as DB\_KEEP\_CACHE\_SIZE.

# Disabling ASMM

- Setting SGA\_TARGET to zero disables autotuning.
- Autotuned parameters are set to their current sizes.
- The SGA size as a whole is unaffected.



Copyright © 2008, Oracle. All rights reserved.

## Disabling ASMM

You can dynamically choose to disable Automatic Shared Memory Management by setting SGA\_TARGET to zero. In this case, the values of all the autotuned parameters are set to the current sizes of the corresponding components, even if the user had earlier specified a different nonzero value for an autotuned parameter.

In the example in the slide, the value of SGA\_TARGET is 8 GB and the value of SHARED\_POOL\_SIZE is 1 GB. If the system has internally adjusted the size of the shared pool component to 2 GB, then setting SGA\_TARGET to zero results in SHARED\_POOL\_SIZE being set to 2 GB, thereby overriding the original user-specified value.

# Manually Resizing Dynamic SGA Parameters

- **For autotuned parameters, manual resizing:**
  - Results in immediate component resize if the new value is greater than the current size
  - Changes the minimum size if the new value is smaller than the current size
- **Resizing manually tuned parameters affects only the tunable portion of the SGA.**

ORACLE

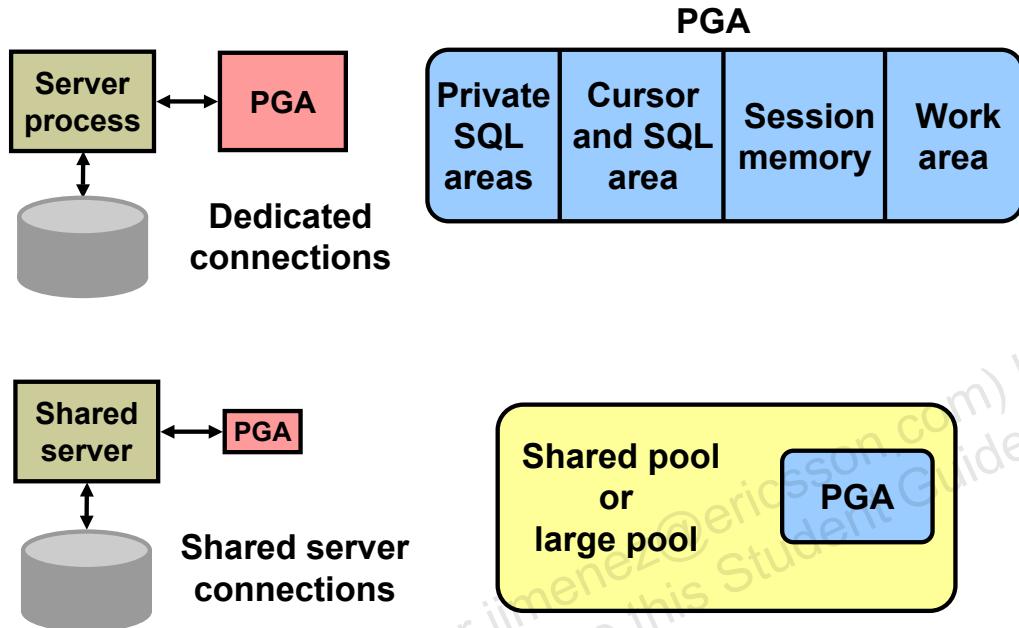
Copyright © 2008, Oracle. All rights reserved.

## Manually Resizing Dynamic SGA Parameters

When an autotuned parameter is resized and SGA\_TARGET is set, the resizing results in an immediate change to the size of the component only if the new value is larger than the current size of the component. For example, if you set SGA\_TARGET to 8 GB and set SHARED\_POOL\_SIZE to 2 GB, ensure that the shared pool has at least 2 GB at all times to accommodate the necessary memory allocations. After this, adjusting the value of SHARED\_POOL\_SIZE to 1 GB has no immediate effect on the size of the shared pool. It allows the automatic memory-tuning algorithm to later reduce the shared pool size to 1 GB if it needs to. Alternatively, if the size of the shared pool is 1 GB to begin with, then adjusting the value of SHARED\_POOL\_SIZE to 2 GB results in the shared pool component growing immediately to a size of 2 GB. The memory used in this resize operation is taken away from one or more autotuned components, and the sizes of the manual components are not affected.

Parameters for manually sized components can be dynamically altered as well, but the difference is that the value of the parameter specifies the precise size of that component immediately. Therefore, if the size of a manual component is increased, extra memory is taken away from one or more automatically sized components. If the size of a manual component is decreased, the memory that is released is given to the automatically sized components.

# Program Global Area (PGA)



**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Program Global Area (PGA)

The Program Global Area (PGA) is a memory region that contains data and control information for a server process. It is nonshared memory created by the Oracle server when a server process is started. Access to it is exclusive to that server process. The total PGA memory allocated by all server processes attached to an Oracle instance is also referred to as the *aggregated PGA* memory allocated by the instance.

Part of the PGA can be located in the SGA when using shared servers.

PGA memory typically contains the following:

### Private SQL Area

A private SQL area contains data such as bind information and run-time memory structures. This information is specific to each session's invocation of the SQL statement; bind variables hold different values, and the state of the cursor is different, among other things. Each session that issues a SQL statement has a private SQL area. Each user that submits the same SQL statement has his or her own private SQL area that uses a single shared SQL area. Thus, many private SQL areas can be associated with the same shared SQL area. The location of a private SQL area depends on the type of connection established for a session. If a session is connected through a dedicated server, private SQL areas are located in the server process's PGA. However, if a session is connected through a shared server, part of the private SQL area is kept in the SGA.

## Program Global Area (continued)

### Cursor and SQL Areas

The application developer of an Oracle precompiler program or OCI program can explicitly open *ursors* or handles to specific private SQL areas, and use them as a named resource throughout the execution of the program. Recursive cursors that the database issues implicitly for some SQL statements also use shared SQL areas.

### Work Area

For complex queries (for example, decision support queries), a big portion of the PGA is dedicated to work areas allocated by memory-intensive operators, such as:

- Sort-based operators, such as ORDER BY, GROUP BY, ROLLUP, and window functions
- Hash-join
- Bitmap merge
- Bitmap create
- Write buffers used by bulk load operations

A sort operator uses a work area (the sort area) to perform the in-memory sort of a set of rows. Similarly, a hash-join operator uses a work area (the hash area) to build a hash table from its left input.

The size of a work area can be controlled and tuned. Generally, bigger work areas can significantly improve the performance of a particular operator at the cost of higher memory consumption.

### Session Memory

Session memory is the memory allocated to hold a session's variables (logon information) and other information related to the session. For a shared server, the session memory is shared and not private.

# Automatic PGA Memory Management

- **Dynamically adjusts the amount of PGA memory dedicated to work areas, on the basis of the `PGA_AGGREGATE_TARGET` parameter**
- **Helps maximize the performance of all the memory-intensive SQL operations**
- **Is enabled by default**

ORACLE®

Copyright © 2008, Oracle. All rights reserved.

## Automatic PGA Memory Management

Ideally, the size of a work area is big enough that it can accommodate the input data and auxiliary memory structures allocated by its associated SQL operator. This is known as the optimal size of a work area. When the size of the work area is smaller than optimal, the response time increases because an extra pass is performed over part of the input data.

Automatic PGA memory management simplifies and improves the way PGA memory is allocated. By default, PGA memory management is enabled. In this mode, the Oracle database dynamically adjusts the size of the portion of the PGA memory dedicated to work areas, based on 20% of the SGA memory size. The minimum value is 10 MB.

When running in automatic PGA memory management mode, sizing of work areas for all sessions becomes automatic and the `*_AREA_SIZE` parameters (such as `SORT_AREA_SIZE`) are ignored by all sessions running in that mode. At any given time, the total amount of PGA memory available to active work areas in the instance is automatically derived from the `PGA_AGGREGATE_TARGET` initialization parameter. This amount is set to the value of `PGA_AGGREGATE_TARGET` minus the amount of PGA memory allocated by other components of the system (for example, PGA memory allocated by sessions). The resulting PGA memory is then assigned to individual active work areas on the basis of their specific memory requirements.

# PGA Management Resources

- **Statistics to manage the PGA\_AGGREGATE\_TARGET initialization parameter, such as PGA cache hit percentage**
- **Views for monitoring the PGA work area include:**
  - v\$sql\_workarea\_histogram
  - v\$pgastat
  - v\$sql\_workarea\_active
  - v\$sql\_workarea
  - v\$tempseg\_usage
- **Views to assist in sizing the PGA work area are:**
  - v\$pga\_target\_advice
  - v\$pga\_target\_advice\_histogram

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## PGA Management Resources

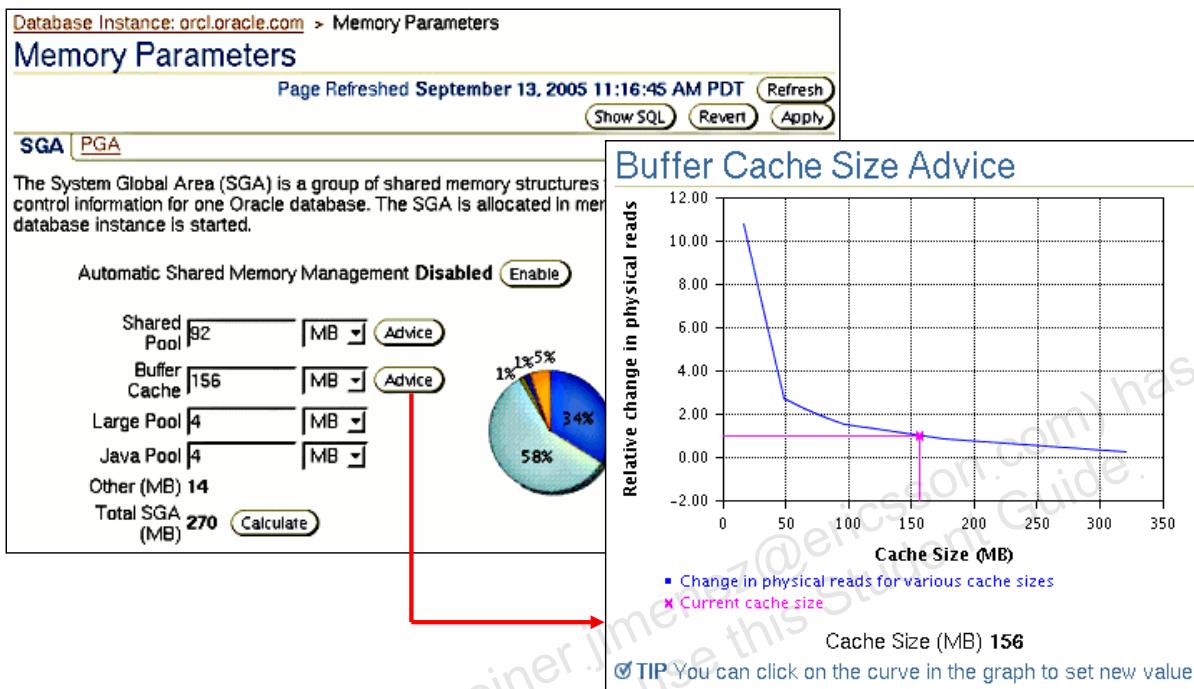
When configuring a new instance, it is difficult to know precisely the appropriate setting for PGA\_AGGREGATE\_TARGET. You can determine this setting in three stages:

1. Make the first estimate for PGA\_AGGREGATE\_TARGET, based on convention. By default, the Oracle database sets this to 20% of the SGA size. However, this initial setting may be too low for a large DSS system.
2. Run a representative workload on the instance and monitor performance, using PGA statistics collected by Oracle, to see whether the maximum PGA size is underconfigured or overconfigured.
3. Tune PGA\_AGGREGATE\_TARGET, using Oracle PGA advice statistics.

For backward compatibility, automatic PGA memory management can be disabled by setting the value of the PGA\_AGGREGATE\_TARGET initialization parameter to 0. When automatic PGA memory management is disabled, the maximum size of a work area can be sized with the associated \*\_AREA\_SIZE parameter, for example:

- SORT\_AREA\_SIZE
- HASH\_AREA\_SIZE
- BITMAP\_MERGE\_AREA\_SIZE
- CREATE\_BITMAP\_AREA\_SIZE

# Using the Memory Advisor to Size the SGA



**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Using the Memory Advisor to Size the SGA (continued)

With the Memory Advisor, you can tune the size of your memory structures. If Automatic Shared Memory Management is enabled, you can use this to tune the total size of the SGA. If ASMM is disabled, you can use this advisor to tune the different components of the SGA.

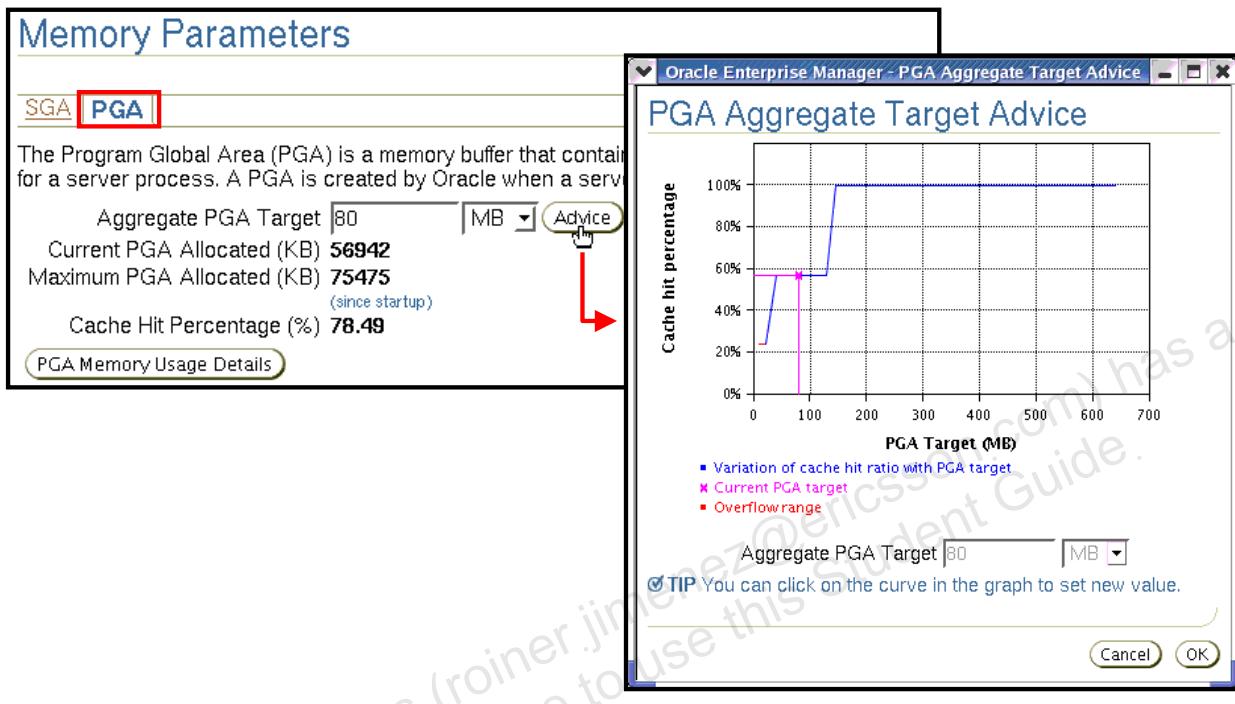
The Memory Advisor comprises three advisors that give you recommendations on the following SGA memory structures:

- Shared pool
- Buffer cache

You can invoke the memory advisors by performing the following steps:

1. Click Advisor Central in the Related Links region on the Database home page.
2. Click Memory Advisor on the Advisor Central page. The Memory Parameters page appears. This page provides a breakdown of memory usage for the SGA.
3. Click Advice next to the Shared Pool value or Buffer Cache value to invoke the respective advisors.

# Using the Memory Advisor to Size the PGA



Copyright © 2008, Oracle. All rights reserved.

ORACLE

## Using the Memory Advisor to Size the PGA

The Memory Advisor can be used to get advice on the size of the PGA. To do this, click the PGA tab, and then click Advice. In this example, the graph shows that raising the Aggregate PGA Target from 80 MB to approximately 140 MB will raise the cache hit percentage from 60% to close to 100%.

## Efficient Memory Usage: Guidelines

- Try to fit the SGA into physical memory.
- Tune for a high buffer cache hit ratio, with the following caveats:
  - Even valid and necessary full table scans lower it.
  - It is possible that unnecessary repeated reads of the same blocks are artificially raising it.
- Use the Memory Advisor.

ORACLE®

Copyright © 2008, Oracle. All rights reserved.

### Efficient Memory Usage: Guidelines

If possible, it is best to fit the SGA into physical memory, which provides the fastest access. Even though the operating system may provide additional virtual memory, that memory, by its nature, can often be swapped out to the disk. On some platforms, you can use the `LOCK_SGA` initialization parameter to lock the SGA into physical memory.

When a SQL statement executes, Oracle data blocks are requested for reading or writing, or both. This is called a logical I/O. As the block is requested, the Oracle database first checks to see whether it already exists in memory. If it is not in memory, it reads it from the disk, which is called a physical I/O. The number of times the block is found already in memory compared to the total number of logical I/Os is referred to as the buffer cache hit ratio. A higher ratio is usually better because that means more blocks are being found in memory without incurring disk I/O.

It is not uncommon to have a buffer cache hit ratio above 99%. But that does not always mean the system is well tuned. If there is a query that is executed more often than necessary, and it constantly requests the same blocks over and over again, the ratio is raised. If it is an inefficient or unnecessary query, then it artificially inflates the ratio. This is because it should not execute in that manner or that often in the first place.

## **Efficient Memory Usage: Guidelines (continued)**

Also, consider the fact that large full table scans (a full reading of the entire table) can lower this ratio because the entire table may be read from the disk; the scan may not take advantage of the fact that some of the blocks may be in the buffer cache. So, if there are some necessary large full table scans in your application, your well-tuned database may always have a low database buffer cache hit ratio.

Use Enterprise Manager's Memory Advisor. This can help you size the SGA on the basis of the activity in your particular database.

## Memory Tuning Guidelines for the Library Cache

- Establish formatting conventions for developers so that SQL statements match in the cache.
- Use bind variables.
- Eliminate unnecessary duplicate SQL.
- Consider using CURSOR\_SHARING.
- Use PL/SQL when possible.
- Cache sequence numbers.
- Pin objects in the library cache.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Memory Tuning Guidelines for the Library Cache

The library cache, which is a part of the shared pool, is where the Oracle database stores all the SQL, PL/SQL, and Java code that get executed. The code goes into this central location so that it can be shared among all users. The benefit of sharing is that all users can take advantage of any work that is already done on behalf of SQL. Therefore, tasks such as parsing the statement and determining the data access path (also known as the “explain plan”) are done only once per statement, no matter how many times the statement is executed, and no matter how many users execute it. A library cache that is too small does not have room for all the statements being executed and, therefore, you cannot take advantage of this sharing of work for some statements. A library cache that is too large causes a burden on the system just to manage its contents.

A library cache may end up being filled with what appear to be different statements when, in fact, they are copies of the same statement. A common cause of this is having slightly different formatting for each statement. There is no match if the string does not compare exactly. Another cause is the use of literals instead of bind variables. If the only difference between two statements is literal values, then, in most cases, each of those statement executions and the overall system would benefit from replacing those literals with bind variables.

## **Memory Tuning Guidelines for the Library Cache (continued)**

The CURSOR\_SHARING initialization parameter can be set to have the system automatically replace literals with bind variables when statements otherwise match. You should typically take advantage of this setting as a temporary measure until the application is corrected to use bind variables where appropriate. As with all of these guidelines, the use of this variable can have other side effects, which you should investigate.

Rather than having the same SQL statement issued from several different places in an application, put the statement or statements into a stored procedure by using PL/SQL. Then just call the procedure. This guarantees that the SQL statement is shared because it exists only in one location. Also, the SQL is already parsed and has an explain plan because it is in an already compiled stored procedure.

Sequence numbers can be cached. Therefore, if there are some sequences with high activity, determine a good setting for the cache size and take advantage of it.

You can use the DBMS\_SHARED\_POOL package to pin objects in the library cache. This reduces the chance of reloading and recompiling objects. Refer to the *PL/SQL Packages and Types Reference* document for more information about how to use that package.

# Summary

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

- **Describe the memory components in the SGA**
- **Implement Automatic Shared Memory Management**
- **Manually configure SGA parameters**
- **Use automatic PGA memory management**



Copyright © 2008, Oracle. All rights reserved.

# Practice Overview: Using ASMM to Correct a Memory Allocation Problem

This practice covers the following topics:

- Diagnosing a memory allocation problem
- Enabling Automatic Shared Memory Management

ORACLE®

Copyright © 2008, Oracle. All rights reserved.

# 9

## Automatic Performance Management

ORACLE®

Copyright © 2008, Oracle. All rights reserved.

# Objectives

Tuning  
Statistics  
SGA Attach  
AWR  
Advisors  
Async. Commit

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

- **Diagnose database performance issues**
- **Configure the Automatic Workload Repository**
- **Access the database advisors**
- **Use the SQL Access Advisor to improve database performance**
- **Use asynchronous COMMIT effectively**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

# Tuning Activities

**The three activities in performance management are:**

- **Performance planning**
- **Instance tuning**
- **SQL tuning**



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Tuning Activities

The three facets of tuning involve performance planning, instance tuning, and SQL tuning.

- Performance planning is the process of establishing the environment: the hardware, software, operating system, network infrastructure, and so on.
- Instance tuning is the actual adjustment of Oracle database parameters and operating system parameters to gain better performance of the Oracle database.
- SQL tuning involves making your application submit efficient SQL. This is an applicationwide, as well as a statement-specific, consideration. Systemwide, you want to be sure that different parts of the application are taking advantage of each other's work and are not competing for resources unnecessarily. In this lesson, you learn about some common actions that you can take to tune specific SQL statements.

**Note:** For more information about performance tuning, refer to the *Oracle Database Performance Tuning Guide*.

# Performance Planning

- **Investment options**
- **System architecture**
- **Scalability**
- **Application design principles**
- **Workload testing, modeling, and implementation**
- **Deploying new applications**



ORACLE®

Copyright © 2008, Oracle. All rights reserved.

## Performance Planning

There are many facets to performance planning. You have to consider the investment in your system architecture: the hardware and software infrastructure needed to meet your requirements. The number of hard drives and controllers has an impact on the speed of data access. Striping data across many devices is often a way to make performance gains. This, of course, requires analysis to determine the value for your given environment, application, and performance requirements.

The ability of an application to scale is also important. This means that you are able to handle more and more users, clients, sessions, or transactions, without incurring a huge impact on overall system performance. The most obvious violator against scalability is serializing operations among users. If all users have to go through a single path one at a time, then as more users are added, there are definitely adverse effects on performance. This is because more and more users line up to go through that path. Poorly written SQL also affects scalability. It requires many users to wait for inefficient SQL to complete; each user competing with the other on a large number of resources that they are not actually in need of.

The principles of application design can greatly affect performance. Simplicity of design, use of views and indexes, and data modeling are all very important.

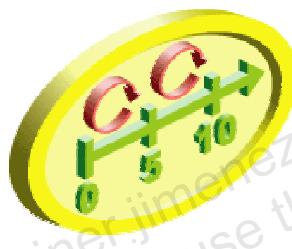
## Performance Planning (continued)

Any application must be tested under a representative production workload. This requires estimating database size and workload, and generating test data and system load.

Performance must be considered as new applications (or new versions of applications) are deployed. Sometimes design decisions are made to maintain compatibility with old systems during the rollout. A new database should be configured (on the basis of the production environment) specifically for the applications that it hosts.

# Instance Tuning

- **Have well-defined goals.**
- **Allocate memory to database structures.**
- **Consider I/O requirements in each part of the database.**
- **Tune the operating system for optimal performance of the database.**



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Instance Tuning

At the start of any tuning activity, it is necessary to have specific goals. A goal such as “Process 500 sales transactions per minute” is easier to work toward than one that says “Make it go as fast as you can, and we’ll know when it’s good enough.”

You must allocate Oracle database memory suitably for your application to attain optimum performance. You have a finite amount of memory to work with. Too little memory allotted to certain parts of the Oracle database can cause inefficient background activity, which you may not even be aware of without doing some analysis.

Disk I/O is often the bottleneck of a database and, therefore, requires a lot of attention at the outset of any database implementation.

The operating system configuration can also affect the performance of an Oracle database. For more information, see the *Oracle Database Installation Guide* for your particular platform.

# Performance Tuning Methodology

## The tuning steps:

- **Tune from the top down. Tune:**
  - The design before tuning the application code
  - The code before tuning the instance
- **Tune the area with the greatest potential benefit.**
  - Identify the longest waits.
  - Identify the largest service times.
- **Stop tuning when the goal is met.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Performance Tuning Methodology

Oracle has developed a tuning methodology based on years of experience. The basic steps are:

- Check the OS statistics and general machine health before tuning the instance to be sure that the problem is in the database.
- Tune from the top down. Start with the design, then the application, and then the instance. For example, try to eliminate the full tables scans causing the I/O contention before tuning the tablespace layout on disk.
- Tune the area with the greatest potential benefit. The tuning methodology presented in this course is simple. Identify the biggest bottleneck and tune it. Repeat this step. All the various tuning tools have some way to identify the SQL statements, resource contention, or services that are taking the most time. The Oracle database provides a time model and metrics to automate the process of identifying bottlenecks.
- Stop tuning when you meet your goal. This step implies that you set tuning goals.

This is a general approach to tuning the database instance and may require multiple passes.

# Statistics Collection

- **Performance tuning depends on the collection of accurate statistics.**
- **There are different types of statistics:**
  - Optimizer statistics
  - System statistics
- **There are different methods of collecting statistics:**
  - Automatically, through `GATHER_STATS_JOB`
  - Manually, with the `DBMS_STATS` package
  - By setting database initialization parameters
  - By importing statistics from another database

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Statistics Collection

Optimizer statistics are collections of data that describe more details about the database and the objects in the database. These statistics are used by the query optimizer to choose the best execution plan for each SQL statement.

System statistics describe the system's hardware characteristics, such as I/O and CPU performance and utilization, to the query optimizer. When choosing an execution plan, the optimizer estimates the I/O and CPU resources required for each query. System statistics enable the query optimizer to more accurately estimate I/O and CPU costs, and thereby choose a better execution plan. System statistics are collected using the `DBMS_STATS.GATHER_SYSTEM_STATS` procedure. When the Oracle database gathers system statistics, it analyzes system activity in a specified period of time. System statistics are not automatically gathered. Oracle Corporation recommends that you use the `DBMS_STATS` package to gather system statistics.

The recommended approach to gathering optimizer statistics is to allow the Oracle database to automatically gather the statistics. The `GATHER_STATS_JOB` job is created automatically at database creation time and is managed by the Scheduler. It gathers statistics on all objects in the database that have either missing or stale optimizer statistics.

## Statistics Collection (continued)

If you choose not to use automatic statistics gathering, then you must manually collect statistics in all schemas, including system schemas. If the data in your database changes regularly, you also need to gather statistics regularly to ensure that the statistics accurately represent characteristics of your database objects. To manually collect statistics, use the DBMS\_STATS package. This PL/SQL package is also used to modify, view, export, import, and delete statistics.

You can also manage optimizer and system statistics collection through database initialization parameters. For example:

- The OPTIMIZER\_DYNAMIC\_SAMPLING parameter controls the level of dynamic sampling performed by the optimizer. You can use dynamic sampling to estimate statistics for tables and relevant indexes when they are not available or are too out of date to trust. Dynamic sampling also estimates single-table predicate selectivities when collected statistics cannot be used or are likely to lead to significant errors in estimation.
- The STATISTICS\_LEVEL parameter controls all major statistics collections or advisories in the database and sets the statistics collection level for the database. The values for this parameter are BASIC, TYPICAL, and ALL.
- The TIMED\_STATISTICS parameter directs the Oracle server to gather wait time for events, in addition to wait counts already available. This data is useful for comparing the total wait time for an event to the total elapsed time between the performance data collections.
- The TIMED\_OS\_STATISTICS parameter specifies the interval (in seconds) at which Oracle collects operating system statistics when a request is made from the client to the server or when a request completes.

Timed system statistics are automatically collected for the database if the STATISTICS\_LEVEL initialization parameter is set to TYPICAL or ALL. If STATISTICS\_LEVEL is set to BASIC, then you must set TIMED\_STATISTICS to TRUE to enable the collection of timed statistics. Note that setting STATISTICS\_LEVEL to BASIC disables many automatic features and is not recommended.

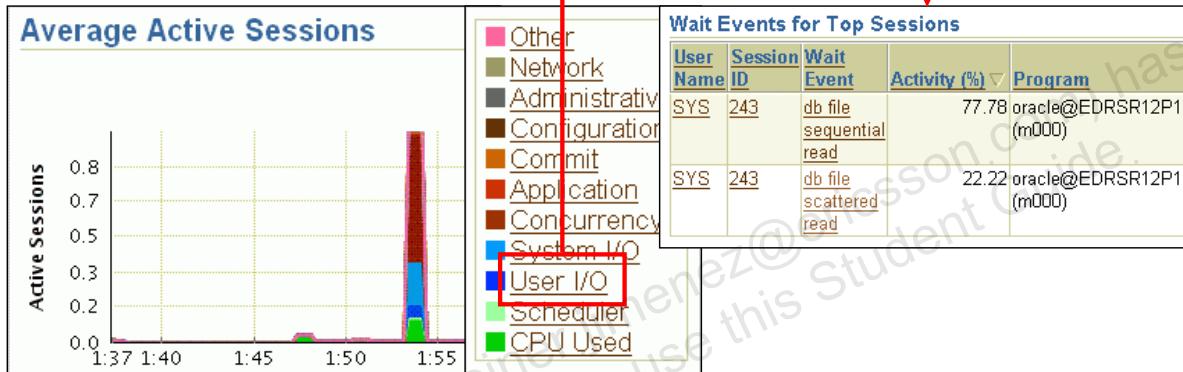
If you explicitly set TIMED\_STATISTICS or TIMED\_OS\_STATISTICS, either in the initialization parameter file or by using the ALTER SYSTEM or ALTER SESSION commands, then the explicitly set value overrides the value derived from STATISTICS\_LEVEL.

You can query the V\$STATISTICS\_LEVEL view to determine which parameters are affected by the STATISTICAL\_LEVEL parameter.

# Oracle Wait Events

Tuning  
 > Statistics  
 SGA Attach  
 AWR  
 Advisors  
 Async. Commit

- A collection of wait events provides information about the sessions or processes that had to wait or must wait for different reasons.
- These events are listed in the V\$EVENT\_NAME view.



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Oracle Wait Events

Wait events are statistics that are incremented by a server process or thread to indicate that it had to wait for an event to complete before being able to continue processing. Wait event data reveals various symptoms of problems that might be impacting performance, such as latch contention, buffer contention, and I/O contention. Remember that these are only symptoms of problems, not the actual causes.

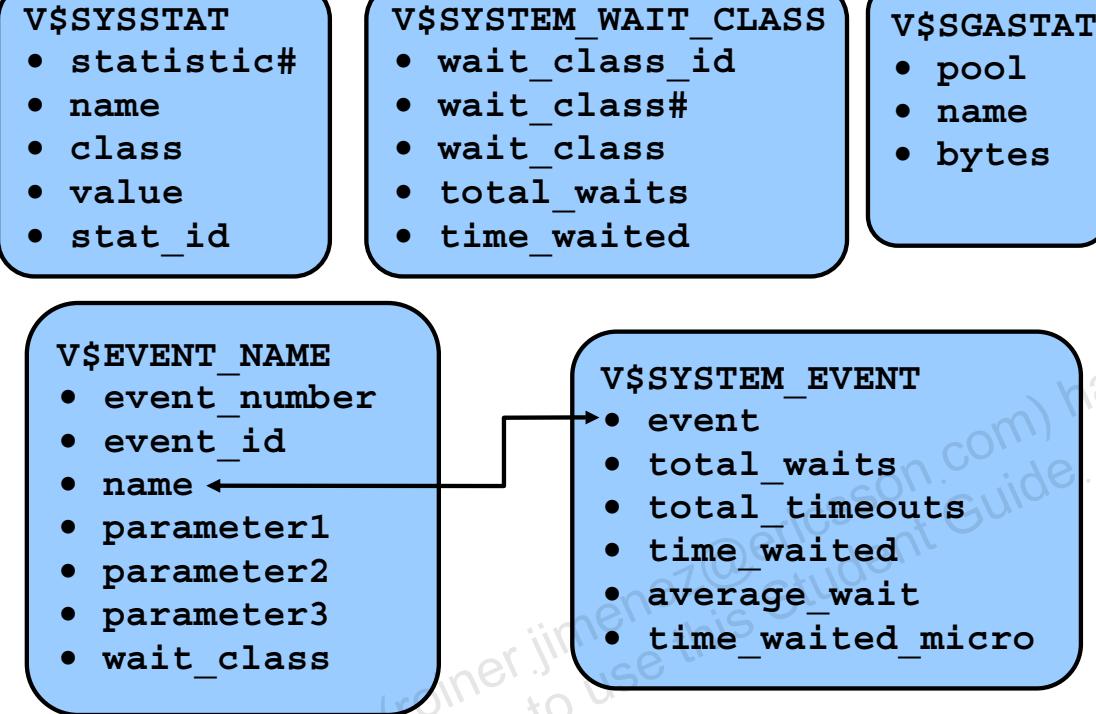
Wait events are grouped into classes. The wait event classes include: Administrative, Application, Cluster, Commit, Concurrency, Configuration, Idle, Network, Other, Scheduler, System I/O, and User I/O.

There are more than 800 wait events in the Oracle database, including free buffer wait, latch free, buffer busy waits, db file sequential read, and db file scattered read.

Using EM, you can view wait events by opening the Performance page and viewing the “Sessions: Waiting and Working” graph, as shown in the slide. By clicking the link for a particular wait event class, you can drill down to the specific wait events by using the Top Sessions interface. In this example, the most significant wait events were file reads.

For a list of the most common Oracle events, refer to the *Oracle Database Reference 10g* documentation.

# System Statistics



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## System Statistics

To effectively diagnose performance problems, statistics must be available. The Oracle database generates many types of cumulative statistics for the system, sessions, and individual SQL statements. The Oracle database also tracks cumulative statistics on segments and services. When analyzing a performance problem in any of these scopes, you typically look at the change in statistics (delta value) over the period of time you are interested in.

### Wait Events Statistics

All the possible wait events are cataloged in the V\$EVENT\_NAME view.

Cumulative statistics for all sessions are stored in V\$SYSTEM\_EVENT, which shows the total waits for a particular event since instance startup.

When you are troubleshooting, you need to know whether a process has waited for any resource.

### Systemwide Statistics

All the systemwide statistics are cataloged in the V\$STATNAME view: About 330 statistics are available in Oracle Database 10g.

The server displays all calculated system statistics in the V\$SYSSTAT view. You can query this view to find cumulative totals since the instance started.

## System Statistics (continued)

### Systemwide Statistics (continued)

For example:

```
SQL> SELECT name, class, value FROM v$sysstat;
NAME CLASS VALUE

...
table scans (short tables) 64 135116
table scans (long tables) 64 250
table scans (rowid ranges) 64 0
table scans (cache partitions) 64 3
table scans (direct read) 64 0
table scan rows gotten 64 14789836
table scan blocks gotten 64 558542
...
...
```

Systemwide statistics are classified by the tuning topic and the debugging purpose. The classes include general instance activity, redo log buffer activity, locking, database buffer cache activity, and so on.

You can also view all wait events for a particular wait class by querying V\$SYSTEM\_WAIT\_CLASS. For example (with formatting applied):

```
SQL> SELECT * FROM V$SYSTEM_WAIT_CLASS
 2 WHERE wait_class LIKE '%I/O%';
CLASS_ID CLASS# WAIT_CLASS TOTAL_WAITS TIME_WAITED

1740759767 8 User I/O 1119152 39038
4108307767 9 System I/O 296959 27929
```

## SGA Global Statistics

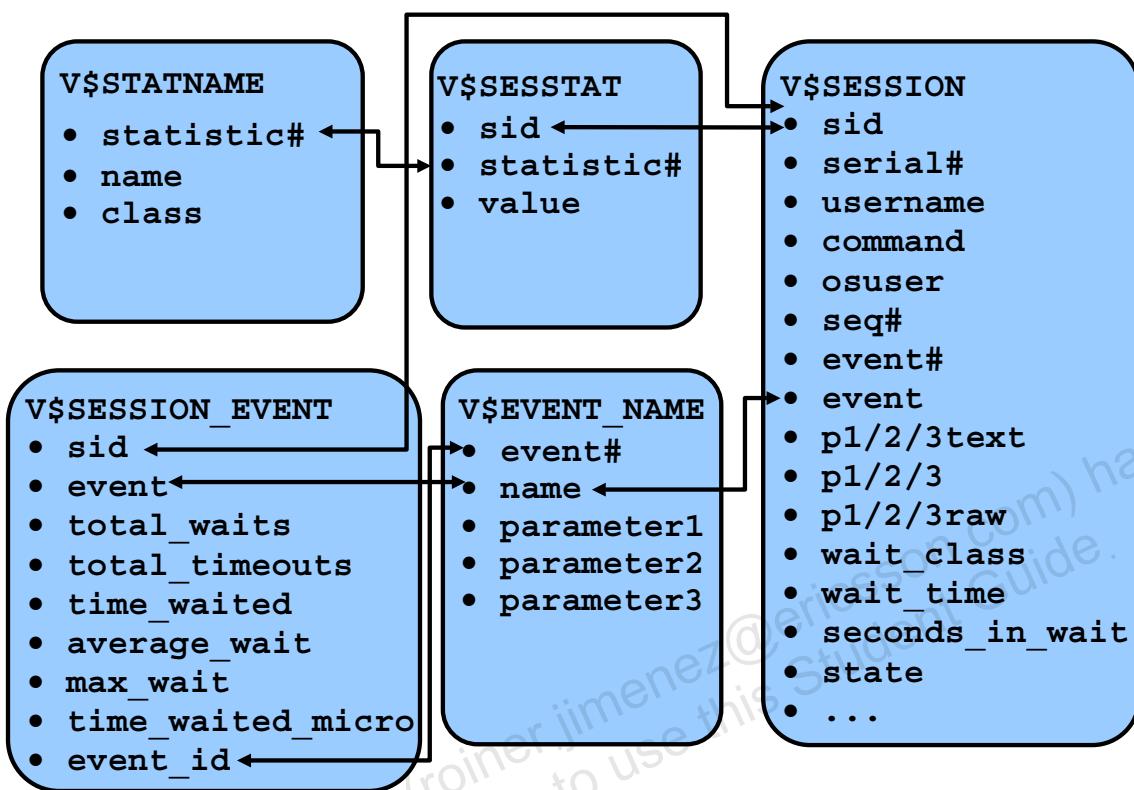
The server displays all calculated memory statistics in the V\$SGASTAT view. You can query this view to find cumulative totals of detailed SGA usage since the instance started. For example:

```
SQL> SELECT * FROM v$sgastat;
POOL NAME BYTES

fixed_sga 7780360
buffer_cache 25165824
log_buffer 262144
shared pool sessions 1284644
shared pool sql area 22376876
...
...
```

The results shown are only a partial display of the output.

# Displaying Session-Related Statistics



Copyright © 2008, Oracle. All rights reserved.

## Displaying Session-Related Statistics

You can display current session information for each user logged on by querying V\$SESSION. For example, you can use V\$SESSION to determine whether a session represents a user session, or was created by a database server process (BACKGROUND).

You can query either V\$SESSION or V\$SESSION\_WAIT to determine the resources or events for which active sessions are waiting.

The Oracle server displays user session statistics in the V\$SESSTAT view. The V\$SESSION\_EVENT view lists information on waits for an event by a session.

Cumulative values for statistics are generally available through dynamic performance views, such as the V\$SESSTAT and V\$SYSSTAT views. Note that the cumulative values in dynamic views are reset when the database instance is shut down.

The V\$MYSTAT view displays the statistics of the current session.

You can also query the V\$SESSMETRIC view to display the performance metric values for all active sessions. This view lists performance metrics such as CPU usage, number of physical reads, number of hard parses, and the logical read ratio.

# Displaying Service-Related Statistics

**For *n*-tier environments, because session statistics are not as helpful, you can see service-level statistics in these views:**

- **V\$SERVICE\_EVENT: Aggregated wait counts and wait times for each service, on a per event basis**
- **V\$SERVICE\_WAIT\_CLASS: Aggregated wait counts and wait times for each service on a wait class basis**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Displaying Service-Related Statistics

In an *n*-tier environment where there is an application server that is pooling database connections, viewing sessions may not provide the information you need to analyze performance. Grouping sessions into service names enables you to monitor performance more accurately. These two views provide the same information that their like-named session counterparts provide, except that the information is presented at the service level rather than at the session level.

V\$SERVICE\_WAIT\_CLASS shows wait statistics for each service, broken down by wait class. V\$SERVICE\_EVENT shows the same information as V\$SERVICE\_WAIT\_CLASS, except that it is further broken down by event ID.

# Troubleshooting and Tuning Views

## Instance/Database

V\$DATABASE  
V\$INSTANCE  
V\$PARAMETER  
V\$SPPARAMETER  
V\$SYSTEM\_PARAMETER  
V\$PROCESS  
V\$BGPRESS  
V\$PX\_PROCESS\_SYSSTAT  
V\$SYSTEM\_EVENT

## Disk

V\$DATAFILE  
V\$FILESTAT  
V\$LOG  
V\$LOG\_HISTORY  
V\$DBFILE  
V\$TEMPFILE  
V\$TEMPSEG\_USAGE  
V\$SEGMENT\_STATISTICS

## Memory

V\$BUFFER\_POOL\_STATISTICS  
V\$LIBRARYCACHE  
V\$SGAINFO  
V\$PGASTAT

## Contention

V\$LOCK  
V\$UNDOSTAT  
V\$WAITSTAT  
V\$LATCH

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Troubleshooting and Tuning Views

The slide lists some of the views you may need to access to determine the cause of performance problems or analyze the current status of your database.

For a complete description of these views, refer to the *Oracle Database Reference Manual*.

## Dictionary Views

- The following dictionary and special views provide useful statistics after using the DBMS\_STATS package:
  - DBA\_TABLES, DBA\_TAB\_COLUMNS
  - DBA\_CLUSTERS
  - DBA\_INDEXES
  - DBA\_TAB\_HISTOGRAMS
- This statistical information is static until you reexecute the appropriate procedures in DBMS\_STATS.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Dictionary Views

When you need to look at data storage in detail, use the DBMS\_STATS package, which collects statistics and populates columns in some DBA\_xxx views.

DBMS\_STATS populates columns in the views concerned with:

- Table data storage within extents and blocks:
  - DBA\_TABLES
  - DBA\_TAB\_COLUMNS
- Cluster data storage within extents and blocks:
  - DBA\_CLUSTERS
- Index data storage within extents and blocks, and index usefulness:
  - DBA\_INDEXES
- Non-indexed and indexed columns data distribution:
  - DBA\_TAB\_HISTOGRAMS

For more information about using the DBMS\_STATS package, refer to the *Oracle Database Performance Tuning Guide*.

Performing an ANALYZE INDEX ... VALIDATE STRUCTURE command populates the INDEX\_STATS and INDEX\_HISTOGRAM views that contain statistics for indexes.

# Diagnosis of Hung or Extremely Slow Databases

Tuning  
Statistics  
> **SGA Attach**  
AWR  
Advisors  
Async. Commit

**Use for problem analysis when the database is performing very slowly, or is hung:**

- **Direct access to SGA for performance monitoring (memory access mode)**
  - V\$SESSION
  - V\$SESSION\_WAIT
  - V\$SYSTEM\_EVENT
  - V\$SYSSTAT
  - **Hang analysis using Enterprise Manager**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Diagnosis of Hung or Extremely Slow Databases

There is functionality available for problem analysis when the database is performing very poorly, or is actually hung.

The system supports the collection of real-time performance statistics directly from the System Global Area (SGA) using optimized or lightweight system-level calls as an alternative to SQL. In Enterprise Manager, this is referred to as the memory access mode. There is one SGA collector thread per Oracle instance, and it is started automatically by the EM agent when it starts monitoring a database instance. The v\$ views shown in the slide are the main performance views used for high-level performance diagnostics and, therefore, are the ones for which direct access to SGA is available. If more extensive drill-down information is required, then you must use SQL to retrieve it. Host information, such as the number of CPUs and the host name, are also collected and made visible through the EM interface.

# Using Memory Access Mode

The screenshot shows the Oracle Database 10g Performance Monitoring interface. At the top, there's a navigation bar with Home, Performance, Administration, and Maintenance tabs. The Performance tab is selected. Below the navigation bar, there's a section titled "Performance monitoring in Memory Access Mode". It contains a graph showing "Runnable Processes" over time (from 10:50am to 11:40am on September 22, 2005). A red line on the graph is labeled "Maximum CPU". To the right of the graph is a legend for "Active Sessions" with categories: Other (pink), Network (light green), Administrative (dark blue), Configuration (orange), Commit (red), Application (yellow), and Concurrency (purple). A red box highlights the "Monitor in Memory Access Mode" link in the "Related Links" sidebar. A red arrow points from this link to the "Monitor in Memory Access Mode" button in the main content area. The content area has a yellow border and contains the following text:

**Monitor in Memory Access Mode**

Performance monitoring in Memory Access Mode is currently disabled for this database instance.

Click the Enable Memory Access Mode button to enable this feature now.

Below this text is a button labeled "Enable Memory Access Mode" with a red circle containing the number 2. A red arrow points from this button to a confirmation dialog box. The confirmation dialog box has a blue border and contains the following text:

**Confirmation**

You are about to enable the performance monitoring in memory access mode. Are you sure you want to enable it for this target?

At the bottom of the confirmation dialog are two buttons: "No" and "Yes" with a red circle containing the number 3. A red arrow points from the "Yes" button to the "Yes" button in the confirmation dialog. The Oracle logo is at the bottom right of the page.

## Using Memory Access Mode

You can access the memory access mode graphic page from the home page in the Related Links section by clicking the “Monitor in Memory Access Mode” link. The link takes you to the Performance page in “Memory Access” view mode. As shown in the slide, you have to “Enable Memory Access Mode” for the first time, and you can disable it later if you want.

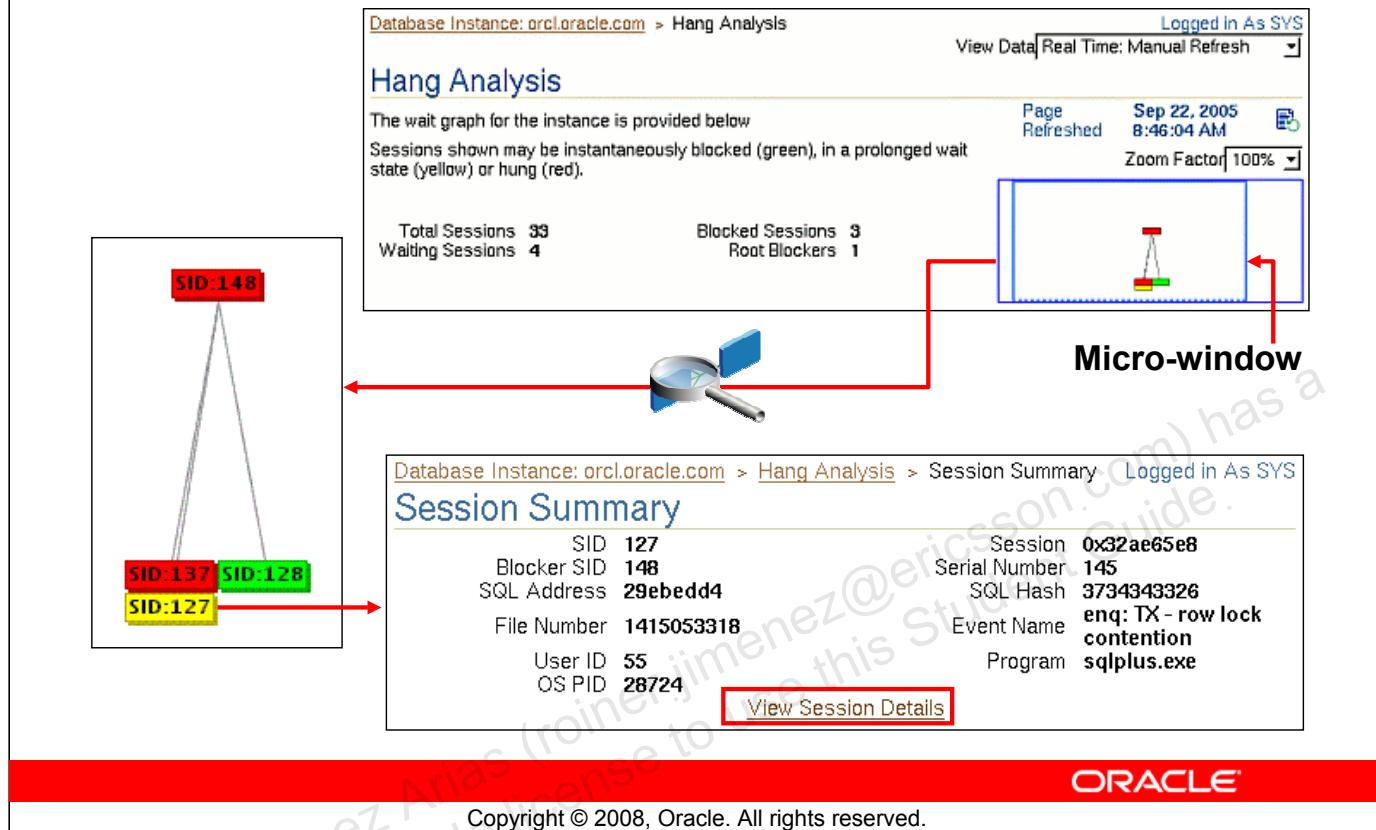
When you want to return to SQL Access view mode, just click the “Switch to SQL Access Mode” button on the Performance page.

Memory Access mode avoids the computation associated with parsing and executing SQL statements, thereby making it robust for severe cases of library cache contention that can prevent the instance from being monitored using SQL. You should switch to Memory Access mode for slow or hung systems.

Pages in Memory Access mode contain data sampled at a higher frequency than pages in SQL mode. Charts may appear to be slightly different from SQL mode for this reason. Consequently, the Memory Access mode page provides better information about where events begin and end, and you may also detect short-duration events that might otherwise be missed.

**Note:** Memory access mode is also called direct SGA attach mode.

# Using the Hang Analysis Page



## Using the Hang Analysis Page

To analyze hangs and system slowdowns, you can use the Hang Analysis page in the Additional Monitoring Links section of the Performance page in EM. You can also use the Blocking Sessions page to display a list of all sessions currently blocking other sessions.

You use the Hang Analysis page to:

- Determine which sessions are causing bottlenecks
- Examine session summaries for blocking or blocked sessions

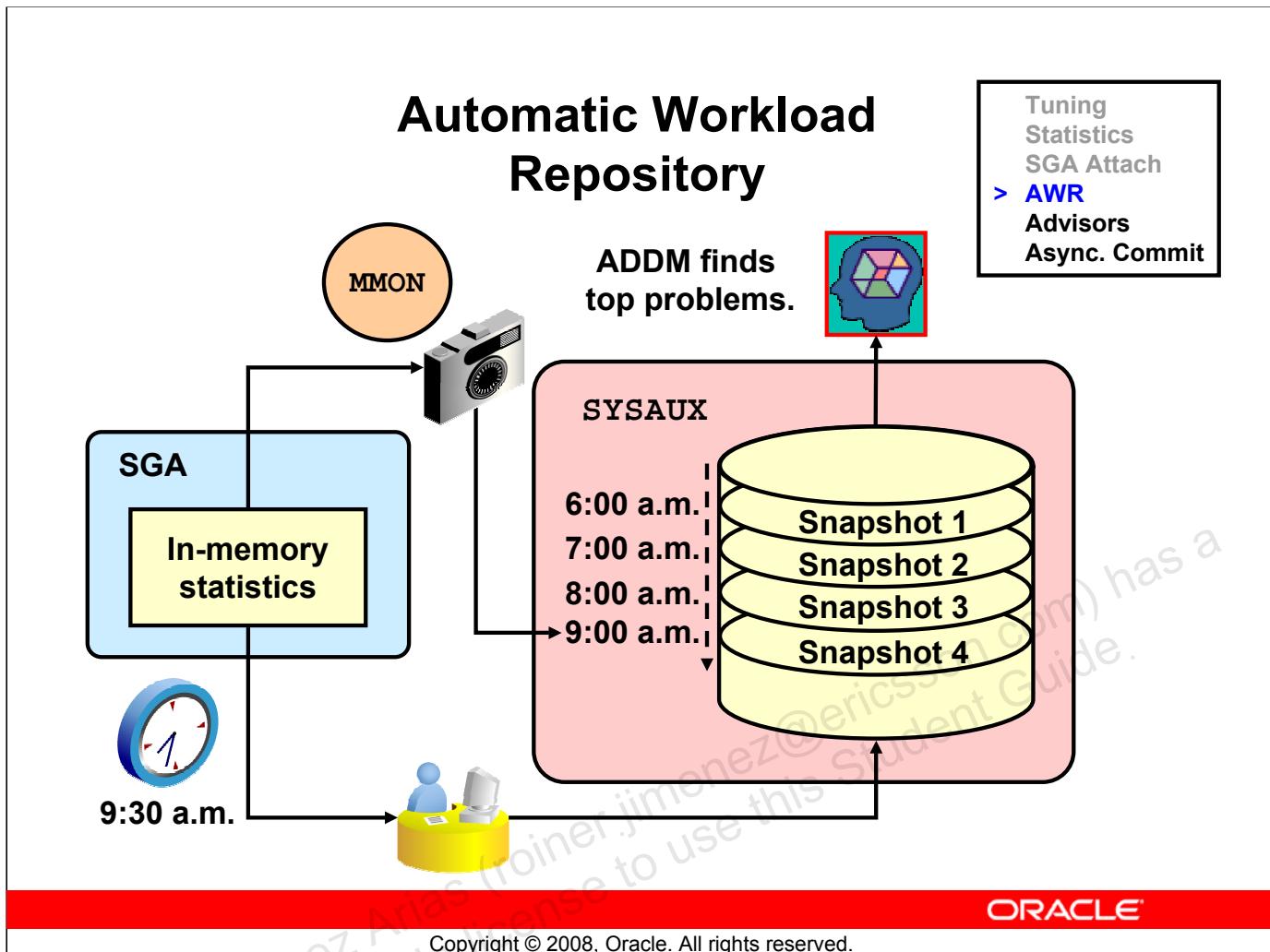
This page provides you with a graphical topology of waiting sessions in the system, with blocked sessions appearing below blocking sessions. Enterprise Manager assesses the situation on the basis of historical activity and determines which sessions are hung or likely to be hung, rather than being only instantaneously in a wait state. To view summary information about a session, click a session ID in the topology. The Session Summary page appears, which displays general information about the selected session. You can click View Session Details on the Session Summary page to get further session information, and determine whether terminating the session is beneficial.

## Using the Hang Analysis Page (continued)

The sessions are displayed in green, yellow, or red depending on the perceived seriousness of the state of the session within the topology.

You can also select a Zoom Factor percentage based on the desired size of the topology. In cases where there are many blocked sessions, you can use a smaller factor to visualize the overall wait information for the system. You can then zoom in on relevant portions and read the topology or drill down further.

You click in an area of the Zoom Factor microwindow to move the topology to the desired viewing position. For instance, if you click the left side of the window, the topology shifts to the right to enable you to fully view the left side of the topology.



Tuning  
Statistics  
SGA Attach  
> **AWR**  
Advisors  
Async. Commit

## Automatic Workload Repository

The Automatic Workload Repository (AWR) is a collection of persistent system performance statistics owned by SYS. The AWR resides in the SYSAUX tablespace.

A *snapshot* is a set of performance statistics captured at a certain time and stored in the AWR. Snapshots are used to compute the rate of change of a statistic. Each snapshot is identified by a snapshot sequence number (`snap_id`) that is unique in the AWR.

Statspack users should start using the workload repository in Oracle Database 10g. However, there is no supported path to migrate Statspack data into the workload repository. Also, the workload repository is not backward compatible with the Statspack schema.

By default, snapshots are generated every 60 minutes. You can adjust this frequency by changing the snapshot `INTERVAL` parameter. Because the database advisors rely on these snapshots, be aware that adjustment of the interval setting can affect diagnostic precision. For example, if the `INTERVAL` is set to 4 hours, you may miss spikes that occur within 60-minute intervals.

You can use the `DBMS_WORKLOAD_REPOSITORY.MODIFY_SNAPSHOT_SETTINGS` stored procedure or Database Control to change the settings that control snapshot collection. In Database Control, click Automatic Workload Repository in the Statistics Management region of the Administration tabbed page. Then, click Edit to make the changes.

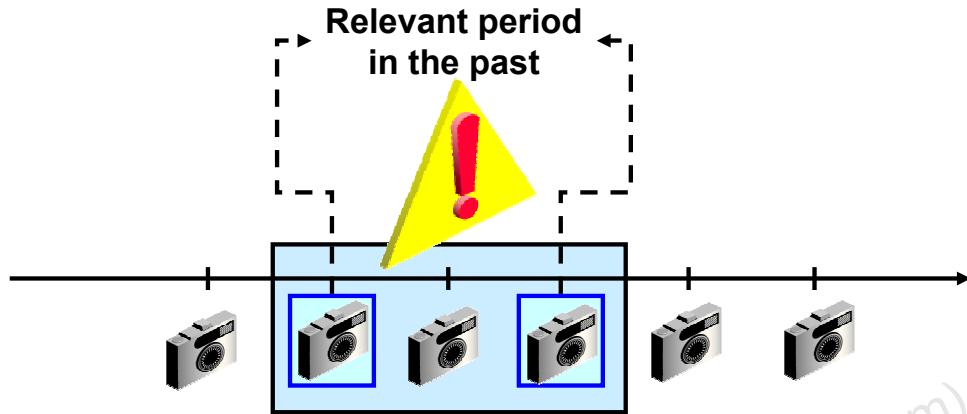
The stored procedure offers more flexibility in defining `INTERVAL` values than does Database Control.

## Automatic Workload Repository (continued)

You can take manual snapshots by using Database Control or the DBMS\_WORKLOAD\_REPOSITORY.CREATE\_SNAPSHOT stored procedure. Taking manual snapshots is supported in conjunction with the automatic snapshots that the system generates. Manual snapshots are expected to be used when you want to capture the system behavior at two specific points in time that do not coincide with the automatic schedule.

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a  
non-transferable license to use this Student Guide.

## AWR Snapshot Baselines



```
DBMS_WORKLOAD_REPOSITORY.CREATE_BASELINE (-
 start_snap_id IN NUMBER,
 end_snap_id IN NUMBER,
 baseline_name IN VARCHAR2);
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### AWR Snapshot Baselines

Baselines are a mechanism for you to tag sets of snapshot data for important periods. A baseline is defined as a pair of snapshots; the snapshots are identified by their snapshot sequence numbers (`snap_id`). Each baseline corresponds to one and only one pair of snapshots.

A baseline can be identified by either a user-supplied name or a system-generated identifier. You can create a baseline by using Database Control or by executing the `DBMS_WORKLOAD_REPOSITORY.CREATE_BASELINE` procedure and specifying a name and a pair of snapshot identifiers. A baseline identifier is assigned to the newly created baseline. Baseline identifiers are unique for the life of a database.

Baselines are used to retain snapshot data. Therefore, snapshots belonging to baselines are retained until the baselines are dropped.

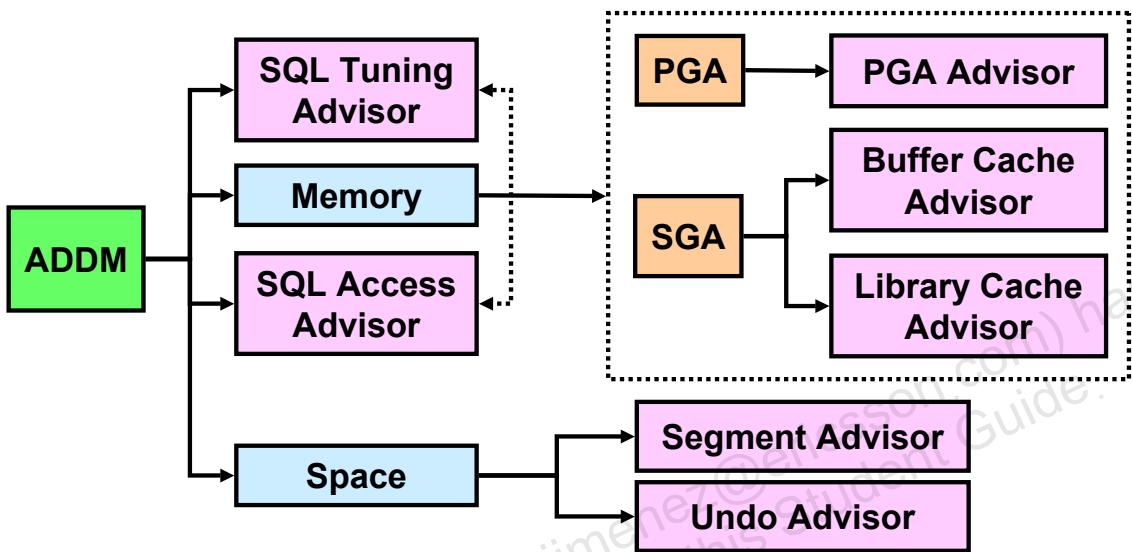
Usually, you set up baselines from some representative periods in the past, to be used for comparisons with current system behavior. You can also set up threshold-based alerts by using baselines from Database Control.

You can get the `snap_ids` directly from `DBA_HIST_SNAPSHOT`, or from Database Control.

**Note:** For more information about the `DBMS_WORKLOAD_REPOSITORY` package, see the *Oracle Database PL/SQL Packages and Types Reference Guide*.

# Advisory Framework: Overview

Tuning  
Statistics  
SGA Attach  
AWR  
> Advisors  
Async. Commit



**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Advisory Framework: Overview

Advisors are server components that provide you with useful feedback about resource utilization and performance for its respective component. The advisors use all the resources previously discussed, and more. Following is the list of available advisors:

- **Automatic Database Diagnostic Monitor (ADDM):** Performs a top-down instance analysis, identifies problems and potential causes, and gives recommendations for fixing the problems. ADDM can potentially call other advisors.
- **SQL Tuning Advisor:** Provides tuning advice for SQL statements
- **SQL Access Advisor:** Deals with schema issues and determines optimal data access paths such as indexes and materialized views
- **PGA Advisor:** Gives detailed statistics for the work areas, and provides recommendations about optimal usage of the Program Global Area (PGA) memory on the basis of workload characteristics
- **SGA Advisor:** Is responsible for tuning and recommending the System Global Area (SGA) size depending on the pattern of access for the various components within the SGA
- **Segment Advisor:** Monitors object space issues and analyzes growth trends
- **Undo Advisor:** Suggests parameter values and the amount of additional space that is needed to support flashback for a specified time

## **Advisory Framework: Overview (continued)**

The major benefits provided by the advisor infrastructure are:

- It uses a uniform interface for all advisors.
- All advisors are invoked and they report results in a consistent manner.

# Database Control and Advisors

**Advisors**

|                                    |                                |                                 |
|------------------------------------|--------------------------------|---------------------------------|
| <a href="#">ADDM</a>               | <a href="#">Memory Advisor</a> | <a href="#">Segment Advisor</a> |
| <a href="#">SQL Tuning Advisor</a> | <a href="#">MTTR Advisor</a>   | <a href="#">Undo Management</a> |
| <a href="#">SQL Access Advisor</a> |                                |                                 |

**Advisor Tasks**

**Search**

Select an advisory type and optionally enter a task name to filter the data that is displayed in your results set.

| Advisory Type | Task Name | Advisor Runs                               |
|---------------|-----------|--------------------------------------------|
| All Types     |           | Last Run <input type="button" value="Go"/> |

**Results**

| Select                           | Advisory Type | Name                                 | Description                                                | User | Status    | Start Time              | End Time             | Expires In (days) |
|----------------------------------|---------------|--------------------------------------|------------------------------------------------------------|------|-----------|-------------------------|----------------------|-------------------|
| <input checked="" type="radio"/> | ADDM          | <a href="#">ADDM:1037512968_1_66</a> | ADDM auto run: snapshots [65, 66], instance 1, database in | SYS  | COMPLETED | Nov 12, 2003 5:30:44 AM | Nov 12, 2003 5:30:44 | 30                |

**ORACLE**

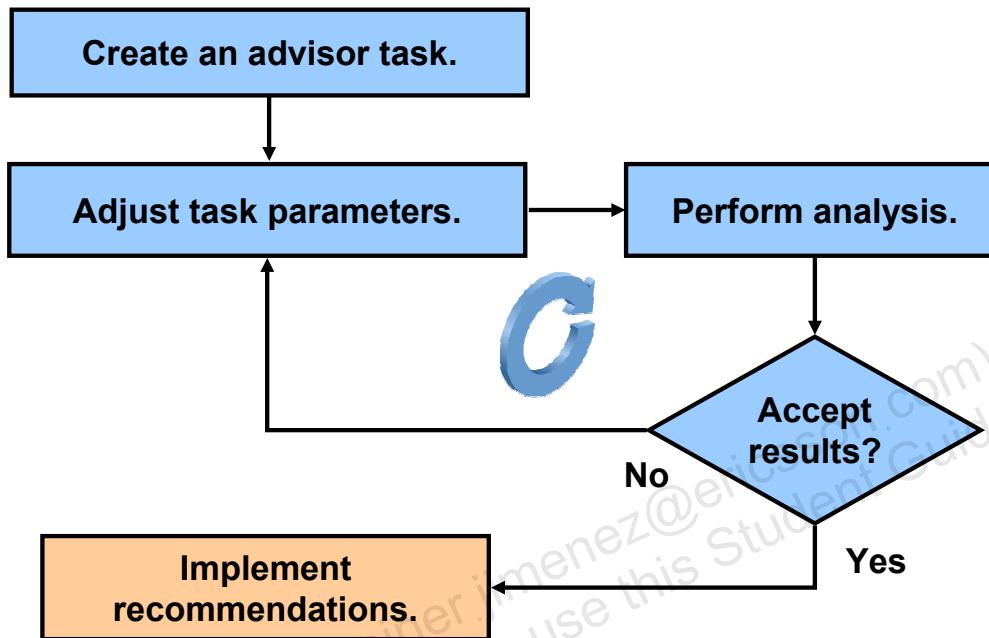
Copyright © 2008, Oracle. All rights reserved.

## Database Control and Advisors

The Advisor Central page is the main page for all advisors. You can reach this page by clicking the Advisor Central link in the Related Links region of the Database home page. However, this is not the only place inside the Database Control Console where advisors can be invoked.

On the Advisor Central page, you can list all the advisor tasks that were registered in the AWR. You can also filter this list by advisor type and for predefined time periods.

# Typical Advisor Tuning Session



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Typical Advisor Tuning Session

A typical tuning session comprises the following steps:

1. **Create an advisor task.**

An advisor task is an executable data area in the advisor repository that manages your tuning efforts.

2. **Adjust appropriate task parameters.**

Parameters are set in the main advisor task. They control the advisor's behavior.

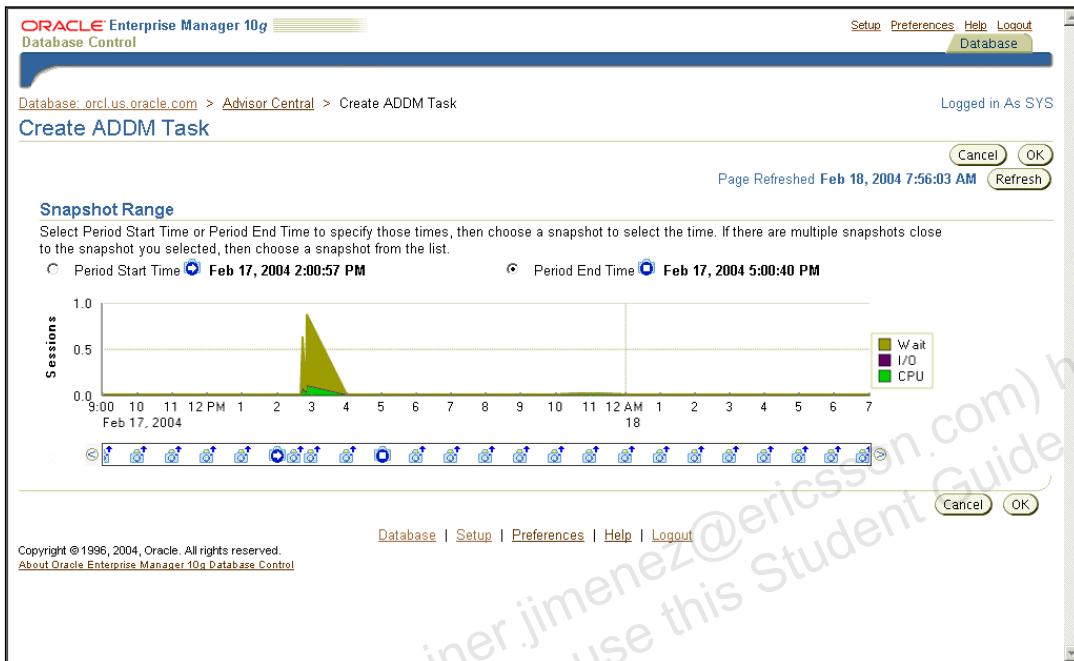
3. **Perform an analysis.**

The execution of the task is a synchronous operation; control is not returned until the operation has completed or a user interruption is detected. At any time, you can interrupt the operation and review the results up to that point in the analysis process. If not satisfied, you can resume the execution for further recommendations, or the task data can be adjusted and execution be restarted.

4. **Review the results.**

The results of the analysis can be reviewed using the built-in views or the procedure shown in the slide. You have the option of accepting, rejecting, or ignoring the recommendations. If a recommendation is rejected, you may want to rerun an analysis using the rejected recommendation as advice for the next analysis operation.

# Manually Invoking ADDM



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Manually Invoking ADDM

By default, ADDM tasks are run for every Oracle database snapshot that is stored in the AWR. However, you can create a custom ADDM task to analyze a period of time that you identify with a starting snapshot and an ending snapshot. To create an ADDM task, perform the following steps:

1. Navigate to the Database home page. In the Related Links section, click Advisor Central.
2. Under Advisors, choose ADDM.
3. Select the Period Start Time option and then click the snapshot that you want to use as the starting point of the period of time. Then, select the End Time option and click the snapshot to use as the terminating point of the time period.
4. Use the ADDM Task Page to view the results of a selected ADDM task.

## Using the SQL Tuning Advisor: Review

- **Use the SQL Tuning Advisor to analyze SQL statements and obtain performance recommendations.**
- **Sources for SQL Tuning Advisor to analyze:**
  - **Top SQL:** Analyzes the top SQL statements currently active
  - **SQL Tuning Sets:** Analyzes a set of SQL statements you provide
  - **Snapshots:** Analyzes a snapshot
  - **Baselines:** Analyzes a baseline

ORACLE

Copyright © 2008, Oracle. All rights reserved.

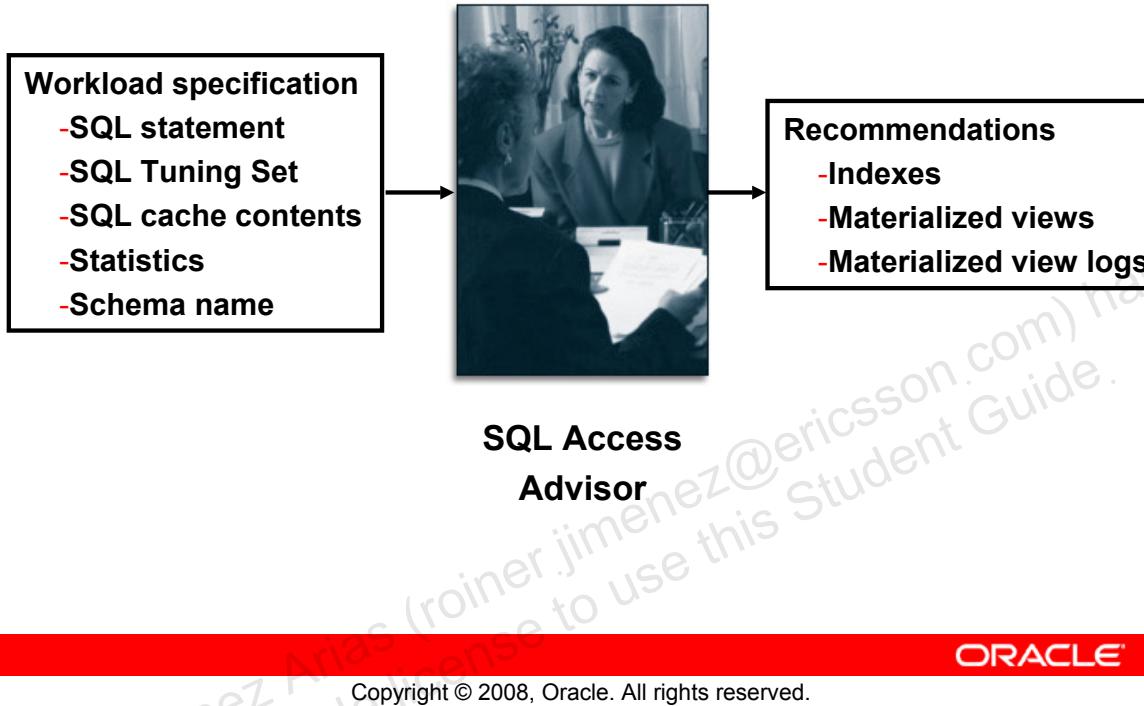
## Using the SQL Tuning Advisor: Review

You can use the SQL Tuning Advisor to analyze SQL statements and obtain performance recommendations. Typically, you run this advisor as an ADDM performance-finding action.

Additionally, you can run the SQL Tuning Advisor when you want to analyze the top SQL statements consuming the most CPU time, I/O, and memory.

The SQL Tuning Advisor is covered in detail in the *Oracle Database 10g: Administration Workshop I* course.

# SQL Access Advisor: Overview



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## SQL Access Advisor: Overview

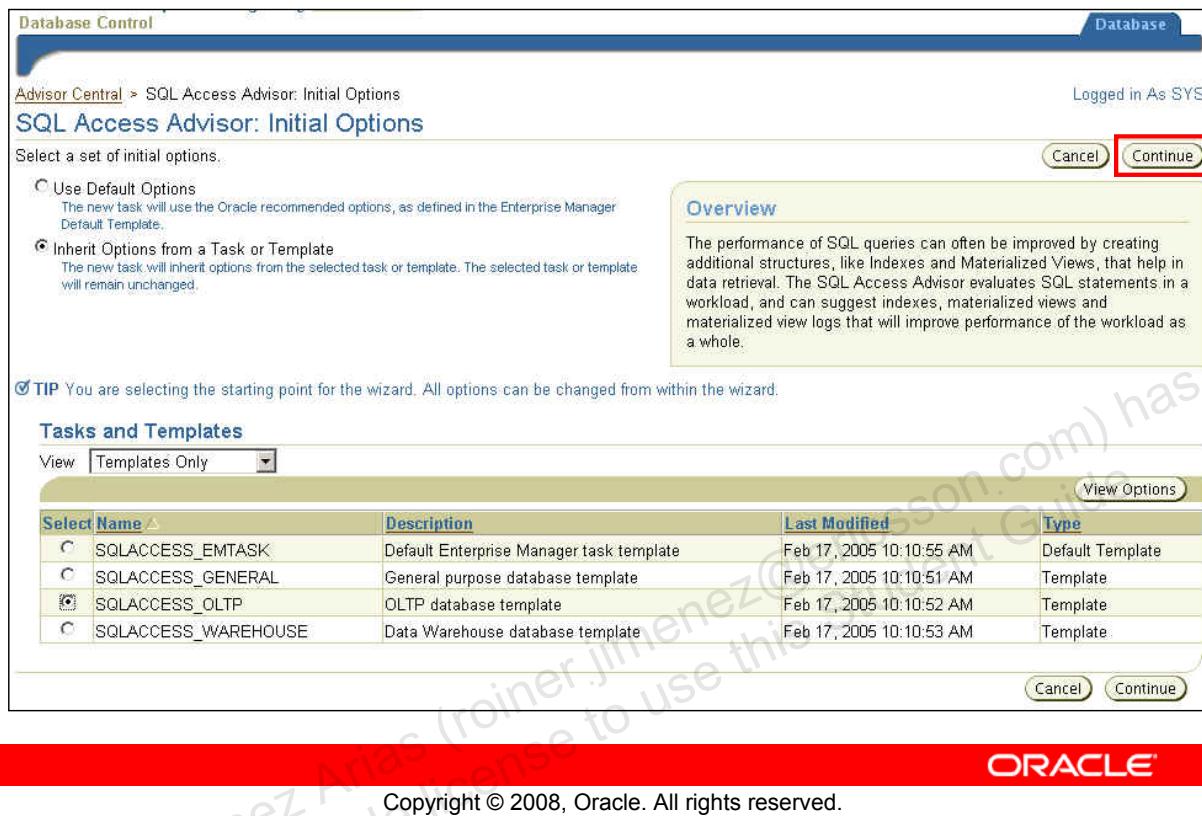
The SQL Access Advisor can recommend the proper set of materialized views, materialized view logs, and indexes for a given workload. Understanding and using these structures is essential when optimizing SQL because they can result in significant performance improvements in data retrieval.

The SQL Access Advisor recommends bitmap, function-based, and B-tree indexes. A bitmap index offers a reduced response time for many types of ad hoc queries and reduced storage requirements compared to other indexing techniques. B-tree indexes are most commonly used in a data warehouse to index unique or near-unique keys.

Another component of the SQL Access Advisor also recommends how to optimize materialized views so that they can be fast refreshable and take advantage of general query rewrite.

**Note:** For more information about materialized views and query rewrite, see the *Oracle Database Performance Tuning Guide*.

# Typical SQL Access Advisor Session



## Typical SQL Access Advisor Session

When starting a SQL Access Advisor session, you can select Use Default Options and start with a predefined set of Advisor options that are recommended. Additionally, you can start a task and have it inherit a set of option values as defined by a template or task by selecting “Inherit Options from a Task or Template.” These include several generic templates designed for general purpose environments, OLTP, and data warehousing databases. You can save custom templates from a previous task and reuse them when needed.

Click Continue to launch the SQL Access Advisor wizard.

**Note:** You can access SQL Access Advisor from the Advisor Central page of Database Control.

# Workload Source

**SQL Access Advisor: Workload Source**

Database EDRSR14P1\_orcl.oracle.com  
Logged in As SYS

Select the source of the workload that you want to use for the analysis. The best workload is one that fully represents all the SQL statements that access the underlying tables.

Current and Recent SQL Activity  
SQL will be selected from the cache.

Import Workload from SQL Repository  
Choose any SQL Tuning Set from the SQL Repository.  
SQL Tuning Set: SYS.MY\_STS\_WORKLOAD

User-Defined Workload; Import SQL from a Table or View  
The table or view must contain at least SQL\_TEXT and USERNAME columns.  
Table: [ ]

Create a Hypothetical Workload from the Following Schemas and Tables  
The advisor can create a hypothetical workload if the tables contain dimension or primary/foreign key constraints.  
Tables: [ ]  
Comma-separated list  
**TIP** Enter "Schema.%" to specify all the tables belonging to a particular schema.

**Filter Options**  
You can apply filters to reduce the scope of the statements found in the workload. Using filters has two advantages. First, it directs the advisor to make recommendations based on a specific subset of statements from the workload, which may lead to better quality recommendations. Second, removing extraneous statements from the workload may greatly reduce processing time.

Cancel Step 1 of 4 Next

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Workload Source

Use the SQL Access Advisor wizard's Workload Source page to provide a defined workload that allows the Access Advisor to make recommendations. Supported workload sources are:

- **Current and Recent SQL Activity:** Uses the current SQL from the cache as the workload
- **Import Workload from SQL Repository:** Enables you to specify a previously created SQL Tuning Set as the workload source
- **User-Defined Workload; Import SQL from a Table or View:** Enables you to receive recommendations for a workload that may not be running on the current database. This provides and implements access recommendations before an application goes live.
- **Create a Hypothetical Workload from the Following Schemas and Tables:** Provides a schema that allows the advisor to search for dimension tables and produce a workload

The scope of the workload can be further reduced by applying filters that you can access in the Filter Options section. With these options, you can reduce the scope of the SQL statements that are present in the workload. The filters are applied to the workload by the advisor to focus the tuning effort. Possible filter options are:

- Top resource consuming SQL statements
- Users, module identifier, or actions
- Tables

# Recommendation Options

**SQL Access Advisor: Recommendation Options**

Database EDRSR14P1\_orcl.oracle.com  
Logged in As SYS

**Recommendation Types**

The advisor may recommend indexes or materialized views to reduce the time it takes to read data. However you must balance this benefit against the cost to maintain the additional structures. Select the type of structures to be recommended by the advisor.

Indexes

Materialized Views

Both Indexes and Materialized Views

Evaluation Only  
Evaluates usage of existing access structures and describes which access structures are currently being used by this workload. No new access structures will be recommended.

**Advisor Mode**

The advisor can run in one of two modes, Limited or Comprehensive. Limited Mode is meant to return quickly after processing the statements with the highest cost, potentially ignoring statements with a cost below a certain threshold. Comprehensive Mode will perform an exhaustive analysis.

Limited Mode  
Analysis will focus on highest cost statements

Comprehensive Mode  
Analysis will be exhaustive

**Advanced Options**

**Workload Categorization**

**Workload Volatility**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Recommendation Options

Use the Recommendations Options page to choose whether to limit the advisor to recommendations based on a single access method. Choose Indexes or Materialized Views or both from the Recommendation Types section. You can choose Evaluation Only to evaluate only existing access structures. In this mode, the advisor does not generate new recommendations but comments on the use of existing structures. This is useful to track the effectiveness of the current index, materialized view, and MV log usage over time.

You can use the Advisor Mode section to run the advisor in one of two modes. These modes affect the quality of recommendations as well as the length of time required for processing. In Comprehensive Mode, the advisor searches a large pool of candidates resulting in recommendations of the highest quality. In Limited Mode, the advisor performs quickly, limiting the candidate recommendations.

# Recommendation Options

**▼Advanced Options**

**Workload Categorization**

Workload Volatility

- Allow Advisor to consider the volatility of referenced objects when forming recommendations  
Performance of update/insert/delete operations is critical (OLTP)
- Do not use volatility data to limit recommendations  
Workload favors read-only operations (Data Warehouse)

Workload Scope

- Partial Workload  
SQL Statements may be missing from the workload that might be adversely affected by removing access structures, so don't include recommendations to drop unused access structures
- Complete Workload  
Workload contains a full representation of the interesting application SQL statements for the targeted tables, so include recommendations to drop unused access structures

**Space Restrictions**

Indexes and materialized views increase performance at the cost of space. When the SQL Access Advisor is invoked with no space limitations it will make the best possible performance recommendations. When reviewing task results, you can decide which recommendations to implement. Do you wish the sum of the recommendation space requirements to fit within a space limit?

- No, show me all recommendations
- Yes, space is limited Space Limit  MB

When a complete workload is specified, you can enter a zero or negative space limit. This will cause the SQL Access Advisor to recommend dropping the maximum number of non-essential access structures such that their total storage size is within the absolute value of this space limitation.

**Tuning Options**

Prioritize tuning of SQL statements by

SQL statements will be analyzed in descending order of the value of the prioritized statistic.

Allow Advisor to consider creation costs when forming recommendations  
If checked, the SQL Access Advisor will weigh the cost of creation of access structures against the frequency and potential improvement of SQL statement execution time. If unchecked, the cost of creation will be ignored.

**Default Storage Locations**

By default indexes will be placed in the schema and tablespace of the table they reference, materialized views will be placed in the schema and tablespace of the first table referenced in the query, and materialized view logs will be placed in the default tablespace of the schema of the table they reference. These fields allow you to change these default locations.

|                                  |                      |                                                                                    |
|----------------------------------|----------------------|------------------------------------------------------------------------------------|
| Index Tablespace                 | <input type="text"/> |  |
| Index Schema                     | <input type="text"/> |  |
| Materialized View Tablespace     | <input type="text"/> |  |
| Materialized View Schema         | <input type="text"/> |  |
| Materialized View Log Tablespace | <input type="text"/> |  |

Step 2 of 4



Copyright © 2008, Oracle. All rights reserved.

## Recommendation Options (continued)

You can choose Advanced Options to show or hide options that enable you to set space restrictions, tuning options, and default storage locations. Use the Workload Categorization section to set options for Workload Volatility and Workload Scope. You can choose to favor read-only operations or you can consider the volatility of referenced objects when forming recommendations. You can also select Partial Workload, which does not include recommendations to drop unused access structures, or Complete Workload, which does include recommendations to drop unused access structures.

Use the Space Restrictions section to specify a hard space limit, which forces the advisor to produce recommendations only with total space requirements that do not exceed the specified limit.

Use the Tuning Options section to specify options that customize the recommendations made by the advisor. Use the “Prioritize Tuning of SQL Statements by” drop-down list to prioritize by Optimizer Cost, Buffer Gets, CPU Time, Disk Reads, Elapsed Time, and Execution Count.

Use the Default Storage Locations section to override the defaults defined for schema and tablespace locations. By default, indexes are placed in the schema and tablespace of the table they reference. Materialized views are placed in the schema and tablespace of the user who executed one of the queries that contributed to the materialized view recommendation.

After you define these parameters, you can schedule and review your tuning task.

# Reviewing Recommendations

**Results for Task: SQLACCESSJFV2**

| Task Name     | Status    | Advisor Mode  |
|---------------|-----------|---------------|
| SQLACCESSJFV2 | COMPLETED | COMPREHENSIVE |

**Overall Workload Performance**

**Potential for Improvement**

**Workload I/O Cost**

Original Cost (2053)  
New Cost (148)

**Recommendations**

Recommendations 3  
Space Requirements 0.11MB  
User Specified Space Limit Unlimited

**SQL Statements**

SQL Statements 4  
Statements remaining after filters were applied

**Query Execution Time Improvement**

No Performance Improvement  
Potential Performance Improvement

**Copyright © 2008, Oracle. All rights reserved.**

## Reviewing Recommendations

Using the Advisor Central page, you can list all the completed SQL Access Advisor tasks. Select the one for which you want to see the recommendations, and then click the View Result button. Use the Results for Task Summary page to get an overview of the advisor findings. The page presents charts and statistics that provide overall workload performance and query execution time potential improvement for the recommendations. You can use the page to show statement counts and recommendation action counts.

To see other aspects of the results for the advisor task, click one of the three other tabs on the page: Recommendations, SQL Statements, or Details.

The Recommendations page displays a chart and a table that show the top recommendations ordered by their percentage improvement to the total cost of the entire workload. The top recommendations have the biggest total performance improvement.

By clicking the Show SQL button, you can see the generated SQL script for the selected recommendations. You can click the corresponding recommendation identifier in the table to see the list of actions that need to be performed in order to implement the recommendation. On the Actions page, you can actually see all the corresponding SQL statements to execute in order to implement the action. For recommendations that you do not want to implement, keep those check boxes deselected. Then, click the Schedule Implementation button to implement the retained actions. This step is executed in the form of a Scheduler job.

## Asynchronous COMMIT

Tuning  
Statistics  
SGA Attach  
AWR  
Advisors  
➤ Async. Commit

- **The default COMMIT behavior is to wait until redo is saved in the redo log files.**
- **The default behavior can now be changed to “not wait.”**
- **Asynchronous COMMIT is useful for high transaction throughput.**
- **However, transactions may be lost.**
  - Machine crashes
  - File I/O problems with redo log files

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Asynchronous COMMIT

When a transaction commits, the log writer (LGWR) process writes redo for the commit, along with the accumulated redo of all changes in the corresponding transaction, to disk. By default, the Oracle database writes the redo to disk before the call returns to the client. This behavior introduces a latency in the commit because the application must wait for the redo to be written to disk.

Suppose you are writing an application that requires very high transaction throughput. If you are willing to trade commit durability for lower commit latency, then you can change the default COMMIT options so that the application does not need to wait for the Oracle database to write data to the online redo logs.

Thus, the redo generated for a transaction may not persist when the commit returns to the user. This opens a small window of vulnerability where the transaction that purportedly committed could be rolled back. The most obvious case is where the machine crashes. In this case, any commit redo buffered in the redo log buffers before being written to the online redo log files is also lost. Another case could be when you experience file I/O problems with the online redo logs at the point where LGWR actually attempts to force any redo buffered in the redo log buffer to disk. If the redo logs are not multiplexed to provide a level of redundancy, then it is possible to lose the commit action.

# Using Asynchronous COMMIT

- Possible combinations:
  - IMMEDIATE, WAIT (this is the default)
  - IMMEDIATE, NOWAIT
  - BATCH, WAIT
  - BATCH, NOWAIT
- System- or session-level examples:

```
ALTER SYSTEM SET COMMIT_WRITE = IMMEDIATE, WAIT
```

```
ALTER SESSION SET COMMIT_WRITE = IMMEDIATE, NOWAIT
```

- COMMIT statement examples:

```
COMMIT WRITE BATCH WAIT
```

```
COMMIT WRITE BATCH NOWAIT
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Using Asynchronous COMMIT

You can change the commit behavior in the following locations:

- COMMIT\_WRITE initialization parameter at the system or session level
- COMMIT statement

The IMMEDIATE option ensures that the redo for the commit of the transaction is written out immediately by the LGWR process; that is, an I/O is initiated. The BATCH option means that the redo is buffered and no I/O is initiated. However, LGWR is still permitted to write the redo to disk in its own time.

The WAIT option ensures that the commit does not return until the redo corresponding to the commit is persisted in the online redo logs. When you use the NOWAIT option, the commit returns without waiting for the commit redo to be written to the online redo logs.

In the absence of any options, the defaults are IMMEDIATE and WAIT, which is consistent with earlier database releases. You cannot specify both BATCH and IMMEDIATE together, nor can you specify both WAIT and NOWAIT together.

After the initialization parameter is set, a COMMIT statement with no options conforms to the options specified in the parameter.

**Note:** The options in the COMMIT statement override the current settings in the initialization parameter.

# **Summary**

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

- **Diagnose database performance issues**
- **Configure the Automatic Workload Repository**
- **Access the database advisors**
- **Use the SQL Access Advisor to improve database performance**
- **Use asynchronous COMMIT effectively**



Copyright © 2008, Oracle. All rights reserved.

# Practice Overview: Using ADDM to Diagnose Performance Problems

**This practice covers the following topics:**

- **Viewing and interpreting ADDM findings to diagnose a performance problem**
- **Implementing those findings**



Copyright © 2008, Oracle. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2010, Oracle and/or its affiliates.

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a  
non-transferable license to use this Student Guide.

# 10

## Managing Schema Objects

ORACLE

Copyright © 2008, Oracle. All rights reserved.

# Objectives

Table Types  
Partition  
IOT  
Cluster  
DBA Tasks

**After completing this lesson, you should be able to manage schema objects and:**

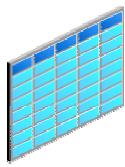
- **Determine appropriate table types for your requirements: heap, partition, IOT, or cluster**
- **Perform related DBA tasks:**
  - Estimating the size of new tables
  - Analyzing growth trends
  - Managing optimizer statistics
  - Reorganizing schema objects online

**ORACLE**

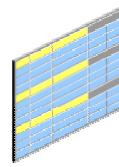
Copyright © 2008, Oracle. All rights reserved.

# Table Types

Heap

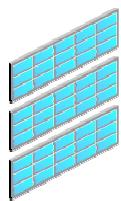


Clustered



| Type                            | Description                                                                       |
|---------------------------------|-----------------------------------------------------------------------------------|
| Ordinary (heap-organized) table | Data is stored as an unordered collection (heap).                                 |
| Partitioned table               | Data is divided into smaller, more manageable pieces.                             |
| Index-organized table (IOT)     | Data (including non-key values) is sorted and stored in a B-tree index structure. |
| Clustered table                 | Related data from one or more tables are stored together.                         |

Partitioned



IOT



Copyright © 2008, Oracle. All rights reserved.

## Table Types

Ordinary “heap-organized” tables are introduced in the *Oracle Database 10g: Administration Workshop I* course.

Partitions are pieces of a table or an index, created to facilitate management of a very large database (VLDB), which could contain several terabytes of data.

Unlike a heap-organized table whose data is stored as an unordered collection (heap), data for an index-organized table (IOT) is stored in a B-tree index structure in a primary key-sorted manner.

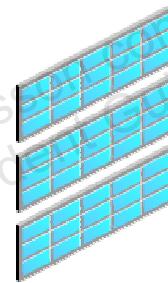
A cluster comprises a table or a group of tables that share the same data blocks because they share common columns and are often used together.

# What Is a Partition and Why Use It?

Table Types  
-> Partition  
IOT  
Cluster  
DBA Tasks

## A partition is:

- A piece of a “very large” table or index
- Stored in its own segment
- Used for improved performance and manageability



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## What Is a Partition and Why Use It?

A partition is a piece of a “very large” table or index, stored in its own segment, so that it can be managed individually. An example of a “very large” table is a data warehouse table of several hundred gigabytes of data. Partitions can be further broken down into subpartitions for finer levels of manageability and improved performance.

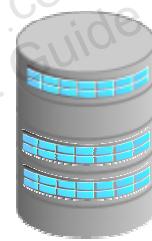
Partitioning can also bring better performance because many queries can ignore partitions that, according to the WHERE clause, do not have the requested rows, thereby reducing the amount of data to be scanned to produce a result set.

Operations on partitioned tables and indexes can be performed in parallel by assigning different parallel execution servers to different partitions of the table or index.

# Partitions

## Characteristics of partitions are:

- **Same logical attributes: Same columns, constraints, and indexes**
- **Different physical attributes: Stored in different tablespaces**
- **Transparent to applications**
- **Several partitioning methods**



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Partitions

Each partition is stored in its own segment and can be managed individually. It can function independently of the other partitions, thereby providing a structure that can be better tuned for availability and performance.

If you are using parallel execution, partitions provide another means of parallelization. Operations on partitioned tables and indexes are performed in parallel by assigning different parallel execution servers to different partitions of the table or index.

Partitions and subpartitions of a table or index all share the same logical attributes. For example, all partitions (or subpartitions) in a table share the same column and constraint definitions, and all partitions (or subpartitions) of an index share the same index options. However, they can have different physical attributes (such as TABLESPACE). Storing partitions in separate tablespaces is advantageous for independent backup and recovery, controlling the mapping to disk drives (balancing IO) and reducing the possibility of data corruption.

Partitioning is transparent to existing applications and standard DML statements run against partitioned tables. However, an application can be programmed to take advantage of partitioning by using partition-extended table or index names in DML.

For more information about partitioned tables and indexes, including partitioning methods, see the *Oracle Database Administrator's Guide*.

# Creating a Partition



## Creating a Partition

Database Control supports various types of partitioning for tables and indexes and simplifies their creation process. Wizards take you through this process and advise you on possible options.

Select Administration > Tables. On the Tables page, click the Create button to create a partitioned table. This displays the “Create Table: Table Organization” page, where you can decide whether to create a heap-organized table or an index-organized table. Then, click the Continue button to access the Create Table page. On this page, you can specify the table name and various other information. Click the Partitions tab and then the Create Partitions button. The Create Partitions: Partitioning Method page opens, which is shown in the slide. This is the entry point to the creation wizard for partitioned tables. Decide on the partitioning type, and then click the Continue button. The wizard guides you through the creation process by defining the partition columns and their specifications.

# Partitioning Methods

- **Range partitioning:** Maps rows based on logical ranges of columns values—for example, months in a year
- **Hash partitioning:** Maps rows based on the hash value of the partitioning key
- **List partitioning:** Maps rows based on a discrete list of values, provided by the DBA
- **Range-hash partitioning:** Maps rows using the range method, and within each range partition, creates hash subpartitions
- **Range-list partitioning:** Maps rows first based on a range of values, then based on discrete values

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Partitioning Methods

- Performance of **range** partitioning is best when the data evenly distributes across the range.
- **Hash** partitioning means automatic, even distribution of data: The DBA has no control of how a specific row maps to a partition. For example:

```
CREATE TABLE regions
 (region_id NUMBER, region_name VARCHAR2 (25))
 PARTITION BY HASH (region_id)
 PARTITIONS 4 STORE IN (tbs1, tbs2, tbs3, tbs4);
```

This example shows a hash-partitioned table. The partitioning column is `id`; four partitions are created and assigned system-generated names, and they are placed in four tablespaces named `tbs1`, `tbs2`, `tbs3`, and `tbs4`.

- Specify a **list** of discrete values for the partitioning column in the description for each partition, if you require explicit control over how rows map to partitions. For example, specify country or state codes.
- The composite partitions, produced by **range-hash** partitioning, are ideal for both historical data and striping.
- The partitions of a **range-list** partitioned table are logical structures only, because their data is stored in the segments of their subpartitions.

# Partition Maintenance

The screenshot shows the Oracle Enterprise Manager 10g Database Control interface. The title bar reads "ORACLE Enterprise Manager 10g Database Control". The top menu includes "Setup", "Preferences", "Help", "Logout", and a "Database" dropdown. The URL in the address bar is "Database Instance: EDRSR14P1\_orcl.oracle.com > Tables > Edit Table: SH.SALES". The status bar indicates "Logged in As SYS". The main content area is titled "Edit Table: SH.SALES". Below it, a navigation bar has tabs: General, Constraints, Segments, Storage, Options, **Partitions**, Statistics, and Indexes. The "Partitions" tab is active. A sub-navigation bar below shows "Partitioning Method: Range" and "Partitioning Columns: TIME\_ID". A search bar for "Search by Partition Name" is followed by a "Go" button. Below this is a table with columns: "Select Partition Name", "High Value - TIME\_ID (DATE)", and "Tablespace". The table contains 14 rows of partition information. At the bottom of the table is a "Add Another Partition" button. Navigation controls at the bottom right include "Previous 1-10 of 28" and "Next 10". The footer is red with the "ORACLE" logo and the text "Copyright © 2008, Oracle. All rights reserved.".

| Select Partition Name                       | High Value - TIME_ID (DATE) | Tablespace |
|---------------------------------------------|-----------------------------|------------|
| <input checked="" type="radio"/> SALES_1995 | 1/1/1996                    | EXAMPLE    |
| <input type="radio"/> SALES_1996            | 1/1/1997                    | EXAMPLE    |
| <input type="radio"/> SALES_H1_1997         | 7/1/1997                    | EXAMPLE    |
| <input type="radio"/> SALES_H2_1997         | 1/1/1998                    | EXAMPLE    |
| <input type="radio"/> SALES_Q1_1998         | 4/1/1998                    | EXAMPLE    |
| <input type="radio"/> SALES_Q2_1998         | 7/1/1998                    | EXAMPLE    |
| <input type="radio"/> SALES_Q3_1998         | 10/1/1998                   | EXAMPLE    |
| <input type="radio"/> SALES_Q4_1998         | 1/1/1999                    | EXAMPLE    |
| <input type="radio"/> SALES_Q1_1999         | 4/1/1999                    | EXAMPLE    |
| <input type="radio"/> SALES_Q2_1999         | 7/1/1999                    | EXAMPLE    |
| <a href="#">Add Another Partition</a>       |                             |            |

## Partition Maintenance

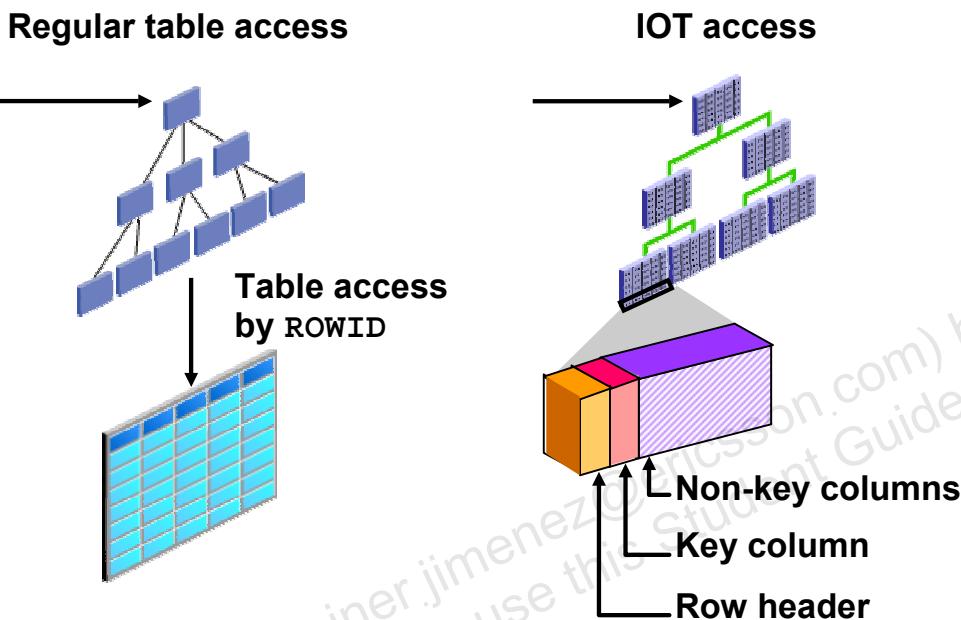
After a partitioned object has been created, you can maintain its partitions by using Database Control.

Select Administration > Tables. Click the partitioned table of interest. This displays the Edit Table page, where you can click the Partitions tab to open the page shown in the slide. You can select a partition of interest and apply a partition maintenance operation to it by selecting the operation from the Actions drop-down list.

**Note:** Use the Advanced Options button to change the storage characteristics of the selected partition.

# Index-Organized Tables

Table Types  
Partition  
> IOT  
Cluster  
DBA Tasks



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Index-Organized Tables

Unlike an ordinary (heap-organized) table whose data is stored as an unordered collection (heap), data for an index-organized table (IOT) is stored in a B-tree index structure in a primary key–sorted manner. Besides storing the primary key column values, each index entry in the IOT B-tree stores the non-key column values as well.

Index-organized tables have full table functionality. They support features such as constraints, triggers, LOB and object columns, partitioning, parallel operations, online reorganization, and replication. You can even create indexes on an index-organized table.

Index-organized tables are ideal for OLTP applications, which require fast primary key access and high availability. Queries and DML on an orders table used in online order processing are predominantly primary-key based, and a heavy volume of DML causes fragmentation that results in a frequent need to reorganize. Because an index-organized table can be reorganized online and without invalidating its secondary indexes, the window of unavailability is greatly reduced or eliminated.

An index-organized table is an alternative to:

- A table indexed on the primary key by using the CREATE INDEX statement
- A cluster table stored in an indexed cluster, which has been created using the CREATE CLUSTER statement that maps the primary key for the table to the cluster key

# Index-Organized Tables and Heap Tables

- **Compared to heap tables, IOTs:**
  - Have faster key-based access to table data
  - Do not duplicate the storage of primary key values
  - Require less storage
  - Use secondary indexes and logical row IDs
  - Have higher availability because table reorganization does not invalidate secondary indexes
- **IOTs have the following restrictions:**
  - Must have a primary key that is not DEFERRABLE
  - Cannot be clustered
  - Cannot use composite partitioning
  - Cannot contain a column of type ROWID or LONG

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Index-Organized Tables and Heap Tables

Index-organized tables do not have regular (physical) row IDs, but use *logical row IDs* instead. Logical row IDs give the fastest possible access to rows in IOTs by using two methods:

- A *physical guess* whose access time is equal to that of physical row IDs
- Access without the guess (or after an incorrect guess); this performs a primary key access of the IOT

The guess is based on knowledge of the file and block that a row resides in. The latter information is accurate when the index is created, but changes if the leaf block splits. If the guess is wrong and the row no longer resides in the specified block, then the remaining portion of the logical row ID entry, the primary key, is used to get the row.

The Oracle database constructs secondary indexes on index-organized tables by using logical row IDs that are based on the table's primary key. Because rows in index-organized tables do not have permanent physical addresses, the physical guesses can become stale when rows are moved to new blocks.

To obtain fresh guesses, you can rebuild the secondary index. Note that rebuilding a secondary index on an index-organized table involves reading the base table, unlike rebuilding an index on an ordinary table.

## **Index-Organized Tables and Heap Tables (continued)**

A quicker, more lightweight means of fixing the guesses is to use the ALTER INDEX . . . UPDATE BLOCK REFERENCES statement. This statement is performed online, while DML is still allowed on the underlying index-organized table.

After you rebuild a secondary index, or otherwise update the block references in the guesses, collect index statistics again.

The UROWID data type enables applications to use logical row IDs in the same way they use physical row IDs—for example, selecting row IDs for later update or as part of a cursor.

UROWID can also be used to store row IDs from other databases, accessed through gateways. The UROWID type can also be used to reference physical row IDs.

# Creating Index-Organized Tables

```

SQL> CREATE TABLE country
 2 (country_id CHAR(2)
 3 CONSTRAINT country_id_nn NOT NULL,
 4 country_name VARCHAR2(40),
 5 currency_name VARCHAR2(25),
 6 currency_symbol VARCHAR2(3),
 7 map BLOB,
 8 flag BLOB,
 9 CONSTRAINT country_c_id_pk
 10 PRIMARY KEY (country_id))
 11 ORGANIZATION INDEX
 12 TABLESPACE indx
 13 PCTTHRESHOLD 20
 14 OVERFLOW TABLESPACE users;

```

Copyright © 2008, Oracle. All rights reserved.

## Creating Index-Organized Tables

Regular B-tree index entries are usually small. They consist of only the primary key value and a row ID value for each row. Many of these small index entries can be stored in a leaf block. This is not necessarily the case for index-organized tables because they store full rows.

Storing large entries in index leaf blocks slows down index searches and scans. You can specify that the rows go into an overflow area by setting a threshold value that represents a percentage of block size.

The primary key column must always be stored in the IOT index blocks as a basis for searching. But you can place non-key values in a separate area—the row overflow area—so that the B-tree itself remains densely clustered.

Index-organized tables differ from ordinary tables only in physical organization. Logically, they are manipulated in the same manner as ordinary tables. You can specify an index-organized table just as you would specify a regular table in `INSERT`, `SELECT`, `DELETE`, and `UPDATE` statements.

All of the alter options available for ordinary tables are available for index-organized tables. This includes `ADD`, `MODIFY`, `DROP COLUMNS`, and `DROP CONSTRAINTS`. However, the primary key constraint for an index-organized table cannot be dropped, deferred, or disabled.

# Clusters

Table Types  
Partition  
IOT  
> Cluster  
DBA Tasks

| ORD_NO | PROD  | QTY | ... |
|--------|-------|-----|-----|
| 101    | A4102 | 20  |     |
| 102    | A2091 | 11  |     |
| 102    | G7830 | 20  |     |
| 102    | N9587 | 26  |     |
| 101    | A5675 | 19  |     |
| 101    | W0824 | 10  |     |

| ORD_NO | ORD_DT    | CUST_CD |
|--------|-----------|---------|
| 101    | 05-JAN-97 | R01     |
| 102    | 07-JAN-97 | N45     |

Unclustered orders and  
order\_item tables

| Cluster Key<br>(ORD_NO) |           |         |  |
|-------------------------|-----------|---------|--|
| 101                     | ORD DT    | CUST CD |  |
|                         | 05-JAN-97 | R01     |  |
| PROD                    | QTY       |         |  |
| A4102                   | 20        |         |  |
| A5675                   | 19        |         |  |
| W0824                   | 10        |         |  |
| 102                     | ORD DT    | CUST CD |  |
|                         | 07-JAN-97 | N45     |  |
| PROD                    | QTY       |         |  |
| A2091                   | 11        |         |  |
| G7830                   | 20        |         |  |
| N9587                   | 26        |         |  |

Clustered orders and  
order\_item tables

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Definition of Clusters

A cluster is a group of one or more tables that share the same data blocks because they share common columns and are often used together in join queries. Storing tables in clusters offers the DBA a method to denormalize data. If you implement clustered tables in your database, you do not need to change any application code that accesses the tables. Clusters are transparent to the end user and programmer.

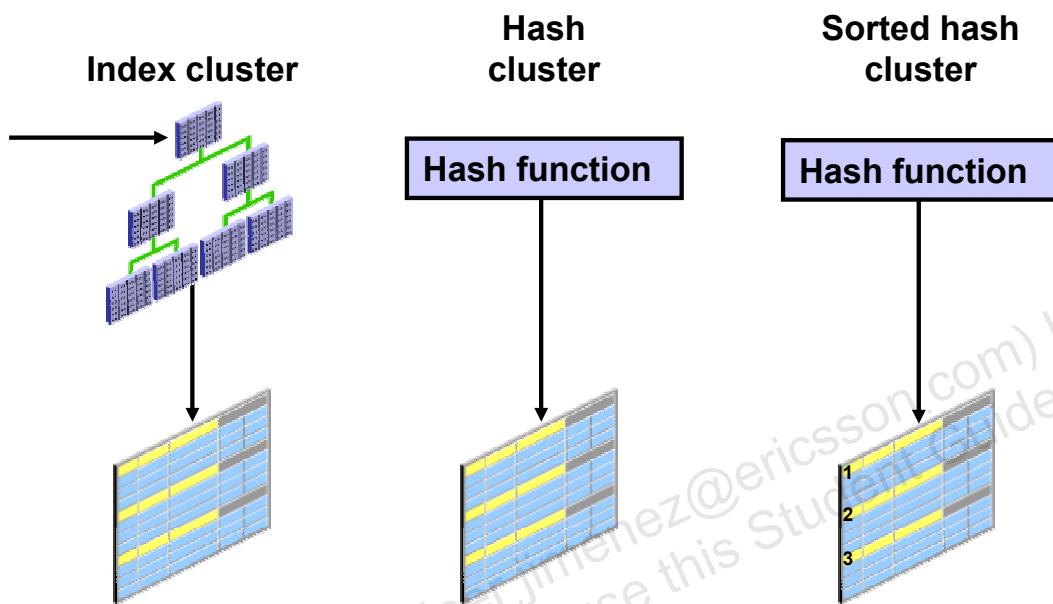
## Performance Benefits of Clusters

- Disk I/O is reduced and access time improved for joins of clustered tables.
- Each cluster key value is stored only once for all the rows of the same key value; therefore, it uses less storage space.

## Performance Consideration

Full table scans are generally slower on clustered tables than on nonclustered tables.

# Cluster Types



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Cluster Types

### Index Clusters

An index cluster uses an index, known as the cluster index, to maintain the data within the cluster. The cluster index must be available to store, access, or maintain data in an index cluster.

The cluster index is used to point to the block that contains the rows with a given key value. The structure of a cluster index is similar to that of a normal index.

Although a normal index does not store null key values, cluster indexes store null keys. There is only one entry for each key value in the cluster index. Therefore, a cluster index is likely to be smaller than a normal index on the same set of key values.

### Hash Clusters

A hash cluster uses a hash algorithm (either user-defined or system-generated) to calculate the location of a row, both for retrieval and for DML operations.

For equality searches that use the cluster key, a hash cluster can provide greater performance gains than an index cluster because there is only one segment to scan (no index access is needed).

## Cluster Types (continued)

### Sorted Hash Clusters

Inside a sorted hash cluster, the rows are organized in lists of sorted rows. Each list corresponds to a particular value of the hash key columns defined by the corresponding sorted hash cluster. Within each list, the rows are stored in the order specified by the sort key columns that are defined by the corresponding sorted hash cluster. This is also the default return order when querying a sorted hash cluster table by using the hash key columns in the predicate.

Sorted hash clusters offer the benefit of eliminating the CPU time and private memory needed to sort the data for queries that require a guaranteed returned order between SQL statements.

# Situations Where Clusters Are Useful

| Criterion                                             | Index | Hash | Sorted hash |
|-------------------------------------------------------|-------|------|-------------|
| <b>Uniform key distribution</b>                       | ✗     | ✗    | ✗           |
| <b>Evenly distributed key values</b>                  |       | ✗    | ✗           |
| <b>Rarely updated key</b>                             | ✗     | ✗    | ✗           |
| <b>Often joined master-detail tables</b>              | ✗     |      |             |
| <b>Predictable number of key values</b>               |       | ✗    | ✗           |
| <b>Queries using equality predicate on key</b>        |       | ✗    | ✗           |
| <b>Data is retrieved in the order it was inserted</b> |       |      | ✗           |

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Situations Where Clusters Are Useful

### When Not to Use Clusters

- If the application uses queries joining tables only occasionally or modifies their common column values frequently. Modifying a row's cluster key value takes longer than modifying the value in an unclustered table because the Oracle database might need to migrate the modified row to another block to maintain the cluster.
- If a full scan is executed often on only one of the clustered tables. This table is stored on more blocks than if it had been created alone.
- If the data for all rows of a cluster key value exceed one or two Oracle blocks. To access an individual row in a clustered key table, the Oracle server reads all blocks containing rows with the same value.

### When Not to Use Hash Clusters

- If the table is constantly growing or if the application frequently modifies the cluster key values
- If your application often performs full table scans and you must allocate a great deal of space to the hash cluster in anticipation of the table growing

Hash and index clusters require a lot of planning before being used. There may be more performance overhead involved for major operations such as bulk (direct path) inserts and rebuilds.

## Sorted Hash Cluster: Overview

- **New data structure used to store data sorted by nonprimary key columns:**
  - Cluster key values are hashed.
  - Rows corresponding to a particular cluster key value are sorted according to the sort key.
- **Used to guarantee that row order is returned by queries without sorting data:**
  - Rows are returned in ascending or descending order for a particular cluster key value.
  - ORDER BY clause is not mandatory to retrieve rows in ascending order.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Sorted Hash Cluster: Overview

When the Oracle database stores data in a heap-organized table, the rows are not stored in a user-controlled order. Rather, the decision about where to place a row is dependent on storage heuristics. The Oracle database does not guarantee the return order of the rows unless the query includes an ORDER BY clause.

Inside a sorted hash cluster, the rows are organized in lists of sorted rows. Each list corresponds to a particular value of the hash key columns defined by the corresponding sorted hash cluster. Within each list, the rows are stored in the order specified by the sort key columns defined by the corresponding sorted hash cluster. This is also the default return order when querying the table by using the hash key columns in the predicate.

Sorted hash clusters offer the benefit of eliminating the CPU time and private memory needed to sort the data for queries that require a guaranteed returned order between SQL statements.

When querying data in a sorted hash-clustered table by cluster key columns with an ORDER BY clause that references only the sort key columns or one of their prefixes, the optimizer avoids the sorting overhead because the rows are returned sorted by the sort key columns. However, for the same kind of queries, if you have an ORDER BY clause on a suffix of the sort key columns or nonsort key columns, additional sorting is required, assuming that no indexes are defined on the table. For this reason, when you create a sorted hash cluster, select its order key columns carefully.

## Sorted Hash Cluster: Example

```

CREATE CLUSTER calls_cluster
(origin_number NUMBER ← Cluster key
, call_timestamp NUMBER SORT } ← Sort key
, call_duration NUMBER SORT)
HASHKEYS 10000
SINGLE TABLE HASH IS origin_number
SIZE 50;

```

```

CREATE TABLE calls
(origin_number NUMBER
, call_timestamp NUMBER
, call_duration NUMBER
, other_info VARCHAR2(30))
CLUSTER calls_cluster(
origin_number,call_timestamp,call_duration
);

```

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

### Sorted Hash Cluster: Example

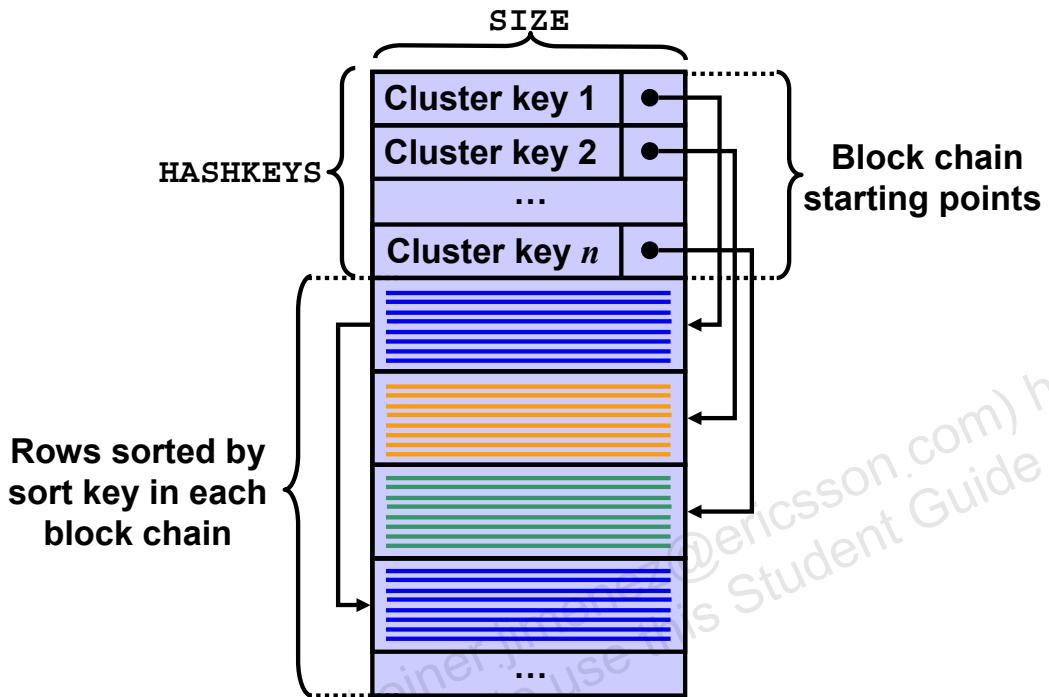
The way you define sorted hash clusters is very similar to the way you create hash clusters.

In the first step, you create the sorted hash cluster. As you can see, the most important difference with a traditional hash cluster is that you need to define the sort key columns in addition to the cluster key columns. Here, the cluster key is ORIGIN\_NUMBER, and the sort key columns are CALL\_TIMESTAMP and CALL\_DURATION. Another difference with traditional hash clusters is that for sorted hash clusters, the SIZE parameter specifies how many metadata entries to store for a particular hash key value. The size of one metadata entry is mainly determined by the size of the cluster key columns.

In the second step, you create the actual table specifying the link to the sorted hash cluster with the CLUSTER clause. You must specify the cluster key columns in the correct order followed by the sort key columns in the correct order. The example in the slide represents the following scenario: A telecommunications company needs to store call records for a fixed number of originating telephone numbers through a telecommunications switch. From each originating telephone number, there can be an unlimited number of telephone calls. Calls are stored as they are made and processed later in a “first-in, first-out” order when bills are generated for each originating telephone number. Each call is identified by a time-stamp number.

**Note:** Although the example uses a single table hash cluster, you can store more than one table in a sorted hash cluster. This is similar to traditional hash clusters.

## Sorted Hash Cluster: Basic Architecture



ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Sorted Hash Cluster: Basic Architecture

In the example in the previous slide, only one table is stored in the sorted hash cluster. Also, the hash function is very simple and no collision is expected. This is because the hash function is determined by the cluster key itself, and each cluster key value is unique. Basically, HASHKEYS represents the number of different originating telephone numbers, and SIZE represents the number of bytes used to store each cluster key metadata.

As you can see, the first part of the sorted hash cluster segment is reserved to store the metadata entries. Each metadata entry contains a link to the list of its corresponding rows. Each list is made up of a series of Oracle blocks that are linked together. Each list is sorted according to the sort key columns.

Whenever you want to retrieve the rows for a corresponding cluster key value, the cluster key value is hashed to its metadata entry location, which gives the sorted list of rows that you are selecting.

# Schema Management Tasks

Table Types  
Partition  
IOT  
Cluster  
> DBA Tasks

## DBA tasks include:

- **Estimating the size of new tables**
- **Analyzing growth trends**
- **Managing optimizer statistics**
- **Reorganizing schema objects online**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

# Estimating Resource Usage

The screenshot shows the Oracle Database 10g 'Create Table' dialog. In the 'General' tab, fields are filled with: Name (TEST\_REGIONS), Schema (HR), Tablespace (EXAMPLE), and Organization (Standard, Heap Organized). A red box highlights the 'Estimate Table Size' button, which is being clicked. An arrow points from this button to a sub-dialog titled 'Estimate Table Size'. This sub-dialog contains the following information:

- To get an accurate estimate for the size of the table, all table columns and storage parameters must be defined. The estimate of the table size is based on: column data types, column sizes, Free Space (PCTFREE). Extent information is used to calculate the space allocation impact on the tablespace.
- Projected Row Count: 44000 (highlighted with a red box)
- Estimated Result of Creating the Table in Tablespace EXAMPLE:
  - Table Size (MB): 1.00
  - Allocation Required in this Tablespace (MB): 1.00
- TIP Does not include space for LOB,IOT Overflow, Nested Table or Partition segments.

At the bottom right of the main dialog is the ORACLE logo.

## Estimating Resource Usage

The resource estimation feature enables you to estimate the amount of resources the creation of a new segment would require. Based on the structure of a table or index, and the number of estimated rows in the table, the Oracle database estimates the amount of disk space likely to be consumed by the object.

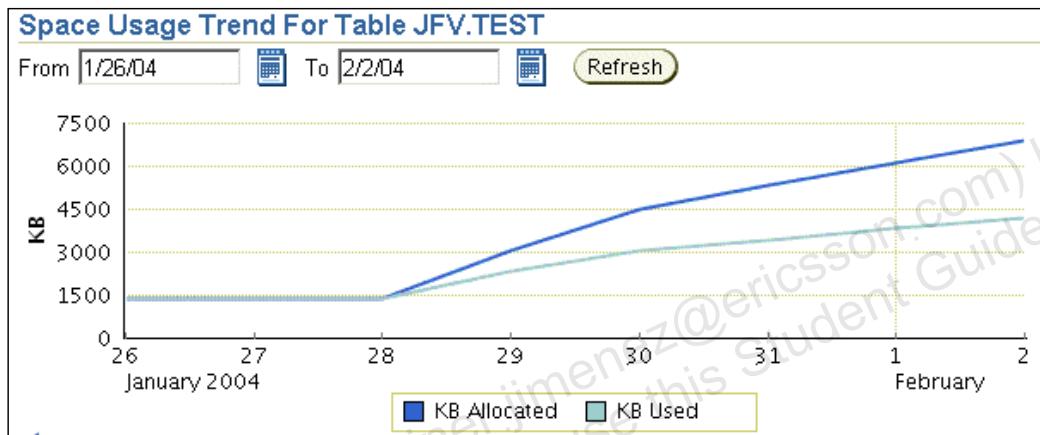
For example, the estimation of a table size is based on the following data: column data types, column sizes, and PCTFREE. Extent information is also used to calculate the space allocation impact on the currently selected tablespace.

To access this feature for tables, from the Administration page, select Tables in the Storage section, and then click Create to create the new table. You can also use the “Create like” functionality. Specify the tablespace and columns with data types and lengths, and then click the “Estimate Table Size” button. Now, specify the “Projected Row Count” and click the “Estimate Table Size” button to get both the estimated table size and the corresponding space allocated in the tablespace.

# Analyzing Growth Trends

## EM growth trend report:

- **Used by the Segment Advisor**
- **Space usage statistics collected into AWR**



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Analyzing Growth Trends

Segment growth reporting makes critical growth data available for advisories and reporting tools such as the Segment Advisor and the Growth Trend Report graph.

Some characteristics of the growth trend report feature include:

- Automatic workload repository data analysis. The workload repository collects the persistent space usage statistics and stores it in a system-defined schema. These statistics are captured at snapshot creation, and when alerts are triggered.
- Indications of past growth trend and predictions of future growth patterns
- Support for locally managed tablespaces only

The growth trend report consists of two components: growth history and growth forecast. The growth history charts past space usage data. The growth forecast predicts future space requirements by using history information and straight-line projection.

In Database Control, select Administration > Tables. Select your desired table, click Edit, and then click the Segments tabbed page.

The screenshot shows you such a report. You can choose the analysis period, and from the calculated graph, you can see the point in time at which you may have to allocate more space to your segment.

# Managing Optimizer Statistics

The screenshot shows the Oracle Enterprise Manager interface for managing optimizer statistics. A red box highlights the 'Actions' dropdown menu, which contains options like 'Create Like', 'Create Index', 'Create Synonym', etc., with 'Manage Optimizer Statistics' selected. A blue circle labeled '1' points to the 'Not analyzed' status of the TEST\_REGIONS table in the list. A blue circle labeled '2' points to the 'Actions' dropdown menu. A blue circle labeled '3' points to the 'Objects' tab in the wizard.

| Select | Schema ▾ | Table Name   | Tablespace | Partitioned | Rows | Last Analyzed                  |
|--------|----------|--------------|------------|-------------|------|--------------------------------|
| 1      | HR       | COUNTRIES    | EXAMPLE    | NO          |      | 25 Jul 7, 2005 2:00:01 AM PDT  |
| 2      | HR       | DEPARTMENTS  | EXAMPLE    | NO          |      | 27 Jul 7, 2005 2:00:03 AM PDT  |
| 3      | HR       | EMPLOYEES    | EXAMPLE    | NO          |      | 107 Jul 7, 2005 2:00:04 AM PDT |
| 4      | HR       | JOB HISTORY  | EXAMPLE    | NO          |      | 19 Jul 7, 2005 2:00:03 AM PDT  |
| 5      | HR       | LOCATIONS    | EXAMPLE    | NO          |      | 23 Jul 7, 2005 2:00:02 AM PDT  |
| 6      | HR       | REGIONS      | EXAMPLE    | NO          |      | 4 Jul 7, 2005 2:00:01 AM PDT   |
| 7      | HR       | TEST_REGIONS | EXAMPLE    | NO          |      | <b>Not analyzed</b>            |

Copyright © 2008, Oracle. All rights reserved.

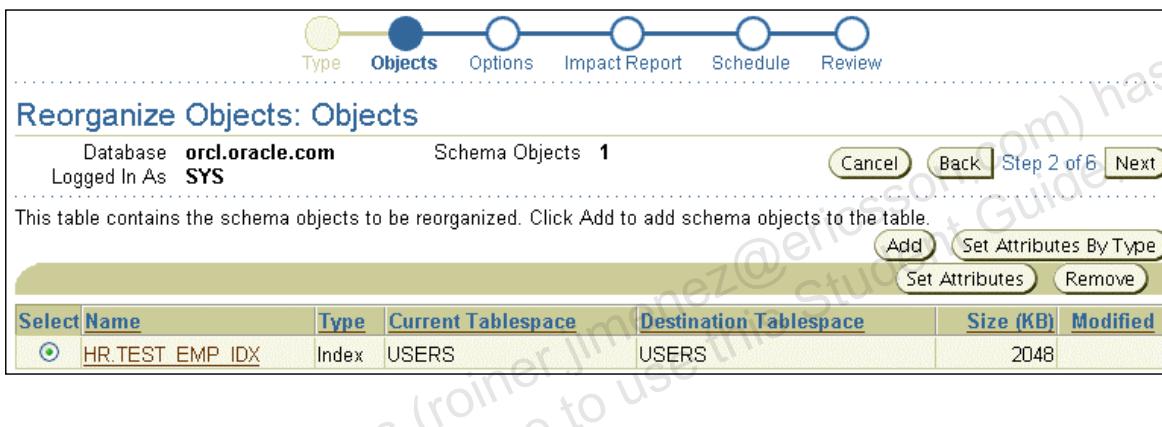
## Managing Optimizer Statistics

Best practices recommend that you gather statistics after major DML activities, such as after inserting 9,000 rows. The list of tables in the example (1) shows you that the TEST\_REGIONS table has not been analyzed.

In Enterprise Manager, choose Administration > Tables. Select “Manage Optimizer Statistics” from the Actions drop-down list (2), and follow the wizard (3) to create an Optimizer Statistics job.

# Reorganizing Schema Objects Online

- **Modifying logical or physical structure of a schema object, such as a table or index**
- **Transparent to users**
- **Space requirements**



## Reorganizing Schema Objects Online

Sometimes, you may need to modify the logical or physical structure of a table to manage storage, improve performance, or accommodate new applications. To perform this task, while the table remains accessible to both queries and DML, use the Reorganize Objects wizard in Enterprise Manager or the DBMS\_REDEFINITION package.

Reorganizing tables online provides a substantial increase in availability compared to traditional methods of redefining tables. The table is locked in exclusive mode only during a very small window that is independent of the size of the table and complexity of the redefinition, and that is completely transparent to users.

This process requires an amount of free space that is approximately equivalent to the space used by the table being redefined. More space may be required if new columns are added.

### Online reorganization can be used for:

- Rebuilding fragmented indexes
- Rebuilding fragmented tables
- Relocating objects to another tablespace
- Re-creating objects with better storage attributes

## Reorganizing Schema Objects Online (continued)

Options to customize your reorganization:

Unauthorized reproduction or distribution prohibited. Copyright© 2010, Oracle and/or its affiliates.

**Reorganize Objects: Options**

Database **orcl.oracle.com**      Schema Objects **1**      Logged In As **SYS**

**Cancel** **Back** **Step 3 of 6** **Next**

**Method**

Some object types can be reorganized online. With an online reorganization the objects have higher availability but the reorganization is slower. Do you want the reorganization to favor speed or availability?

Speed (offline) - object availability is not a concern  
 Availability (online) - object availability is important  
 Use ROWID method - adds a hidden column to tables

**Scratch Tablespace**

Reorganizations are performed inside the database and require sufficient free space. The scratch tablespace is used for intermediate storage of objects during reorganization.

Use current tablespace  
 Use scratch tablespace  
Tablespace

**▼ Hide Advanced Options**

**Object Parameters**

Use parallel execution when possible  
Parallel Degree  Default  Value   
 Rebuild indexes without logging for faster reorganization  
 Update any existing cost-based optimizer statistics  
     Compute statistics based on all the rows of the selected objects  
     Estimate statistics based on some of the rows of the selected objects. This method is faster but the statistic is less accurate.  
    Estimate Percentage  Default - Oracle determines the best sample size for good statistics  
                           Percentage   
 Specify number of rows inserted between commits when reorganizing tables with LONG columns  
Number of Rows

**Session Parameters**

Sort Area Size (KB)   
(Sort area size has a significant impact on the performance of index creation.)

# Reorganizing Objects: Impact Report

Reorganize Objects: Impact Report

Database **orcl.oracle.com** Schema Objects **1** Logged In As **SYS** **Cancel** **Back** **Step 4 of 6** **Next**

**Script Generation Summary**

Most Serious Message Severity **INFORMATION**  
Generation Started **Aug 21, 2005 5:48:58 PM**  
Generation Completed **Aug 21, 2005 5:49:01 PM**

**Script Generation Information**

The following table provides information about the objects and resources examined during script generation and lists details of any warnings or errors detected.

| Object Name | Object Type | Message Severity | Message Type | Message                                                                                                                                                         |
|-------------|-------------|------------------|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| USERS       | TABLESPACE  | INFORMATION      | Plan         | Sufficient free space in Tablespace USERS. Starting Freespace with automatic extension: 33544176KB. Ending Freespace: 33546160KB. Lowest Freespace: 33544112KB. |
| HR          | USER        | INFORMATION      | Plan         | Sufficient tablespace quota for User HR.                                                                                                                        |

**Printable Page**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Reorganizing Objects: Impact Report

Before any commands are executed, the impact report provides you with errors or warnings and an overview of planned activities, such as rebuilding an index.

# Reorganizing Objects: Review

The screenshot shows the 'Reorganize Objects: Review' dialog box. At the top, there is a navigation bar with four steps: 'Previous', 'Impact Report', 'Schedule', and 'Review'. The 'Review' step is highlighted with a blue circle. Below the navigation bar, the database name is listed as 'orcl.oracle.com' and the schema objects are '1'. The user is logged in as 'SYS'. The job name is 'REORGANIZE\_ORCL.ORACLE.COM\_21' and the job schedule is 'Run Immediately'. There are buttons for 'Cancel', 'Back', 'Step 6 of 6', and 'Submit Job'. A 'Script' section contains a summary of the commands that will be used for reorganization. It includes a 'Save Full Script' button. Below this, there are two options: 'Script Summary' (selected) and 'Full Script'. The 'Full Script' option shows the generated PL/SQL code:

```
-- Target database: orcl.oracle.com
-- Script generated at: 21-AUG-2005 17:48
ALTER INDEX "HR"."TEST_EMP_IDX" REBUILD
BEGIN DBMS_STATS.GATHER_INDEX_STATS("HR", "TEST_EMP_IDX", estimate_percent=>NULL); END;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Reorganizing Objects: Review

For a detailed understanding of the commands that will be executed, select Full Script and view all the relevant statements. If you prefer a simpler overview of this reorganization, select Script Summary.

## Basic Steps for Manual Online Reorganization

1. Verify that the table is a candidate for online reorganization.
2. Create an interim table.
3. Start the redefinition process.
4. Copy dependent objects. (This automatically creates any triggers, indexes, grants, and constraints on the interim table.)
5. Query the DBA\_REDEFINITION\_ERRORS view to check for errors.
6. Optionally, synchronize the interim table.
7. Complete the redefinition.
8. Drop the interim table.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Basic Steps for Manual Online Reorganization

Commands and procedures used:

1. DBMS\_REDEFINITION.CAN\_REDEF\_TABLE
2. CREATE TABLE ...
3. DBMS\_REDEFINITION.START\_REDEF\_TABLE
4. DBMS\_REDEFINITION.COPY\_TABLE\_DEPENDENTS and DBMS\_REDEFINITION.CONS\_ORIG\_PARAMS
5. SELECT object\_name, base\_table\_name, ddl\_txt FROM DBA\_REDEFINITION\_ERRORS;
6. DBMS\_REDEFINITION.SYNC\_INTERIM\_TABLE
7. DBMS\_REDEFINITION.FINISH\_REDEF\_TABLE
8. DROP TABLE ... PURGE

# **Summary**

**In this lesson, you should have learned how to manage schema objects and:**

- **Determine appropriate table types for your requirements**
- **Perform related DBA tasks:**
  - Estimating the size of new tables
  - Analyzing growth trends
  - Managing optimizer statistics
  - Reorganizing schema objects online

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

# Practice Overview: Managing Schema Objects

**This practice covers the following topics:**

- **Monitoring table and index space usage**
- **Managing optimizer statistics**
- **Reorganizing table and index**



Copyright © 2008, Oracle. All rights reserved.

# 11

## Managing Storage

ORACLE®

Copyright © 2008, Oracle. All rights reserved.

# Objectives

- Space Management
- Proactive Monitoring
- Seg. Advisor & Seg.Shrink
- Resumable Allocation
- Transportable TBS and DB

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

- **Describe how the Oracle database automatically manages space**
- **Proactively monitor and manage tablespace space usage**
- **Use the Segment Advisor**
- **Reclaim wasted space from tables and indexes by using the segment shrink functionality**
- **Manage resumable space allocation**
- **Describe the concepts of transportable tablespaces and databases**



Copyright © 2008, Oracle. All rights reserved.

## Space Management: Overview

**Space is automatically managed by the Oracle database. It generates alerts about potential problems and recommends possible solutions. Features include:**

- **Oracle Managed Files (OMF)**
- **Free-space management with bitmaps (“locally managed”) and automatic data file extension**
- **Proactive space management (default thresholds and server-generated alerts)**
- **Space reclamation (shrinking segments, online table redefinition)**
- **Capacity planning (growth reports)**



Copyright © 2008, Oracle. All rights reserved.

### Space Management: Overview

With Oracle Managed Files (OMF), you can specify operations in terms of database objects rather than file names. For more details, see the lesson titled “Introduction.”

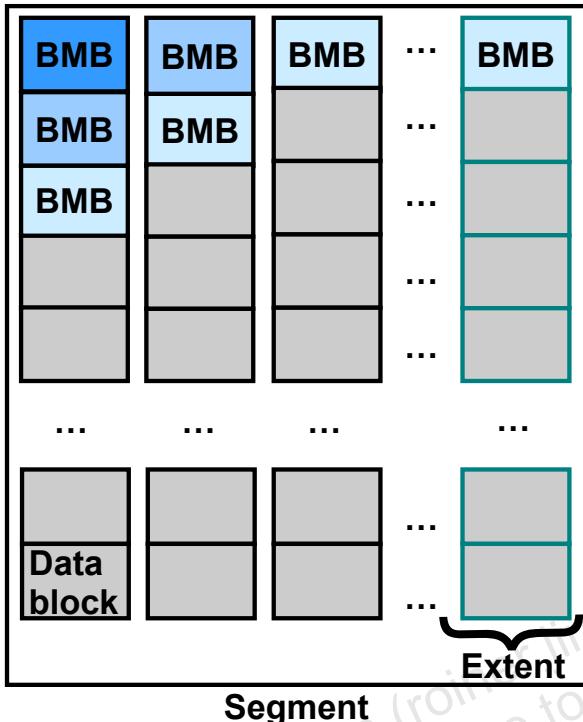
The Oracle database manages free space within a table with bitmaps. This is known as a “locally managed” tablespace. (Dictionary-managed tablespaces are supported only for backward compatibility.) The bitmapped implementation eliminates much space-related tuning of tables, while providing improved performance during peak loads. Additionally, the Oracle database provides automatic extension of data files, so the files can grow automatically based on the amount of data in the files.

When you create a database, proactive space monitoring is enabled by default. (This causes no performance impact.) The Oracle database monitors space utilization during normal space allocation and deallocation operations and alerts you if the free space availability falls below the predefined thresholds (which you can override). Advisors and wizards assist you with space reclamation.

For capacity planning, the Oracle database provides space estimates based on table structure and number of rows and a growth trend report based on historical space utilization stored in the Automatic Workload Repository (AWR).

The *Oracle Database 10g: Administration Workshop I* course provides an introduction to space and storage concepts, related utilities, and DBA tasks. Through this or other means, you should be familiar with the basic concepts and storage features.

# Free Space Management



- Automatic
- Enabled by the use of locally managed tablespaces
- Tracked by bitmaps in segments

## Benefits:

- More flexible space utilization
- Run-time adjustment
- Multiple process search of BMBs

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Free Space Management

Free space can be managed automatically inside database segments. The in-segment free or used space is tracked with bitmaps. You specify Automatic Segment Space Management, when you create a locally managed tablespace. Your specification then applies to all segments subsequently created in this tablespace.

Automatic space management segments have a set of bitmap blocks (BMBs) describing the space utilization of the data blocks in that segment. BMBs are organized in a tree hierarchy. The root level of the hierarchy, which contains the references to all intermediate BMBs, is stored in the segment header. The leaves of this hierarchy represent the space information for a set of contiguous data blocks that belong to the segment. The maximum number of levels inside this hierarchy is three.

Benefits of using automatic space management (compared to manual space management, which uses “freelist” data structures and is synonymous with “dictionary-managed” tablespaces):

- Better space utilization, especially for the objects with highly varying row sizes
- Better run-time adjustment to variations in concurrent access
- Better multi-instance behavior in terms of performance or space utilization

Therefore, less work for you, the DBA.

# Types of Segments

- **A segment is a set of extents allocated for a certain logical structure. The different types of segments are:**
  - Data segment
  - Index segment
  - Temporary segment
  - Rollback segment
- **Segments are dynamically allocated by the database.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Types of Segments

**Data segment:** Each nonclustered table has a data segment. All table data is stored in the extents of the data segment. For a partitioned table, each partition has a data segment. Each cluster has a data segment. The data of every table in the cluster is stored in the cluster's data segment.

**Index segment:** Each index has an index segment that stores all of its data. For a partitioned index, each partition has an index segment.

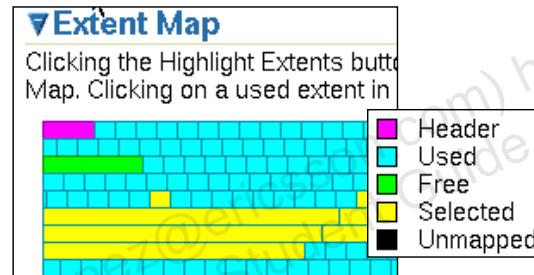
**Temporary segment:** A temporary segment is created by the Oracle database when a SQL statement needs a temporary database area to complete execution. When the statement finishes execution, the extents in the temporary segment are returned to the system for future use.

The Oracle database dynamically allocates space when the existing extents of a segment become full. Because extents are allocated as needed, the extents of a segment may or may not be contiguous on disk.

**Rollback segment:** Rollback segments exist to support transaction rollback and read consistency if you are not using automatic undo management mode. Oracle recommends you use undo tablespaces to manage undo rather than rollback segments.

# Allocating Extents

- **Searching the data file's bitmap for the required number of adjacent free blocks**
- **Sizing extents with storage clauses:**
  - UNIFORM
  - AUTOALLOCATE
- **Viewing extent map**
- **Obtaining deallocation advice**



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Allocating Extents

With locally managed tablespaces, the Oracle database looks for free space to allocate to a new extent by first determining a candidate data file in the tablespace and then searching the data file's bitmap for the required number of adjacent free blocks. If that data file does not have enough adjacent free space, then the Oracle database looks in another data file.

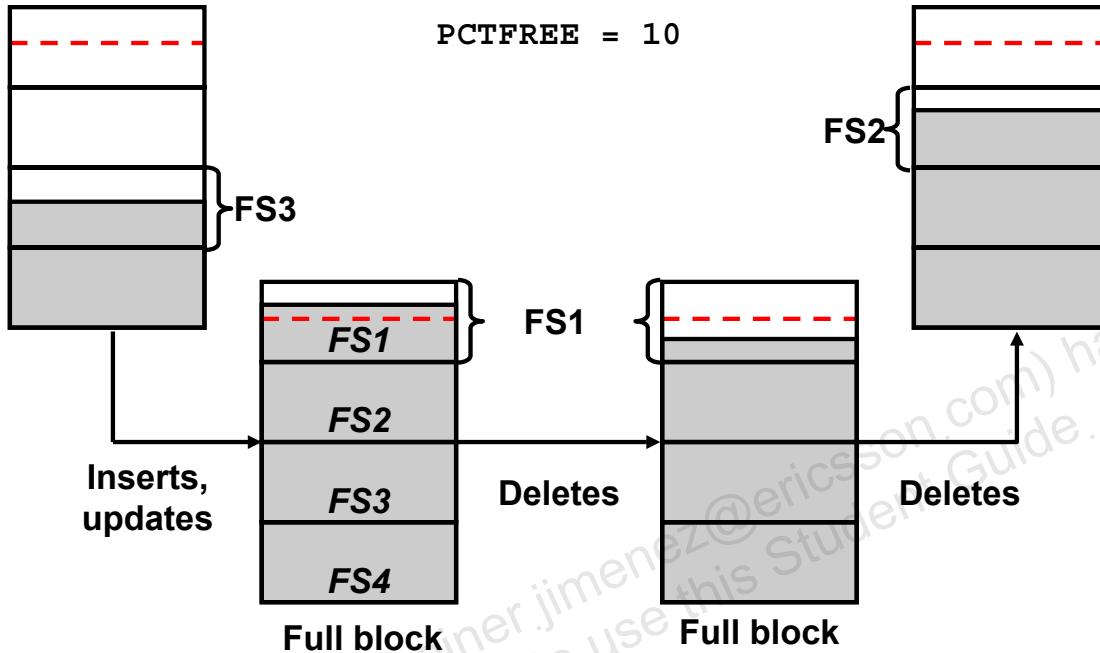
Two clauses affect the sizing of extents:

- With the UNIFORM clause, the database creates all extents of a uniform size that you specified (or a default size) for any objects created in the tablespace.
- With the AUTOALLOCATE clause, the database determines the extent-sizing policy for the tablespace.

To view the extent map in Enterprise Manager, choose Administration > Tablespaces > View Tablespace > Show Tablespace Contents.

The Oracle database provides a Segment Advisor that helps you determine whether an object has space available for reclamation on the basis of the level of space fragmentation within the object.

# Block Space Management



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Block Space Management

Space management involves the management of free space at the block level. With Automatic Segment Space Management, each block is divided into four sections, named FS1 (between 0 and 25% of free space), FS2 (25% to 50% free), FS3 (50% to 75% free), and FS4 (75% to 100% free).

Depending on the level of free space in the block, its status is automatically updated. That way, depending on the length of an inserted row, you can tell whether a particular block can be used to satisfy an insert operation. Note that a “full” status means that a block is no longer available for inserts.

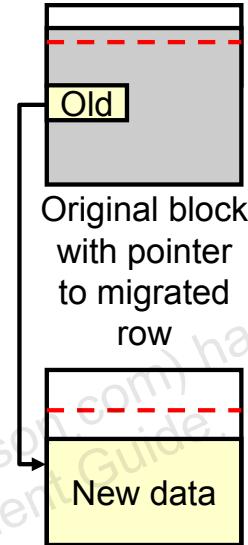
In the slide example, the block on the left is an FS3 block because it has between 50% and 75% free space. After some insert and update statements, PCTFREE is reached (the dashed line) and it is no longer possible to insert new rows in that block. The block is now considered as a “full” or FS1 block. The block is considered for insertion again, as soon as its free space level drops below the next section. In the above case, it gets status FS2 as soon as the free space is more than 25%.

**Note:** Large object (LOB) data types (BLOB, CLOB, NCLOB, and BFILE) do not use the PCTFREE storage parameter. For more information, see the *Oracle Database Application Developer's Guide - Large Objects 10g Release 2 (10.2)*.

# Row Chaining and Migration

## Example:

- **On update: Row length increases, exceeding the available free space in the block.**
- **Data needs to be stored in a new block.**
- **Original physical identifier of row (ROWID) is preserved.**
- **The Oracle database needs to read two blocks to retrieve data.**
- **The Segment Advisor finds segments containing the migrated rows.**



**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Row Chaining and Migrating

In two circumstances, the data for a row in a table may be too large to fit into a single data block. In the first case, the row is too large to fit into one data block when it is first inserted. In this case, the Oracle database stores the data for the row in a chain of data blocks (one or more) reserved for that segment. Row chaining most often occurs with large rows, such as rows that contain a column of data type LONG or LONG RAW. Row chaining in these cases is unavoidable.

However, in the second case, a row that originally fit into one data block is updated, so that the overall row length increases, and the block's free space is already completely filled. In this case, the Oracle database migrates the data for the entire row to a new data block, assuming that the entire row can fit in a new block. The database preserves the original row piece of a migrated row to point to the new block containing the migrated row. The ROWID of a migrated row does not change.

When a row is chained or migrated, I/O performance associated with this row decreases because the Oracle database must scan more than one data block to retrieve the information for the row. The Segment Advisor finds the segments containing migrated rows that result from an UPDATE.

# Proactive Tablespace Monitoring

**Edit Tablespace: EXAMPLE**

Actions Add Datafile Go Show SQL Revert Apply

General Storage Thresholds

Name EXAMPLE

Bigfile tablespace No

| Extent Management                                                                            | Type                                                                                                       | Status                                                                                                                                    |
|----------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| <input checked="" type="radio"/> Locally Managed<br><input type="radio"/> Dictionary Managed | <input checked="" type="radio"/> Permanent<br><input type="checkbox"/> Set as default permanent tablespace | <input checked="" type="radio"/> Read Write<br><input type="radio"/> Read Only<br><input type="radio"/> Offline<br>Offline Mode<br>Normal |
|                                                                                              | <input type="radio"/> Temporary<br><input type="checkbox"/> Set as default                                 |                                                                                                                                           |

**Datafiles**

Select Name example01

**Space Used (%)**

A warning or critical alert will be generated if the percentage of space used exceeds the corresponding threshold.

Use Database Default Thresholds [Modify](#)  
 Warning (%) 85  
 Critical (%) 97

Specify Thresholds  
 Warning (%) 85  
 Critical (%) 97

Disable Thresholds

Space Management > Proactive Monitoring Seg. Advisor & Seg.Shrink Resumable Allocation Transportable TBS and DB

General Storage Thresholds

**Extent Allocation**

Allocation Type Automatic

**Segment Space Management**

Type Automatic

**Enable logging**

Yes  
 Generate redo logs for creation of tables, indexes and partitions, and for subsequent inserts. Recoverable

No  
 Redo log entries are smaller, the above operations are not logged and not recoverable.

**Block information**

Block Size (B) 8192

ORACLE

Copyright © 2008, Oracle. All rights reserved.

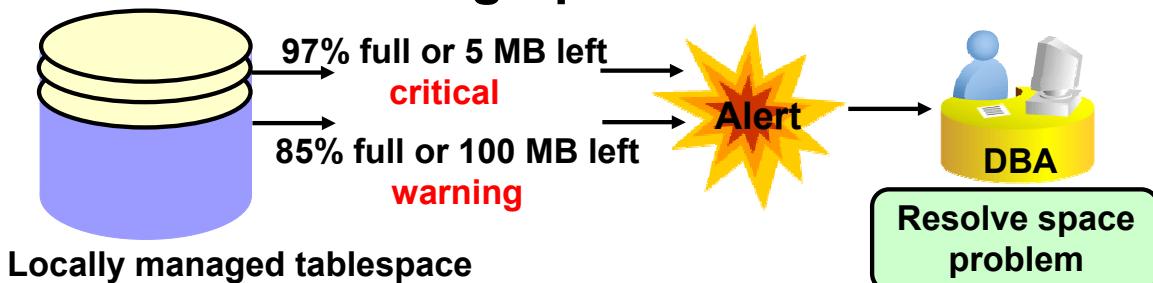
## Proactive Tablespace Monitoring

Tablespace disk space usage is proactively managed by the database in the following ways:

- Through the use of database alerts, you are informed when a tablespace runs low on available disk space as well as when particular segments are running out of space. You can then provide the tablespace with more disk space, thus avoiding out-of-space conditions.
- Information gathered is stored in the Automatic Workload Repository (AWR) and is used to perform growth trend analysis and capacity planning of the database.

To view and modify tablespace information in Enterprise Manager, select Administration > Tablespaces. Select the tablespace of your choice and click the Edit button.

## Thresholds and Resolving Space Problems



### Resolve space problem by:

- Adding or resizing data file
- Setting AUTOEXTEND ON
- Shrinking objects
- Reducing UNDO\_RETENTION
- Checking for long-running queries in temporary tablespaces

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Thresholds and Resolving Space Problems

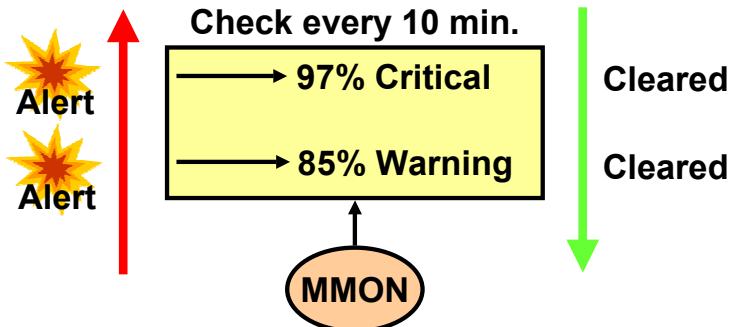
Tablespace thresholds are defined either as full or as available space in the tablespace. Critical and warning thresholds are the two thresholds that apply to a tablespace. The DBMS\_SERVER\_ALERT package contains procedures to set and get the threshold values. When the tablespace limits are reached, an appropriate alert is raised. The threshold is expressed in terms of a percentage of the tablespace size or in remaining bytes free. It is calculated in memory. You can have both a percentage and a byte-based threshold defined for a tablespace. Either or both of them may generate an alert.

The ideal setting for the warning threshold trigger value results in an alert that is early enough to ensure that there is enough time to resolve the problem before it becomes critical, but late enough so that you are not bothered when space is not a problem.

The alert indicates that the problem can be resolved by doing one or more of the following:

- Adding more space to the tablespace by adding a file or resizing existing files, or making an existing file autoextendable
- Freeing up space on disks that contain any autoextendable files
- Shrinking sparse objects in the tablespace

# Monitoring Tablespace Space Usage



- **Read-only and offline tablespaces:** Do not set up alerts.
- **Temporary tablespace:** Threshold corresponds to space currently used by sessions.
- **Undo tablespace:** Threshold corresponds to space used by active and unexpired extents.
- **Autoextensible files:** Threshold is based on the maximum file size.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

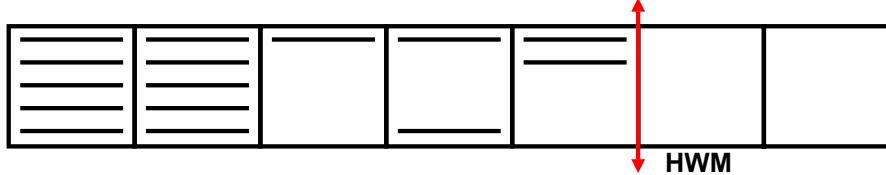
## Monitoring Tablespace Space Usage

The database tracks space utilization while performing regular space management activities. This information is aggregated every 10 minutes by the MMON process. An alert is triggered when the threshold for a tablespace has been reached or cleared.

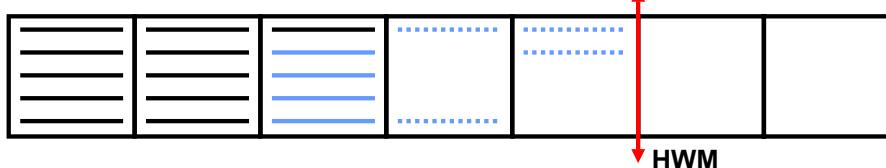
- Alerts should not be flagged on tablespaces that are in read-only mode, or tablespaces that were taken offline, because there is not much to do for them.
- In temporary tablespaces, the threshold value has to be defined as a limit on the used space in the tablespace.
- For undo tablespaces, an extent is reusable if it does not contain active or unexpired undo. For the computation of threshold violation, the sum of active and unexpired extents is considered as used space.
- For tablespaces with autoextensible files, the thresholds are computed according to the maximum file size you specified, or the maximum OS file size.

# Shrinking Segments

Space Management  
Proactive Monitoring  
-> Seg. Advisor & Seg.Shrink  
Resumable Allocation  
Transportable TBS and DB

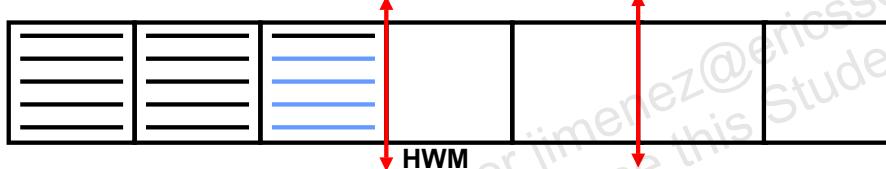


① `ALTER TABLE employees SHRINK SPACE COMPACT;`



DML operations and queries can be issued during compaction.

② `ALTER TABLE employees SHRINK SPACE;`



DML operations are blocked when the HWM is adjusted.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

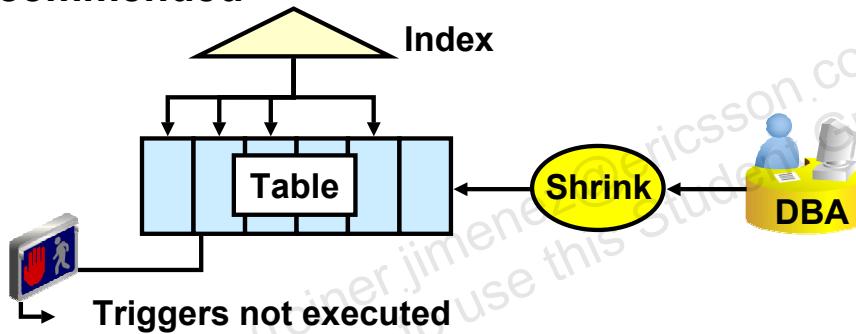
## Shrinking Segments

The diagram in the slide describes the two phases of a table shrink operation. The first phase does the compaction. During this phase, rows are moved to the left part of the segment as much as possible. Internally, rows are moved by packets to avoid locking issues. After the rows have been moved, the second phase of the shrink operation is started. During this phase, the high-water mark (HWM) is adjusted and the unused space is released.

The COMPACT clause is useful if you have long-running queries that might span the shrink operation and attempt to read from blocks that have been reclaimed. When you specify the SHRINK SPACE COMPACT clause, the progress of the shrink operation is saved in the bitmap blocks of the corresponding segment. This means that the next time a shrink operation is executed on the same segment, the Oracle database remembers what has been done already. You can then reissue the SHRINK SPACE clause without the COMPACT clause during off-peak hours to complete the second phase.

# Results of Shrink Operation

- Improved performance and space utilization
- Indexes maintained
- Triggers not executed
- Number of migrated rows may be reduced
- Rebuilding secondary indexes on IOTs recommended



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Results of Shrink Operation

Shrinking a sparsely populated segment improves the performance of scan and DML operations on that segment. This is because there are fewer blocks to look at after the segment has been shrunk. This is especially true for:

- Full table scans (fewer and denser blocks)
- Better index access (fewer I/Os on range ROWID scans due to a more compact tree)

Also, by shrinking sparsely populated segments, you enhance the efficiency of space utilization inside your database because more free space is made available for objects in need.

Index dependency is taken care of during the segment shrink operation. The indexes are in a usable state after shrinking the corresponding table. Therefore, no further maintenance is needed.

The actual shrink operation is handled internally as an INSERT/DELETE operation. However, any DML triggers are not executed because the data itself is not changed.

As a result of a segment shrink operation, it is possible that the number of migrated rows is reduced. However, you should not always depend on reducing the number of migrated rows after a segment has been shrunk. This is because a segment shrink operation may not touch all the blocks in the segment. Therefore, it is not guaranteed that all the migrated rows are handled.

**Note:** It is recommended to rebuild secondary indexes on an index-organized table (IOT) after a shrink operation.

## Space Reclamation with ASSM

- **Online and in-place operation**
- **Applicable only to segments residing in ASSM tablespaces**
- **Candidate segment types:**
  - **Heap-organized tables and index-organized tables**
  - **Indexes**
  - **Partitions and subpartitions**
  - **Materialized views and materialized view logs**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Space Reclamation with ASSM

A shrink operation is an online and in-place operation because it does not need extra database space to be executed.

- You cannot execute a shrink operation on segments managed by free lists. Segments in automatic segment-space managed tablespaces can be shrunk. However, the following objects stored in ASSM tablespaces cannot be shrunk:
  - Tables in clusters
  - Tables with LONG columns
  - Tables with on-commit materialized views
  - Tables with ROWID-based materialized views
  - IOT mapping tables
  - Tables with function-based indexes
- ROW MOVEMENT must be enabled for heap-organized segments.

**Note:** Automatic Segment Space Management (ASSM) is the default type of segment space management for all new, permanent, locally managed tablespaces in Oracle Database 10g Release 2.

# Segment Advisor: Overview

The screenshot shows the Segment Advisor: Scope page. At the top, there is a navigation bar with four tabs: Scope (which is highlighted in blue), Objects, Schedule, and Review. Below the navigation bar, there is a section titled "Automatic Segment Advisor Information" with a blue info icon. It contains text about the Automatic Segment Advisor job and a link to view detected issues. Underneath this is a section titled "Segment Advisor Recommendations". The main content area has a title "Segment Advisor: Scope" and displays the database name "orcl.oracle.com" and the user "Logged In As SYS". On the right side, there are "Cancel", "Step 1 of 4", and "Next" buttons. A callout box labeled "Overview" provides a detailed explanation of how the segment advisor determines unused space for shrinking. At the bottom of the page, there is a red footer bar with the "ORACLE" logo and the copyright notice "Copyright © 2008, Oracle. All rights reserved."

## Segment Advisor: Overview

The Segment Advisor determines whether an object is a good candidate for a shrink operation. The advisor also finds the segments containing migrated rows that result from an UPDATE. (Beginning with Oracle Database 10.2, the Segment Advisor jobs are automatically run for you.) The advisor makes recommendations based on the amount of unused space that can be released, and takes into consideration estimated future space requirements by using criteria from the gathered information about segment growth trends.

After the recommendations are made, you can choose to implement the recommendations. The shrink advisor can be invoked at the segment or tablespace level.

The EM Database Control Console is the interface to the Segment Advisor. You can access the Segment Advisor from several places within EM:

- Advisor Central page
- Tablespaces page
- Schema object pages

The Database Control Console provides the option to select various inputs and schedule a job that calls the Segment Advisor to get shrink advice. The Segment Advisor wizard can be invoked with no context, in the context of a tablespace, or in the context of a schema object.

The Segment Advisor makes recommendation on the basis of sampled analysis, historical information, and future growth trends.

# Segment Advisor

**Segment Advisor: Review**

|                                   |                                                |              |     |        |          |      |             |        |
|-----------------------------------|------------------------------------------------|--------------|-----|--------|----------|------|-------------|--------|
| Database                          | orcl.oracle.com                                | Logged In As | SYS | Cancel | Show SQL | Back | Step 4 of 4 | Submit |
| Task Name                         | SEGMENTADV_3712320                             |              |     |        |          |      |             |        |
| Task Description                  | Get shrink advice based on object growth trend |              |     |        |          |      |             |        |
| Time Limit for Analysis (mins)    | Unlimited                                      |              |     |        |          |      |             |        |
| Advisory Results Retention (days) | 30                                             |              |     |        |          |      |             |        |

**Selected Objects**

| Tablespace | Type      |
|------------|-----------|
| EXAMPLE    | PERMANENT |

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Segment Advisor

From the Administration page, select Tablespaces in the Storage section. On the Tablespaces page, select the tablespace on which you want to perform the shrink analysis, and then select Run Segment Advisor in the Actions drop-down list. Click Go to open the Segment Advisor initial page. You must choose “comprehensive” or “limited” analysis mode. In comprehensive mode, the analysis is longer because the advisor is sampling the segments to identify the right targets.

Keep clicking Continue to answer the various questions of the advisor. You end up on the Segment Advisor: Review page, where you can review the details of your analysis. The Segment Advisor analysis is run as a scheduled job, so you can review the scheduled task from the Advisor Central page. When completed, you can review the advisor’s recommendations.

**Note:** In the Segment Advisor, you can specify the duration of the analysis. This enables you to limit the time the advisor takes to produce recommendations. Generally speaking, a longer analysis period produces more comprehensive results. The results are stored in the AWR and can be viewed later. Use the “Number of days to retain” option to instruct the Oracle database how long these results should be preserved before being purged from the AWR.

## Segment Advisor (continued)



Execute task script

### Review: Show SQL

[Return](#)

#### Create task and objects script

```
DECLARE

taskname varchar2(100);
taskdesc varchar2(128);
task_id number;
object_id number;
timeLimit varchar2(25);
numDaysToRetain varchar2(25);
objectName varchar2(100);
objectType varchar2(100);

BEGIN
taskname := 'SEGMENTADV_3712320';
taskdesc := 'Get shrink advice based on object growth trend';
numDaysToRetain := '30';
dbms_advisor.create_task('Segment Advisor', ?, taskname, taskdesc, NULL);
dbms_advisor.create_object(taskname, 'TABLESPACE', 'EXAMPLE', ' ', ' ',
NULL, object_id);
dbms_advisor.set_task_parameter(taskname, 'RECOMMEND_ALL', 'TRUE');
dbms_advisor.set_task_parameter(taskname, 'DAYS_TO_EXPIRE',
numDaysToRetain);
END;
```

#### Execute task script

[Return to Top](#)

```
DECLARE
taskname varchar2(100);
BEGIN
taskname := 'SEGMENTADV_3712320';
dbms_advisor.reset_task(taskname);
dbms_advisor.execute_task(taskname);
END;
```

# Implementing Recommendations

Database Instance: orcl.oracle.com > Advisor Central > Segment Advisor Task: SEGMENTADV\_2730408

## Segment Advisor Task: SEGMENTADV\_2730408

The following table contains the minimum reclaimable space summary for the evaluated segments in that tablespace. Based on growth trends, the advisor takes into consideration estimated future space requirements. Oracle recommends shrinking or reorganizing these segments to release wasted space. Select the Recommendation Details button to view and implement the recommendations.

Task Name **SEGMENTADV\_2730408**

Started Aug 25, 2005 10:04:55 AM

Status **COMPLETED**

Ended Aug 25, 2005 10:05:09 AM

Running Time (seconds) **14**

Time Limit (mins) **UNLIMITED**

[Recommendation Details](#)

| Select                              | Tablespace | Recommendations | Tablespace Size (MB) | Evaluated Space (%) | Reclaimable Space (MB) | Extent Management | Segment Space Management |
|-------------------------------------|------------|-----------------|----------------------|---------------------|------------------------|-------------------|--------------------------|
| <input checked="" type="checkbox"/> | TBSALERT   | 3               | 120.00               | 57.50               | 34.46 LOCAL            | AUTO              |                          |

### Recommendation Details for Tablespace: TBSALERT

The following table contains the reclaimable space information for the evaluated segments in the selected tablespace. Based on growth trends, the advisor takes into consideration estimated future space requirements. Oracle recommends shrinking or reorganizing these segments to release wasted space. Select the segment to implement the recommendation.

Task Name **SEGMENTADV\_2730408**

Started Aug 25, 2005 10:04:55 AM

Status **COMPLETED**

Ended Aug 25, 2005 10:05:09 AM

Running Time (seconds) **14**

Time Limit (mins) **UNLIMITED**

| Schema                              | Segment                     | Partition  | Minimum Reclaimable Space (MB) | Search                 |
|-------------------------------------|-----------------------------|------------|--------------------------------|------------------------|
| <a href="#">Implement</a>           |                             |            |                                |                        |
| <a href="#">Select All</a>          | <a href="#">Select None</a> |            |                                |                        |
| Select                              | Schema                      | Segment    | Recommendation                 | Reclaimable Space (MB) |
| <input checked="" type="checkbox"/> | SYS                         | EMPLOYEES1 | <a href="#">Shrink</a>         | 11.40                  |
| <input checked="" type="checkbox"/> | SYS                         | EMPLOYEES2 | <a href="#">Shrink</a>         | 22.00                  |
| <input checked="" type="checkbox"/> | SYS                         | EMPLOYEES3 | <a href="#">Shrink</a>         | 11.51 TABLE            |

```
alter table "SYS"."EMPLOYEES1" shrink space
alter table "SYS"."EMPLOYEES2" shrink space
alter table "SYS"."EMPLOYEES3" shrink space
```

**ORACLE®**

Copyright © 2008, Oracle. All rights reserved.

## Implementing Recommendations

After the Segment Advisor completes its job, you can view the recommendation details and implement them directly.

**Note:** Before shrinking a heap-organized table, you must enable row movement on that table. You can do this with Database Control from the Options tab on the Edit Table page.

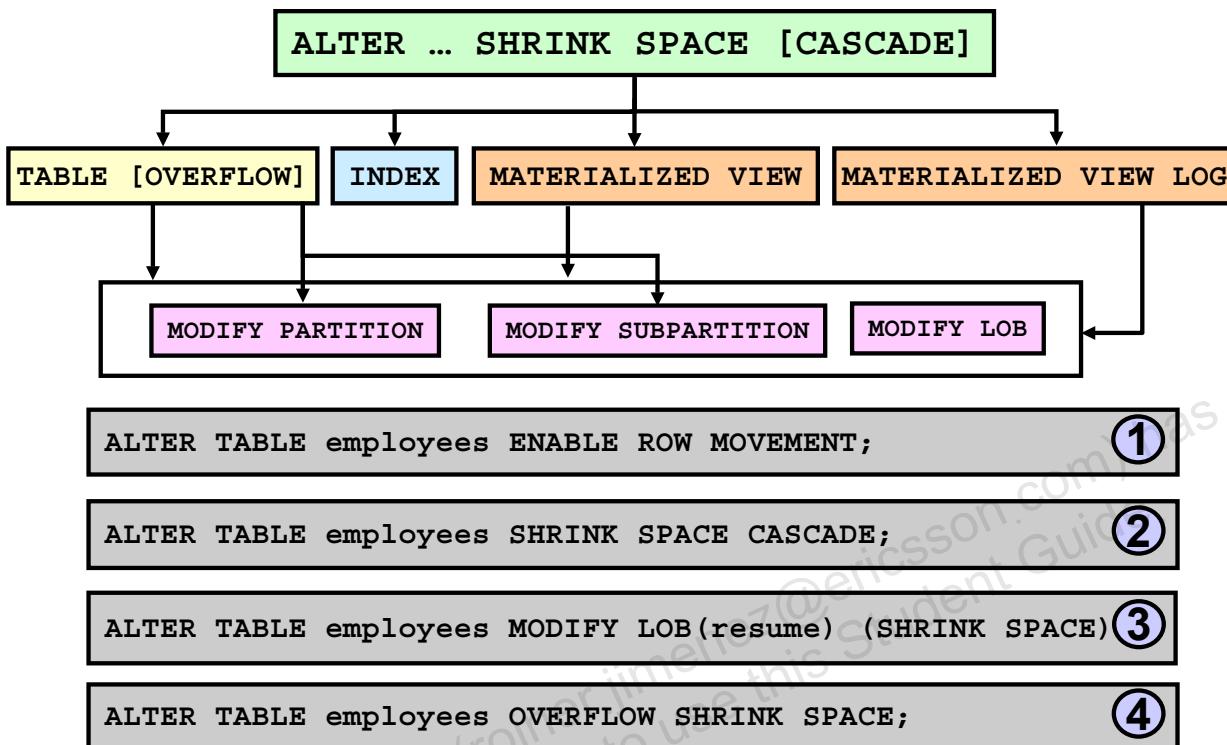
# Database Control and Segment Shrink

The screenshot shows the Oracle Database Control interface. In the top navigation bar, the 'Actions' dropdown is set to 'Shrink Segment' and the 'Go' button is highlighted with a red box and arrow. The main table lists three tables: COUNTRIES, DEPARTMENTS, and EMPLOYEES. The EMPLOYEES row is selected and highlighted with a red box. Below the table is a 'Shrink Segment Options' dialog. It shows the segment name as 'HR.EMPLOYEES' and the object type as 'Table'. A note states: 'The shrink operation compacts fragmented space and, optionally, frees the space. The shrink operation will take some time and will be scheduled as a job.' The 'Shrink Options' section contains two radio buttons: 'Compact Segments and Release Space' (selected) and 'Compact Segments'. The 'Segment Selection' section has one radio button selected: 'Shrink HR.EMPLOYEES Only'. The 'Dependent Segments' table lists two segments: 'EMPLOYEES' (TABLE, EXAMPLE) and 'FMP\_EMAIL\_IUK' (INDEX, EXAMPLE). At the bottom right of the dialog is a 'Continue' button, also highlighted with a red box.

## Database Control and Segment Shrink

Alternatively (to implementing the Segment Advisor recommendations), you can shrink individual segments. For example, from the Database Control home page, click the Tables link in the Schema section. On the Tables page, select your table, and then select Shrink Segment in the Actions drop-down list. Then click the Go button. This brings you to the Shrink Segment page, where you can choose the dependent segments to shrink. You have the opportunity to compact only or to compact and release the space. You can also choose the CASCADE option. When done, click the Continue link. This submits the shrink statements as a scheduled job.

# Shrinking Segments by Using SQL



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Shrinking Segments by Using SQL

Because a shrink operation may cause ROWIDs to change in heap-organized segments, you must enable row movement on the corresponding segment before executing a shrink operation on that segment. Row movement by default is disabled at segment level. To enable row movement, the `ENABLE ROW MOVEMENT` clause of the `CREATE TABLE` or `ALTER TABLE` command is used. This is illustrated in the first example in the slide.

Use the `ALTER` command to invoke segment shrink on an object. The object's type can be one of the following: table (heap- or index-organized), partition, subpartition, LOB (data and index segment), index, materialized view, or materialized view log.

Use the `SHRINK SPACE` clause to shrink space in a segment. If `CASCADE` is specified, the shrink behavior is cascaded to all the dependent segments that support a shrink operation, except materialized views, LOB indexes, and IOT (index-organized tables) mapping tables. The `SHRINK SPACE` clause is illustrated in the second example.

In an index segment, the shrink operation coalesces the index before compacting the data.

Example 3 shows a command that shrinks a LOB segment, given that the `RESUME` column is a `CLOB`.

Example 4 shows a command that shrinks an IOT overflow segment belonging to the `EMPLOYEES` table.

**Note:** For more information, refer to the *Oracle Database SQL Reference* guide.

# Managing Resumable Space Allocation

Space Management  
Proactive Monitoring  
Seg. Advisor & Seg.Shrink  
> **Resumable Allocation**  
Transportable TBS and DB

## A resumable statement:

- **Enables you to suspend large operations instead of receiving an error**
- **Gives you a chance to fix the problem while the operation is suspended, rather than starting over**
- **Is suspended for the following conditions:**
  - Out of space
  - Maximum extents reached
  - Space quota exceeded

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Managing Resumable Space Allocation

The Oracle database provides a means for suspending, and later resuming, the execution of large database operations in the event of space allocation failures. This enables you to take corrective action instead of the Oracle database server returning an error to the user. After the error condition is corrected, the suspended operation automatically resumes. This feature is called “resumable space allocation.” The statements that are affected are called “resumable statements.”

A statement executes in resumable mode only when the resumable statement feature has been enabled for the system or session.

Suspending a statement automatically results in suspending the transaction. Thus all transactional resources are held through the suspension and resuming of a SQL statement. When the error condition disappears (for example, as a result of user intervention or perhaps sort space released by other queries), the suspended statement automatically resumes execution.

A suspension time-out interval is associated with resumable statements. A resumable statement that is suspended for the time-out interval (the default is 7,200 seconds (2 hours)) reactivates itself and returns the exception to the user. A resumable statement can be suspended and resumed multiple times during execution.

**Note:** A maximum extents reached error happens only with dictionary-managed tablespaces.

# Using Resumable Space Allocation

- **Queries, DML operations, and certain DDL operations can be resumed if they encounter an out-of-space error.**
- **A resumable statement can be issued through SQL, PL/SQL, SQL\*Loader, or the Oracle Call Interface (OCI).**
- **To run a resumable statement, you must first enable resumable statements for your session.**

```
ALTER SESSION ENABLE RESUMABLE;

INSERT INTO sales_new SELECT * FROM sh.sales;

ALTER SESSION DISABLE RESUMABLE;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Using Resumable Space Allocation

Resumable space allocation is possible only when statements are executed within a session that has resumable mode enabled. There are two means of enabling and disabling resumable space allocation:

- Issue the ALTER SESSION ENABLE RESUMABLE command.
- Set the RESUMABLE\_TIMEOUT initialization parameter to a nonzero value with an ALTER SESSION or ALTER SYSTEM statement.

When enabling resumable mode for a session or the database, you can specify a time-out period, after which a suspended statement errors out if no intervention has taken place. The RESUMABLE\_TIMEOUT initialization parameter indicates the number of seconds before a time-out occurs. You can also specify the time-out period with the following command:

```
ALTER SESSION ENABLE RESUMABLE TIMEOUT 3600;
```

The value of TIMEOUT remains in effect until it is changed by another ALTER SESSION ENABLE RESUMABLE statement, it is changed by another means, or the session ends. The default time-out interval when using the ENABLE RESUMABLE TIMEOUT clause to enable resumable mode is 7,200 seconds, or 2 hours.

## Using Resumable Space Allocation (continued)

You can also give a name to resumable statements. For example:

```
ALTER SESSION ENABLE RESUMABLE TIMEOUT 3600
NAME 'multitab insert';
```

The name of the statement is used to identify the resumable statement in the DBA\_RESUMABLE and USER\_RESUMABLE views.

For example:

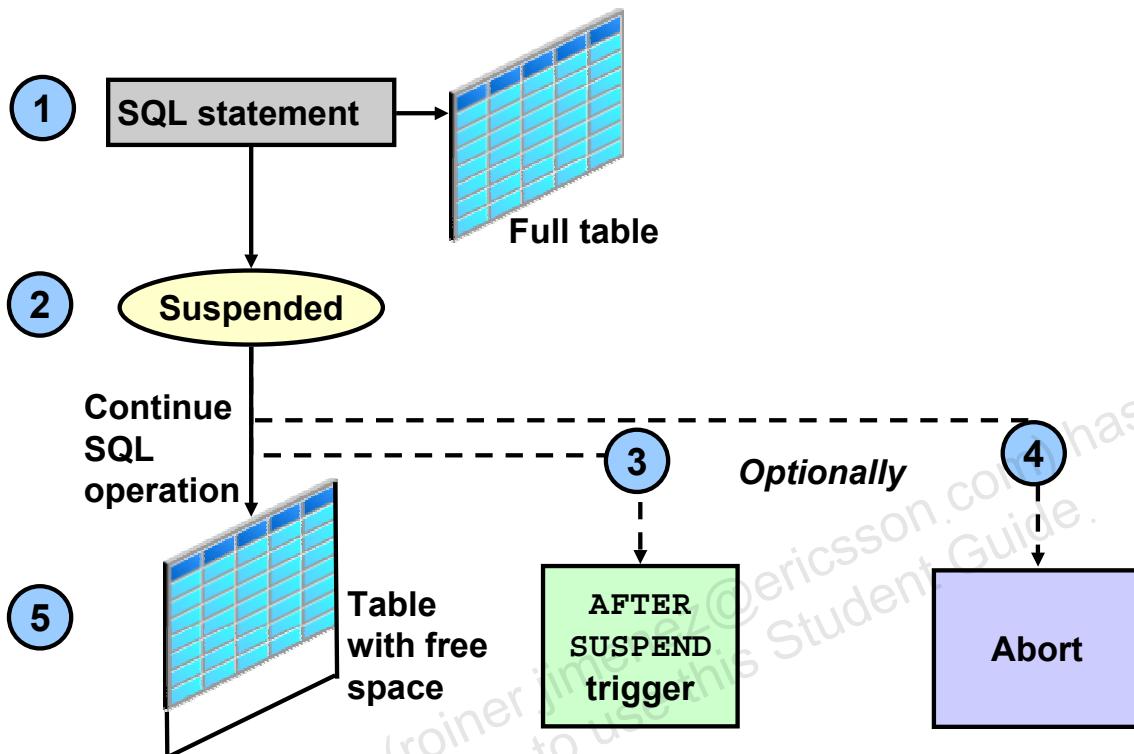
```
SELECT name, sql_text FROM user_resumable;
```

| NAME            | SQL_TEXT                                     |
|-----------------|----------------------------------------------|
| multitab insert | INSERT INTO oldsales SELECT * FROM sh.sales; |

To automatically configure resumable statement settings for individual sessions, you can create and register a database-level LOGON trigger that alters a user's session. The trigger issues commands to enable resumable statements for the session, specifies a time-out period, and associates a name with the resumable statements issued by the session.

Because suspended statements can hold up some system resources, users must be granted the RESUMABLE system privilege before they are allowed to enable resumable space allocation and execute resumable statements.

# Resuming Suspended Statements



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Resuming Suspending Statements

Example:

1. An INSERT statement encounters an error saying the table is full.
2. The INSERT statement is suspended, and no error is passed to client.
3. Optionally, an AFTER SUSPEND trigger is executed.
4. Optionally, the SQLERRROR exception is activated to abort the statement.
5. If the statement is not aborted and free space is successfully added to the table, the INSERT statement resumes execution.

## Detecting a Suspended Statement

When a resumable statement is suspended, the error is not raised to the client. In order for corrective action to be taken, the Oracle database provides alternative methods for notifying users of the error and for providing information about the circumstances.

## Resuming Suspended Statements (continued)

### Possible Actions During Suspension

When a resumable statement encounters a correctable error, the system internally generates the AFTER SUSPEND system event. Users can register triggers for this event at both the database and schema level. If a user registers a trigger to handle this system event, the trigger is executed after a SQL statement has been suspended. SQL statements executed within an AFTER SUSPEND trigger are always nonresumable and are always autonomous. Transactions started within the trigger use the SYSTEM rollback segment. These conditions are imposed to overcome deadlocks and reduce the chance of the trigger experiencing the same error condition as the statement.

Within the trigger code, you can use the USER\_RESUMABLE or DBA\_RESUMABLE views, or the DBMS\_RESUMABLE.SPACE\_ERROR\_INFO function to get information about the resumable statements.

When a resumable statement is suspended:

- The session invoking the statement is put into a wait state. A row is inserted into V\$SESSION\_WAIT for the session with the EVENT column containing “statement suspended, wait error to be cleared”.
- An operation-suspended alert is issued on the object that needs addition resources for the suspended statement to complete.

### Ending a Suspended Statement

When the error condition is resolved (for example, as a result of DBA intervention or perhaps sort space released by other queries), the suspended statement automatically resumes execution and the “resumable session suspended” alert is cleared.

A suspended statement can be forced to activate the SERVERERROR exception by using the DBMS\_RESUMABLE.ABORT() procedure. This procedure can be called by a DBA, or by the user who issued the statement. If the suspension time-out interval associated with the resumable statement is reached, the statement aborts automatically and an error is returned to the user.

# Transporting Tablespaces

...  
Resumable  
Allocation  
> **Transportable TBS**  
Transportable DB

## Concept: Cross-platform transportable tablespaces:

- Simplify data distribution between data warehouse and data marts
- Allow database migration from one platform to another
- Supported platforms:

|                               |                   |                                  |
|-------------------------------|-------------------|----------------------------------|
| Solaris[tm] OE (32-bit)       | HP-UX (64-bit)    | Microsoft Windows IA (64-bit)    |
| Solaris[tm] OE (64-bit)       | HP Tru64 UNIX     | IBM zSeries Based Linux          |
| Microsoft Windows IA (32-bit) | HP-UX IA (64-bit) | Linux 64-bit for AMD             |
| Linux IA (32-bit)             | Linux IA (64-bit) | Apple Mac OS                     |
| AIX-Based Systems (64-bit)    | HP Open VMS       | Microsoft Windows 64-bit for AMD |
|                               |                   | Solaris Operating System (x86)   |

**ORACLE®**

Copyright © 2008, Oracle. All rights reserved.

## Transporting Tablespaces

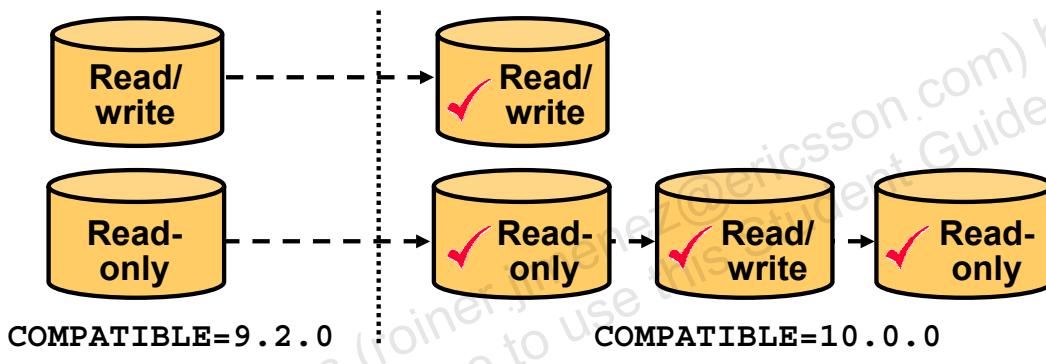
You can use the transportable tablespace feature to move data across platform boundaries. This simplifies the distribution of data from a data warehouse environment to data marts, which often run on smaller platforms. It also allows a database to be migrated from one platform to another by rebuilding the dictionary and transporting the user tablespaces.

To be able to transport data files from one platform to another, you must ensure that both the source system and the target system are running on one of the supported platforms (see slide).

**Note:** The cross-platform transportable tablespace feature requires both platforms to be using the same character sets.

## Concept: Minimum Compatibility Level

- Both source and target databases must have COMPATIBLE set to 10.0.0 or higher.
- Data file headers are platform-aware.
- Before transporting, make sure that all read-only and offline files are platform-aware.



ORACLE

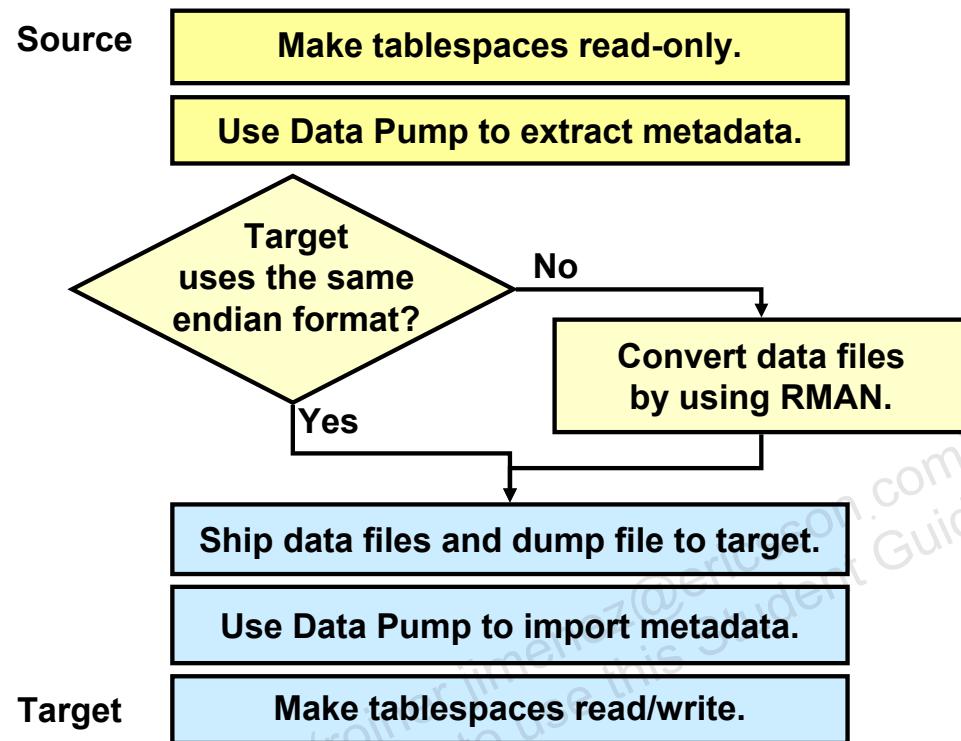
Copyright © 2008, Oracle. All rights reserved.

### Concept: Minimum Compatibility Level

Both source and target databases need to advance their database COMPATIBLE initialization parameter to 10.0.0 or greater before they can use the cross-platform transportable tablespace feature.

When data files are first opened in Oracle Database 10g with COMPATIBLE set to 10.0.0 (or greater), the files are made platform-aware. This is represented by the check marks in the diagram. Each file identifies the platform that it belongs to. These files have identical on-disk formats for file header blocks that are used for file identification and verification. Read-only and offline files get the compatibility advanced only after they are made read/write or are brought online. This implies that tablespaces that are read-only in databases before Oracle Database 10g must be made read/write at least once before they can use the cross-platform transportable feature.

# Transportable Tablespace Procedure



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Transportable Tablespace Procedure

To transport a tablespace from one platform to another (source to target), data files belonging to the tablespace set must be converted to a format that can be understood by the target or destination database. Although with Oracle Database 10g, disk structures conform to a common format, it is possible for the source and target platforms to use different endian formats (byte ordering). When going to a different endian platform, you must use the CONVERT command of the RMAN utility to convert the byte ordering. This operation can be performed on either the source or the target platforms. For platforms that have the same endian format, no conversion is needed.

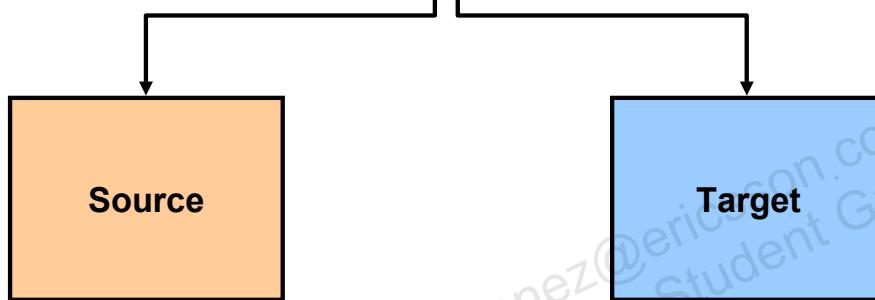
The slide graphic depicts the possible steps to transport tablespaces from a source platform to a target platform. However, it is possible to perform the conversion after shipping the files to the target platform. The last two steps must be executed on the target platform.

Basically, the procedure is the same as when using previous releases of the Oracle database except when both platforms use different endian formats. It is assumed that both platforms are cross-transportable compliant.

**Note:** Byte ordering can affect the results when data is written and read. For example, the 2-byte integer value 1 is written as 0x0001 on a big-endian system (such as Sun SPARC Solaris) and as 0x0100 on a little-endian system (such as an Intel-compatible PC).

## Determining the Endian Format of a Platform

```
SELECT tp.endian_format
FROM v$transportable_platform tp,
 v$database d
WHERE tp.platform_name = d.platform_name;
```



ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Determining the Endian Format of a Platform

You can query V\$TRANSPORTABLE\_PLATFORM to determine whether the endian ordering is the same on both platforms. V\$DATABASE has two columns that can be used to determine your own platform name and platform identifier.

Use the query in the slide on both platforms, and then compare the results. On a Sun SPARC Solaris system, the SELECT statement produces the following output:

```
ENDIAN_FORMAT

Big
```

On a Microsoft Windows Intel-based platform, the SELECT statement produces the following output:

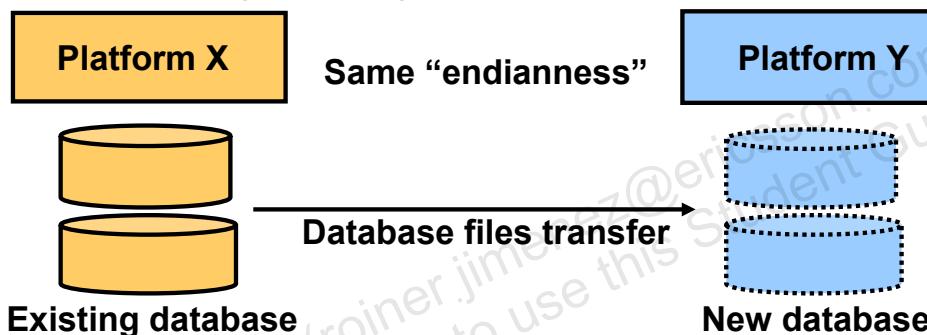
```
ENDIAN_FORMAT

Little
```

# Transporting Databases

...  
Resumable  
Allocation  
Transportable TBS  
> Transportable DB

- Generalize the transportable tablespace feature.
- Data can easily be distributed from a data warehousing environment to data marts, which are usually on smaller platforms.
- A database can be migrated from one platform to another very quickly.



**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Transporting Databases

With the transportable tablespace feature, moving data across different platforms becomes much faster. However, metadata still needs to be unloaded, because the system tablespace cannot be transported.

The purpose of the database transport feature is to provide a fast and easy way to transport a database across different platforms with the same endian format. However, the source platform and the target platform can have different disk alignments. For example, HP-UX and Solaris both have big endian, but the disk alignment is eight on HP-UX and four on Solaris.

To transport databases from one platform to another, you must ensure that both the source system and the target system are running on one of the platforms that are listed in V\$TRANSPORTABLE\_PLATFORM and that both have the same endian format. For example, you can transport a database running on Linux IA (32-bit) to one of the Windows platforms.

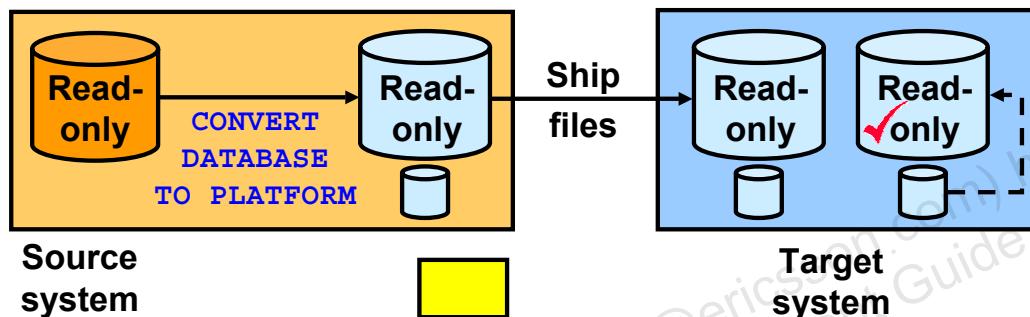
If one or both of the databases uses Automatic Storage Management (ASM), you may need to use the DBMS\_FILE\_TRANSFER package to ftp the files.

Unlike transportable tablespace, where there is a target database to plug data into, this feature creates a new database on the target platform. The newly created database contains the same data as the source database. Except for things such as database name, instance name, and location of files, the new database also has the same settings as the source database.

**Note:** Transporting database is faster than using Data Pump to move data.

# Database Transportation Procedure: Source System Conversion

Open database in **READ ONLY** mode  
and **COMPATIBLE=10.0.0**



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Database Transportation Procedure: Source System Conversion

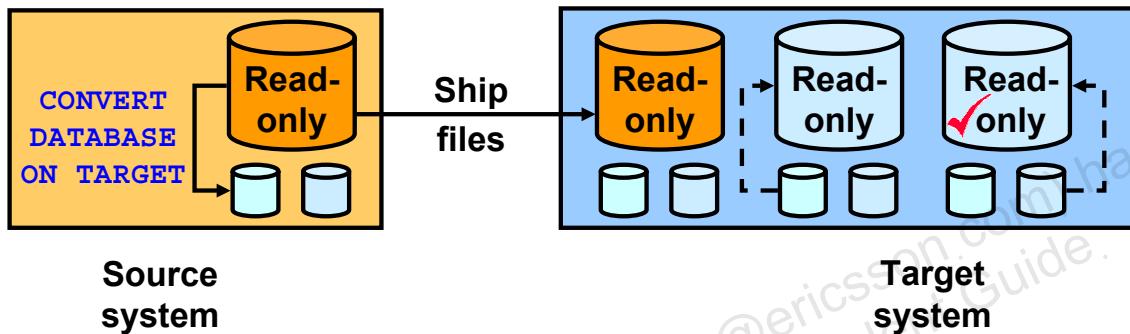
Before you can transport your database, you must open it in **READ ONLY** mode. Then use RMAN to convert the necessary data files of the database.

When you do the conversion on the source platform, the new RMAN command **CONVERT DATABASE** generates a script containing the correct **CREATE CONTROLFILE RESETLOGS** command that is used on the target system to create the new database. The **CONVERT DATABASE** command then converts all identified data files so that they can be used on the target system. You then ship the converted data files and the generated script to the target platform. By executing the generated script on the target platform, you create a new copy of your database.

**Note:** The source database must be running with the **COMPATIBLE** initialization parameter set to **10.0.0** or higher. All identified tablespaces must have been **READ WRITE** at least once since **COMPATIBLE** was set to **10.0.0** or higher.

# Database Transportation Procedure: Target System Conversion

Open database in **READ ONLY** mode  
and **COMPATIBLE=10.0.0**



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Database Transportation Procedure: Target System Conversion

Before you can transport your database, you must open it in **READ ONLY** mode. Then use RMAN to convert the necessary data files of the database.

When you do the conversion on the target platform, the **CONVERT DATABASE** command (which is executed on the source system) generates only two scripts used on the target system to convert the data files, and to re-create the control files for the new database. Then, you ship the identified data files and both scripts to the target platform. After this is done, execute both scripts in the right order. The first one uses the existing **CONVERT DATAFILE RMAN** command to do the conversion, and the second issues the **CREATE CONTROLFILE RESETLOGS SQL** command with the converted data files to create the new database.

**Note:** The source database must be running with the **COMPATIBLE** initialization parameter set to **10.0.0** or higher. All identified tablespaces must have been **READ WRITE** at least once since **COMPATIBLE** was set to **10.0.0** or higher.

# Database Transportation: Considerations

- **Create the password file on the target platform.**
- **Transport the BFILEs used in the source database.**
- **The generated pfile and transport script use OMF.**
- **Use DBNEWID to change the DBID.**



Copyright © 2008, Oracle. All rights reserved.

## Database Transportation: Considerations

Redo logs, control files, and tempfiles are not transported. They are re-created for the new database on the target platform. As a result, the new database on the target platform must be opened with the RESETLOGS option.

If a password file is used, it is not transported and you need to create it on the target platform. This is because the types of file names allowed for the password file are OS specific. However, the output of the CONVERT DATABASE command lists all the usernames and their system privileges, and advises to re-create the password file and add entries for these users on the target platform.

The CONVERT DATABASE command lists all the directory objects and objects that use BFILE data types or external tables in the source database. You may need to update these objects with new directory and file names. If BFILEs are used in the database, you have to transport the BFILEs.

The generated pfile and transport script use Oracle Managed Files (OMF) for database files. If you do not want to use OMF, you must modify the pfile and transport script.

The transported database has the same DBID as the source database. You can use the DBNEWID utility to change the DBID. In the transport script as well as the output of the CONVERT DATABASE command, you are prompted to use the DBNEWID utility to change the database ID.

# Summary

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

- **Use the Oracle database to automatically manage space**
- **Proactively monitor and manage tablespace space usage**
- **Use the Segment Advisor**
- **Reclaim wasted space from tables and indexes by using the segment shrink functionality**
- **Manage resumable space allocation**
- **Describe the concepts of transportable tablespaces and databases**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Practice Overview: Managing Storage

**This practice covers the following topics:**

- **Using threshold alerts to proactively manage tablespaces**
- **Using the Segment Advisor to shrink space**
- **Viewing alerts and alert history in SQL\*Plus and Enterprise Manager**



Copyright © 2008, Oracle. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2010, Oracle and/or its affiliates.

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a  
non-transferable license to use this Student Guide.

# 12

## Automatic Storage Management

ORACLE®

Copyright © 2008, Oracle. All rights reserved.

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a  
non-transferable license to use this Student Guide.

# Objectives

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

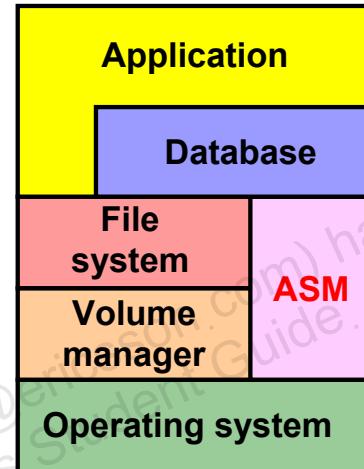
- **Identify the features of Automatic Storage Management (ASM)**
- **Set up initialization parameter files for ASM and database instances**
- **Execute SQL commands with ASM file names**
- **Start up and shut down ASM instances**
- **Administer ASM disk groups**
- **Use RMAN to migrate your database to ASM**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

# Automatic Storage Management: Review

- Portable and high-performance cluster file system
- Manages Oracle database files
- Data spread across disks to balance load
- Integrated mirroring across disks
- Solves many storage management challenges



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Automatic Storage Management: Review

Automatic Storage Management (ASM) provides a vertical integration of the file system and the volume manager that is specifically built for the Oracle database files. ASM can provide management for single SMP machines, or across multiple nodes of a cluster for Oracle Real Application Clusters (RAC) support.

ASM distributes I/O load across all available resources to optimize performance while removing the need for manual I/O tuning. ASM helps DBAs to manage a dynamic database environment by allowing them to increase the database size without having to shut down the database to adjust the storage allocation.

ASM can maintain redundant copies of data to provide fault tolerance, or it can be built on top of vendor-supplied reliable storage mechanisms. Data management is done by selecting the desired reliability and performance characteristics for classes of data rather than with human interaction on a per file basis.

ASM capabilities save DBAs' time by automating manual storage and thereby increasing their ability to manage larger databases and more of them with increased efficiency.

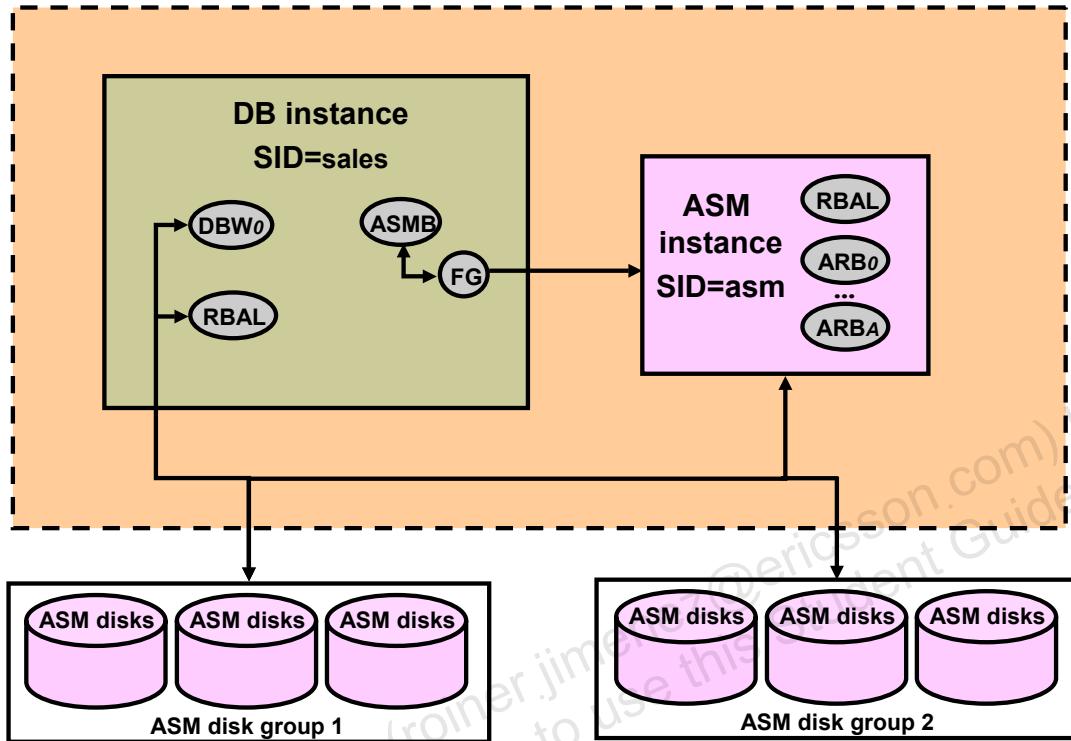
## **Automatic Storage Management: Review (continued)**

ASM divides files into allocation units (AUs) and spreads the AUs for each file evenly across all the disks. ASM uses an index technique to track the placement of each AU. When your storage capacity changes, ASM does not restripe all of the data, but moves an amount of data proportional to the amount of storage added or removed to evenly redistribute the files and maintain a balanced load across the disks. This is done while the database is active.

You can increase the speed of a rebalance operation, or lower it to reduce the impact on the I/O subsystem. ASM provides mirroring protection without the need to purchase a third-party Logical Volume Manager. One unique advantage of ASM is that the mirroring is applied on a file basis, rather than on a volume basis. Therefore, the same disk group can contain a combination of files protected by mirroring, along with those that are not protected at all.

ASM supports data files, log files, control files, archive logs, RMAN backup sets, and other Oracle database file types. ASM supports Real Application Clusters and eliminates the need for a Cluster Logical Volume Manager or a Cluster File System.

# ASM General Architecture



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## ASM General Architecture

To use ASM, you must start a special instance, called an ASM instance, before you start your database instance. ASM instances do not mount databases, instead they manage the metadata needed to make ASM files available to ordinary database instances. Both ASM instances and database instances have access to some common set of disks called disk groups. Database instances access the contents of ASM files directly, communicating with an ASM instance only to get information about the layout of these files.

An ASM instance contains two new background processes. One coordinates rebalance activity for disk groups. It is called RBAL. The second one performs the actual rebalance AU movements. There can be many of these at a time, and they are called ARB0, ARB1, and so forth. An ASM instance also has some of the same background processes as a database instance, including SMON, PMON, LGWR, DBWR, and CKPT.

## ASM General Architecture (continued)

Each database instance using ASM has two new background processes called ASMB and RBAL. RBAL performs global opens of the disks in the disk groups. At database instance startup, ASMB connects as a foreground process into the ASM instance. Communication between the database and the ASM instance is performed via this bridge. This includes physical file changes such as data file creation and deletion. Over this connection, periodic messages are exchanged to update statistics and to verify that both instances are healthy.

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a  
non-transferable license to use this Student Guide.

## ASM Instance Tasks

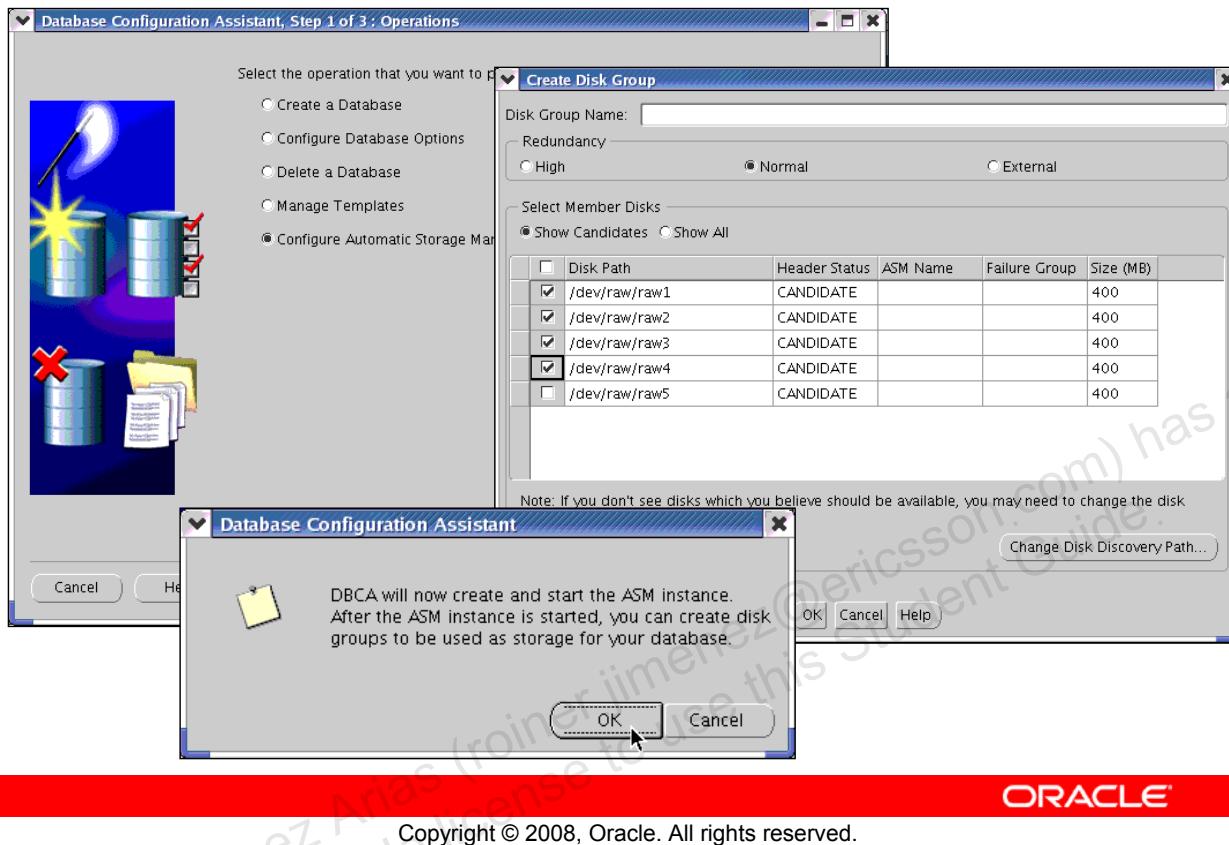
**The following are tasks that you need to be able to perform in order to use an ASM instance:**

- **Create the ASM instance**
- **Set the initialization parameters**
- **Start the ASM instance**
- **Manage the ASM instance**
- **Shut down the ASM instance**



Copyright © 2008, Oracle. All rights reserved.

# Creating an ASM Instance



## Creating an ASM Instance

You create an ASM instance by running the Database Configuration Assistant (DBCA). On the first screen, choose the option to Configure Automatic Storage Management, and follow the steps. The ASM instance is created and started for you. Then you are guided through the process of defining disk groups for the instance.

As part of the ASM instance creation process, the DBCA automatically creates an entry into the `oratab` file. This entry is used for discovery purposes. On the Windows platform where a services mechanism is used, the DBCA automatically creates an Oracle Service and the appropriate registry entry to facilitate the discovery of ASM instances.

When an ASM instance is configured, the DBCA creates an ASM instance parameter file and an ASM instance password file.

If you were to first create an ASM-enabled database, the DBCA determines whether an ASM instance already exists on your host. If ASM instance discovery returns an empty list, the DBCA creates a new ASM instance.

# ASM Instance Initialization Parameters

```
INSTANCE_TYPE = ASM
DB_UNIQUE_NAME = +ASM
ASM_POWER_LIMIT = 1
ASM_DISKSTRING = '/dev/rdsk/*s2', '/dev/rdsk/c1*'
ASM_DISKGROUPS = dgroupA, dgroupB
LARGE_POOL_SIZE = 8MB
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## ASM Instance Initialization Parameters

- INSTANCE\_TYPE should be set to ASM for ASM instances.
- DB\_UNIQUE\_NAME specifies the service provider name for which this ASM instance manages disk groups. The default value of +ASM must be modified only if you run multiple ASM instances on the same node.
- ASM\_POWER\_LIMIT controls the speed for a rebalance operation. Values range from 1 through 11, with 11 being the fastest. If omitted, this value defaults to 1. The number of slaves is derived from the parallelization level specified in a manual rebalance command (POWER), or by the ASM\_POWER\_LIMIT parameter.
- ASM\_DISKSTRING is an operating system-dependent value used by ASM to limit the set of disks considered for discovery.
- ASM\_DISK\_GROUPS is the list of names of disk groups to be mounted by an ASM instance at startup, or when the ALTER DISKGROUP ALL MOUNT command is used.

The INSTANCE\_TYPE parameter is the only parameter that you must define. All other ASM parameters have default values that are suitable for most environments.

**Note:** If the ASM environment has been created using the command line instead of EM, then the disk groups must be created before they can be mounted.

# Database Instance Parameter Changes

```
...
INSTANCE_TYPE = RDBMS
LOG_ARCHIVE_FORMAT
DB_BLOCK_SIZE
DB_CREATE_ONLINE_LOG_DEST_n
DB_CREATE_FILE_DEST
DB_RECOVERY_FILE_DEST
CONTROL_FILES
LOG_ARCHIVE_DEST_n
LOG_ARCHIVE_DEST
STANDBY_ARCHIVE_DEST
LARGE_POOL_SIZE = 8MB
...
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Database Instance Parameter Changes

INSTANCE\_TYPE defaults to RDBMS and specifies that this instance is an RDBMS instance.

LOG\_ARCHIVE\_FORMAT is ignored if LOG\_ARCHIVE\_DEST is set to an incomplete ASM file name, such as +dGroupA. If LOG\_ARCHIVE\_DEST is set to an ASM directory (for example, +dGroupA/myarchlogdir/), then LOG\_ARCHIVE\_FORMAT is used and the files are non-OMF. Unique file names for archived logs are automatically created by the Oracle database.

The following parameters accept the multifile creation context form of ASM file names as a destination:

- DB\_CREATE\_ONLINE\_LOG\_DEST\_n
- DB\_CREATE\_FILE\_DEST
- DB\_RECOVERY\_FILE\_DEST
- CONTROL\_FILES
- LOG\_ARCHIVE\_DEST\_n
- LOG\_ARCHIVE\_DEST
- STANDBY\_ARCHIVE\_DEST

**Note:** Because allocation unit maps for ASM files are allocated from the LARGE\_POOL, you must set the LARGE\_POOL\_SIZE initialization parameter to at least 8 MB, preferably higher.

# Starting Up an ASM Instance

```
$ export ORACLE_SID='+ASM'
$ sqlplus /nolog
SQL> CONNECT / AS sysdba
Connected to an idle instance.
SQL> STARTUP;
ASM instance started
Total System Global Area 83886080 bytes
Fixed Size 1260216 bytes
Variable Size 57460040 bytes
ASM Cache 25165824 bytes
ASM diskgroups mounted
```

ORACLE

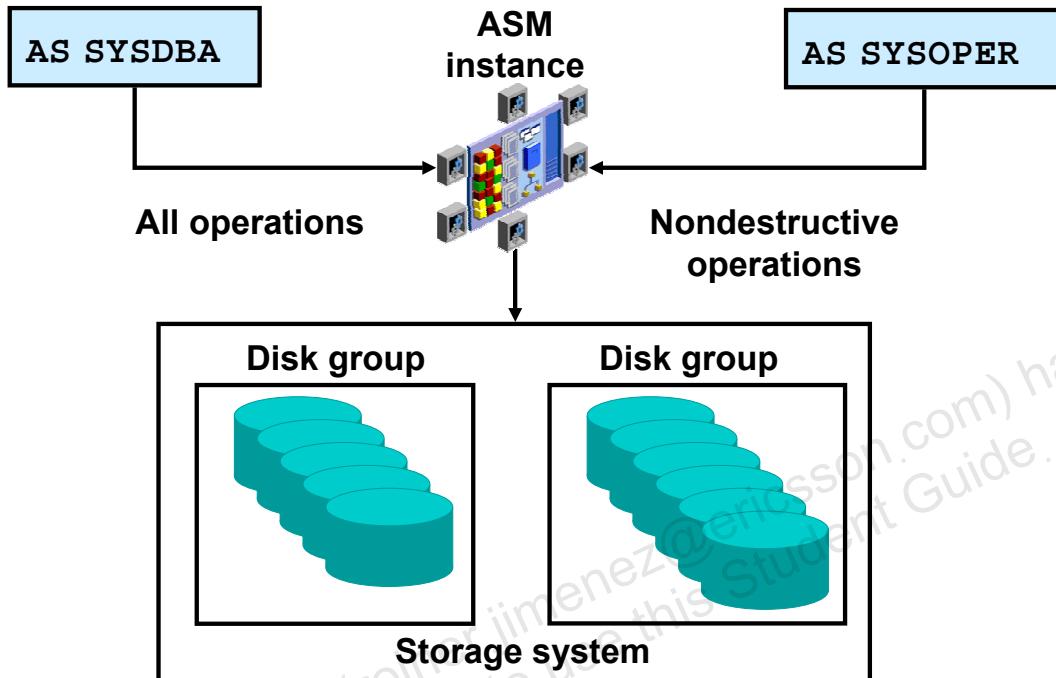
Copyright © 2008, Oracle. All rights reserved.

## Starting Up an ASM Instance

ASM instances are started similarly to database instances except that the initialization parameter file contains an entry like `INSTANCE_TYPE=ASM`. When this parameter is set to the value `ASM`, it informs the Oracle executable that an ASM instance is starting, not a database instance. Also, the `ORACLE_SID` variable must be set to the ASM instance name. When the ASM instance starts up, the mount stage attempts to mount the disk groups specified by the `ASM_DISKGROUPS` initialization parameter rather than mounting a database, as is done with non-ASM instances.

Other `STARTUP` clauses have comparable interpretation for ASM instances as they do for database instances. `OPEN` is invalid for an ASM instance. `NOMOUNT` starts up the ASM instance without mounting any disk group.

## Accessing an ASM Instance



ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Accessing an ASM Instance

ASM instances do not have a data dictionary, so the only way to connect to one is by using OS authentication, that is, SYSDBA or SYSOPER. To connect remotely, a password file must be used. Normally, the SYSDBA privilege is granted through the use of an operating system group. On UNIX, this is typically the dba group. By default, members of the dba group have SYSDBA privilege on all instances on the node, including the ASM instance. Users who connect to the ASM instance with the SYSDBA privilege have administrative access to all disk groups in the system. The SYSOPER privilege is supported in ASM instances and limits the set of allowable SQL commands to the minimum required for basic operation of an already configured system.

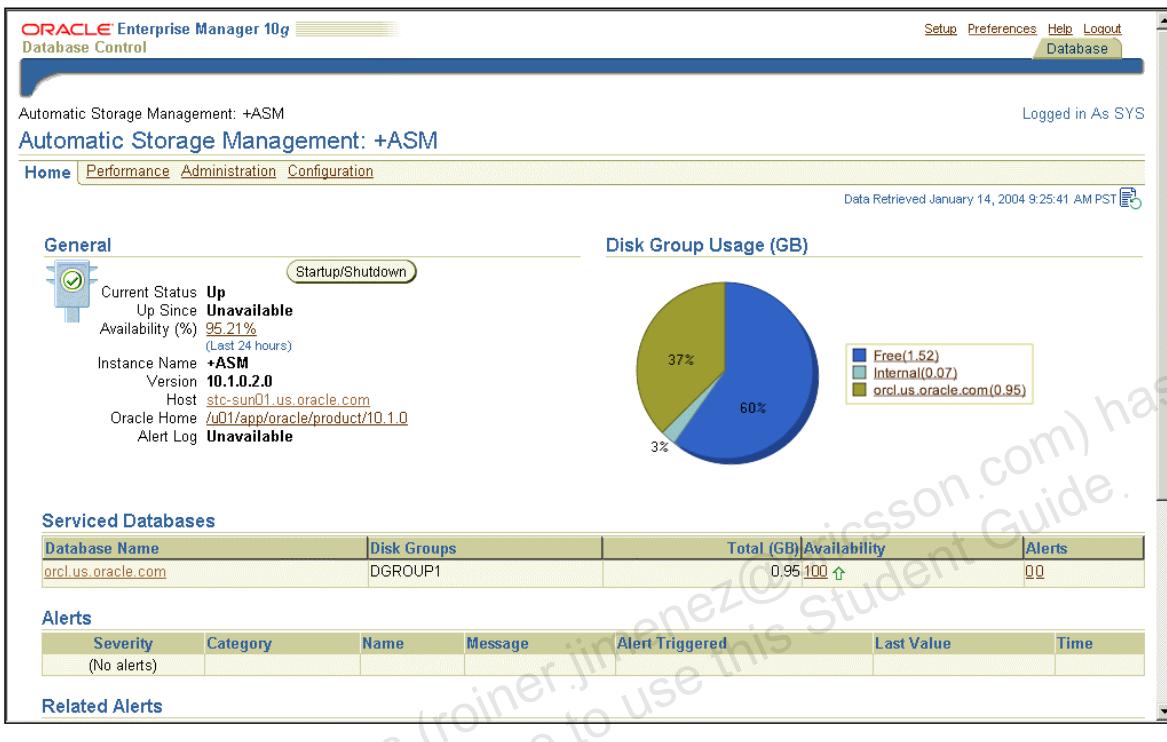
## Accessing an ASM Instance (continued)

The following commands are available to SYSOPER users:

- STARTUP/SHUTDOWN
- ALTER DISKGROUP MOUNT/DISMOUNT
- ALTER DISKGROUP ONLINE/OFFLINE DISK
- ALTER DISKGROUP REBALANCE
- ALTER DISKGROUP CHECK
- SELECT all V\$ASM\_\* views

All other commands, such as CREATE DISKGROUP, ADD/DROP/RESIZE DISK, and so on, require the SYSDBA privilege and are not allowed with the SYSOPER privilege.

# ASM Home Page

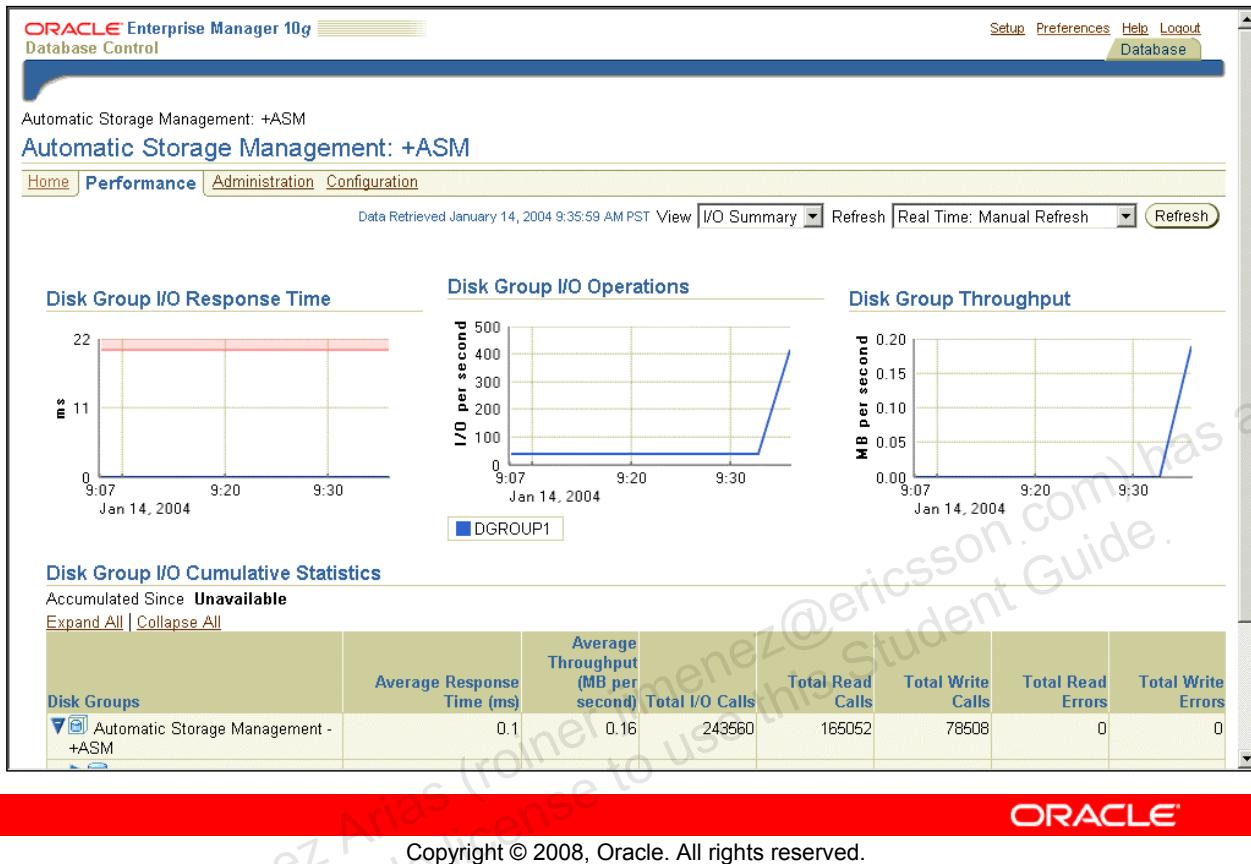


## ASM Home Page

Enterprise Manager provides a user-friendly graphical interface to the Oracle database management, administration, and monitoring tasks. Oracle Database 10g extends the existing functionality to transparently support the management, administration, and monitoring of Oracle databases that use ASM storage. It also adds support for the new management tasks required for administration of ASM instance and ASM disk groups.

The ASM home page shows the status of the ASM instance along with the metrics and alerts generated by the collection mechanisms. It also provides the startup and shutdown functionality. Clicking the Alert link takes the user to an alert details page. The DiskGroup Usage chart shows space used by each client database along with free space.

# ASM Performance Page



## ASM Performance Page

The Performance tab of the Automatic Storage Management page shows the I/O response time and throughput for each disk group. You can further drill down to view disk-level performance metrics.

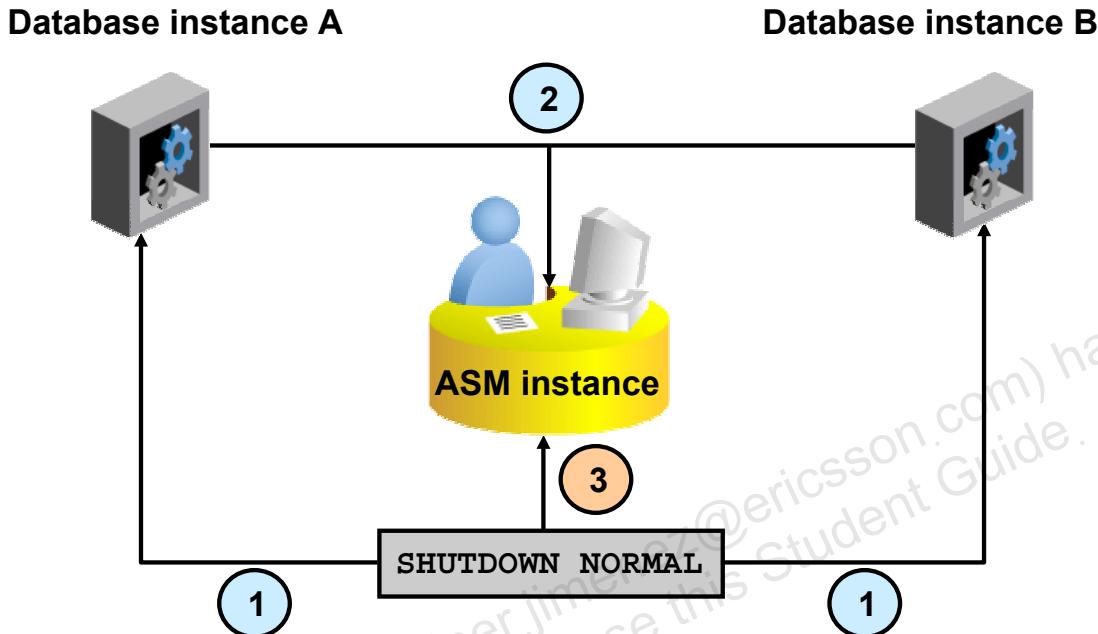
# ASM Configuration Page

The screenshot shows the Oracle Enterprise Manager 10g Database Control interface. The title bar reads "ORACLE Enterprise Manager 10g Database Control". The main header says "Automatic Storage Management: +ASM". The navigation bar includes "Home", "Performance", "Administration", and "Configuration" (which is selected). The sub-header "Configuration Parameters" is displayed. Under "Disk Discovery Path", the value is "/u01/asmdisks/\*" with a note: "TIP Limits the set of disks considered for discovery when a new disk is added to a Disk Group. The disk string should match the path of the disk, not the directory containing the disk. For example: /dev/rdsk/\*.". Under "Auto Mount Disk Groups", the value is "DGROUP1" with a note: "TIP The list of the Disk Group names to be mounted by the ASM at startup or when ALTER DISKGROUP ALL MOUNT command is used.". Under "Rebalance Power", the value is "1" with a note: "TIP Affects the speed of disk group rebalancing. Higher values use more I/O bandwidth and complete rebalance more quickly. Lower values cause rebalance to take longer, but use less I/O bandwidth. Values range from 1 to 11.". At the bottom, there are "Revert" and "Apply" buttons, along with links to "Database", "Setup", "Preferences", "Help", and "Logout". The footer contains the "ORACLE" logo and the copyright notice "Copyright © 1996, 2003, Oracle. All rights reserved. About Oracle Enterprise Manager 10g Database Control".

## ASM Configuration Page

The Configuration tab of the Automatic Storage Management page enables you to view or modify the initialization parameters of the ASM instance.

# Shutting Down an ASM Instance



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Shutting Down an ASM Instance

When you attempt to shutdown an ASM instance in the NORMAL, IMMEDIATE, or TRANSACTIONAL modes, it will only succeed if there are no database instances connected to the ASM instance. If there is at least one connected instance, you will receive the following error:

```
ORA-15097: cannot SHUTDOWN ASM instance with connected RDBMS instance
```

If you perform a SHUTDOWN ABORT on the ASM instance, it will shutdown, and it will require recovery at the time of the next startup. Any connected database instances will also eventually shutdown, reporting the following error:

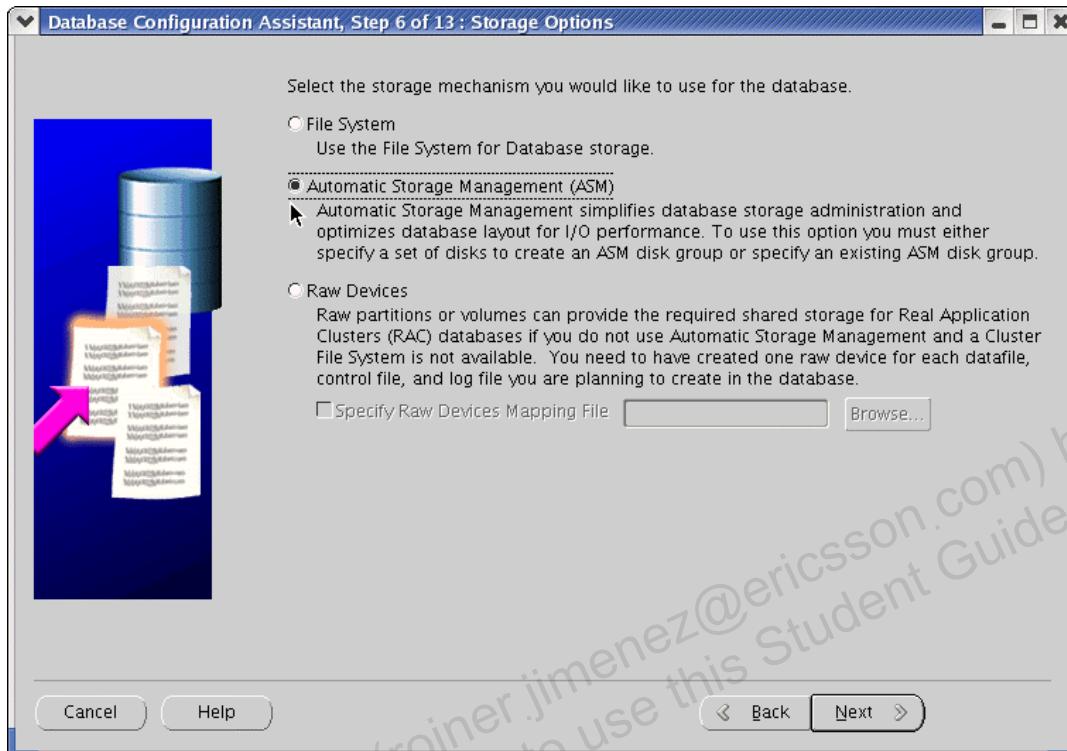
```
ORA-15064: communication failure with ASM instance
```

In a single ASM instance configuration, if the ASM instance fails while disk groups are open for update, then after the ASM instance reinitializes, it reads the disk group's log and recovers all transient changes. With multiple ASM instances sharing disk groups, if one ASM instance fails, another ASM instance automatically recovers transient ASM metadata changes caused by the failed instance. The failure of a database instance does not affect ASM instances. The ASM instance should be started automatically whenever the host is rebooted. ASM instance is expected to use the automatic startup mechanism supported by the underlying operating system. Note that file system failure usually crashes a node.

## **Shutting Down an ASM Instance (continued)**

As shown in the slide, the proper way to shut down instances that are using ASM is to first shut down any instances that are using the ASM instance. Those instances notify the ASM instance that they are no longer using the ASM instance. Then the ASM instance can be shut down normally.

## DBCA and Storage Options



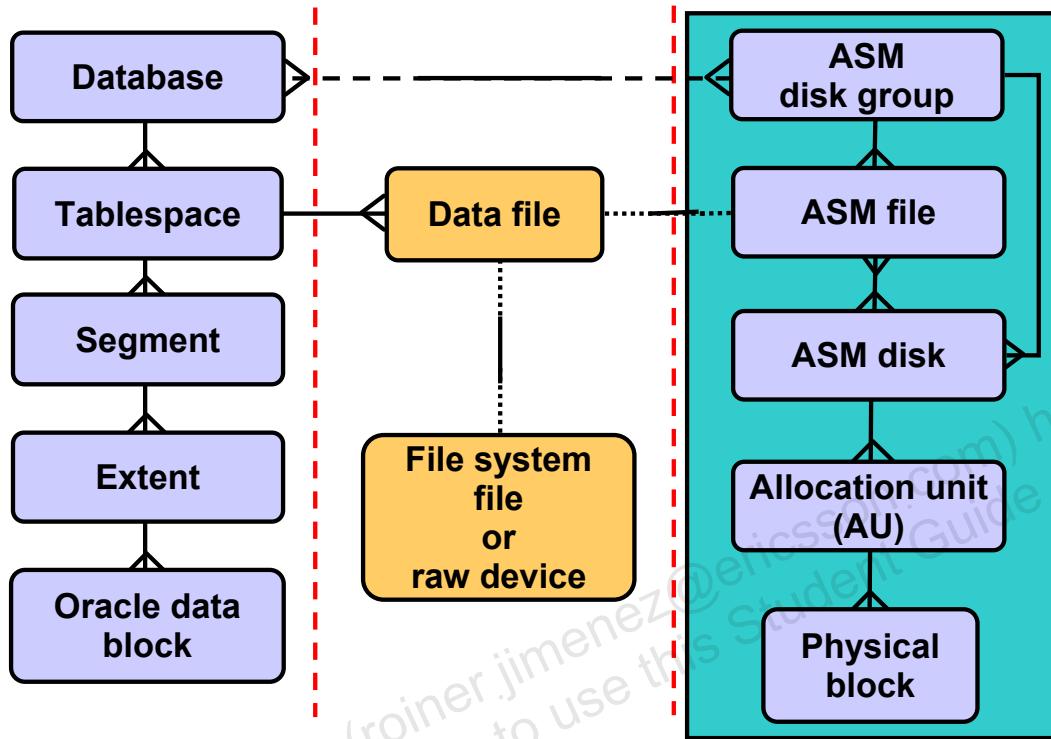
ORACLE

Copyright © 2008, Oracle. All rights reserved.

### DBCA and Storage Options

In order to support ASM as a storage option, this screen appears in the Database Configuration Assistant (DBCA) when creating a database. This allows you to choose the storage options: file system, ASM, or raw devices.

# ASM Storage: Concepts



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## ASM Storage: Concepts

ASM does not eliminate any preexisting database functionality. Existing databases are able to operate as they always have. You can create new files as ASM files and leave existing files to be administered in the old way, or you can eventually migrate them to ASM.

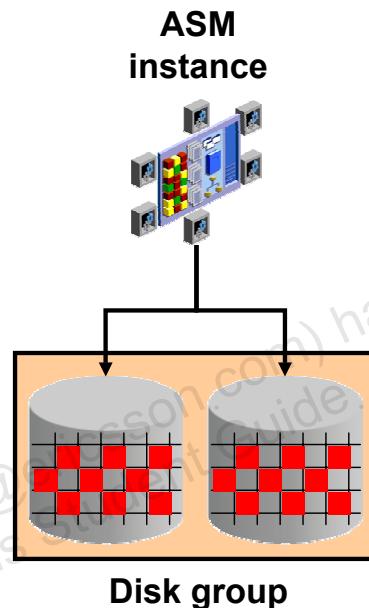
The diagram depicts the relationships that exist between the various storage components inside an Oracle database with ASM available. The left and middle parts of the diagram show the relationships that exist in previous releases. On the right are the new concepts introduced by ASM.

Database files can be stored as ASM files. At the top of the new hierarchy are ASM disk groups. Any single ASM file is contained in only one disk group. However, a disk group may contain files belonging to several databases, and a single database may use storage from multiple disk groups. As you can see, one disk group is made up of multiple ASM disks, and each ASM disk belongs to only one disk group. ASM files are always spread across all the ASM disks in the disk group. ASM disks are partitioned in allocation units (AU) of one megabyte each. An allocation unit is the smallest contiguous disk space that ASM allocates. ASM does not allow an Oracle block to be split across allocation units.

**Note:** This graphic deals with only one type of ASM file: data file. However, ASM can be used to store other database file types.

# ASM Disk Groups

- **A pool of disks managed as a logical unit**
- **Partitions total disk space into uniform sized units**
- **Spreads each file evenly across all disks**
- **Uses coarse- or fine-grain striping on the basis of file type**
- **Administers disk groups, not files**



ORACLE

Copyright © 2008, Oracle. All rights reserved.

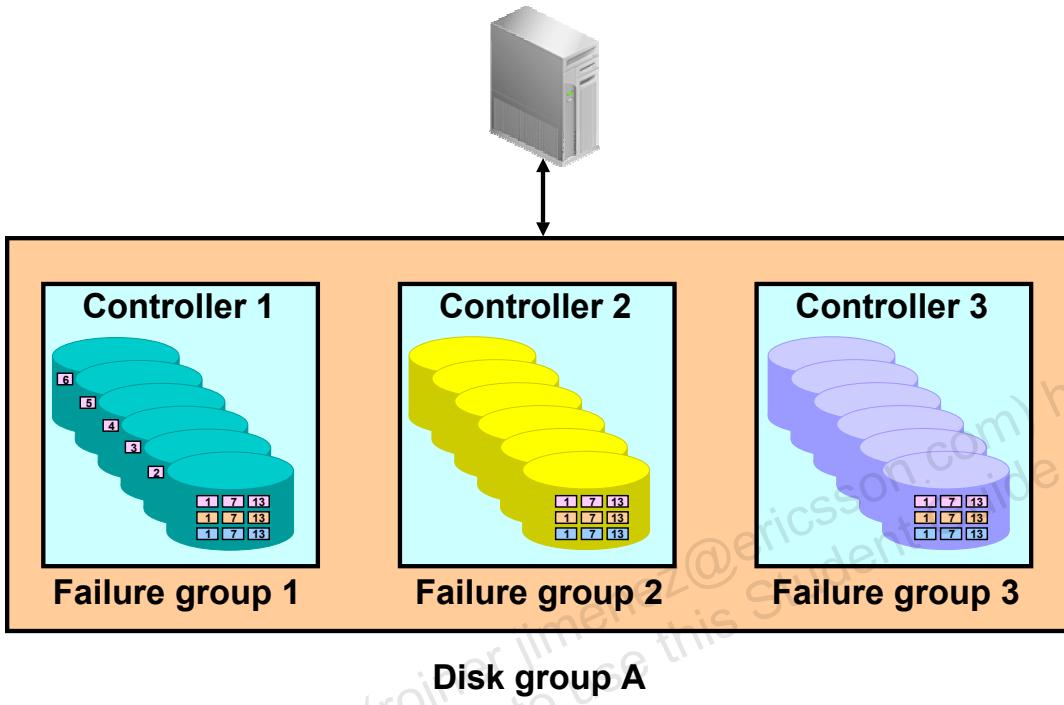
## ASM Disk Groups

A disk group is a collection of disks managed as a logical unit. Storage is added and removed from disk groups in units of ASM disks. Every ASM disk has an ASM disk name, which is a name common to all nodes in a cluster. The ASM disk name abstraction is required because different hosts can use different names to refer to the same disk.

ASM always spreads files evenly in 1 MB allocation unit (AU) chunks across all the disks in a disk group. This is called *coarse* striping. That way, ASM eliminates the need for manual disk tuning. However, disks in a disk group should have similar size and performance characteristics to obtain optimal I/O. For most installations there is only a small number of disk groups. For instance, one disk group for a work area, and one for a recovery area. For files, such as log files, that require low latency, ASM provides fine-grained (128 KB) striping. *Fine* striping stripes each AU. Fine striping breaks up medium-sized I/O operations into multiple smaller I/O operations that execute in parallel. While the number of files and disks increase, you only have to manage a constant number of disk groups. From a database perspective, disk groups can be specified as the default location for files created in the database.

**Note:** Each disk group is self-describing, containing its own file directory and disk directory.

# Failure Group



ORACLE

Copyright © 2008, Oracle. All rights reserved.

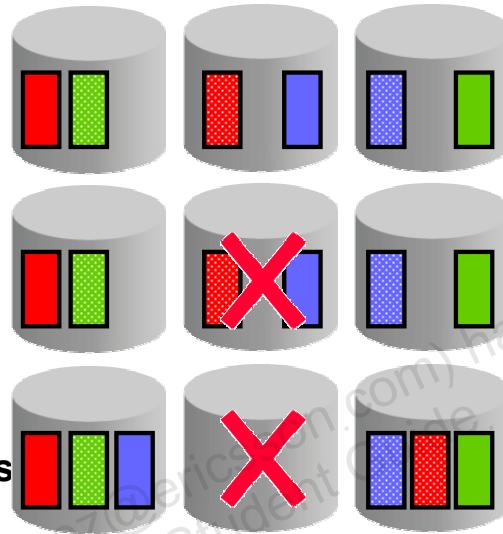
## Failure Group

A failure group is a set of disks, inside one particular disk group, sharing a common resource whose failure needs to be tolerated. An example of a failure group is a string of SCSI disks connected to a common SCSI controller. A failure of the controller leads to all the disks on its SCSI bus becoming unavailable, although each of the individual disks is still functional.

What constitutes a failure group is site-specific. It is largely based upon failure modes that a site is willing to tolerate. By default, ASM assigns each disk to its own failure group. When creating a disk group or adding a disk to a disk group, administrators may specify their own grouping of disks into failure groups. After failure groups are identified, ASM can optimize file layout to reduce the unavailability of data due to the failure of a shared resource.

# Disk Group Mirroring

- **Mirror at AU level**
- **Mix primary and mirror AUs on each disk**
- **External redundancy:**  
**Defers to hardware mirroring**
- **Normal redundancy:**
  - Two-way mirroring
  - At least two failure groups
- **High redundancy:**
  - Three-way mirroring
  - At least three failure groups



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Disk Group Mirroring

ASM has three disk group types that support different types of mirroring:

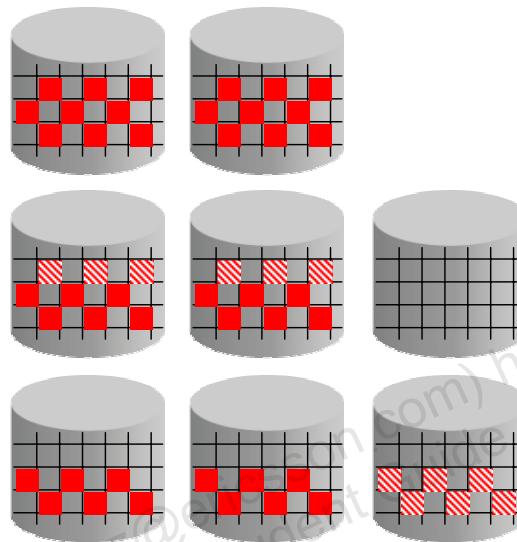
- **External redundancy:** Do not provide mirroring. Use an external-redundancy disk group if you use hardware mirroring or if you can tolerate data loss as the result of a disk failure. Failure groups are not used with these types of disk groups.
- **Normal-redundancy:** Support two-way mirroring
- **High-redundancy:** Provide triple mirroring

ASM does not mirror disks; rather, it mirrors allocation units. As a result, you need only spare capacity in your disk group. When a disk fails, ASM automatically reconstructs the contents of the failed disk on the surviving disks in the disk group by reading the mirrored contents from the surviving disks. This spreads the I/O hit from a disk failure across several disks.

When ASM allocates a primary AU of a file to one disk in a disk group, it allocates a mirror copy of that AU to another disk in the disk group. Primary AUs on a given disk can have their mirror copies on one of several partner disks in the disk group. ASM ensures that a primary AU and its mirror copy never reside in the same failure group. If you define failure groups for your disk group, ASM can tolerate the simultaneous failure of multiple disks in a single failure group.

# Disk Group Dynamic Rebalancing

- **Automatic online rebalance whenever storage configuration changes**
- **Only move data proportional to storage added**
- **No need for manual I/O tuning**
- **Online migration to new storage**
- **Configurable load on system using `ASM_POWER_LIMIT`**



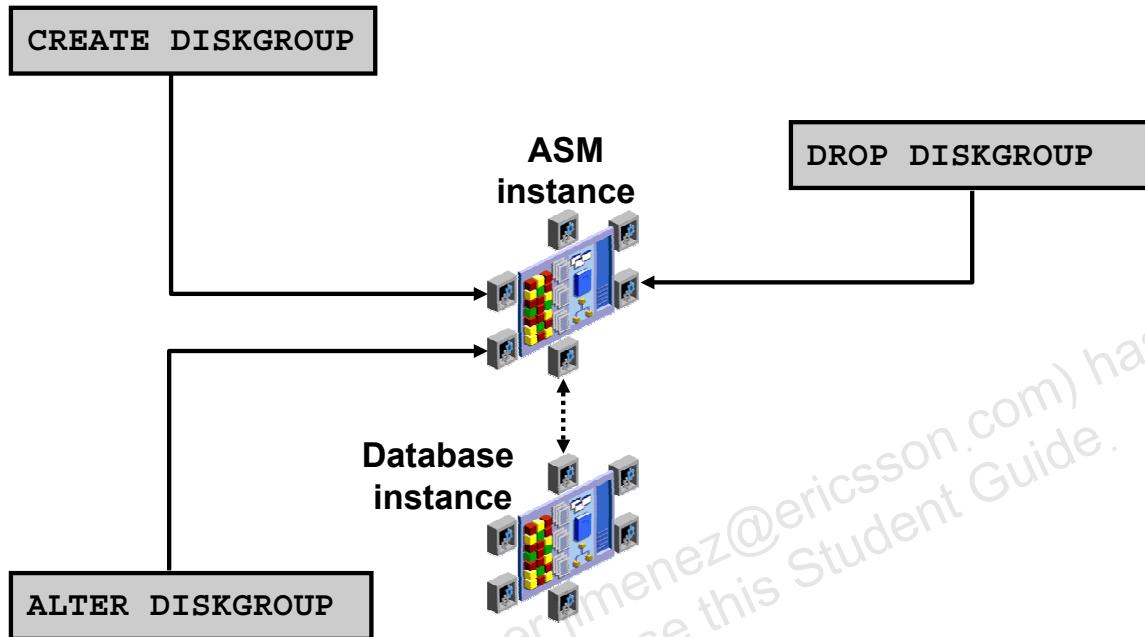
ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Disk Group Dynamic Rebalancing

- With ASM, the rebalance process is very easy and happens without any intervention from the DBA or system administrator. ASM automatically rebalances a disk group whenever disks are added or dropped.
- By using index techniques to spread AUs on the available disks, ASM does not need to restripe all of the data, but instead needs to only move an amount of data proportional to the amount of storage added or removed to evenly redistribute the files and maintain a balanced I/O load across the disks in a disk group.
- With the I/O balanced whenever files are allocated and whenever the storage configuration changes, the DBA never needs to search for hot spots in a disk group and manually move data to restore a balanced I/O load.
- It is more efficient to add or drop multiple disks at the same time so that they are rebalanced as a single operation. This avoids unnecessary movement of data. With this technique, it is easy to achieve online migration of your data. All you need to do is add the new disks in one operation and drop the old ones in one operation.
- You can control how much of a load the rebalance operation has on the system by setting the `ASM_POWER_LIMIT` initialization variable. Its range of values is 0 through 11. The lower the number, the lighter the load, whereas a higher setting has more of a load, and finishes sooner. A setting of 0 places rebalance operations on hold. The default value is 1.

# Managing Disk Groups



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Managing Disk Groups

The main goal of an ASM instance is to manage disk groups and protect their data. ASM instances also communicate file layout to database instances. In this way, database instances can directly access files stored in disk groups.

There are several new disk group administrative commands. They all require the SYSDBA privilege and must be issued from an ASM instance.

You can add new disk groups. You can also modify existing disk groups to add new disks, remove existing ones, and perform many other operations. You can remove existing disk groups.

# ASM Administration Page

**Disk Groups**

| Select                           | Name    | State   | Redundancy | Size (MB) | Used (MB) | Used (%) | Member Disks | Pending Operations |
|----------------------------------|---------|---------|------------|-----------|-----------|----------|--------------|--------------------|
| <input checked="" type="radio"/> | DGROUP1 | MOUNTED | EXTERN     | 2600      | 1046      | 40.23    | 130          | 0                  |

**Disk Group: DGROUP1**

General Performance Templates Files

Name DGROUP1  
State MOUNTED  
Redundancy EXTERN  
Total (GB) 2.54 GB  
Free (GB) 1.52 GB  
Pending Operations 0

**Disk Group Usage (GB)**

**Disk Group Usage History (GB)**

No data is currently available.

**Member Disks**

| Select ASM Disk Name             | By Failure Group | Path                 | Read/Write Errors | State  | Size (GB) | Used (GB) | Used (%) |
|----------------------------------|------------------|----------------------|-------------------|--------|-----------|-----------|----------|
| <input checked="" type="radio"/> | DGROUP1_0000     | /u01/asmdisks/disk00 | 0                 | NORMAL | 0.20      | 0.08      | 40.5     |
| <input type="radio"/>            | DGROUP1_0001     | /u01/asmdisks/disk01 | 0                 | NORMAL | 0.20      | 0.08      | 39.5     |

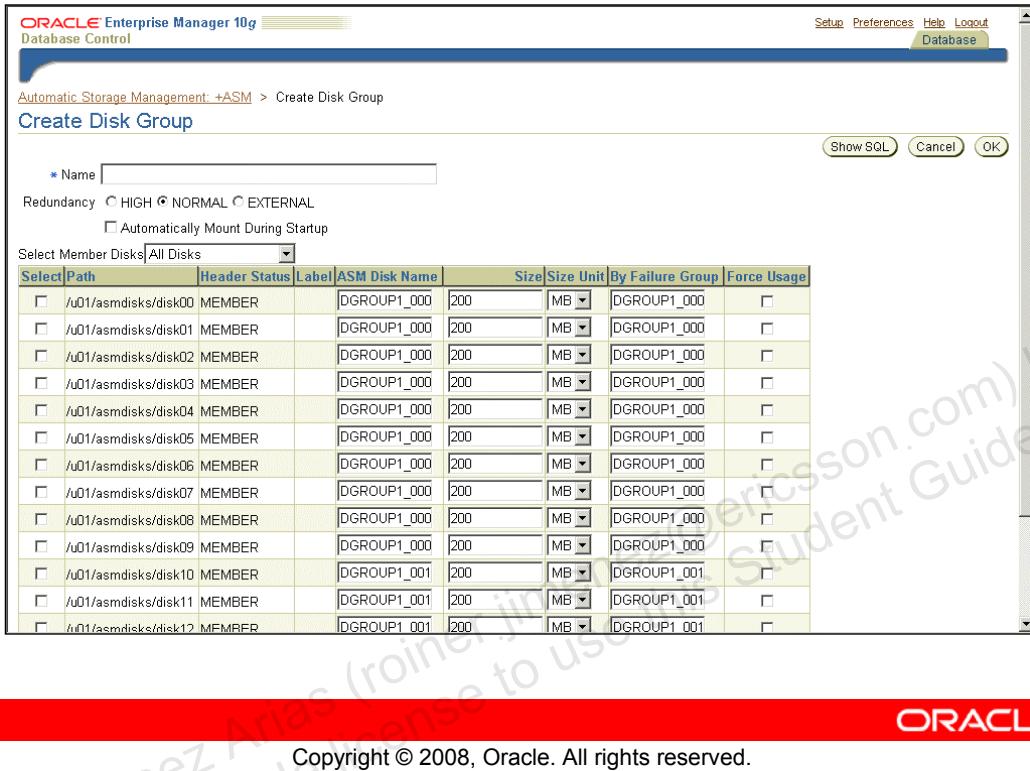
**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## ASM Administration Page

When you click the Administration tabbed page of the Automatic Storage Management page, you can see the disk groups listed in the V\$ASM\_DISKGROUP view. From here, you can create, edit, or drop a disk group. You can also perform disk group operations such as mount, dismount, rebalance, check, and repair on a selected disk group.

# Create Disk Group Page



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Create Disk Group Page

Clicking Create on the Disk Group Overview page displays the Create Disk Group page. You can enter the disk group name, redundancy mechanism, and the list of disks that you would like to include in the new disk group.

The list of disks is obtained from the V\$ASM\_DISK fixed view. By default, only the disks that can be assigned to a disk group show up. Those are the ones with a status of one of the following:

- **CANDIDATE:** The disk has never been assigned to an ASM disk group.
- **FORMER:** The disk was once assigned to an ASM disk group, but is not now.
- **PROVISIONED:** ASMLib is being used, and this disk is not yet assigned to a disk group.

**Note:** ASMLib is an API that interfaces with other vendors' storage arrays. See the *Database Administrator's Guide* documentation for more information about ASMLib.

# Creating and Dropping Disk Groups

```
CREATE DISKGROUP dgroupA NORMAL REDUNDANCY
FAILGROUP controller1 DISK
 '/devices/A1' NAME diskA1 SIZE 120G FORCE,
 '/devices/A2',
 '/devices/A3'

FAILGROUP controller2 DISK
 '/devices/B1',
 '/devices/B2',
 '/devices/B3';
```

```
DROP DISKGROUP dgroupA INCLUDING CONTENTS;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Creating and Dropping Disk Groups

Assume that ASM disk discovery identified the following disks in the /devices directory: A1, A2, A3, B1, B2, and B3. Also, assume that disks A1, A2, and A3 are on a separate SCSI controller from disks B1, B2, and B3. The first example in the slide illustrates how to configure a disk group called DGROUPA with two failure groups: CONTROLLER1 and CONTROLLER2.

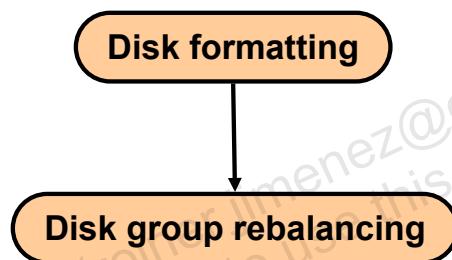
The example also uses the default redundancy characteristic, NORMAL REDUNDANCY, for the disk group. You can optionally provide a disk name and size for the disk. If you do not supply this information, ASM creates a default name and attempts to determine the size of the disk. If the size cannot be determined, an error is returned. FORCE indicates that a specified disk should be added to the specified disk group even though the disk is already formatted as a member of an ASM disk group. Using the FORCE option for a disk that is not formatted as a member of an ASM disk group returns an error.

As shown by the second statement in the slide, you can delete a disk group along with all its files. To avoid accidental deletions, the INCLUDING CONTENTS option must be specified if the disk group still contains any files besides internal ASM metadata. The disk group must be mounted in order for it to be dropped. After ensuring that none of the disk group files are open, the group and all its drives are removed from the disk group. Then the header of each disk is overwritten to eliminate the ASM formatting information.

# Adding Disks to Disk Groups

```
ALTER DISKGROUP dgroupA ADD DISK
 '/dev/rdsk/c0t4d0s2' NAME A5,
 '/dev/rdsk/c0t5d0s2' NAME A6,
 '/dev/rdsk/c0t6d0s2' NAME A7,
 '/dev/rdsk/c0t7d0s2' NAME A8;
```

```
ALTER DISKGROUP dgroupA ADD DISK '/devices/A*';
```



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Adding Disks to Disk Groups

This example shows how to add disks to a disk group. You execute an ALTER DISKGROUP ADD DISK command to add the disks. The first statement adds four new disks to the DGROUPA disk group.

The second statement demonstrates the interactions of discovery strings. Consider the following configuration:

/devices/A1 is a member of disk group DGROUPA.  
/devices/A2 is a member of disk group DGROUPA.  
/devices/A3 is a member of disk group DGROUPA.  
/devices/A4 is a candidate disk.

The second command adds A4 to the DGROUPA disk group. It ignores the other disks, even though they match the discovery string, because they are already part of the DGROUPA disk group. As shown by the diagram, when you add a disk to a disk group, the ASM instance ensures that the disk is addressable and usable. The disk is then formatted and rebalanced. The rebalance process is time consuming because it moves AUs from every file onto the new disk.

## **Adding Disks to Disk Groups (continued)**

**Note:** Rebalancing does not block any database operations. The main impact that a rebalance process has is on the I/O load on the system. The higher the power of the rebalance, the more I/O load it puts on the system. Thus less I/O bandwidth is available for database I/Os.

## Miscellaneous ALTER Commands

Remove a disk from dgroupA:

```
ALTER DISKGROUP dgroupA DROP DISK A5;
```

Add and drop a disk in a single command:

```
ALTER DISKGROUP dgroupA
 DROP DISK A6
 ADD FAILGROUP fred
 DISK '/dev/rdsk/c0t8d0s2' NAME A9;
```

Cancel a disk drop operation:

```
ALTER DISKGROUP dgroupA UNDROP DISKS;
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Miscellaneous ALTER Commands

The first statement in the slide shows how to remove one of the disks from the DGROUPA disk group. The second statement shows how you can add and drop a disk in a single command. The big advantage in this case is that rebalancing is not started until the command completes. The third statement shows how to cancel a disk drop operation. The UNDROP command operates only on pending drops of disks; it has no effect on drops that have completed.

The following statement rebalances the DGROUPE disk group, if necessary:

```
ALTER DISKGROUP dgroupB REBALANCE POWER 5;
```

This command is generally not necessary because it is automatically done as disks are added, dropped, or resized. However, it is useful if you want to use the POWER clause to override the default speed defined by the initialization parameter ASM\_POWER\_LIMIT. You can change the power level of an ongoing rebalance operation by reentering the command with a new level. A power level of zero causes rebalancing to halt until the command is either implicitly or explicitly reinvoked.

The following statement dismounts DGROUPE:

```
ALTER DISKGROUP dgroupA DISMOUNT;
```

The MOUNT and DISMOUNT options allow you to make one or more disk groups available or unavailable to the database instances.

## Miscellaneous ALTER Commands (continued)

Use the following statement to verify the internal consistency of disk group metadata and to repair any error found:

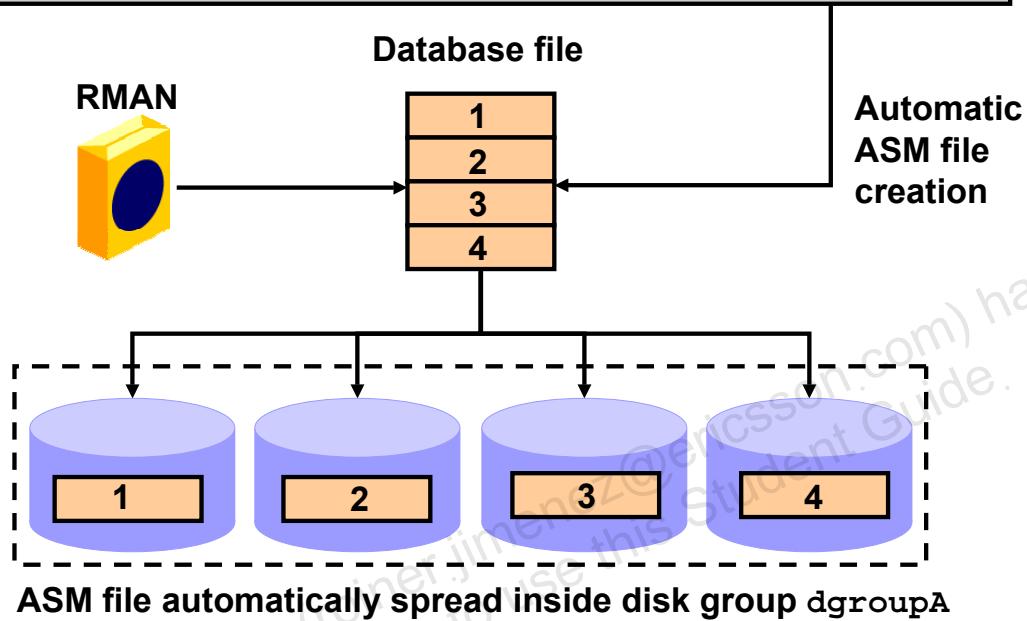
```
ALTER DISKGROUP dgroupA CHECK ALL;
```

It is also possible to use the NOREPAIR clause if you just want to be alerted about errors. While the example requests a check across all disks in the disk group, checking can be specified on a file or an individual disk. This command requires that the disk group be mounted. If any error is found, a summary error message is displayed and the details of the detected error are reported in the alert log.

**Note:** Of these six examples, the first four trigger a disk group rebalancing, and the last two do not.

# ASM Files

```
CREATE TABLESPACE sample DATAFILE '+dgroupA';
```



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## ASM Files

When you specify an ASM disk group as the data file name for a tablespace, ASM files are created in the disk group to provide storage for the tablespace.

When an ASM file is created, certain file attributes are permanently set. Among these are its protection policy, and its striping policy. ASM files are Oracle Managed Files. Any file that is created by ASM is automatically deleted when it is no longer needed.

With ASM, file operations are specified in terms of database objects. Administration of databases never requires knowing the name of a file, though the name of the file is exposed through some data dictionary views, or the ALTER DATABASE BACKUP CONTROLFILE TO TRACE command. Because each file in a disk group is physically spread across all disks in the disk group, a backup of a single disk is not useful. Database backups of ASM files must be made with RMAN.

**Note:** ASM does not manage binaries, alert logs, trace files, or password files.

## ASMCMD Utility

```
SQL> CREATE TABLESPACE tbsasm DATAFILE '+DGROUP1' SIZE 100M;
Tablespace created.
```

```
SQL> CREATE TABLESPACE hrapps DATAFILE '+DGROUP1' SIZE 10M;
Tablespace created.
```

```
$ asmcmd
ASMCMD> ls -l DGROUP1/ORCL/DATAFILE
Type Redund Striped Time Sys Name
DATAFILE MIRROR COARSE OCT 05 21:00:00 Y HRAPPS.257.570923611
DATAFILE MIRROR COARSE OCT 05 21:00:00 Y TBSASM.256.570922917
ASMCMD>
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### ASMCMD Utility

ASMCMD is a command-line utility that you can use to easily view and manipulate files and directories within ASM disk groups. It can list the contents of disk groups, perform searches, create and remove directories, and display space utilization, among other things.

**Note:** For more information about ASMCMD, see the *Oracle Database Utilities* documentation.

# Migrating Your Database to ASM Storage

- 1. Shut down your database cleanly.**
- 2. Shut down the database and modify your server parameter file to use Oracle Managed Files (OMF).**
- 3. Edit and execute the following RMAN script:**

```
STARTUP NOMOUNT;
RESTORE CONTROLFILE FROM '/u1/c1.ctl';
ALTER DATABASE MOUNT;
BACKUP AS COPY DATABASE FORMAT '+dgroup1';
SWITCH DATABASE TO COPY;
SQL "ALTER DATABASE RENAME '/u1/log1' TO '+dgroup1' ";
Repeat RENAME command for all online redo log members
...
ALTER DATABASE OPEN RESETLOGS;
SQL "ALTER DATABASE TEMPFILE '/u1/temp1' DROP";
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Migrating Your Database to ASM Storage

Because ASM files cannot be accessed through normal operating system interfaces, RMAN is the only means for copying ASM files. Although files in a tablespace may be both ASM files and non-ASM files as a result of the tablespace history, RMAN commands enable non-ASM files to be relocated to an ASM disk group. You can use the following procedure to relocate your entire database to an ASM disk group: (It is assumed that you are using a server parameter file.)

1. Obtain the file names of the current control files and online redo logs by using V\$CONTROLFILE and V\$LOGFILE.
2. Shut down the database consistently. Modify the server parameter file of your database as follows:
  - Set the necessary OMF destination parameters to the desired ASM disk group.
  - Remove the CONTROL\_FILES parameter.

## Migrating Your Database to ASM Storage (continued)

3. Edit and run the RMAN command file, which backs up the database, switches the current data files to the backups, and renames the online redo logs. You can move only tablespaces or data files by using the BACKUP AS COPY command.
4. Delete the old database files.

**Note:** If you create an OMF control file, and if there is a server parameter file, then a CONTROL\_FILES initialization parameter entry is created in the server parameter file.

See the *Oracle Database Backup and Recovery Advanced User's Guide* for details about how to migrate a database to ASM.

## Summary

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

- **Describe the concepts of Automatic Storage Management (ASM)**
- **Set up initialization parameter files for ASM and database instances**
- **Execute SQL commands with ASM file names**
- **Start up and shut down ASM instances**
- **Administer ASM disk groups**
- **Use RMAN to migrate your database to ASM**



Copyright © 2008, Oracle. All rights reserved.

# Practice Overview: Using Automatic Storage Management

**This practice covers the following topics:**

- **Creating and starting an ASM instance**
- **Creating and using ASM disk groups**
- **Migrating a tablespace to ASM storage**



Copyright © 2008, Oracle. All rights reserved.

# 13

## Managing Resources

ORACLE®

Copyright © 2008, Oracle. All rights reserved.

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a  
non-transferable license to use this Student Guide.

# Objectives

> Concepts  
Resource Plan  
Consumer Group  
Plan Directives  
Mapping  
Plan activation  
Monitoring

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

- **Configure the Database Resource Manager**
- **Access and create resource plans**
- **Create consumer groups**
- **Specify directives for allocating resources to consumer groups**
- **Map consumer groups to plans**
- **Activate a resource plan**
- **Monitor the Resource Manager**

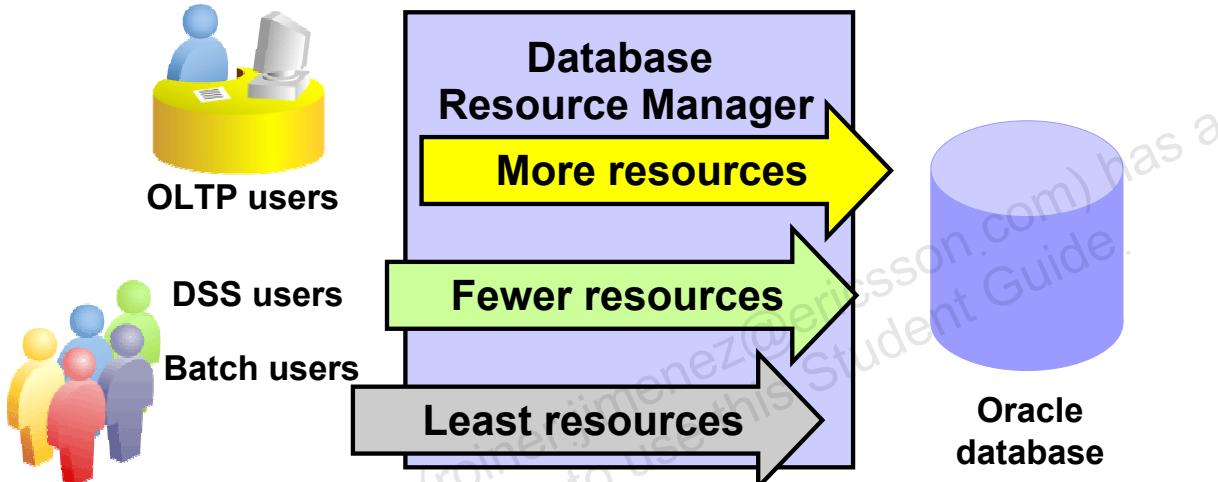
**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

# Database Resource Manager: Overview

## Use the Resource Manager to:

- Manage mixed workload
- Control system performance



Copyright © 2008, Oracle. All rights reserved.

ORACLE

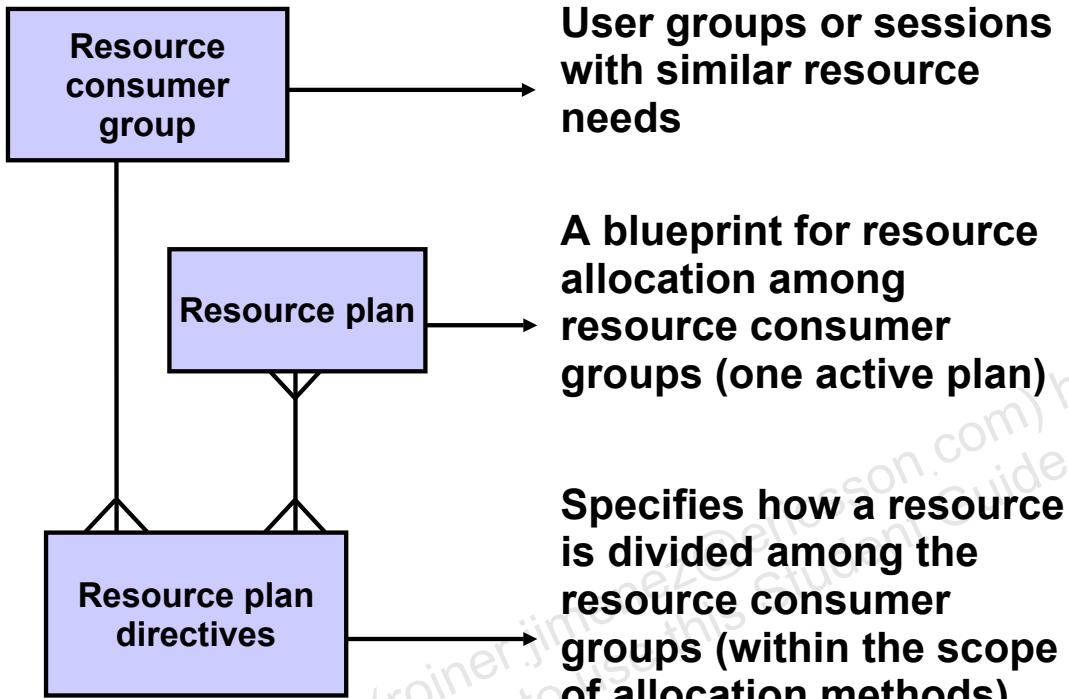
## Database Resource Manager: Overview

By using the Database Resource Manager (also called the Resource Manager), you have more control over the allocation of machine resources than is normally possible through operating system resource management alone. If resource management decisions are made by operating system, it can lead to problems such as:

- Excessive overhead resulting from operating system context switching of Oracle database server processes when the number of server processes is high
- Suspension of a database server processes that is holding a latch
- Unequal distribution of resources among all Oracle database processes, and an inability to prioritize one task over another
- Inability to manage database-specific resources, such as parallel execution servers and active sessions

The Database Resource Manager controls the distribution of resources among various sessions by controlling the execution schedule inside the database. By controlling which sessions run and for how long, the Database Resource Manager can ensure that resource distribution matches the plan directive and, therefore, the business objectives. With the Database Resource Manager, you can guarantee groups of users a minimum amount of processing resources regardless of the load on the system and the number of users.

# Database Resource Manager Concepts



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Database Resource Manager Concepts

Administering systems by using the Database Resource Manager involves the use of resource plans, resource consumer groups, and resource plan directives.

A **resource consumer group** defines a set of users or sessions that have similar requirements for using system and database resources.

A **resource plan** specifies how the resources are distributed among various resource consumer groups. The Database Resource Manager also allows for creation of plans within plans, called **subplans**.

**Resource plan directives** specify how a particular resource is shared among consumer groups or subplans. You associate resource consumer groups and subplans with a particular resource plan through plan directives.

**Resource allocation methods** determine what policy to use when allocating for any particular resource. Resource allocation methods are used by resource plans and resource consumer groups.

# Why Use Resource Manager

- You can manage database and operating system resources, such as:
  - CPU usage
  - Degree of parallelism
  - Number of active sessions
  - Undo generation
  - Operation execution time
  - Idle time
- You can also specify criteria that, if met, cause the automatic switching of sessions to another consumer group.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Why Use Resource Manager

The Database Resource Manager provides several means of allocating resources:

- **CPU Method:** Enables you to specify how CPU resources are allocated among consumer groups and subplans
- **Degree of Parallelism Limit:** Enables you to control the maximum degree of parallelism for any operation within a consumer group
- **Active Session Pool with Queuing:** Allows you to limit the number of concurrent active sessions for a consumer group or subplan. If a group exceeds the maximum allowed number of sessions, new sessions are placed in a queue where they wait for an active session to complete. You can also specify a time limit on how long a session will wait before exiting with an error.
- **Undo Pool:** Enables you to control the total amount of undo that can be generated by a consumer group or subplan. Whenever the total undo space exceeds the amount specified by UNDO\_POOL, no further INSERT, UPDATE, or DELETE commands are allowed until undo space is freed by another session in the same group or the undo pool is increased for the consumer group. If the consumer group's quota is exceeded during the execution of a DML statement, the operation aborts and returns an error. Queries are still allowed, even if a consumer group has exceeded its undo threshold.

## Why Use Resource Manager (continued)

- **Execution Time Limit:** Allows you to specify a maximum execution time allowed for an operation. The Oracle database uses cost-based optimizer statistics to estimate how long an operation will take. If it is longer than the maximum time allowed (`MAX_EST_EXEC_TIME`), the operation returns an error and is not started. If a resource consumer group has more than one plan directive with `MAX_EST_EXEC_TIME` specified, the Resource Manager chooses the most restrictive of all incoming values.
- **Idle Time Limit:** Enables you to specify an amount of time for which a session can be idle, after which it will be terminated (`MAX_IDLE_TIME`). You can further restrict the Resource Manager to only terminate sessions that are blocking other sessions (`MAX_IDLE_TIME_BLOCKER`).

# Accessing Resource Plans

## Enterprise Manager and PL/SQL

### Resource Plans



Concepts  
> **Resource Plan**  
Consumer Group  
Plan Directives  
Mapping  
Plan activation  
Monitoring

Object Type **Resource Plan**

#### Search

Select an object type and optionally enter an object name to filter the data that is displayed in your results set.

Object Name



By default, the search returns all uppercase matches beginning with the string you entered. To run an exact or case-sensitive match, double quote the search string. You can use the wildcard symbol (%) in a double quoted string.

**Create**

| Select                           | Plan             | Status | Description                           | Actions                                                                  |
|----------------------------------|------------------|--------|---------------------------------------|--------------------------------------------------------------------------|
| <input checked="" type="radio"/> | INTERNAL PLAN    |        | Default Plan                          | <b>Activate</b><br>Activate<br>Create Like<br>Deactivate<br>Generate DDL |
| <input type="radio"/>            | INTERNAL QUIESCE |        | Plan to internally quiesce            |                                                                          |
| <input type="radio"/>            | SYSTEM PLAN      |        | Plan to give system sessions priority |                                                                          |

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Accessing Resource Plans

### Using Enterprise Manager

The Enterprise Manager Database Control Console provides an easy-to-use graphical interface for configuring resource plans, consumer groups, and so on.

### Using the DBMS\_RESOURCE\_MANAGER Package

This PL/SQL package comprises many procedures, including the following:

- **CREATE\_PLAN:** Names a resource plan and specifies its allocation methods
- **UPDATE\_PLAN:** Updates a resource plan's comment
- **DELETE\_PLAN:** Deletes a resource plan and its directives

## Example: SYSTEM\_PLAN

| Resource Consumer Group | Allocation Methods |        |        |
|-------------------------|--------------------|--------|--------|
|                         | CPU_P1             | CPU_P2 | CPU_P3 |
| SYS_GROUP               | 100%               | 0%     | 0%     |
| OTHER_GROUPS            | 0%                 | 100%   | 0%     |
| LOW_GROUP               | 0%                 | 0%     | 100%   |

Copyright © 2008, Oracle. All rights reserved.

### Example: SYSTEM\_PLAN

The SYSTEM\_PLAN resource plan is one of the default plans provided for you. It contains directives for the following provided consumer groups:

- **SYS\_GROUP:** The initial consumer group for the SYS and SYSTEM users
- **OTHER\_GROUPS:** Used for all sessions that belong to consumer groups that are not part of the active resource plan. There must be a plan directive for OTHER\_GROUPS in any active plan.
- **LOW\_GROUP:** A group with lower priority than SYS\_GROUP and OTHER\_GROUPS in this plan. You must decide which user sessions will be part of LOW\_GROUP. Initially, no user is associated with this consumer group. The switch privilege is granted to PUBLIC for this group.

The initial consumer group of a user is the consumer group to which any session created by that user initially belongs. If you have not set the initial consumer group for a user, the user's initial consumer group will automatically be DEFAULT\_CONSUMER\_GROUP.

The SYSTEM\_PLAN and associated resource consumer groups can be used or not used. It can be a template for new resource plans; it can be modified or deleted. Use it as appropriate for your environment.

# Creating a New Resource Plan

Database Instance: orcl.oracle.com > Resource Plans > Create Resource Plan      Logged in As SYS

## Create Resource Plan

Show SQL Cancel OK

**General** Parallelism Session Pool Undo Pool Maximum Execution Time Consumer Group Switching Idle Time

\* Plan DAY\_PLAN  
Description Resource plan for weekday use  
 Activate this plan  
 Automatic Plan Switching Enabled

**Selected Groups/Subplans** Modify

**Group/Subp** OTHER\_GRO

```
DBMS_RESOURCE_MANAGER.SWITCH_PLAN
(PLAN_NAME => 'DAY_PLAN',
 SID => 'ORCL',
 ALLOW_SCHEDULER_PLAN_SWITCHES => true);
```

Copyright © 2008, Oracle. All rights reserved. ORACLE

## Creating a New Resource Plan

To create a new plan, you need to configure several Resource Manager objects.

Select Administration > Plans, and then click Create or choose “Create Like” and click Go.

The Scheduler can automatically change the Resource Manager plan at the Scheduler window boundaries. Deselect the default “Automatic Plan Switching Enabled,” if this is unacceptable.

For more information about the Scheduler, see the lesson titled “Automating Tasks with the Scheduler.”

# Creating Consumer Groups

Concepts  
Resource Plan  
Consumer Group  
Plan Directives  
Mapping  
Plan activation  
Monitoring

| Select Consumer Group                                     | Mandatory | Description                                        |
|-----------------------------------------------------------|-----------|----------------------------------------------------|
| <input checked="" type="radio"/> AUTO TASK CONSUMER GROUP | NO        | System maintenance task consumer group             |
| <input type="radio"/> DEFAULT CONSUMER GROUP              | YES       | consumer group for users not assigned to any group |
| <input type="radio"/> LOW GROUP                           | NO        | Group of low priority sessions                     |
| <input type="radio"/> SYS GROUP                           | YES       | Group of system sessions                           |

**Create Resource Consumer Group**

**General** Roles

\* Consumer Group: APPUSER

Description:

Scheduling Policy: Round Robin

**Selected Users**

Select User: No items found

```
DBMS_RESOURCE_MANAGER.CREATE_CONSUMER_GROUP (
 CONSUMER_GROUP => 'APPUSER',
 CPU_MTH => 'ROUND-ROBIN',
 COMMENT => '');
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Creating Consumer Groups

To manage a resource consumer group, select Administration > Resource Consumer Groups. Choose the action that you want to perform. You can create a new resource consumer group or you can select a resource consumer group from the results list and choose one of the actions from the Actions drop-down list.

Use the Resource Consumer Groups page to create or edit the consumer group and description, and to select users assigned to the group. You can add or delete members to the Selected Users table.

The General property sheet (shown in the slide) is one of the two pages that make up the Create Resource Consumer Group and Edit Resource Consumer Group pages. You can define or edit which database roles are associated with the specified resource consumer group by moving to the other property sheet (Roles).

You specify a resource allocation method for the distribution of CPU among sessions in the consumer group. “Round Robin” scheduling ensures that sessions are fairly executed. Therefore, the default allocation method is Round Robin. The “Run to Completion” allocation method specifies that sessions with the largest active time are scheduled ahead of other sessions. The equivalent functionality is achieved by the DBMS\_RESOURCE\_MANAGER.CREATE\_CONSUMER\_GROUP procedure with the CPU\_MTH option.

# Assigning Users to Consumer Groups

The screenshot shows the Oracle Database 10g User Management interface. The top navigation bar includes 'Database Instance: orcl.oracle.com > Users > Edit User: PM' and 'Logged in As SYS'. Below the navigation is a toolbar with 'Actions' dropdown, 'Create Like', 'Go', 'Show SQL', 'Revert', and 'Apply' buttons. A menu bar at the top has tabs for 'General', 'Roles', 'System Privileges', 'Object Privileges', 'Quotas', 'Consumer Groups', 'Switching Privileges' (which is selected), and 'Proxy Users'. A note below the tabs states: 'Resource consumer groups are groups of users, or sessions, that are grouped together based on their processing needs. If a user is granted permission to switch to a particular consumer group, then that user can switch their current consumer group to the new consumer group.' A large table titled 'Consumer Group' lists three entries: APPUSER, LOW\_GROUP, and SYS\_GROUP. Each row has a checkbox under 'Admin Option'. A 'Default Consumer Group' dropdown is set to 'None'. An 'Edit List' button is located in the top right corner of the table area. At the bottom of the page is a red footer bar with the 'ORACLE' logo and the text 'Copyright © 2008, Oracle. All rights reserved.'

## Assigning Users to Groups

Before enabling the Database Resource Manager, users must be assigned to resource consumer groups. The user's default consumer group is the one to which any session created by that user initially belongs. If it is not set for a user, the user's initial consumer group defaults to DEFAULT\_CONSUMER\_GROUP.

You must directly grant to the user, or to PUBLIC, the switch privilege to a consumer group before it can be the user's default consumer group. The switch privilege cannot come from a role granted to that user.

```
DBMS_RESOURCE_MANAGER.SET_INITIAL_CONSUMER_GROUP (
 user => 'PM', consumer_group => 'APPUSER');
```

The DBMS\_RESOURCE\_MANAGER\_PRIVS package contains the procedure to assign resource consumer groups to users. Granting the switch privilege to a user enables the user to switch to a different consumer group.

```
DBMS_RESOURCE_MANAGER_PRIVS.GRANT_SWITCH_CONSUMER_GROUP (
 grantee_name => 'PM',
 consumer_group => 'APPUSER',
 grant_option => FALSE);
```

You do not use a pending area for any of these procedures.

# Specifying Resource Plan Directives

Concepts  
Resource Plan  
Consumer Group  
> Plan Directives  
Mapping  
Plan activation  
Monitoring

**Edit Resource Plan: SYSTEM\_PLAN**

General Parallelism Session Pool Undo Pool Maximum Execution Time Consumer Group Switching Idle Time

Plan SYSTEM\_PLAN

Description Plan to give system sessions priority

Activate this plan  Automatic Plan Switching Enabled

**Selected Groups/Subplans**

| Group/Subplan | Level 1 | Level 2 | Level 3 | Level 4 | Level 5 | Level 6 | Level 7 | Level 8 |
|---------------|---------|---------|---------|---------|---------|---------|---------|---------|
| LOW_GROUP     | 0       | 0       | 100     | 0       | 0       | 0       | 0       | 0       |
| OTHER_GROUPS  | 0       | 100     | 0       | 0       | 0       | 0       | 0       | 0       |
| SYS_GROUP     | 100     | 0       | 0       | 0       | 0       | 0       | 0       | 0       |

**CPU\_MTH values**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Specifying Resource Plan Directives

If you do not use Enterprise Manager to create the resource plan or resource consumer groups, you must first create a *pending area*. This is a scratch area that enables you to stage your changes and to validate them before they are made active.

In Enterprise Manager, there are several property pages, which you can use for specifying plan directives:

1. On the General page, associate consumer groups with plans and specify how much CPU each consumer group or subplan gets with the CPU\_MTH value.
2. Specify a parallel degree limit to control the maximum degree of parallelism for any operation within a consumer group.
3. You can control the maximum number of concurrently active sessions allowed within a consumer group. An entire parallel execution session is counted as one active session.
4. You can control the amount of total undo that can be generated by a consumer group.
5. You can specify a maximum execution time allowed for an operation.
6. You can control resources by specifying criteria that, if met, cause the automatic switching of sessions to another consumer group.
7. You can specify an amount of time that a session can be idle, after which it will be terminated. You can further restrict such termination to only sessions that are blocking other sessions.

**Note:** The following slides provide further details about directives using the tab numbers indicated in this slide. Refer to this slide when you see "Directive Tab n" on subsequent slides.

# Resource Allocation Methods for Resource Plans

| Parameter                        | Possible Values                       |
|----------------------------------|---------------------------------------|
| <b>CPU_MTH</b>                   | <b>EMPHASIS</b>                       |
|                                  | <b>RATIO</b>                          |
| <b>PARALLEL_DEGREE_LIMIT_MTH</b> | <b>PARALLEL_DEGREE_LIMIT_ABSOLUTE</b> |
| <b>ACTIVE_SESS_POOL_MTH</b>      | <b>ACTIVE_SESS_POOL_ABSOLUTE</b>      |
| <b>QUEUING_MTH</b>               | <b>FIFO_TIMEOUT</b>                   |

Copyright © 2008, Oracle. All rights reserved.

## Resource Allocation Methods for Resource Plans

Resource allocation methods determine how the Resource Manager allocates a particular resource to a resource consumer group or resource plan. You specify values for the following resource allocation methods when creating the resource plan.

There are two ways of specifying the CPU distribution with the **CPU\_MTH** parameter:

- **EMPHASIS**, the default method, is for multilevel plans that use percentages to specify how CPU is distributed among consumer groups.
- **RATIO** is for single-level plans that use ratios to specify how CPU is distributed.

**PARALLEL\_DEGREE\_LIMIT\_MTH** limits the maximum degree of parallelism of any operation. This method can be specified only for resource consumer groups, not subplans. The **ABSOLUTE** method is the possible value, specifying how many processes may be assigned to an operation. If there are multiple plan directives referring to the same subplan or consumer group, the minimum of all the possible values is used as the parallel degree limit for that subplan or consumer group.

**ACTIVE\_SESS\_POOL\_MTH** limits the number of active sessions. All other sessions are inactive and wait in a queue to be activated. **ACTIVE\_SESS\_POOL\_ABSOLUTE** is the default and only method available.

**QUEUING\_MTH** controls the order in which queued inactive sessions execute. **FIFO\_TIMEOUT** is the default and only method available.

## Comparison of EMPHASIS and RATIO

| EMPHASIS                                                                              | RATIO                                                                                                         |
|---------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------|
| The value specifies the maximum percentage of CPU resources a consumer group can use. | The value specifies a number that indicates the ratio of CPU resources to be allocated to the consumer group. |
| You can allocate resources for up to 8 different levels.                              | You can specify values for only one level.                                                                    |
| The sum of percentages at any given level must be less than or equal to 100.          | You must use integer values, but there is no limit on the sum of values.                                      |
| Default value is NULL.                                                                | Default value is NULL.                                                                                        |

Copyright © 2008, Oracle. All rights reserved.

### Comparison of EMPHASIS and RATIO

The EMPHASIS CPU allocation method determines how much emphasis is given to sessions in different consumer groups in a resource plan. CPU usage is assigned levels from 1 through 8, with level 1 having the highest priority. Percentages specify how to allocate CPU to each consumer group at each level.

The following rules apply for the EMPHASIS resource allocation method:

- CPU resources are distributed at a given level on the basis of the specified percentages. The percentage of CPU specified for a resource consumer group is a maximum for how much that consumer group can use at a given level.
- Consumer resources that are not used at a given level are made available to consumer groups at next level. For example, if the consumer groups at Level 1 use only 60% of the available resources, the additional 40% is made available to consumer groups at Level 2.
- The sum of percentages at any given level must be less than or equal to 100.
- Any levels that have no plan directives explicitly specified have a default of 0% for all subplans or consumer groups.
- The EMPHASIS resource allocation method avoids starvation problems, where consumers with lower priorities are not given the opportunity to run.

## **Comparison of EMPHASIS and RATIO (continued)**

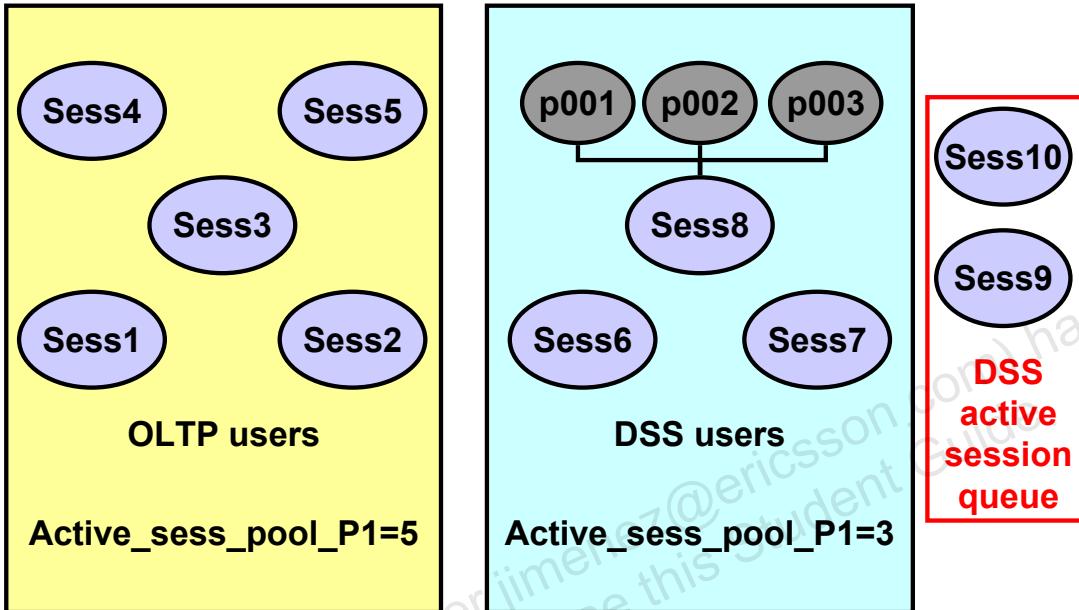
The RATIO policy is a single-level CPU allocation method. Instead of percentages, you specify numbers corresponding to the ratio of CPU you want to give to the consumer group. For example, given three consumer groups OLTP\_USERS, DSS\_USERS, and BATCH\_USERS, you can specify the following ratios:

- **OLTP\_USERS**: 4
- **DSS\_USERS**: 3
- **BATCH\_USERS**: 2
- **OTHER**: 1

This is similar to saying that OLTP users should get 40% of the resources, DSS users should get 30% of the resources, BATCH users should get 20% of the resources, and all other consumer groups should get 10% of the available resources.

If there are no consumers in the OTHER or DSS\_USERS consumer groups currently utilizing CPU resources, then the OLTP\_USERS consumer group would get two-thirds of the available resources and the BATCH\_USERS consumer group would get the other third.

# Active Session Pool Mechanism



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Active Session Pool Mechanism

Using the active session pool feature, you can control the maximum number of concurrently active sessions per resource consumer group. With this functionality, a DBA can indirectly control the amount of resources that any resource consumer group uses because resource consumption is proportional to the number of active sessions. Using an active session pool can help to reduce the number of servers taking resources in the system, thus avoiding inefficient paging, swapping, and other resource depletion (such as memory) resulting from attempting to run too many jobs simultaneously.

After the active session pool is filled with active sessions, the Resource Manager queues all subsequent sessions attempting to become active until other active sessions complete or become inactive. An active session is one currently involved in a transaction, query, or parallel operation. Individual parallel slaves are not counted as sessions; the entire parallel operation counts as one active session.

There is only one queue per resource consumer group and the queuing method is first in, first out (FIFO) with a timeout. The queue is implemented as a memory structure and cannot be queried directly.

# Setting the Active Session Pool

Database Instance: orcl.oracle.com > Resource Plans > Edit Resource Plan: SYSTEM\_PLAN Logged in As SYS

Edit Resource Plan: SYSTEM\_PLAN

Actions Create Like Go Show SQL Revert Apply

General Parallelism Session Pool Undo Pool Maximum Execution Time Consumer Group Switching Idle Time

Specify a limit on the maximum number of concurrently active sessions for a consumer group. All other sessions will wait in an activation queue.

| Group        | Maximum Number of Active Sessions | Activation Queue Timeout (sec) |
|--------------|-----------------------------------|--------------------------------|
| APPUSER      | UNLIMITED                         | UNLIMITED                      |
| LOW_GROUP    | UNLIMITED                         | UNLIMITED                      |
| OTHER_GROUPS | UNLIMITED                         | UNLIMITED                      |
| SYS_GROUP    | UNLIMITED                         | UNLIMITED                      |



Copyright © 2008, Oracle. All rights reserved.

## Setting the Active Session Pool

You can easily configure the Active Session Pool settings for a resource plan by using Enterprise Manager.

## Maximum Estimated Execution Time

- **The Database Resource Manager can estimate the execution time of an operation proactively.**
- **You can specify a maximum estimated execution time for an operation at the resource consumer group level.**
- **Operation does not start if the estimate is longer than MAX\_EST\_EXEC\_TIME. (ORA-07455)**
- **The benefit of this feature is the elimination of the exceptionally large job that uses too many system resources.**
- **The default is UNLIMITED.**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Maximum Estimated Execution Time

You can define the maximum estimated execution time any operation can take at any given time by setting the resource plan directive's MAX\_EST\_EXEC\_TIME parameter. When this parameter is set, the Database Resource Manager estimates the time a specific job will take. If the operation's estimate is more than MAX\_EST\_EXEC\_TIME, then the operation does not start and the ORA-07455 error is issued. This eliminates any exceptionally large jobs that would utilize too many system resources.

If a resource consumer group has more than one plan directive referring to it, it may have more than one MAX\_EST\_EXEC\_TIME. The Database Resource Manager then chooses the most restrictive of all incoming values.

The estimated execution time for a given statement is calculated using the statistics from the cost-based optimizer.

**Directive**  
**Tab 6**

# Consumer Group Switching

Database Instance: orcl.oracle.com > Resource Plans > Edit Resource Plan: SYSTEM\_PLAN Logged in As SYS

Edit Resource Plan: SYSTEM\_PLAN

Actions Create Like Go Show SQL Revert Apply

General Parallelism Session Pool Undo Pool Maximum Execution Time Consumer Group Switching Idle Time

Specify the maximum time that a session can execute in a consumer group before a selected action is taken. To select the action taken, you may choose to kill the session, or to cancel the current SQL operation, or to switch to another consumer group. If you choose to switch to another consumer group, specify whether to switch back at the end of the call by using the 'switch back to original group after call' checkbox. Choosing to switch back to the original group after a call is useful for 3-tier applications where the middle tier server is using session pooling.

| Group        | Maximum Execution Time (sec) | Switch back to original group after call? | Action To Take                           | Use estimate?                       |
|--------------|------------------------------|-------------------------------------------|------------------------------------------|-------------------------------------|
| APPUSER      | UNLIMITED                    | <input type="checkbox"/>                  | Kill This Session                        | <input checked="" type="checkbox"/> |
| LOW_GROUP    | UNLIMITED                    | <input type="checkbox"/>                  | Cancel SQL                               | <input type="checkbox"/>            |
| OTHER_GROUPS | UNLIMITED                    | <input type="checkbox"/>                  | Switch to Group APPUSER                  | <input type="checkbox"/>            |
| SYS_GROUP    | UNLIMITED                    | <input type="checkbox"/>                  | Switch to Group AUTO_TASK_CONSUMER_GROUP | <input type="checkbox"/>            |
|              |                              |                                           | Switch to Group BATCH_GROUP              | <input type="checkbox"/>            |
|              |                              |                                           | Switch to Group DEFAULT_CONSUMER_GROUP   | <input type="checkbox"/>            |
|              |                              |                                           | Switch to Group LOW_GROUP                | <input type="checkbox"/>            |
|              |                              |                                           | Switch to Group OTHER_GROUPS             | <input type="checkbox"/>            |
|              |                              |                                           | Switch to Group SYS_GROUP                | <input type="checkbox"/>            |

Copyright © 2008, Oracle. All rights reserved.

ORACLE

## Consumer Group Switching

The EM Database Control Console provides a graphical interface for configuring automatic consumer group switching. You can configure this option when creating a new resource plan, or by editing an existing plan.

**Note:** You cannot set SWITCH\_TIME\_IN\_CALL when using the Enterprise Manager interface. You can set SWITCH\_TIME, which is displayed as the Execution Time in the screenshot in the slide.

# Switching Back to the Initial Consumer Group at the End of Call

Database Instance: EDRSR15P1 orcl.oracle.com > Resource Plans > Create Resource Plan

Create Resource Plan

General Parallelism Session Pool Undo Pool Maximum Execution Time Consumer Group Switching Idle Time

Specify the maximum time that a session can execute in a consumer group before a selected action is taken. To select the action taken, you may choose to kill the session, or to cancel the current SQL operation, or to switch to another consumer group. If you choose to switch to another consumer group, specify whether to switch back at the end of the call by using the 'switch back to original group after call' checkbox. Choosing to switch back to the original group after a call is useful for 3-tier applications where the middle tier server is using session pooling.

| Group         | Maximum Execution Time (sec) | Switch back to original group after call? | Action To Take                                                                                                                                                                                                                           | Use estimate?            |
|---------------|------------------------------|-------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|
| DSS_GROUP     | 600                          | <input checked="" type="checkbox"/>       | Kill This Session<br>Cancel SQL<br>Switch to Group DEFAULT_CONSUMER_GROUP<br>Switch to Group DSS_GROUP<br><b>Switch to Group LONGRUN_GROUP</b><br>Switch to Group LOW_GROUP<br>Switch to Group OTHER_GROUPS<br>Switch to Group SYS_GROUP | <input type="checkbox"/> |
| LONGRUN_GROUP | UNLIMITED                    | <input type="checkbox"/>                  |                                                                                                                                                                                                                                          | <input type="checkbox"/> |
| OTHER_GROUPS  | UNLIMITED                    | <input type="checkbox"/>                  |                                                                                                                                                                                                                                          | <input type="checkbox"/> |

Copyright © 2008, Oracle. All rights reserved.

## Switching Back to the Initial Consumer Group at the End of Call

You can specify that at the end of every top call, a session is returned back to its initial consumer group by selecting the “Switch back to original group after call?” check box in the Create Resource Plan window. The initial consumer group is the group that a session would be in had it just logged in. The top call is defined as treating an entire PL/SQL block as one call or, similarly, treating SQL statements that are issued separately by the client as separate calls.

This functionality is mostly beneficial for three-tier applications where the middle-tier server implements session pooling. In this case, the middle tier tends to do one call for an end user and then use the same session for a call for a different end user. Therefore, the boundaries of work are really calls, and the actions of a prior end user should not affect the next end user.

## Switching Back to the Initial Consumer Group at the End of Call

### Call 1

```
DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE
 (PLAN => 'Day_Plan',
 GROUP_OR_SUBPLAN => 'DSS_GROUP',
 CPU_P1 => 100, CPU_P2 => 0,
 SWITCH_GROUP => 'LONGRUN_GROUP',
 SWITCH_TIME_IN_CALL => 600);
```

### Call 2

```
DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE
 (PLAN => 'Day_Plan',
 GROUP_OR_SUBPLAN => 'LONGRUN_GROUP',
 CPU_P1 => 0, CPU_P2 => 100);
```

At call end

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Switching Back to the Initial Consumer Group at the End of Call (continued)

Using the previous Enterprise Manager scenario example, but with the DBMS\_RESOURCE\_MANAGER.CREATE\_PLAN\_DIRECTIVE procedure, when a user logs in, the Resource Manager places the session in the DSS\_GROUP consumer group because of its initial consumer group mapping. The user then executes a query. When you set a value (in seconds) for the SWITCH\_TIME\_IN\_CALL parameter, you are setting a timer on the call before an action is taken, dictated by the SWITCH\_GROUP parameter. At the end of the top call, the Resource Manager automatically switches the user back to the initial consumer group.

**Note:** You cannot specify both the SWITCH\_TIME\_IN\_CALL and SWITCH\_TIME parameters within the same directive. The SWITCH\_TIME parameter is primarily intended for client/server applications.

**Directive**  
Tab 7

## Setting Idle Timeouts

Database Instance: EDRSR15P1 orcl.oracle.com > Resource Plans > Edit Resource Plan: DAY\_PLAN

Edit Resource Plan: DAY\_PLAN

Actions Create Like Go

General Parallelism Session Pool Undo Pool Maximum Execution Time Consumer Group Switching Idle Time

Plan DAY\_PLAN

Description Limit Idle Time Example

| Group        | Max Idle Time (sec) | Max Idle Time if Blocking Another Session (sec) |
|--------------|---------------------|-------------------------------------------------|
| DSS_GROUP    | 600                 | 300                                             |
| OTHER_GROUPS | UNLIMITED           | UNLIMITED                                       |

```
DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE
(PLAN => 'DAY_PLAN',
 GROUP_OR_SUBPLAN => 'DSS_GROUP',
 COMMENT => 'Limit Idle Time Example',
 MAX_IDLE_TIME => 600,
 MAX_IDLE_BLOCKER_TIME => 300);
```

Copyright © 2008, Oracle. All rights reserved.

### Setting Idle Timeouts

You use the resource plan's Idle Time tab to set the maximum idle timeouts for a resource plan. "Max Idle Time (sec)" and "Max Idle Time if Blocking Another Session (sec)" are the respective equivalents of the MAX\_IDLE\_TIME and MAX\_IDLE\_BLOCKER\_TIME resource directives in the DBMS\_RESOURCE\_MANAGER.CREATE\_PLAN\_DIRECTIVE procedure. They are both specified in seconds.

MAX\_IDLE\_TIME specifies the time that a session is neither executing nor waiting for I/O. When the session exceeds the specified limit, the PMON process forcibly kills the session and cleans up its state. In addition to limiting the maximum idle time for a session, you can also limit the amount of time that an idle session can block another session. You impose this limit by setting the MAX\_IDLE\_BLOCKER\_TIME resource directive to the number of seconds to allow a session to be idle while blocking another session. You can also specify a value of UNLIMITED to indicate that no maximum time has been set. The default is NULL, which means unlimited. These settings give you a more granular control than profiles, whose single value cannot distinguish between blocking and nonblocking sessions.

In the slide example, the PMON process kills sessions that are idle for longer than 600 seconds. The PMON process also kills sessions that are idle for more than 300 seconds and are blocking other sessions. PMON checks these limits once every minute and if it finds a session that has exceeded one of the limits, it forcibly kills the session and cleans up all its resources.

# Resource Consumer Group Mapping

Concepts  
Resource Plan  
Consumer Group  
Plan Directives  
>> **Mapping**  
Plan activation  
Monitoring

ORACLE Enterprise Manager 10g Database Control

Database Instance: EDRSR15P1 orcl.oracle.com > Resource Consumer Group Mapping

**Resource Consumer Group Mapping**

**General** **Priorities** SQL

Configure the Resource Manager to automatically assign consumer groups to sessions

**Oracle User Map**

| Select Consumer Group                      | Oracle User |
|--------------------------------------------|-------------|
| <input checked="" type="radio"/> SYS_GROUP | SYSTEM      |
| <input type="radio"/> SYS_GROUP            | SYS         |
| <a href="#">Add Another Row</a>            |             |

**Client DS User Map**

| Select Consumer Group           | Client OS |
|---------------------------------|-----------|
| <a href="#">Add Another Row</a> |           |

**Attribute Mappings**

| Service, Module, and Action |
|-----------------------------|
| Service and Module          |
| Module and Action           |
| Module                      |
| Service                     |
| Oracle User                 |
| Client Program              |
| Client OS User              |
| Client Machine              |

**General** **Priorities**

Reorder the list of mappings to set priorities. Mappin

Copyright © 2008, Oracle. All rights reserved.

## Resource Consumer Group Mapping

You can configure the Database Resource Manager to automatically assign consumer groups to sessions by providing mappings between session attributes and consumer groups. Further, you can prioritize the mappings so as to indicate which mapping has precedence in case of conflicts. There are two types of session attributes: login attributes and run-time attributes. The login attributes (the last five in the Attribute Mappings list shown in the slide) are meaningful only at session login time, when the Database Resource Manager determines the initial consumer group of the session. In contrast, a session that has already logged in can later be reassigned to another consumer group on the basis of its run-time attributes.

From the Database Control home page, navigate to the Administration tabbed page, and then click the Resource Consumer Group Mappings link in the Resource Manager section. For each of the attributes, set up a mapping that consists of a way to identify a session (for example, user name), and a consumer group. Add or remove rows for each of the resource consumer group categories, as required, and enter text identifying the user, client, module, or service in the corresponding group. You can establish a priority ordering between conflicting mappings of the attributes by using the Priorities tab. You can set the priority from the most important to the least important by using the navigational arrows (as highlighted). The mappings at the top of the list have the highest priority.

Using EM Database Control, you can easily view the SQL generated from your actions by clicking the Show SQL button.

## Resource Consumer Group Mapping (continued)

**Resource Consumer Group Mapping**

[Show SQL](#) [Revert](#) [Apply](#)

**General** [Priorities](#)

Configure the Resource Manager to automatically assign consumer groups to sessions by providing mappings between session attributes and consumer groups.

**Oracle User Map**

| Select                           | Consumer Group | Oracle User | Remove |
|----------------------------------|----------------|-------------|--------|
| <input checked="" type="radio"/> | SYS_GROUP      | SYS         |        |
| <input type="radio"/>            | SYS_GROUP      | SYSTEM      |        |

[Add Another Row](#)

**Client OS User Map**

| Select | Consumer Group | Client OS User |
|--------|----------------|----------------|
|        | No items found |                |

[Add Another Row](#)

**Client Program Map**

| Select | Consumer Group | Client Program |
|--------|----------------|----------------|
|        | No items found |                |

[Add Another Row](#)

**Client Machine Map**

| Select | Consumer Group | Client Machine |
|--------|----------------|----------------|
|        | No items found |                |

[Add Another Row](#)

**Service Map**

| Select | Consumer Group | Service |
|--------|----------------|---------|
|        | No items found |         |

[Add Another Row](#)

**Module Map**

| Select | Consumer Group | Module |
|--------|----------------|--------|
|        | No items found |        |

[Add Another Row](#)

**Module and Action Map**

| Select | Consumer Group | Module and Action |
|--------|----------------|-------------------|
|        | No items found |                   |

[Add Another Row](#)

# Activating a Resource Plan for an Instance

Concepts  
Resource Plan  
Consumer Group  
Plan Directives  
Mapping  
➤ Plan activation  
Monitoring

The screenshot shows the 'Resource Plans' page in Oracle Enterprise Manager. At the top, it says 'Database: orcl.us.oracle.com > Resource Plans' and 'Logged in As SYS'. Below that is a search bar with a 'Name' input field and a 'Go' button. A note below the search bar says: 'To run an exact match search or to run a case sensitive search, double quote the search criteria. The wildcard (%) symbol can still be used in a double quoted search string.' The main area is titled 'Results' and contains a table of resource plans:

| Select                           | Plan             | Status | Description                                   |
|----------------------------------|------------------|--------|-----------------------------------------------|
| <input type="radio"/>            | BUGDB_PLAN       |        | Resouce plan for the bug                      |
| <input type="radio"/>            | INTERNAL_PLAN    |        | Default Plan                                  |
| <input type="radio"/>            | INTERNAL QUIESCE |        | Plan to internally quiesce system             |
| <input type="radio"/>            | MAILDB_PLAN      |        | for mail delivery operations                  |
| <input type="radio"/>            | MYDB_PLAN        | ACTIVE | Planning for running the post office database |
| <input checked="" type="radio"/> | SYSTEM_PLAN      |        | Plan to give system sessions priority         |

A dropdown menu labeled 'Actions' is open over the last row, showing options: 'Activate', 'Create Like', 'Deactivate', and 'Generate DDL'. The 'Activate' option is highlighted. To the right of the table is a 'Create' button.

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Activating a Resource Plan for an Instance

You can use the “Resource Plans” page of Enterprise Manager to manage resource plans. To activate a plan, select the plan you want to make active, choose “Activate” from the Actions drop-down list, and then click Go. The plan you selected is then made the current top plan for the instance.

### Using the RESOURCE\_MANAGER\_PLAN Initialization Parameter

The plan for an instance is defined using the RESOURCE\_MANAGER\_PLAN database initialization parameter. This parameter specifies the top plan to be used for this instance. If no plan is specified, the Resource Manager is not activated for the instance.

You can activate, deactivate, or change the current top plan by using an ALTER SYSTEM statement. When a resource plan is changed using this command, the change takes effect instantly.

If the parameter is set in a parameter file, and the plan specified is not defined in the database, then the database cannot be opened with that parameter file. The following error is returned:

ORA-07452: specified resource manager plan does not exist in the data dictionary

If this error is encountered, the parameter must be modified to show a correct value before the instance can be restarted.

# Database Resource Manager Information

| View Name                     | Information                           |
|-------------------------------|---------------------------------------|
| DBA_RSRC_PLANS                | Plans and status                      |
| DBA_RSRC_PLAN_DIRECTIVES      | Plan directives                       |
| DBA_RSRC_CONSUMER_GROUPS      | Consumer groups                       |
| DBA_RSRC_CONSUMER_GROUP_PRIVS | Users/roles                           |
| DBA_RSRC_GROUP_MAPPINGS       | Consumer group mapping                |
| DBA_RSRC_MAPPING_PRIORITY     | Mapping priority                      |
| DBA_USERS                     | Column<br>initial_rsrc_consumer_group |
| DBA_RSRC_MANAGER_SYSTEM_PRIVS | Users/roles                           |

Copyright © 2008, Oracle. All rights reserved.

## Database Resource Manager Information

Several data dictionary views are available to check the resource plans, consumer groups, and plan directives that are declared in the instance. This section discusses some useful information that can be obtained from these views. For more detailed information about the contents of each of these views, refer to the *Oracle Database Reference* manual.

Use the following query to obtain information about resource plans defined in the database:

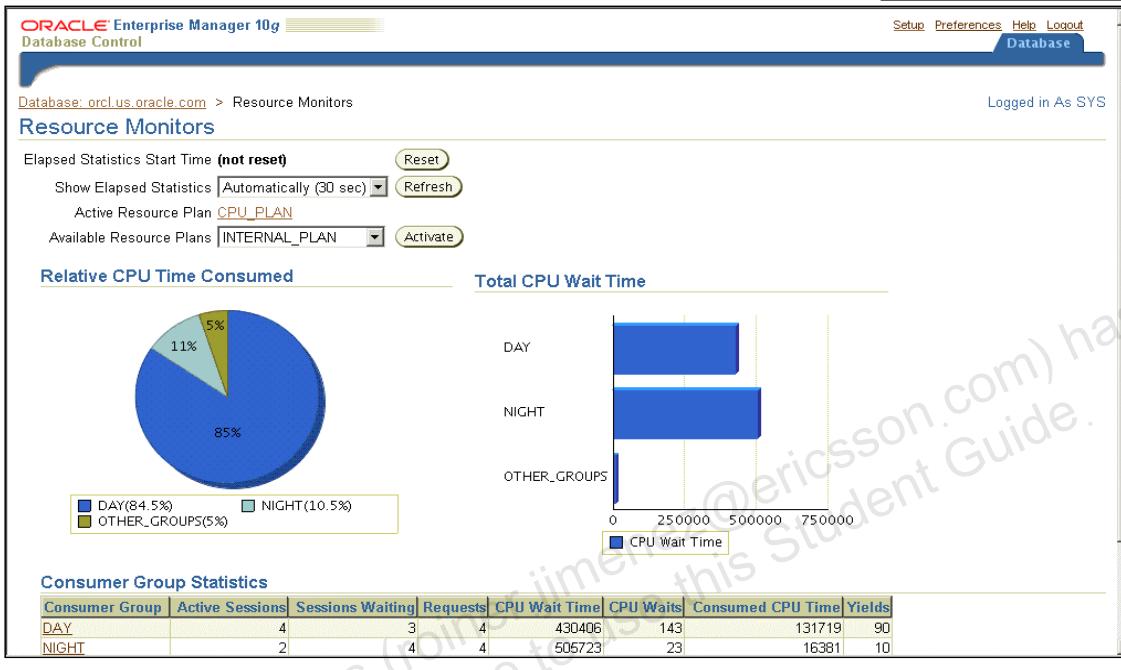
```
SQL> SELECT plan, num_plan_directives, status, mandatory
 2 FROM dba_rsrc_plans;
 PLAN NUM_PLAN_DIRECTIVES STATUS MAN
----- -----
SYSTEM_PLAN 3 ACTIVE NO
INTERNAL QUIESCE 2 ACTIVE YES
INTERNAL_PLAN 1 ACTIVE YES
BUGDB_PLAN 4 ACTIVE NO
MAILDB_PLAN 3 ACTIVE NO
MYDB_PLAN 3 ACTIVE NO
```

A status of ACTIVE indicates that the plan has been submitted and can be used, whereas a status of PENDING shows that the plan has been created, but is still in the pending area.

If the mandatory column is assigned a value of YES, then the plan cannot be deleted.

# Monitoring the Resource Manager

Concepts  
Resource Plan  
Consumer Group  
Plan Directives  
Mapping  
Plan activation  
> Monitoring



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Monitoring the Resource Manager

You can monitor the functioning of the Database Resource Manager at the session level. It is integrated with Automatic Database Diagnostic Monitor (ADDM).

There are different ways to manage and monitor the Resource Manager by using EM Database Control. On the Administration tabbed page, click the Monitors link in the Resource Manager section.

The Resource Monitors page displays a grouping of statistics and charts that depict the current state of the active resource plan. You can view the statistics for the currently active plan, and you can select a plan from the list and activate that plan.

The Consumer Group Statistics table lists a series of statistics for the consumer groups that are part of the current resource plan.

**Note:** When you activate a plan by using the Resource Monitors page, you must exit the page and then choose Resource Monitors again to update the page and view the statistics for the newly activated plan.

# Monitoring the Resource Manager

- **V\$SESSION:** Contains the **resource\_consumer\_group** column that shows the current group for a session
- **V\$RSRC\_PLAN:** A view that shows the active resource plan
- **V\$RSRC\_CONSUMER\_GROUP:** A view that contains statistics for all active groups

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Monitoring the Resource Manager (continued)

### CPU Utilization

There are at least three different views in the system that can provide you with information about the CPU utilization inside the Oracle database:

- V\$RSRC\_CONSUMER\_GROUP shows CPU utilization statistics on a per consumer group basis, if you are running the Oracle Database Resource Manager. This view displays data related to currently active resource consumer groups.
- V\$SYSSTAT shows the Oracle database CPU usage for all sessions. The statistic “CPU used by this session” shows the aggregate CPU used by all sessions.
- V\$SESSTAT shows the Oracle database CPU usage per session. You can use this view to determine which particular session is using the most CPU.

### The V\$RSRC\_CONSUMER\_GROUP View

The following is a quick description of some of the columns in this view:

- **name:** Name of the consumer group
- **active\_sessions:** Number of currently active sessions in this consumer group
- **execution\_waiters:** Number of active sessions waiting for a time slice
- **requests:** Cumulative number of requests executed in this consumer group
- **cpu\_wait\_time:** Cumulative amount of time that sessions waited for CPU
- **consumed\_cpu\_time:** Cumulative amount of CPU time consumed by all sessions

## Monitoring the Resource Manager (continued)

There is no view that shows the active session pool queue directly, but you can get some information from:

- **V\$SESSION**: The `current_queue_duration` column shows how long a session has been queued, or 0 (zero) if the session is not currently queued.
- **V\$RSRC\_CONSUMER\_GROUP**: The `queue_length` column shows the number of sessions currently queued per consumer group.

## Summary

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

- **Configure the Database Resource Manager**
- **Access and create resource plans**
- **Create consumer groups**
- **Specify directives for allocating resources to consumer groups**
- **Map consumer groups to plans**
- **Activate a resource plan**
- **Monitor the Resource Manager**



Copyright © 2008, Oracle. All rights reserved.

## Practice Overview: Using the Resource Manager

**This practice covers the following topics:**

- **Creating a resource consumer group**
- **Specifying CPU resource allocation directives for consumer groups**
- **Associating users with a resource consumer group**
- **Activating a resource plan**
- **Testing in SQL\*Plus**
- **Deactivating a resource plan**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2010, Oracle and/or its affiliates.

Roiner Jimenez Arias (roiner.jimenez@ericsson.com) has a  
non-transferable license to use this Student Guide.

# 14

## Automating Tasks with the Scheduler

ORACLE

Copyright © 2008, Oracle. All rights reserved.

# Objectives

Key Comp.  
& Steps  
Schedules  
Job Chains  
Adv.Concepts

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

- **Simplify management tasks by using the Scheduler**
- **Create a job, program, and schedule**
- **Monitor job execution**
- **Use a time-based or event-based schedule for executing Scheduler jobs**
- **Use job chains to perform a series of related tasks**
- **Use advanced Scheduler concepts to prioritize jobs**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Objectives

For information about the various Scheduler components and their interaction, see the *Oracle Database Administrator's Guide*.

For detailed information about the DBMS\_SCHEDULER package, see the *Oracle Database PL/SQL Packages and Types Reference*.

# Simplifying Management Tasks

Performing a series of month-end tasks on the last day of each month

Running a dequeue procedure as soon as a message is enqueued

Replicating table data via materialized view refreshes

Running a daily job to back up database

Computing table and index statistics twice a day

Starting the batch load as soon as the file arrives on the file system

Generating an hourly report on invalid server access attempts

Rebuilding an index when finished rebuilding the current index

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Simplifying Management Tasks

Many tasks in the Oracle environment need job-scheduling capabilities. Routine database maintenance and application logic require jobs to be scheduled and run periodically. Business-to-business (B2B) applications require scheduling for their business events. DBAs need to schedule regular maintenance jobs in specified time windows.

Oracle Database 10g provides advanced scheduling capabilities through the database Scheduler, which is a collection of functions and procedures in the DBMS\_SCHEDULER package. The Scheduler can be invoked in any SQL environment, or through Enterprise Manager.

The Scheduler enables database administrators and application developers to control when and where various tasks take place in the database environment. These tasks can be time consuming and complicated; using the Scheduler, you can manage and plan these tasks.

Scheduler jobs can be started based on time or when a specified event occurs, and the Scheduler can raise events when a job's state changes (for example, from RUNNING to COMPLETE). You can also use a named series of programs that are linked together for a combined objective.

# A Simple Job

**WHEN**

**WHAT**

Copyright © 2008, Oracle. All rights reserved.

## A Simple Job

A job has two key components: action, “what” needs to be done and schedule, “when” action occurs. The “what” is expressed in the Command region of the screenshot shown in the slide and the `job_type` and `job_action` parameters.

```

BEGIN
 sys.dbms_scheduler.create_job(
 job_name => '"HR"."CREATE_LOG_TABLE_JOB"',
 job_type => 'PLSQL_BLOCK', job_action => 'begin
 execute immediate (''create table session_history(
 snap_time TIMESTAMP WITH LOCAL TIME ZONE,
 num_sessions NUMBER)'')'; end;',
 start_date => systimestamp at time zone 'America/New_York',
 job_class => 'DEFAULT_JOB_CLASS',
 comments => 'Create the SESSION_HISTORY table',
 auto_drop => FALSE, enabled => TRUE);
END;

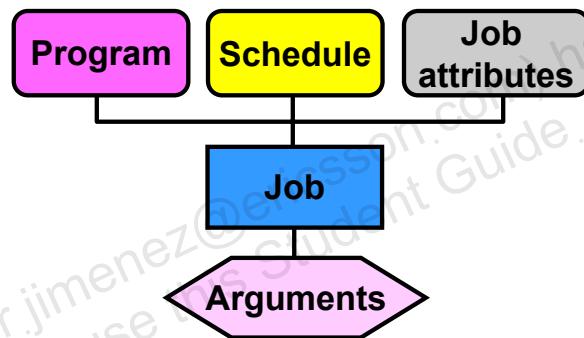
```

A job is also defined by “when” the desired action needs to take place. The “when” is expressed in a “schedule,” which can be based on time (see the `start_date` parameter) or events, or be dependent on the outcome of other jobs. These options are discussed in this lesson.

# Key Components and Steps

To simplify management tasks with the Scheduler, perform the following steps:

1. Create a program.
2. Create and use a schedule.
3. Create and submit a job.
4. Monitor a job.



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Key Components and Key Steps

The Scheduler offers a modularized approach for managing tasks within the Oracle database. By breaking down a task into its components, such as time, location, and database object, the Scheduler offers you an easier way to manage your database environment. The Scheduler uses three basic components:

- A **job** specifies what needs to be executed and when. For example, the “what” could be a PL/SQL procedure, a native binary executable, a Java application, or a shell script. You can specify the program (what) and schedule (when) as part of the job definition, or you can use an existing program or schedule instead. You can use arguments for a job to customize its run-time behavior.
- A **schedule** specifies when and how many times a job is executed. A schedule can be based on time or an event. You can define a schedule for a job by using a series of dates, an event, or a combination of the two, along with additional specifications to denote repeating intervals. You can store the schedule for a job separately and then use the same schedule for multiple jobs.
- A **program** is a collection of metadata about a particular executable, script, or procedure. An automated job executes some task. Using a program enables you to modify the job task, or the “what,” without modifying the job itself. You can define arguments for a program, enabling users to modify the run-time behavior of the task.

The screenshot shows the Oracle Enterprise Manager 10g Database Control interface. On the left, a sidebar titled "Database Scheduler" lists "Jobs", "Chains", "Schedules", "Programs" (which is highlighted with a red box), "Job Classes", "Windows", "Window Groups", and "Global Attributes". A red arrow points from the "Programs" link to the main content area. The main content area has a title "Scheduler Programs" and a sub-title "Following are the programs that define what are to be executed in the jobs.". It displays a table with columns "Select Name", "Schema", "Enabled", "Type", and "Description". Two rows are shown: "APPLY\_PROGRAM" (Schema: DMSYS, Enabled: Yes, Type: STORED PROCEDURE, Description: Used for applying a mining model using JDM API) and "AUTO\_SPACE\_ADVISOR\_PROG" (Schema: SYS, Enabled: Yes, Type: STORED PROCEDURE, Description: auto space advisor maintenance program). A red box highlights the "Create" button in the top right corner of the table header. Below this, another window titled "Create Program" is open, showing fields for "Name" (empty), "Schema" (HR), "Enabled" (Yes selected), "Description" (empty), "Type" (PLSQL\_BLOCK selected), and "Source" (containing a PL/SQL block). The source code is as follows:

```

BEGIN
DBMS_SCHEDULER.CREATE_PROGRAM(
 program_name => 'CALC_STATS2',
 program_action =>
'HR.UPDATE_HR_SCHEMA_STATS',
 program_type =>
'STORED_PROCEDURE',
 enabled => TRUE);
END;
/

```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## 1. Creating a Program

Use the CREATE\_PROGRAM procedure to create a program. Creating a program is an optional part of using the Scheduler. You can also encode the action to be performed within an anonymous PL/SQL block in the CREATE\_JOB procedure. By creating the program separately, you can define the action once, and then reuse this action within multiple jobs. This enables you to change the schedule for a job without having to re-create the PL/SQL block. You can also customize the job by specifying argument values.

To create a program in your own schema, you need the CREATE\_JOB privilege. A user with the CREATE ANY JOB privilege can create a program in any schema.

A program is created in a disabled state by default (unless the enabled parameter is set to TRUE). A disabled program cannot be executed by a job until it is enabled. You can specify that a program should be created in the enabled state by specifying a value of TRUE for enabled.

The program action is a string specifying a procedure, executable name, or a PL/SQL anonymous block, depending on what program\_type is set to.

If you have a procedure called UPDATE\_HR\_SCHEMA\_STATS that collects the statistics for the hr schema, then you can create a program to call this procedure.

In Enterprise Manager, select Administration > Programs and click the Create button.

## 2. Creating and Using Schedules

The screenshot shows the 'Create Schedule' dialog box. Key fields include:

- Name:** stats\_schedule
- Owner:** HR
- Time Zone:** America/Los\_Angeles
- Schedule Type:** Standard
- Repeating:** By Hours, Interval (Hours) 1
- Available to Start:** Immediately
- Not Available After:** No End Date

The generated PL/SQL code is:

```

BEGIN
 DBMS_SCHEDULER.CREATE_SCHEDULE(
 schedule_name => 'stats_schedule',
 start_date => SYSTIMESTAMP,
 end_date => SYSTIMESTAMP + 30,
 repeat_interval =>
 'FREQ=HOURLY;INTERVAL=1',
 comments => 'Every hour');
END;
/

```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### 2. Creating and Using Schedules

By using a schedule (instead of specifying the execution times for a job within the job definition), you can manage the scheduled execution of multiple jobs without having to update multiple job definitions. If a schedule is modified, then each job that uses that schedule automatically uses the new schedule.

Use the CREATE\_SCHEDULE procedure in the DBMS\_SCHEDULER PL/SQL package to create a schedule.

The `start_date` represents the date on which the schedule becomes active. The schedule cannot refer to any dates before this date. The schedule is not valid after the `end_date`.

You can schedule repeated executions by supplying a calendaring expression for `repeat_interval`. This calendaring expression is used to generate the next date of the schedule. Dates falling after the `end_date` time are not included in the schedule.

In the example shown in the slide, a schedule named `STATS_SCHEDULE` is created, specifying a repeat interval of every hour, starting now, and continuing for 30 days.

You can use Enterprise Manager to create schedules as shown in the slide.

### 3. Creating and Running a Job

**Create Job**

**General** [Schedule](#) [Options](#)

\* Name  

\* Owner  

Enabled  Yes  No

Description

Logging Level    
Specify logging requirements for the job

Job Class   [Create Job Class](#)

Auto Drop    
Specify whether the job should be dropped after completion

Restartable    
Specify whether the job can be restarted manually or in the event of failure

**Command**

Select the command type for the job, then enter the command requirements.

Command Type **Program** [Change Command Type](#)

Program Name **HR.LOG\_SESS\_COUNT\_PRGM**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### 3. Creating and Running a Job

A job is a combination of a schedule and a description of what to do, along with any additional arguments that are required by the job.

The program or “Command” can be a precreated PL/SQL or Java program, an anonymous PL/SQL block, or an executable that is run from the operating system’s command line.

The schedule for a job can be a predefined schedule (created with the DBMS\_SCHEDULER.CREATE\_SCHEDULE procedure) or defined as part of the job creation.

The schedule specifies attributes about when the job is run, such as:

- A start time, which defines when the job is picked for execution and an end time, which specifies the time after which the job is no longer valid and is not scheduled any more
- An expression specifying a repeating interval for the job
- A complex schedule created by combining existing schedules
- A condition or change in state, called an event, that must be met before the job is started

There are many attributes that you can set for a job. Attributes control how the job executes.

To run a job in Enterprise Manager, select Administration > Jobs.

## 4. Monitoring a Job

`SELECT job_name, status, error#, run_duration  
FROM USER_SCHEDULER_JOB_RUN_DETAILS;`

| JOB_NAME          | STATUS  | ERROR# | RUN_DURATION  |
|-------------------|---------|--------|---------------|
| GATHER_STATS_JOB  | SUCCESS | 0      | +000 00:08:20 |
| PART_EXCHANGE_JOB | FAILURE | 6576   | +000 00:00:00 |

Scheduler Jobs

Page Refreshed Sep 20, 2005 9:43:59 AM [Refresh](#) [Create](#)

All [Running](#) [History](#)

[Purge All Logs](#) [View Job Status](#) [Purge Log](#) [View Job Definition](#)

[Previous 25](#) 51-75 of 3647  [Next 25](#)

| Select                           | Status | Name               | Owner  | Completion Date                 | Run Duration (minutes) |
|----------------------------------|--------|--------------------|--------|---------------------------------|------------------------|
| <input checked="" type="radio"/> | ✓      | LOG SESSIONS JOB   | HR     | Sep 19, 2005 11:22:00 AM -07:00 | 0.0                    |
| <input type="radio"/>            | ✓      | RLM\$SCHDNEGACTION | EXFSYS | Sep 19, 2005 11:21:06 AM -07:00 | 0.0                    |
| <input type="radio"/>            | ✓      | LOG SESSIONS JOB   | HR     | Sep 19, 2005 11:19:00 AM -07:00 | 0.0                    |
| <input type="radio"/>            | ✓      | LOG SESSIONS JOB   | HR     | Sep 19, 2005 11:16:00 AM -07:00 | 0.0                    |
| <input type="radio"/>            | ✓      | LOG SESSIONS JOB   | HR     | Sep 19, 2005 11:13:00 AM -07:00 | 0.0                    |

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## 4. Monitoring a Job

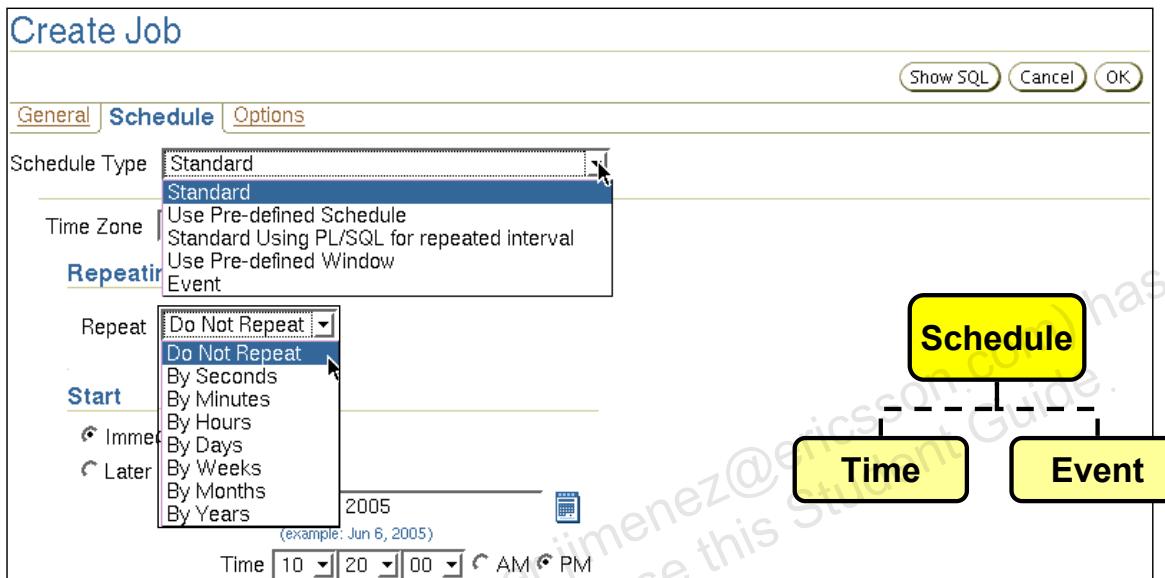
The DBA\_SCHEDULER\_JOB\_RUN\_DETAILS view has a row for each job instance. Each row contains information about the job execution for that instance.

[DBA|ALL]\_SCHEDULER\_JOB\_RUN\_DETAILS views have the following columns:

- **LOG\_ID:** The unique identifier of the log entry
- **LOG\_DATE:** The time stamp of the log entry
- **OWNER:** The job owner
- **JOB\_NAME:** The name of the job
- **STATUS:** The status of the job execution
- **ERROR#:** The number of the first error encountered
- **REQ\_START\_DATE:** The time at which the job was scheduled to start
- **ACTUAL\_START\_DATE:** The time at which the job was actually started
- **RUN\_DURATION:** The duration of execution of the job
- **INSTANCE\_ID:** The instance upon which the job ran
- **SESSION\_ID:** The session the job ran within
- **SLAVE\_PID:** The process ID of the slave process used to perform the job execution
- **CPU\_USED:** The amount of CPU used for the job run
- **ADDITIONAL\_INFO:** Additional information about the job run

# Using a Time-Based or Event-Based Schedule

Key Comp.  
& Steps  
Schedules  
Job Chains  
Adv.Concepts



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Using a Time-Based or Event-Based Schedule

To specify a time-based schedule for a job, you can specify either a calendaring expression or a datetime expression

When using a calendaring expression, the next start time for a job is calculated using the repeat interval and the start date of the job. When using datetime expressions, the specified expression determines the next time that the job should run. If no repeat interval is specified, the job runs only once on the specified start date.

If a job uses an event-based schedule, the job runs when the event is raised. At a high level, an event can be viewed as a change in state. An event occurs when a Boolean condition changes its state from FALSE to TRUE, or TRUE to FALSE.

The Scheduler uses Oracle Streams Advanced Queuing (AQ) to raise and consume events.

**Note:** The Scheduler does not guarantee that a job executes on the exact time because the system may be overloaded and thus resources may be unavailable.

# Creating a Time-Based Job



## Example:

**Create a job that calls a backup script every night at 11:00, starting tonight.**

```
BEGIN
 DBMS_SCHEDULER.CREATE_JOB(
 job_name=>'HR.DO_BACKUP',
 job_type => 'EXECUTABLE',
 job_action =>
 '/home/usr/dba/rman/nightly_incr.sh',
 start_date=> SYSDATE,
 repeat_interval=>'FREQ=DAILY;BYHOUR=23',
 /* next night at 11:00 PM */
 comments => 'Nightly incremental backups');
END;
/
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Creating a Time-Based Job

Use the CREATE\_JOB procedure of the DBMS\_SCHEDULER package to create a job. Jobs are created disabled by default and they become active and scheduled only when they are explicitly enabled. All job names are of the form: [schema.] name.

You should use SYSTIMESTAMP and specify a time zone so that when the time changes because of daylight saving time, your job adjusts its execution time automatically.

By default, a job is created in the current schema. You can create a job in another schema by specifying the name of the schema, as shown in the example in the slide. The job owner is the user in whose schema the job is created, whereas the job creator is the user who created the job. Jobs are executed with the privileges of the job owner. The national language support (NLS) environment of the job when it runs is the same as that which was present at the time the job was created.

The job\_type parameter indicates the type of task to be performed by the job. The possible values are:

- **PLSQL\_BLOCK:** An anonymous PL/SQL block
- **STORED\_PROCEDURE:** A named PL/SQL, Java, or external procedure
- **EXECUTABLE:** A command that can be executed from the operating system command line

## Creating a Time-Based Job (continued)

The `job_action` parameter can be the name of the procedure to run, the name of a script or operating system command, or an anonymous PL/SQL code block, depending on the value of the `job_type` parameter.

In the example in the slide, `job_type` is specified as `EXECUTABLE` and `job_action` is the full OS-dependent path of the desired external executable plus optionally any command-line arguments.

An external job is a job that runs outside the database. All external jobs run as a low-privileged guest user, as has been determined by the database administrator while configuring external job support. Because the executable is run as a low-privileged guest account, you should verify that it has access to necessary files and resources. Most, but not all, platforms support external jobs. For platforms that do not support external jobs, creating or setting the attribute of a job or a program to type `EXECUTABLE` returns an error.

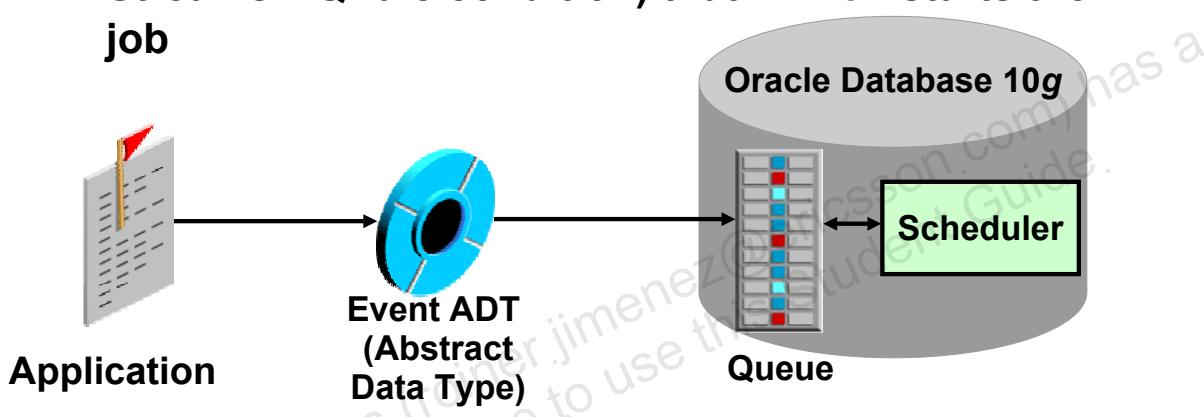
Refer to your Oracle database platform-specific documentation for more information about configuring the environment to run external programs with the Scheduler. For example, you may need to reference one or more of the following books:

- *Oracle Database Platform Guide 10g for Windows*
- *Oracle Database Installation Guide for UNIX Systems*
- *Oracle Database Release Notes 10g for AIX-Based Systems*
- *Oracle Database Release Notes 10g for hp HP-UX PA-RISC (64-bit*

# Creating an Event-Based Schedule

To create an event-based job, you must set:

- A queue specification (where your application enqueues messages to start a job)
- An event condition (same syntax as an Oracle Streams AQ rule condition) that if TRUE starts the job



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Creating an Event-Based Schedule

Jobs can be triggered based on events. An application can notify the Scheduler to start a job by enqueueing a message onto an Oracle Streams AQ queue. A job started in this way is referred to as an event-based job. To create an event-based job, you must set the following two additional attributes with the CREATE\_JOB procedure:

- **queue\_spec**: A queue specification that includes the name of the queue where your application enqueues messages to raise job start events, or in the case of a secure queue, the <queue\_name>, <agent\_name> pair
- **event\_condition**: A conditional expression based on message properties that must evaluate to TRUE for the message to start the job. The expression must use the same syntax as an Oracle Streams AQ rule condition. You can include user data properties in the expression, provided that the message payload is a user-defined object type, and that you prefix object attributes in the expression with tab.user\_data.

You can either specify queue\_spec and event\_condition as in-line job attributes, or create an event-based schedule with these attributes and then create a job that references this schedule.

# Creating Event-Based Schedules with Enterprise Manager

**Schedule**

Time Zone

Schedule Type

**Event Parameters**

\* Queue Name

\* Agent Name

\* Condition

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Creating Event-Based Schedules with Enterprise Manager

The Create Schedule page enables you to choose between a standard, time-based schedule and an event-based schedule. If you choose an event-based schedule, then the interface changes and you can specify the queue name, agent name, and event condition, in addition to the other schedule attributes.

### Note

The Scheduler runs the event-based job for each occurrence of an event that matches `event_condition`. However, events that occur while the job is already running are ignored; the event gets consumed, but does not trigger another run of the job.

### References:

- See the *Oracle Streams Advanced Queuing User's Guide and Reference* for information about how to create queues and enqueue messages.
- For more information about Oracle Streams AQ rules and event conditions, see the `DBMS_AQADM.ADD_SUBSCRIBER` procedure in the *Oracle Database PL/SQL Packages and Types Reference 10g Release 2* manual.

# Creating an Event-Based Job

**Example: Create a job that runs if a batch load data file arrives on the file system before 9:00 a.m.**

```

BEGIN
 DBMS_SCHEDULER.CREATE_JOB (
 job_name=>'ADMIN.PERFORM_DATA_LOAD',
 job_type => 'EXECUTABLE',
 job_action => '/home/usr/dba/rman/report_failure.sh',
 start_date => SYSTIMESTAMP,
 event_condition => 'tab.user_data.object_owner =
 ''HR'' and tab.user_data.object_name = ''DATA.TXT''
 and tab.user_data.event_type = ''FILE_ARRIVAL''
 and tab.user_data.event_timestamp < 9',
 queue_spec => 'HR.LOAD_JOB_EVENT_Q');
END;

```

Copyright © 2008, Oracle. All rights reserved.

## Creating an Event-Based Job

To specify event information as job attributes, you use an alternate syntax of CREATE\_JOB that includes the queue\_spec and event\_condition attributes. The job can include event information in-line as job attributes or can specify event information by pointing to an event schedule. The example shown in the slide uses an in-line event-based schedule.

The example in the slide shows a job that is started when a file arrives on the operating system, as long as the file arrives before 9:00 a.m. Assume that the message payload is an object with four attributes named object\_owner, object\_name, event\_type, and event\_timestamp.

The example uses a user-defined event. Therefore, before this job can be started, when the file arrives on the file system, a program or procedure must enqueue the event object type with the proper information into the specified event queue. The HR.LOAD\_JOB\_EVENT\_Q queue must be of the same type as the event object type used for notifying the Scheduler of an event occurrence. That is, the HR.LOAD\_JOB\_EVENT\_Q queue must be a typed queue where the type has four attributes named object\_owner, object\_name, event\_type, and event\_timestamp.

For more information about how to create queues and enqueue messages, refer to the *Oracle Streams Advanced Queuing User's Guide and Reference* documentation.

# Event-Based Scheduling

## Event types:

- **User- or application-generated events**
- **Scheduler-generated events**

## Events raised by Scheduler jobs:

- |                 |                       |
|-----------------|-----------------------|
| • JOB_STARTED   | • JOB_SCH_LIM_REACHED |
| • JOB_SUCCEEDED | • JOB_DISABLED        |
| • JOB_FAILED    | • JOB_CHAIN_STALLED   |
| • JOB_BROKEN    | • JOB_ALL_EVENTS      |
| • JOB_COMPLETED | • JOB_RUN_COMPLETED   |
| • JOB_STOPPED   | • JOB_OVER_MAX_DUR    |

## Example of raising an event:

```
DBMS_SCHEDULER.SET_ATTRIBUTE('hr.do_backup',
 'raise_events', DBMS_SCHEDULER.JOB_FAILED);
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Event-Based Scheduling

You can create a job that directly references an event as the means to start the job, instead of assigning a schedule to the job. There are two types of events:

- **User- or application-generated events:** An application can raise an event to be consumed by the Scheduler. The Scheduler reacts to the event by starting a job. An example of such events: a running job completes; a file arrives on the file system; an account within the database is locked; and the inventory reaches a low threshold.
- **Scheduler-generated events:** The Scheduler can raise an event to indicate state changes that occur within the Scheduler itself. For example, the Scheduler can raise an event when a job starts, when a job completes, when a job exceeds its allotted run time, and so on. The consumer of the event is an application that performs some action in response to the event.

You can configure a job so that the Scheduler raises an event when the job's state changes. You do this by setting the `raise_events` job attribute. By default, a job does not raise any state change events until you alter the `raise_events` attribute for a job. To alter this attribute, you must first create the job by using the `CREATE_JOB` procedure and then use the `SET_ATTRIBUTE` procedure to modify the attribute's default value. The example shows that the `hr.do_backup` job is altered, so that it raises an event if the job fails.

## Event-Based Scheduling (continued)

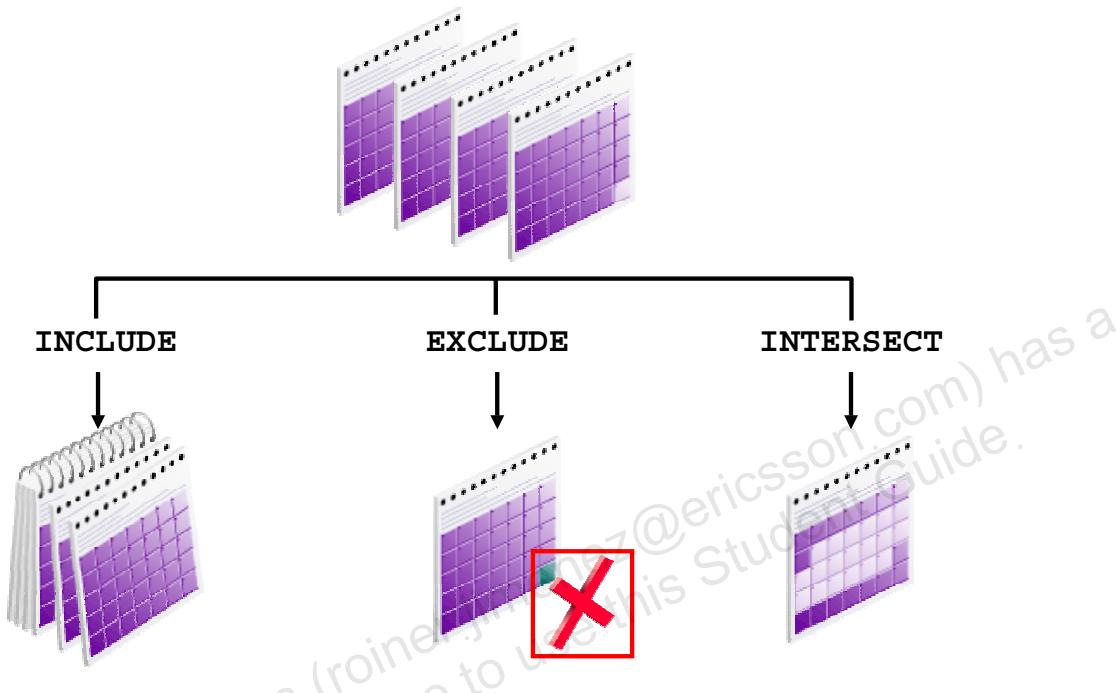
After you enable job state change events for a job, the Scheduler raises these events by enqueueing messages onto the default event queue `SYS.SCHEDULER$_EVENT_QUEUE`.

The default Scheduler event queue is a secure queue. Depending on your application, you may have to configure the queue to enable certain users to perform operations on it. See the *Oracle Streams Concepts and Administration* documentation for information about secure queues.

The default Scheduler event queue is intended primarily for Scheduler-generated events. Oracle does not recommend the use of this queue for user applications, or user-defined events.

| Event Type                       | Description                                                                                                                                                                                                                                                                     |
|----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>JOB_STARTED</code>         | The job is started.                                                                                                                                                                                                                                                             |
| <code>JOB_SUCCEEDED</code>       | The job is successfully completed.                                                                                                                                                                                                                                              |
| <code>JOB_FAILED</code>          | The job failed, either by raising an error or by abnormally terminating.                                                                                                                                                                                                        |
| <code>JOB_BROKEN</code>          | The job is disabled and changed to the <code>BROKEN</code> state, because it exceeded the number of failures defined by the <code>MAX_FAILURES</code> job attribute.                                                                                                            |
| <code>JOB_COMPLETED</code>       | The job is completed, because it reached the values set by the <code>MAX_RUNS</code> or <code>END_DATE</code> job attributes.                                                                                                                                                   |
| <code>JOB_STOPPED</code>         | The job is stopped by a call to the <code>STOP_JOB</code> procedure.                                                                                                                                                                                                            |
| <code>JOB_SCH_LIM_REACHED</code> | The job's schedule limit is reached. The job is not started, because the delay in starting the job exceeded the value of the <code>SCHEDULE_LIMIT</code> job attribute.                                                                                                         |
| <code>JOB_DISABLED</code>        | The job is disabled by the scheduler or by a call to the <code>SET_ATTRIBUTE</code> procedure.                                                                                                                                                                                  |
| <code>JOB_CHAIN_STALLED</code>   | A job running a chain is put into the <code>CHAIN_STALLED</code> state. A running chain becomes stalled if there are no steps running or scheduled to run and the chain <code>EVALUATION_INTERVAL</code> is set to <code>NULL</code> . The chain waits for manual intervention. |
| <code>JOB_ALL_EVENTS</code>      | <code>JOB_ALL_EVENTS</code> is not an event, but a constant, that provides an easy way for you to enable all events.                                                                                                                                                            |
| <code>JOB_RUN_COMPLETED</code>   | A job run is completed. It either failed, succeeded, or is stopped.                                                                                                                                                                                                             |
| <code>JOB_OVER_MAX_DUR</code>    | The job has run over the maximum time it was set to be allowed to run.                                                                                                                                                                                                          |

# Creating Complex Schedules



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Creating Complex Schedules

A schedule is an object in the database. When you create schedules, they are automatically saved. You can use combinations of schedules to create more complex schedules. By combining schedules, you can add specific dates to or exclude specific dates from a calendaring expression.

You can use the following options when defining the repeat interval for a schedule:

- **INCLUDE:** Adds a list of dates to the calendaring expression results
- **EXCLUDE:** Removes a list of dates from the calendaring expression results
- **INTERSECT:** Uses only the dates that are common to two or more schedules

When creating schedules to be used in combinations, you can code the list of dates by including hard-coded dates of the form [ YYYY ] MMDD or by including named schedules created with the CREATE\_SCHEDULE procedure. For example, you can specify a list of dates by using the following values for the repeat interval of a schedule:

0115,0315,0325,0615,quarter\_end\_dates,1215

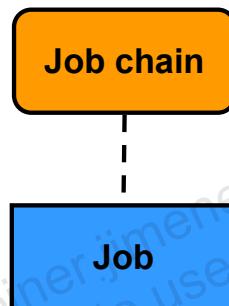
This string represents the dates January 15, March 15, March 25, June 15, December 15, and the list of dates specified by the QUARTER\_END\_DATES schedule.

If you do not specify the optional year component for hard-coded dates in your schedule, the dates are included for every year.

# Creating Job Chains

Key Comp.  
& Steps  
Schedules  
> **Job Chains**  
Adv.Concepts

1. **Create a chain object.**
2. **Define chain steps.**
3. **Define chain rules.**
4. **Starting the chain:**
  - Enable the chain.
  - Create a job that points to the chain.



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Creating Job Chains

A chain is a named series of programs that are linked together for a combined objective. This is known as “dependency scheduling.” An example of a chain may be the following:

*Run program A and then program B, but only run program C if programs A and B complete successfully, otherwise run program D.*

Each position within a chain of interdependent programs is referred to as a step.

To create and use a chain, you complete the following steps in order. All procedures mentioned are part of the DBMS\_SCHEDULER package, unless noted otherwise.

1. **Create a chain** by using the CREATE\_CHAIN procedure. The chain name can be optionally qualified with a schema name (for example, *myschema.myname*).
2. **Define (one or more) chain steps.** Defining a step gives it a name and specifies what happens during the step. Each step can point to one of the following:
  - A program
  - Another chain (a nested chain)
  - An event

You define a step that points to a program or nested chain by calling the DEFINE\_CHAIN\_STEP procedure.

## Creating Job Chains (continued)

(Defining Steps, continued)

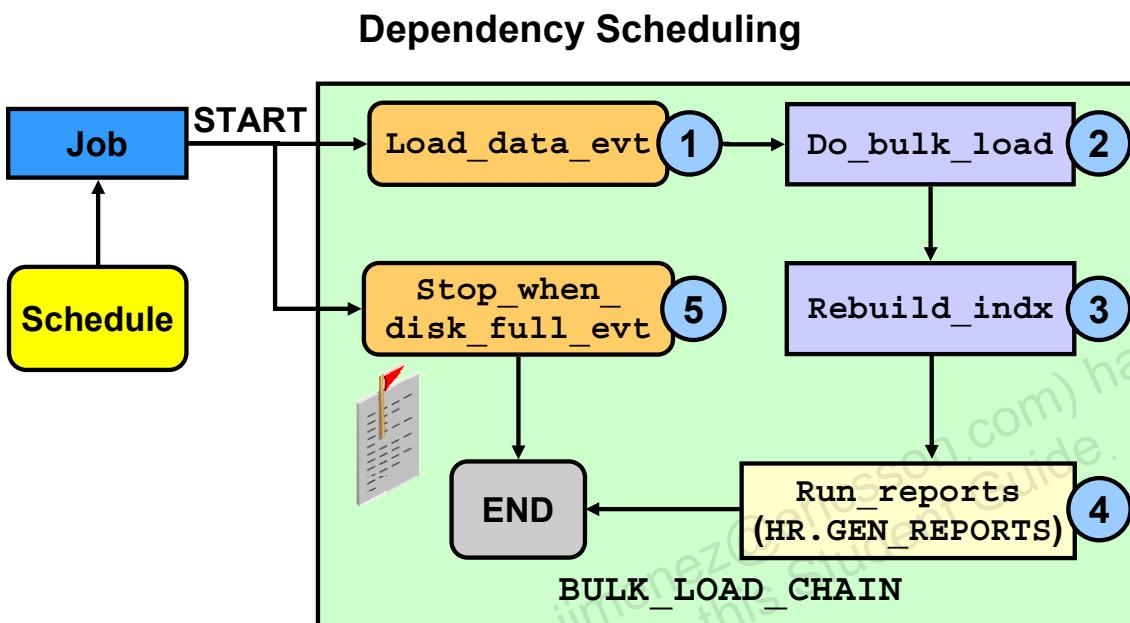
To define a step that waits for an event to occur, you use the `DEFINE_CHAIN_EVENT_STEP` procedure. Procedure arguments can point to an event schedule or can include an in-line queue specification and event condition. A step that points to an event waits until the specified event is raised. If the event occurs, the step completes successfully.

3. After creating the chain object, you **define chain rules**. Chain rules define when steps run, and define dependencies between steps. Each rule has a *condition* and an *action*:
  - If the condition evaluates to TRUE, the action is performed. The condition can contain any syntax that is valid in a SQL WHERE clause. Conditions are usually based on the outcome of one or more previous steps. For example, you may want one step to run if the two previous steps succeeded, and another to run if either of the two previous steps failed.
  - The action specifies what is to be done as a result of the rule being triggered. A typical action is to run a specified step. Possible actions include starting or stopping a step. You can also choose to end the execution of the job chain, returning either a value or a step name and error code.

All rules added to a chain work together to define the overall behavior of the chain. When the job starts and at the end of each step, all rules are evaluated to see what action or actions occur next. You add a rule to a chain with the `DEFINE_CHAIN_RULE` procedure. You call this procedure once for each rule that you want to add to the chain.

4. **Starting the chain** involves two actions:
  - Enable a chain with the `ENABLE` procedure. (A chain is always created disabled, so you can add steps and rules to the chain before it is executed by any job.) Enabling an already enabled chain does not return an error.
  - To run a chain, you must create a job of type '`CHAIN`'. The job action must refer to the chain name. You can use either event-based or time-based schedules for this job.

## Example of a Chain



ORACLE

Copyright © 2008, Oracle. All rights reserved.

### Example of a Chain

As an example of a chain, consider all the tasks and conditions that occur during a bulk data load. First, you must have data to load. Then load the data, observing the file system to make sure that you do not run out of space during the load. After the data load completes, you need to rebuild the indexes defined on the updated tables. Then you run reports against the newly loaded data.

This is an example of dependency scheduling.

# 1. Creating a Chain Object

**Create Chain**

\* Name

\* Owner

Enabled  Yes  No

Description

**Steps**

| Select                           | Step Name               | Type           | Object Name            | Actions |
|----------------------------------|-------------------------|----------------|------------------------|---------|
| <input checked="" type="radio"/> | load_data_evt           | EVENT_SCHEDULE | HR.FILE_ARRIVAL_EVENT  |         |
| <input type="radio"/>            | do_bulk_load            | PROGRAM        | HR.LOAD_DATA_PROG      |         |
| <input type="radio"/>            | rebuild_indx            | PROGRAM        | HR.REBUILD_INDEXES     |         |
| <input type="radio"/>            | run_reports             | SUBCHAIN       | HR.GEN_REPORTS         |         |
| <input type="radio"/>            | stop_when_disk_full_evt | EVENT_SCHEDULE | HR.DISK_FULL_EVT_SCHED |         |
| <b>Add 5 Steps</b>               |                         |                |                        |         |

**Rules**

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## 1. Creating a Chain Object

On the Administration page, select Chains from the Scheduler region. On the Scheduler Chains page, you can create or edit a job chain. On the Create Chain page, you must enter the name and owner of the job chain. You can then choose whether the newly created job chain should be enabled or not. You can enable the chain at a later time.

PL/SQL Example:

```
DBMS_SCHEDULER.CREATE_CHAIN (
 chain_name => 'bulk_load_chain',
 rule_set_name => NULL, evaluation_interval => NULL,
 comments => 'Load data and run reports');
```

After naming the job chain and optionally providing a description, you then enter the job chain steps, one at a time. You can create job chain steps of the following type:

- A step that runs a program (PROGRAM)
- A step that is another job chain (SUBCHAIN)
- A step that uses a stored event-based schedule (EVENT\_SCHEDULE)

Only one program can run during a step. Every step in a chain must be defined before the chain can be enabled and used.

## 2. Defining Chain Steps

```

DBMS_SCHEDULER.DEFINE_CHAIN_EVENT_STEP (
 chain_name => 'bulk_load_chain',
 step_name => 'load_data_evt',
 event_condition => 'tab.user_data.object_owner =
 ''HR'' and tab.user_data.object_name =
 ''DATA.TXT'' and tab.user_data.event_type =
 ''FILE_ARRIVAL'''',
 queue_spec => 'HR.LOAD_JOB_EVENT_Q');

DBMS_SCHEDULER.DEFINE_CHAIN_STEP (
 chain_name => 'bulk_load_chain',
 step_name => 'do_bulk_load',
 program_name => 'hr.load_data_prog');

DBMS_SCHEDULER.DEFINE_CHAIN_STEP (
 chain_name => 'bulk_load_chain',
 step_name => 'rebuild_indx',
 program_name => 'hr.rebuild_indexes');

```

1

2

3

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### 2. Defining Chain Steps

1. The `load_data_evt` step uses an event-based schedule. The `load_data_evt` step waits for a `FILE_ARRIVAL` event to be placed in `HR.LOAD_JOB_EVENT_Q`. This event notifies the Scheduler that the `DATA.TXT` data file has arrived on the file system. Because the `queue_spec` argument does not include an agent name, the specified queue is not a secure queue.
2. The `do_bulk_load` step performs a bulk data load into the schema tables.
3. The `rebuild_indx` step rebuilds the indexes for the tables following a data load.
4. The `run_reports` step runs a report against the new data. `HR.GEN_REPORTS` is another job chain.

```

DBMS_SCHEDULER.DEFINE_CHAIN_STEP (
 chain_name => 'bulk_load_chain',
 step_name => 'run_reports',
 program_name => 'hr.gen_reports');

```

5. The `stop_when_disk_full_evt` step ends the chain execution if the system runs out of disk space.

```

DBMS_SCHEDULER.DEFINE_CHAIN_EVENT_STEP (
 chain_name => 'bulk_load_chain',
 step_name => 'stop_when_disk_full_evt'
 event_schedule_name => 'disk_full_sched')

```

### 3. Defining Chain Rules

**Rules**

| Select                           | Name              | Condition                             | Action                                       | Description                                         |
|----------------------------------|-------------------|---------------------------------------|----------------------------------------------|-----------------------------------------------------|
| <input checked="" type="radio"/> | System_Assigned_1 | 1=1                                   | START load_data_evt, stop_when_disk_full_evt | Start the chain, by waiting for the file to arrive  |
| <input type="radio"/>            | System_Assigned_2 | :load_data_evt.COMPLETED="TRUE"       | START do_bulk_load                           |                                                     |
| <input type="radio"/>            | System_Assigned_3 | :do_bulk_load.STATE="SUCCEEDED"       | START rebuild_indx                           |                                                     |
| <input type="radio"/>            | System_Assigned_4 | :rebuild_indx.COMPLETED="TRUE"        | START recalc_stats                           | recalc stats after indexes rebuilt                  |
| <input type="radio"/>            | System_Assigned_5 | :recalc_stats.STATE="SUCCEEDED"       | START run_reports                            | Start generating reports ONLY if stats recalculated |
| <input type="radio"/>            | System_Assigned_6 | :recalc_stats.STATE != "SUCCEEDED"    | END 10                                       | Exit chain (10) if stats recalc fails               |
| <input type="radio"/>            | System_Assigned_7 | :stop_when_disk_full.COMPLETED="TRUE" | END 99                                       | Exit chain (99) if disk full                        |
| <input type="radio"/>            | System_Assigned_8 | :run_reports.STATE="SUCCEEDED"        | END                                          | Exit normally when successful                       |

Evaluation Interval (minutes)

**Add**   **Edit**   **Delete**

Copyright © 2008, Oracle. All rights reserved.

### 3. Defining Chain Rules

After you have specified the job chain steps, you can create rules for the job chain. Chain rules define when steps run, and define dependencies between steps. Each rule has a condition and an action. If the condition evaluates to TRUE, the action is performed. The condition can also contain the Scheduler's chain syntax.

To create a rule in EM, you click the Add button in the Rules region.

When entering the rule conditions, character strings should be entered in single quotation marks. The GUI interface automatically escapes the quotation marks as needed when executing the commands.

#### PL/SQL Example

```
DBMS_SCHEDULER.DEFINE_CHAIN_RULE (
 chain_name => 'bulk_load_chain',
 condition => 'TRUE',
 action => 'START load_data_evt,stop_when_disk_full_evt',
 rule_name => 'dataload_rule1',
 comments => 'start the chain');
```

**Note:** This PL/SQL example differs from the EM one in that it uses a user-defined rule name, whereas the EM example shows the default generated name.

## 4. Starting the Chain

```
BEGIN
 DBMS_SCHEDULER.ENABLE ('bulk_load_chain');
END;
/
```

```
BEGIN
 DBMS_SCHEDULER.CREATE_JOB (
 job_name => 'bulk_load_chain_job',
 job_type => 'CHAIN',
 job_action => 'bulk_load_chain',
 repeat_interval => 'freq=daily;byhour=7;
 byminute=5;bysecond=0',
 enabled => TRUE);
END;
/
```

ORACLE

Copyright © 2008, Oracle. All rights reserved.

### 4. Starting the Chain

After you have finished creating and modifying the job chain, you enable it by calling the ENABLE procedure.

To run a chain, you must create a job. Set the `job_type` to 'CHAIN' and the job action to the name of the job chain you want to execute. The rest of the arguments are configured as you would configure them for any other type of job.

# Monitoring Job Chains

**Scheduler Chains**

Page Refreshed Mar 21, 2005 7:39:20 AM [Refresh](#) [Create](#)

| Select                           | Name           | Owner | No of Rules | No of Steps | Enabled | Description                         |
|----------------------------------|----------------|-------|-------------|-------------|---------|-------------------------------------|
| <input checked="" type="radio"/> | GEN REPORTS    | HR    | 8           | 7           | FALSE   | Generate reports based on HR data   |
| <input type="radio"/>            | BULK LOAD DATA | HR    | 6           | 5           | FALSE   | Load data from file and run reports |

**Related Links**

|                                   |                             |                               |
|-----------------------------------|-----------------------------|-------------------------------|
| <a href="#">Global Attributes</a> | <a href="#">Job Classes</a> | <a href="#">Jobs</a>          |
| <a href="#">Programs</a>          | <a href="#">Schedules</a>   | <a href="#">Window Groups</a> |
| <a href="#">Windows</a>           |                             |                               |

|                                              |
|----------------------------------------------|
| [DBA   ALL   USER] _SCHEDULER_CHAINS         |
| [DBA   ALL   USER] _SCHEDULER_CHAIN_RULES    |
| [DBA   ALL   USER] _SCHEDULER_CHAIN_STEPS    |
| [DBA   ALL   USER] _SCHEDULER_RUNNING_CHAINS |

ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Monitoring Job Chains

The ALL\_SCHEDULER\_CHAINS view contains information about the chain owner and name; the rule set name and rule set owner for the chain; the number of rules; the number of steps; whether or not the chain is enabled; whether or not the chain uses an evaluation interval; and whether or not the chain uses a user-defined rule set.

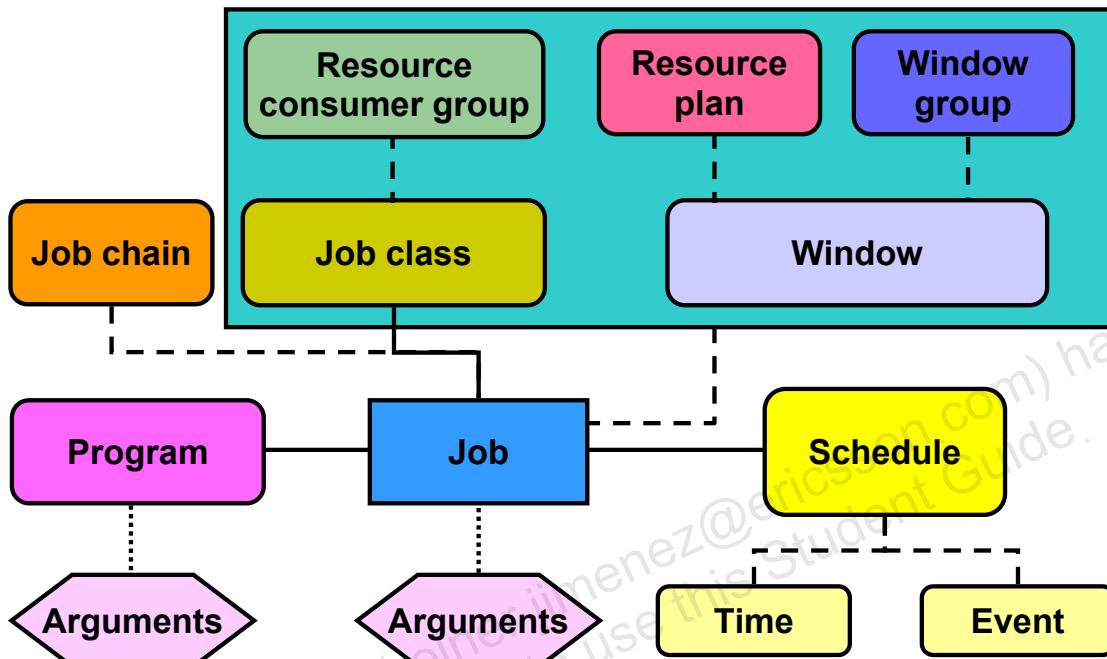
The ALL\_SCHEDULER\_CHAIN\_RULES view displays information such as the name and owner of the chain for which the rule was defined; the rule name, owner, and condition; and the action to be performed if the condition evaluates to TRUE.

The ALL\_SCHEDULER\_CHAIN\_STEPS displays information such as the name and owner of the chain for which the step was created; the step name; the program name and owner; whether the step should be skipped or not; and whether or not the step should be paused after it completes.

The ALL\_SCHEDULER\_RUNNING\_CHAINS view contains information such as the chain name and owner; the name and owner of the job that points to the chain; the name of the steps in the chain and their current state; errors encountered by the chain step; the time at which the chain step started and ended; how long it took the step to complete; and the name of the job running the step, if it is current executing.

# Advanced Scheduler Concepts

Key Comp.  
& Steps  
Schedules  
Job Chains  
> Adv.Concepts



ORACLE

Copyright © 2008, Oracle. All rights reserved.

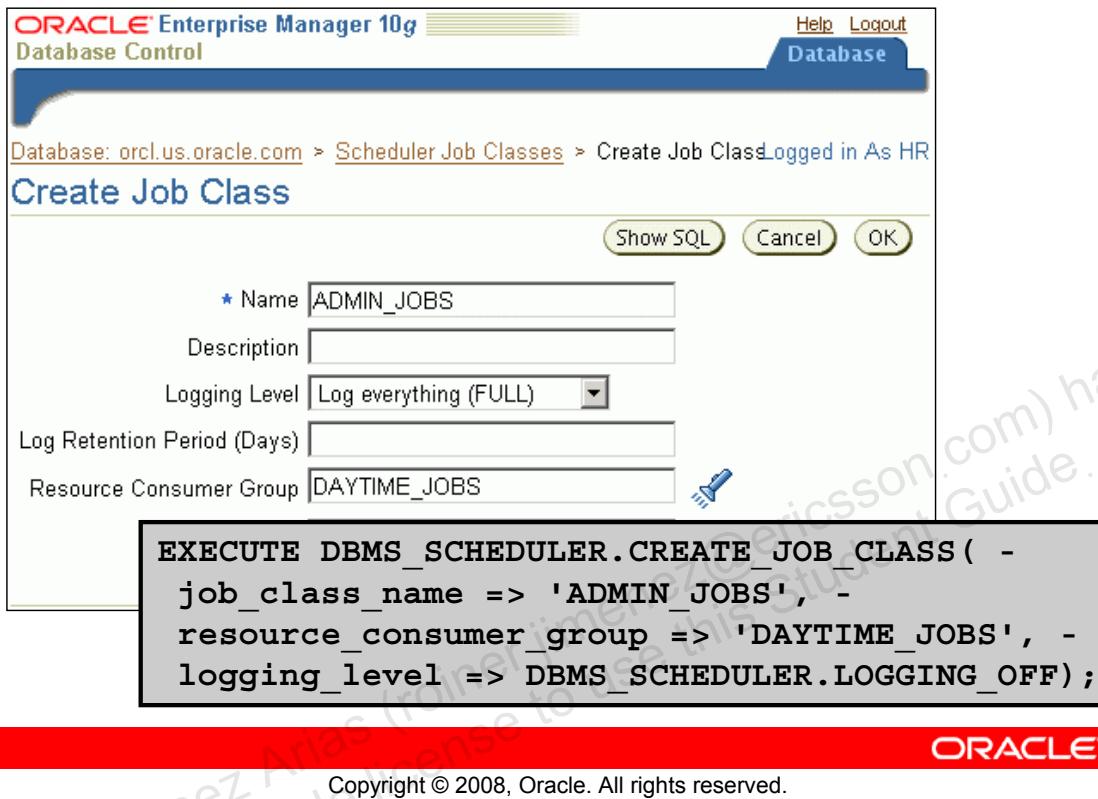
## Advanced Scheduler Concepts

Using the advanced Scheduler concepts, you can have more advanced control of aspects of scheduling, such as prioritizing jobs. The components are summarized below, and are discussed in detail in the following slides.

- A **window** is represented by an interval of time with a well-defined beginning and end, and is used to activate different resource plans at different times. This allows you to change resource allocation during a time period such as time of day or time of the sales year.
- A **window group** represents a list of windows, and allows for easier management of windows. You can use a window or window group as the schedule for a job to ensure that the job runs only when a window and its associated resource plan are active.
- A **job class** defines a category of jobs that share common resource usage requirements and other characteristics. A job class groups jobs into larger entities.
- A **resource consumer group** associated with the job class determines the resources that are allocated to the jobs in the job class.
- A **resource plan** enables users to prioritize resources (most notably CPU) among resource consumer groups.

**Note:** For more information about resource consumer groups and resource plans, see the lesson titled “Managing Resources.”

# Creating a Job Class



## Creating a Job Class

You can use the CREATE\_JOB\_CLASS procedure of the DBMS\_SCHEDULER package to create a job class. A class always belongs to the sys schema. To create a class, you must have the MANAGE SCHEDULER privilege.

After a job class has been created, you can specify jobs as members of this job class when you create the jobs, or after the jobs are created, by using the SET\_ATTRIBUTE procedure of the DBMS\_SCHEDULER package.

If you are using Enterprise Manager Database Control, use the Job Class page to create or edit a job class that you will assign to a job. Enter the name of the job class and the resource consumer group with which this job class is associated. You can select the resource consumer group by using the search function.

There is a default job class named DEFAULT\_JOB\_CLASS that is created with the database. If a job is not associated with a job class, then the job belongs to this default job class.

If a resource consumer group is not specified when a job class is created, the job class maps to the DEFAULT\_CONSUMER\_GROUP resource consumer group. Jobs in the default job class or in a job class associated with the default resource consumer group might not be allocated enough resources to complete their tasks when the Resource Manager is enabled.

# Creating a Window

**Create a window for the month of December that uses the END\_OF\_YEAR resource plan and is active every night from 6:00 p.m. to 6:00 a.m. Eastern Standard Time (EST).**

```
BEGIN
 DBMS_SCHEDULER.CREATE_WINDOW(
 window_name => 'DEC_NIGHTS',
 resource_plan => 'END_OF_YEAR',
 start_date => '01-DEC-03 06.00.00 PM EST',
 repeat_interval => 'FREQ=DAILY; BYHOUR=18',
 duration => '0 12:00:00',
 end_date => '31-DEC-03 06.00.00 AM EST',
 comments => 'Every day at 6:00 PM');
END;
/
```



Copyright © 2008, Oracle. All rights reserved.

## Creating a Window

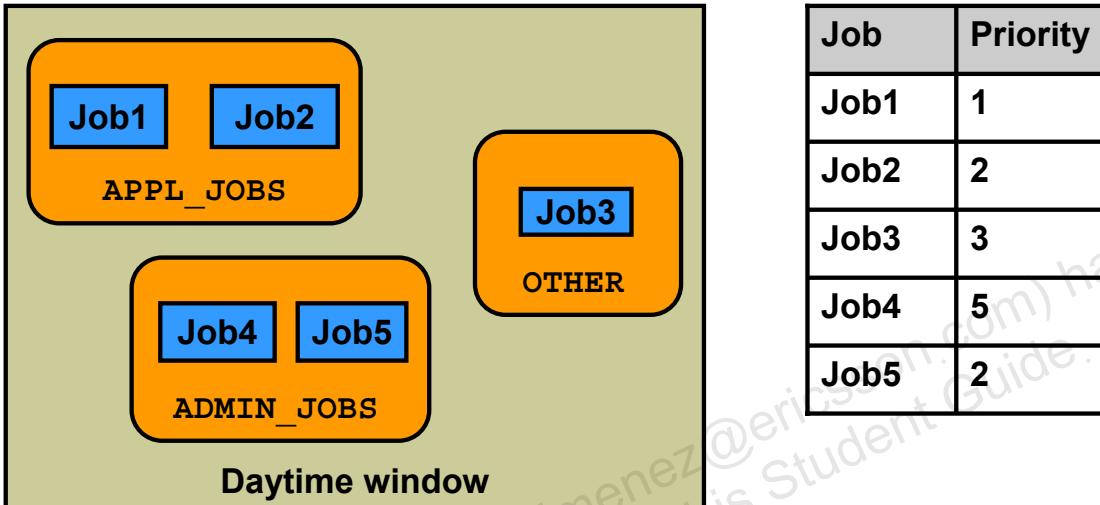
The priority of jobs changes over a period of time. For example, you might want to allocate a high percentage of the database resources to data warehouse loading jobs at night and allocate a higher percentage of the resources during the day to the application jobs. To accomplish this, you can change the database resource plan by using a Scheduler window.

The purpose of a window is to specify which resource plan is active for a specific time period. The window is represented by an interval of time, such as “every day, from 8:00 a.m. to 6:00 p.m.” The recurring time window is defined as a schedule specifying a pattern of start dates and duration (in minutes).

A window is *open* if it is in effect. Only one window can be in effect at any given time. In the example shown in the slide:

- The window opens (becomes active) at 6:00 p.m. on December 1, 2003.
- The duration, specified as an INTERVAL DAY TO SECOND data type, indicates that the window closes at 6:00 a.m. on December 2, 2003.
- The next time the window opens is calculated using the value for REPEAT\_INTERVAL, which evaluates to 6:00 p.m. on December 2, 2003.
- At 6:00 a.m. on December 31, 2003, the window closes and is disabled. While the DEC\_NIGHTS window is open, the resources allocated to the jobs are determined by the guidelines specified in the END\_OF\_YEAR resource plan.

# Prioritizing Jobs Within a Window



ORACLE

Copyright © 2008, Oracle. All rights reserved.

## Prioritizing Jobs Within a Window

When creating multiple jobs in a database, you need a way to align the job processing with your business requirements and specify which jobs have the highest priority. For a particular window, you may have several classes of jobs running, each with its own priority.

Job classes are used to categorize jobs. The job class maps to a resource consumer group. The active resource plan determines the resources allocated to each resource consumer group, and thus to each job class.

There are two levels at which jobs can be prioritized: at the class level and at the job level.

- The first prioritization is at the class level, using resource plans. Prioritization among jobs of different classes is done purely on a class resource allocation basis.
- The second prioritization is within the class, using the job priority attribute of the job.

Prioritization levels are relevant only when two jobs within the same class are supposed to start at the same time. The job with the higher priority starts first.

Prioritization is not guaranteed for jobs in different job classes. For example, a high-priority job in the APPL\_JOBS job class might not get started before a low-priority job in the ADMIN\_JOBS job class, even if they share the same schedule. If the APPL\_JOBS job class has a lower level of resource available, then the high-priority job in that class has to wait for resources to become available, even if there are resources available to lower-priority jobs in a different job class.

# **Summary**

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

- **Simplify management tasks by using the Scheduler**
- **Create a job, program, and schedule**
- **Monitor job execution**
- **Use a time-based or event-based schedule for executing Scheduler jobs**
- **Use job chains to perform a series of related tasks**
- **Use advanced Scheduler concepts to prioritize jobs**



Copyright © 2008, Oracle. All rights reserved.

# Practice Overview: Automating Tasks with the Scheduler

**This practice covers the following topics:**

- **Creating a job that runs a program outside the database**
- **Creating a program and a schedule**
- **Creating a job that uses a program and a schedule**
- **Altering the program and schedule for the job and observing the behavior change of the job**
- **Monitoring job runs**

**ORACLE**

Copyright © 2008, Oracle. All rights reserved.

## Practice Overview

**Note:** This practice uses both the Enterprise Manager Database Control and SQL\*Plus.