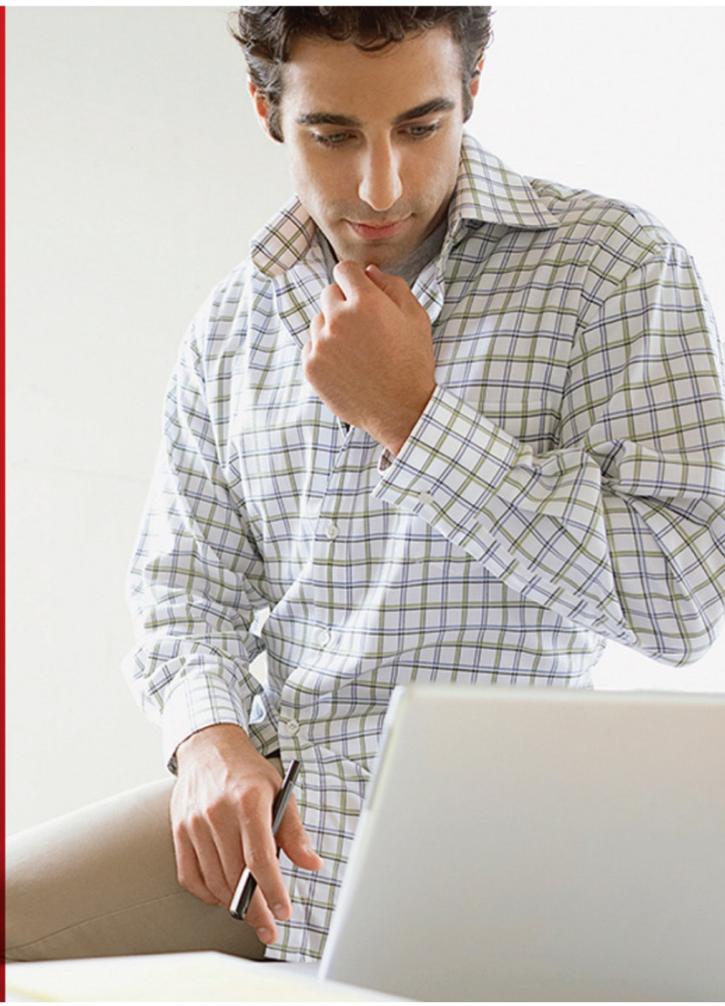




Hardware and Software
Engineered to Work Together



Oracle Database 12c: Administration Workshop

Student Guide – Volume II
D78846GC20
Edition 2.0 | December 2014 | D89299

Learn more from Oracle University at oracle.com/education/

Authors

Donna K. Keesling

James L. Spiller

Technical Contributors and Reviewers

Daryl Balaski

Rainer Bien

Maria Billings

Andy Fortunak

Joel Goodman

Daniela Hansell

Pat Huey

Dominique Jeunot

Gwen Lazenby

Ira Singer

Lori Tritz

Branislav Valny

Harald Van Breederode

Editors

Vijayalakshmi Narasimhan

Malavika Jinka

Publishers

Veena Narasimhan

Jayanthy Keshavamurthy

Copyright © 2014, 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.

Contents

1 Introduction

Objectives 1-2
Course Objectives 1-3
Suggested Schedule 1-4
Oracle Database Innovation 1-5
Enterprise Cloud Computing 1-6
Course Examples: HR Sample Schema 1-7
Summary 1-8

2 Exploring Oracle Database Architecture

Objectives 2-2
Oracle Database Server Architecture: Overview 2-3
Oracle Database Instance Configurations 2-4
Connecting to the Database Instance 2-5
Oracle Database Memory Structures 2-6
Shared Pool 2-8
Database Buffer Cache 2-10
Redo Log Buffer 2-11
Large Pool 2-12
Java Pool 2-13
Streams Pool 2-14
Program Global Area (PGA) 2-15
In-Memory Column Store: Introduction 2-16
In-Memory Column Store: Overview 2-18
Full Database In-Memory Caching 2-20
Quiz 2-22
Process Architecture 2-24
Process Structures 2-26
Database Writer Process (DBWn) 2-28
Log Writer Process (LGWR) 2-30
Checkpoint Process (CKPT) 2-32
System Monitor Process (SMON) 2-33
Process Monitor Process (PMON) 2-34
Recoverer Process (RECO) 2-35
Listener Registration Process (LREG) 2-36

Archiver Processes (ARCn) 2-37
Database Storage Architecture 2-38
Logical and Physical Database Structures 2-40
Segments, Extents, and Blocks 2-42
Tablespaces and Data Files 2-43
SYSTEM and SYSAUX Tablespaces 2-44
Oracle Container Database: Introduction 2-45
Multitenant Architecture 2-46
Automatic Storage Management 2-47
ASM Storage Components 2-48
Interacting with an Oracle Database: Memory, Processes, and Storage 2-49
Quiz 2-51
Summary 2-52
Practice: Overview 2-53

3 Oracle Database Management Tools

Objectives 3-2
Oracle Database Management Tools: Introduction 3-3
Using SQL*Plus 3-4
Calling SQL*Plus from a Shell Script 3-5
Calling a SQL Script from SQL*Plus 3-6
Oracle SQL Developer: Connections 3-7
Oracle SQL Developer: DBA Actions 3-8
Oracle Enterprise Manager Database Express Architecture 3-9
Configuring Enterprise Manager Database Express 3-10
Logging In to Oracle Enterprise Manager Database Express 3-11
Using the Database Home Page 3-12
Using Enterprise Manager Database Express Menus 3-13
Oracle Enterprise Manager Cloud Control Components 3-14
Controlling the Enterprise Manager Cloud Control Framework 3-15
Starting the Enterprise Manager Cloud Control Framework 3-16
Stopping the Enterprise Manager Cloud Control Framework 3-17
Types of Enterprise Manager Cloud Control Targets 3-18
Enterprise Manager Cloud Control 3-19
Using Enterprise Manager Cloud Control 3-20
Quiz 3-21
Summary 3-22
Practice: Overview 3-23

4 Managing the Database Instance

- Objectives 4-2
- Initialization Parameter Files 4-3
- Types of Initialization Parameters 4-5
- Initialization Parameters: Examples 4-6
- Using SQL*Plus to View Parameters 4-10
- Changing Initialization Parameter Values 4-13
- Changing Parameter Values: Examples 4-15
- Quiz 4-16
- Starting Up an Oracle Database Instance: NOMOUNT 4-17
- Starting Up an Oracle Database Instance: MOUNT 4-18
- Starting Up an Oracle Database Instance: OPEN 4-19
- Startup Options: Examples 4-20
- Shutdown Modes 4-21
- Shutdown Options 4-22
- Shutdown Options: Examples 4-25
- Viewing the Alert Log 4-26
- Using Trace Files 4-28
- Administering the DDL Log File 4-30
- Understanding the Debug Log File 4-31
- Using Dynamic Performance Views 4-32
- Dynamic Performance Views: Usage Examples 4-33
- Dynamic Performance Views: Considerations 4-34
- Data Dictionary: Overview 4-35
- Data Dictionary Views 4-36
- Data Dictionary: Usage Examples 4-39
- Quiz 4-40
- Summary 4-41
- Practice: Overview 4-42

5 Configuring the Oracle Network Environment

- Objectives 5-2
- Oracle Net Services: Overview 5-3
- Oracle Net Listener: Overview 5-4
- Establishing Oracle Network Connections 5-5
- Connecting to an Oracle Database 5-6
- Name Resolution 5-7
- Establishing a Connection 5-8
- User Sessions 5-9
- Naming Methods 5-11
- Easy Connect 5-12

Local Naming 5-13
Directory Naming 5-14
External Naming Method 5-15
Tools for Configuring and Managing Oracle Net Services 5-16
Defining Oracle Net Services Components 5-17
Using Enterprise Manager Cloud Control 5-18
Using Oracle Net Manager 5-19
Using Oracle Net Configuration Assistant 5-20
Using the Listener Control Utility 5-21
Listener Control Utility Syntax 5-22
Advanced Connection Options 5-24
Testing Oracle Net Connectivity 5-26
Comparing Dedicated Server and Shared Server Configurations 5-27
User Sessions: Dedicated Server Process 5-28
User Sessions: Shared Server Processes 5-29
SGA and PGA Usage 5-30
Shared Server Configuration Considerations 5-31
Configuring Communication Between Databases 5-32
Connecting to Another Database 5-33
Quiz 5-34
Summary 5-36
Practice: Overview 5-37

6 Administering User Security

Objectives 6-2
Database User Accounts 6-3
Predefined Administrative Accounts 6-5
Administrative Privileges 6-6
Protecting Privileged Accounts 6-7
Authenticating Users 6-8
Administrator Authentication 6-10
OS Authentication and OS Groups 6-11
Managing Users 6-13
Creating a User 6-14
Unlocking a User Account and Resetting the Password 6-15
Privileges 6-16
System Privileges 6-17
Revoking System Privileges with ADMIN OPTION 6-19
Granting Object Privileges 6-20
Object Privileges 6-21
Revoking Object Privileges with GRANT OPTION 6-22

Using Roles to Manage Privileges	6-23
Assigning Privileges to Roles and Assigning Roles to Users	6-24
Predefined Roles	6-25
Creating a Role	6-26
Secure Roles	6-27
Assigning Roles to Users	6-28
Privilege Analysis	6-29
Privilege Analysis Flow	6-30
Profiles and Users	6-31
Implementing Password Security Features	6-33
Creating a Password Profile	6-35
Supplied Password Verification Functions	6-36
Modifications to the Default Profile	6-37
Assigning Quotas to Users	6-38
Applying the Principle of Least Privilege	6-40
Quiz	6-42
Summary	6-46
Practice: Overview	6-47

7 Managing Database Storage Structures

Objectives	7-2
How Table Data Is Stored	7-3
Database Block: Contents	7-4
Exploring the Storage Structure	7-5
Creating a New Tablespace	7-6
Tablespaces Created by Default: Overview	7-12
Altering a Tablespace	7-14
Adding a Data File to a Tablespace	7-16
Making Changes to a Data File	7-17
Dropping Tablespaces	7-18
Viewing Tablespace Information	7-19
Oracle Managed Files (OMF)	7-20
Quiz	7-22
Enlarging the Database	7-23
Moving or Renaming an Online Data File	7-24
Summary	7-26
Practice: Overview	7-27

8 Managing Space

- Objectives 8-2
- Space Management: Overview 8-3
- Block Space Management 8-4
- Row Chaining and Migration 8-5
- Quiz 8-7
- Free Space Management Within Segments 8-8
- Types of Segments 8-9
- Allocating Extents 8-10
- Understanding Deferred Segment Creation 8-11
- Viewing Deferred Segment Information 8-12
- Controlling Deferred Segment Creation 8-13
- Restrictions and Exceptions 8-14
- Additional Automatic Functionality 8-15
- Quiz 8-16
- Table Compression: Overview 8-17
- Compression for Direct-Path Insert Operations 8-18
- Advanced Row Compression for DML Operations 8-20
- Specifying Table Compression 8-21
- Using the Compression Advisor 8-22
- Using the DBMS_COMPRESSION Package 8-23
- Proactive Tablespace Monitoring 8-24
- Thresholds and Resolving Space Problems 8-25
- Monitoring Tablespace Space Usage 8-26
- Shrinking Segments 8-27
- Results of Shrink Operation 8-28
- Reclaiming Space Within ASSM Segments 8-29
- Using the Segment Advisor 8-30
- Automatic Segment Advisor 8-31
- Shrinking Segments by Using SQL 8-32
- Shrinking Segments by Using Enterprise Manager 8-33
- Managing Resumable Space Allocation 8-34
- Using Resumable Space Allocation 8-35
- Resuming Suspended Statements 8-37
- What Operations Are Resumable? 8-39
- Quiz 8-40
- Summary 8-41
- Practice: Overview 8-42

9 Managing Undo Data

- Objectives 9-2
- Undo Data: Overview 9-3
- Transactions and Undo Data 9-5
- Storing Undo Information 9-6
- Comparing Undo Data and Redo Data 9-7
- Managing Undo 9-8
- Configuring Undo Retention 9-9
- Categories of Undo 9-10
- Guaranteeing Undo Retention 9-11
- Changing an Undo Tablespace to a Fixed Size 9-12
- Temporary Undo: Overview 9-13
- Temporary Undo: Benefits 9-14
- Enabling Temporary Undo 9-15
- Monitoring Temporary Undo 9-16
- Viewing Undo Information 9-17
- Viewing Undo Activity 9-18
- Using the Undo Advisor 9-19
- Quiz 9-20
- Summary 9-21
- Practice: Overview 9-22

10 Managing Data Concurrency

- Objectives 10-2
- Locks 10-3
- Locking Mechanism 10-4
- Data Concurrency 10-5
- DML Locks 10-7
- Enqueue Mechanism 10-8
- Lock Conflicts 10-9
- Possible Causes of Lock Conflicts 10-10
- Detecting Lock Conflicts 10-11
- Resolving Lock Conflicts 10-12
- Resolving Lock Conflicts by Using SQL 10-13
- Deadlocks 10-14
- Quiz 10-15
- Summary 10-17
- Practice: Overview 10-18

11 Implementing Oracle Database Auditing

Objectives 11-2
Database Security 11-3
Monitoring for Compliance 11-5
Types of Activities to be Audited 11-6
Mandatorily Audited Activities 11-7
Understanding Auditing Implementation 11-8
Administering the Roles Required for Auditing 11-9
Database Auditing: Overview 11-10
Understanding the Audit Architecture 11-11
Enabling Unified Auditing 11-12
Configuring Auditing 11-13
Using Enterprise Manager Cloud Control 11-14
Creating a Unified Audit Policy 11-15
Creating an Audit Policy: System-Wide Audit Options 11-16
Creating an Audit Policy: Object-Specific Actions 11-17
Creating an Audit Policy: Specifying Conditions 11-18
Enabling and Disabling Audit Policies 11-19
Altering a Unified Audit Policy 11-20
Viewing Audit Policy Information 11-21
Setting the Write Mode for Audit Trail Records 11-22
Value-Based Auditing 11-23
Fine-Grained Auditing 11-25
FGA Policy 11-26
Audited DML Statement: Considerations 11-28
FGA Guidelines 11-29
Archiving and Purging the Audit Trail 11-30
Purging Audit Trail Records 11-31
Quiz 11-32
Summary 11-33
Practice: Overview 11-34

12 Backup and Recovery: Concepts

Objectives 12-2
DBA Responsibilities 12-3
Categories of Failure 12-5
Statement Failure 12-6
User Process Failure 12-7
Network Failure 12-8
User Error 12-9
Flashback Technology 12-10

Instance Failure 12-12
Understanding Instance Recovery: Checkpoint (CKPT) Process 12-13
Understanding Instance Recovery: Redo Log Files and Log Writer 12-14
Understanding Instance Recovery 12-15
Phases of Instance Recovery 12-16
Tuning Instance Recovery 12-17
Using the MTTR Advisor 12-18
Media Failure 12-19
Comparing Complete and Incomplete Recovery 12-20
Complete Recovery Process 12-21
Point-in-Time Recovery Process 12-22
Oracle Data Protection Solutions 12-24
Quiz 12-25
Summary 12-26

13 Backup and Recovery: Configuration

Objectives 13-2
Configuring for Recoverability 13-3
Configuring the Fast Recovery Area 13-4
Monitoring the Fast Recovery Area 13-5
Multiplexing Control Files 13-6
Redo Log Files 13-8
Multiplexing the Redo Log 13-9
Creating Archived Redo Log Files 13-10
Archiver (ARCn) Process 13-11
Archived Redo Log Files: Naming and Destinations 13-12
Configuring ARCHIVELOG Mode 13-14
Quiz 13-15
Summary 13-16
Practice: Overview 13-17

14 Performing Database Backups

Objectives 14-2
Backup Solutions: Overview 14-3
Oracle Secure Backup 14-4
User-Managed Backup 14-5
Understanding Backup Terminology 14-6
Understanding Types of Backups 14-7
RMAN Backup Types 14-8
Using Recovery Manager (RMAN) 14-10
Configuring Backup Settings 14-11

Oracle-Suggested Backup	14-13
Selecting a Backup Strategy	14-14
Backing Up the Control File to a Trace File	14-15
Managing Backups	14-16
Using RMAN Commands to Create Backups	14-17
Quiz	14-18
Summary	14-19
Practice: Overview	14-20

15 Performing Database Recovery

Objectives	15-2
Opening a Database	15-3
Keeping a Database Open	15-5
Data Recovery Advisor	15-6
Loss of a Control File	15-8
Loss of a Redo Log File	15-9
Loss of a Data File in NOARCHIVELOG Mode	15-11
Loss of a Noncritical Data File in ARCHIVELOG Mode	15-12
Loss of a System-Critical Data File in ARCHIVELOG Mode	15-13
Quiz	15-14
Summary	15-15
Practice: Overview	15-16

16 Moving Data

Objectives	16-2
Moving Data: General Architecture	16-3
Oracle Data Pump: Overview	16-4
Oracle Data Pump: Benefits	16-5
Directory Objects for Data Pump	16-7
Creating Directory Objects	16-8
Data Pump Export and Import Clients: Overview	16-9
Data Pump Utility: Interfaces and Modes	16-10
Performing a Data Pump Export by Using Enterprise Manager Cloud Control	16-11
Performing a Data Pump Import	16-12
Data Pump Import: Transformations	16-13
Using Enterprise Manager Cloud Control to Monitor Data Pump Jobs	16-14
SQL*Loader: Overview	16-15
SQL*Loader Control File	16-17
Loading Methods	16-19
Loading Data by Using Enterprise Manager Cloud Control	16-20
SQL*Loader Express Mode	16-21

External Tables 16-23
External Table: Benefits 16-24
Defining an External Tables with ORACLE_LOADER 16-25
External Table Population with ORACLE_DATAPUMP 16-26
Using External Tables 16-27
Data Dictionary 16-28
Quiz 16-29
Summary 16-31
Practice: Overview 16-32

17 Database Maintenance

Objectives 17-2
Database Maintenance 17-3
Viewing the Alert History 17-4
Terminology 17-5
Automatic Workload Repository (AWR): Overview 17-6
AWR Infrastructure 17-7
Automatic Workload Repository 17-8
AWR Baselines 17-10
Accessing the AWR Page 17-11
Managing the AWR 17-12
Statistic Levels 17-13
Automatic Database Diagnostic Monitor (ADDM) 17-14
ADDM Findings in Enterprise Manager Cloud Control 17-15
ADDM Findings in Enterprise Manager Database Express 17-16
Advisory Framework 17-17
Viewing the Advisor Central Page in Enterprise Manager Cloud Control 17-19
Using Packages to Invoke the Advisors 17-20
Automated Maintenance Tasks 17-21
Automated Maintenance Tasks Configuration 17-23
Server-Generated Alerts 17-24
Setting Metrics Thresholds 17-25
Reacting to Alerts 17-26
Alert Types and Clearing Alerts 17-27
Quiz 17-28
Summary 17-29
Practice: Overview 17-30

18 Managing Performance

- Objectives 18-2
- Performance Monitoring 18-3
- Tuning Activities 18-5
- Performance Planning 18-6
- Instance Tuning 18-8
- Performance Tuning Methodology 18-9
- Performance Tuning Data 18-10
- Using the Enterprise Manager Database Express Performance Hub Page 18-11
- Using the Enterprise Manager Cloud Control Performance Home Page 18-13
- Monitoring Session Performance 18-14
- Performance Monitoring: Top Sessions 18-15
- Displaying Session-Related Statistics 18-16
- Performance Monitoring: Top Services 18-17
- Displaying Service-Related Statistics 18-18
- Viewing Wait Events 18-19
- Oracle Wait Events 18-20
- Memory Management: Overview 18-21
- Managing Memory Components 18-22
- Efficient Memory Usage: Guidelines 18-23
- Automatic Memory Management: Overview 18-25
- Oracle Database Memory Parameters 18-26
- Enabling Automatic Memory Management (AMM) by Using Enterprise Manager Cloud Control 18-27
- Monitoring Automatic Memory Management 18-28
- Automatic Shared Memory Management: Overview 18-30
- Enabling Automatic Shared Memory Management (ASMM) 18-31
- Understanding Automatic Shared Memory Management 18-32
- Automatic Shared Memory Advisor 18-33
- Enabling Automatic Shared Memory Management 18-34
- Disabling Automatic Shared Memory Management 18-35
- Using V\$PARAMETER to View Memory Component Sizes 18-36
- Managing the Program Global Area (PGA) 18-37
- Dynamic Performance Statistics 18-39
- Troubleshooting and Tuning Views 18-41
- Quiz 18-42
- Summary 18-44
- Practice: Overview 18-45

19 Managing Performance: SQL Tuning

Objectives 19-2
SQL Tuning 19-3
Oracle Optimizer: Overview 19-4
Optimizer Statistics 19-5
Optimizer Statistics Collection 19-6
Using the Optimizer Statistics Console 19-8
Setting Global Preferences by Using Enterprise Manager Cloud Control 19-9
Gathering Optimizer Statistics Manually 19-10
Setting Optimizer Statistics Preferences 19-12
Concurrent Statistics Gathering 19-14
Viewing Statistics Information 19-15
SQL Plan Directives 19-17
Adaptive Execution Plans 19-18
Using the SQL Advisors 19-19
Automatic SQL Tuning Results 19-20
Implementing Automatic Tuning Recommendations 19-21
SQL Tuning Advisor: Overview 19-22
Using the SQL Tuning Advisor 19-23
SQL Tuning Advisor Recommendations 19-25
Duplicate SQL 19-26
SQL Access Advisor: Overview 19-27
Using the SQL Access Advisor 19-28
Workload Source 19-29
Recommendation Options 19-30
Reviewing Recommendations 19-32
SQL Performance Analyzer: Overview 19-33
SQL Performance Analyzer: Use Cases 19-34
Using SQL Performance Analyzer 19-35
Quiz 19-36
Summary 19-40
Practice: Overview 19-41

20 Using Database Resource Manager

Objectives 20-2
Database Resource Manager: Overview 20-3
Database Resource Manager: Concepts 20-4
Using the Resource Manager 20-5
Default Plan for Maintenance Windows 20-7
Default Plan 20-8
Creating a Simple Resource Plan 20-9

Creating a Complex Resource Plan 20-10
Specifying Resource Plan Directives 20-12
Resource Allocation Methods for Resource Plans 20-13
Comparison of EMPHASIS and RATIO 20-14
Active Session Pool Mechanism 20-16
Specifying Thresholds 20-17
Setting Idle Timeouts 20-19
Limiting CPU Utilization at the Database Level 20-20
Limiting CPU Utilization at the Server Level: Instance Caging 20-22
Instance Caging: Examples 20-23
Monitoring Instance Caging 20-24
Runaway Queries and Resource Manager 20-25
Resource Consumer Group Mapping 20-27
Activating a Resource Plan 20-29
Database Resource Manager Information 20-30
Viewing Resource Manager Statistics 20-31
Monitoring the Resource Manager 20-32
Quiz 20-34
Summary 20-35
Practice: Overview 20-36

21 Using Oracle Scheduler to Automate Tasks

Objectives 21-2
Simplifying Management Tasks 21-3
Understanding a Simple Job 21-4
Oracle Scheduler Core Components 21-5
Using Oracle Scheduler 21-6
Quiz 21-8
Persistent Lightweight Jobs 21-9
Using a Time-Based or Event-Based Schedule 21-10
Creating a Time-Based Job 21-11
Creating an Event-Based Schedule 21-13
Creating Event-Based Schedules by Using Enterprise Manager
 Cloud Control 21-14
Creating an Event-Based Job 21-15
Event-Based Scheduling 21-16
Creating Complex Schedules 21-18
Quiz 21-19
Using Email Notification 21-20
Adding and Removing Email Notifications 21-21
Creating Job Chains 21-22

Example of a Chain 21-24
Using Advanced Scheduler Features 21-25
Job Classes 21-26
Windows 21-28
Prioritizing Jobs Within a Window 21-29
Creating a Job Array 21-30
Quiz 21-32
Creating a File Watcher and an Event-Based Job 21-33
Enabling File Arrival Events from Remote Systems 21-35
Scheduling Remote Database Jobs 21-36
Creating Remote Database Jobs 21-37
Scheduling Multiple Destination Jobs 21-38
Viewing Scheduler Meta Data 21-39
Quiz 21-41
Summary 21-42
Practice: Overview 21-43

Appendix A: Working with Oracle Support

Objectives A-2
Working with Oracle Support A-3
Using My Oracle Support A-4
Researching an Issue A-6
Logging Service Requests A-8
Accessing My Oracle Support Community A-10
Managing Patches A-11
Applying a Patch Release A-12
Enterprise Manager Cloud Control: My Oracle Support Integration A-13
Using the Patch Advisor A-14
Online Patching: Overview A-15
Installing an Online Patch A-16
Benefits of Online Patching A-17
Conventional Patching and Online Patching A-18
Online Patching Considerations A-19
Quiz A-20
Using the Support Workbench A-21
Accessing the Support Workbench A-22
Viewing Problem Details A-23
Viewing Incident Details A-24
Packaging and Uploading Diagnostic Data to Oracle Support A-25
Tracking the Service Request and Implementing Repairs A-26
Summary A-27

12

Backup and Recovery: Concepts

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

- Identify the types of failure that can occur in an Oracle database
- Describe instance recovery
- Describe complete and incomplete recovery



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

DBA Responsibilities

- Protect the database from failure wherever possible
- Increase the mean time between failures (MTBF)
- Protect critical components by using redundancy
- Decrease the mean time to recover (MTTR)
- Minimize the loss of data



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The database administrator (DBA) is typically responsible for ensuring that the database is open and available when users need it. To achieve that goal, the DBA (working with the system administrator):

- Anticipates and works to avoid common causes of failure
- Works to increase the mean time between failures (MTBF) that negatively affect availability
- Ensures that hardware is as reliable as possible, that critical components are protected by redundancy, and that operating system maintenance is performed in a timely manner. Oracle Database provides advanced configuration options to increase MTBF, including:
 - Real Application Clusters
 - Oracle Data Guard
- Decreases the mean time to recover (MTTR) by practicing recovery procedures in advance and configuring backups so that they are readily available when needed

- Minimizes the loss of data. DBAs who follow accepted best practices can configure their databases so that no committed transaction is ever lost. Entities that assist in guaranteeing this include:
 - Archive log files (discussed later in this lesson)
 - Flashback technology
 - Standby databases and Oracle Data Guard (discussed in the *Oracle Database 12c: Data Guard Administration* course)

Categories of Failure

Failures can generally be divided into the following categories:

- Statement failure
- User process failure
- Network failure
- User error
- Instance failure
- Media failure



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

- **Statement failure:** A single database operation (select, insert, update, or delete) fails.
- **User process failure:** A single database session fails.
- **Network failure:** Connectivity to the database is lost.
- **User error:** A user successfully completes an operation, but the operation (dropping a table or entering bad data) is incorrect.
- **Instance failure:** The database instance shuts down unexpectedly.
- **Media failure:** A loss of any file that is needed for database operation (that is, the files have been deleted or the disk has failed).

Statement Failure

Typical Problems	Possible Solutions
Attempts to enter invalid data into a table	Work with users to validate and correct data.
Attempts to perform operations with insufficient privileges	Provide appropriate object or system privileges.
Attempts to allocate space that fail	<ul style="list-style-type: none">Enable resumable space allocation.Increase owner quota.Add space to tablespace.
Logic errors in applications	Work with developers to correct program errors.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

When a single database operation fails, DBA involvement may be necessary to correct errors with user privileges or database space allocation. DBAs may also need to assist in troubleshooting, even for problems that are not directly in their task area. This can vary greatly from one organization to another. For example, in organizations that use off-the-shelf applications (that is, organizations that have no software developers), the DBA is the only point of contact and must examine logic errors in applications.

To understand logic errors in applications, you should work with developers to understand the scope of the problem. Oracle Database tools may provide assistance by helping to examine audit trails or previous transactions.

Note: In many cases, statement failures are by design and desired. For example, security policies and quota rules are often decided upon in advance. If a user gets an error while trying to exceed his or her limits, it may be desired for the operation to fail and no resolution may be necessary.

User Process Failure

Typical Problems	Possible Solutions
A user performs an abnormal disconnect.	A DBA's action is not usually needed to resolve user process failures. Instance background processes roll back uncommitted changes and release locks.
A user's session is abnormally terminated.	
A user experiences a program error that terminates the session.	Watch for trends. 



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

User processes that abnormally disconnect from the instance may have uncommitted work in progress that needs to be rolled back. The Process Monitor (PMON) background process periodically polls server processes to ensure that their sessions are still connected. If PMON finds a server process whose user is no longer connected, PMON recovers from any ongoing transactions; it also rolls back uncommitted changes and releases any locks that are held by the failed session.

A DBA's intervention should not be required to recover from user process failure, but the administrator must watch for trends. One or two users disconnecting abnormally is not a cause for concern. A small percentage of user process failures may occur from time to time. But consistent and systemic failures indicate other problems. A large percentage of abnormal disconnects may indicate a need for user training (which includes teaching users to log out rather than just terminate their programs). It may also be indicative of network or application problems.

Network Failure

Typical Problems	Possible Solutions
Listener fails.	Configure a backup listener and connect-time failover.
Network Interface Card (NIC) fails.	Configure multiple network cards.
Network connection fails.	Configure a backup network connection.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The best solution to network failure is to provide redundant paths for network connections. Backup listeners, network connections, and network interface cards reduce the chance that network failures will affect system availability.

User Error

Typical Causes	Possible Solutions
User inadvertently deletes or modifies data.	Roll back transaction and dependent transactions or rewind table.
User drops a table.	Recover table from recycle bin. Recover table from a backup.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Users may inadvertently delete or modify data. If they have not yet committed or exited their program, they can simply roll back.

You can use Oracle LogMiner to query your online redo logs and archived redo logs through an Enterprise Manager or SQL interface. Transaction data may persist in online redo logs longer than it persists in undo segments; if you have configured archiving of redo information, redo persists until you delete the archived files. Oracle LogMiner is discussed in the *Oracle Database Utilities* reference.

Users who drop a table can recover it from the recycle bin by flashing back the table to before the drop.

If the recycle bin has already been purged, or if the user dropped the table with the PURGE option, the dropped table can still be recovered by using point-in-time recovery (PITR) if the database has been properly configured.

In Oracle Database 12c, RMAN enables you to recover one or more tables or table partitions to a specified point in time without affecting the remaining database objects.

Note: Flashback technologies, PITR, and table recovery are discussed in the *Oracle Database 12c: Backup and Recovery Workshop* course and in the *Oracle Database Backup and Recovery User's Guide*.

Flashback Technology

Use Flashback technology for:

- Viewing past states of data
- Winding data back and forth in time
- Assisting users in error analysis and recovery



For error analysis:

Oracle Flashback Query

Oracle Flashback Versions Query

Oracle Flashback Transaction Query

For error recovery:

Oracle Flashback Transaction Backout

Oracle Flashback Table

Oracle Flashback Drop

Oracle Flashback Database

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle Database includes Oracle Flashback technology: a group of features that support viewing past states of data—and winding data back and forth in time—without requiring restoring the database from backup. With this technology, you help users analyze and recover from errors. For users who have committed erroneous changes, use the following to analyze the errors:

- **Flashback Query:** View committed data as it existed at some point in the past. The SELECT command with the AS OF clause references a time in the past through a time stamp or system change number (SCN).
- **Flashback Version Query:** View committed historical data for a specific time interval. Use the VERSIONS BETWEEN clause of the SELECT command (for performance reasons with existing indexes).
- **Flashback Transaction Query:** View all database changes made at the transaction level.

Possible solutions to recover from user error:

- **Flashback Transaction Backout:** Rolls back a specific transaction and dependent transactions
- **Flashback Table:** Rewinds one or more tables to their contents at a previous time without affecting other database objects

- **Flashback Drop:** Reverses the effects of dropping a table by returning the dropped table from the recycle bin to the database along with dependent objects such as indexes and triggers
- **Flashback Database:** Returns the database to a past time or SCN

Instance Failure

Typical Causes	Possible Solutions
Power outage	Restart the instance by using the STARTUP command. Recovering from instance failure is automatic, including rolling forward changes in the redo logs and then rolling back any uncommitted transactions.
Hardware failure	Investigate the causes of failure by using the alert log, trace files, and Enterprise Manager.
Failure of one of the critical background processes	
Emergency shutdown procedures	



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

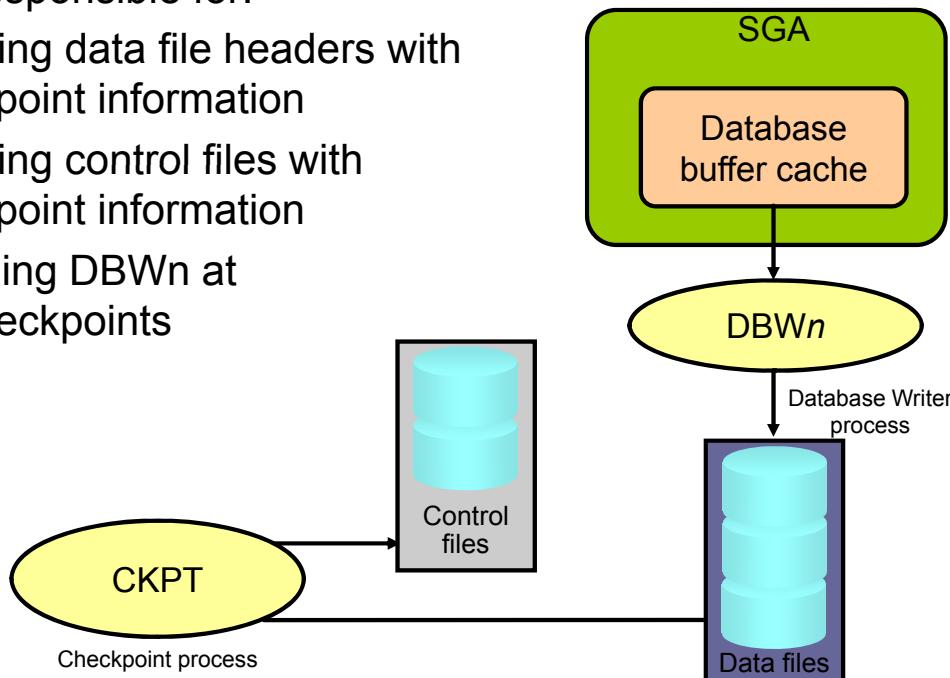
Instance failure occurs when the database instance is shut down before synchronizing all database files. An instance failure can occur because of hardware or software failure or through the use of the emergency SHUTDOWN ABORT and STARTUP FORCE shutdown commands.

Administrator involvement in recovering from instance failure is rarely required if Oracle Restart is enabled and is monitoring your database. Oracle Restart attempts to restart your database instance as soon as it fails. If manual intervention is required, then there may be a more serious problem that prevents the instance from restarting, such as a memory CPU failure.

Understanding Instance Recovery: Checkpoint (CKPT) Process

CKPT is responsible for:

- Updating data file headers with checkpoint information
- Updating control files with checkpoint information
- Signaling DBWn at full checkpoints



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To understand instance recovery, you need to understand the functioning of certain background processes.

Every three seconds (or more frequently), the CKPT process stores data in a control file to document the modified data blocks that DBWn has written from the SGA to disk. This is called an “incremental checkpoint.” The purpose of a checkpoint is to identify that place in the online redo log file where instance recovery is to begin (which is called the “checkpoint position”).

In the event of a log switch, the CKPT process also writes this checkpoint information to the headers of data files.

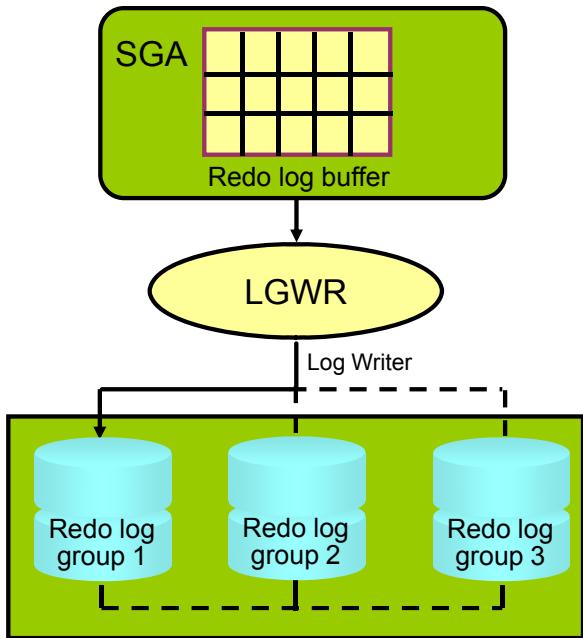
Checkpoints exist for the following reasons:

- To ensure that modified data blocks in memory are written to the disk regularly so that data is not lost in case of a system or database failure
- To reduce the time required for instance recovery (Only the online redo log file entries following the last checkpoint need to be processed for recovery.)
- To ensure that all committed data has been written to data files during shutdown

The checkpoint information written by the CKPT process includes checkpoint position, system change number (SCN), location in the online redo log file to begin recovery, information about logs, and so on.

Note: The CKPT process does not write data blocks to the disk or redo blocks to the online redo log files.

Understanding Instance Recovery: Redo Log Files and Log Writer



Redo log files:

- Record changes to the database
- Should be multiplexed to protect against loss

Log Writer (LGWR) writes:

- At commit
- When one-third full
- Every three seconds
- Before DBW n writes
- Before clean shutdowns

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Redo log files record changes to the database as a result of transactions and internal Oracle server actions. (A transaction is a logical unit of work consisting of one or more SQL statements run by a user.) Redo log files protect the database from loss of integrity because of system failures caused by power outages, disk failures, and so on. Redo log files should be multiplexed to ensure that the information stored in them is not lost in the event of a disk failure.

The redo log consists of groups of redo log files. A group consists of a redo log file and its multiplexed copies. Each identical copy is said to be a member of that group, and each group is identified by a number. The Log Writer (LGWR) process writes redo records from the redo log buffer to all members of a redo log group until the files are filled or a log switch operation is requested.

It then switches and writes to the files in the next group. Redo log groups are used in a circular fashion.

Best practice tip: If possible, multiplexed redo log files should reside on different disks.

Understanding Instance Recovery

Automatic instance or crash recovery:

- Is caused by attempts to open a database whose files are not synchronized on shutdown
- Uses information stored in redo log groups to synchronize files
- Involves two distinct operations:
 - Rolling forward: Redo log changes (both committed and uncommitted) are applied to data files.
 - Rolling back: Changes that are made but not committed are returned to their original state.

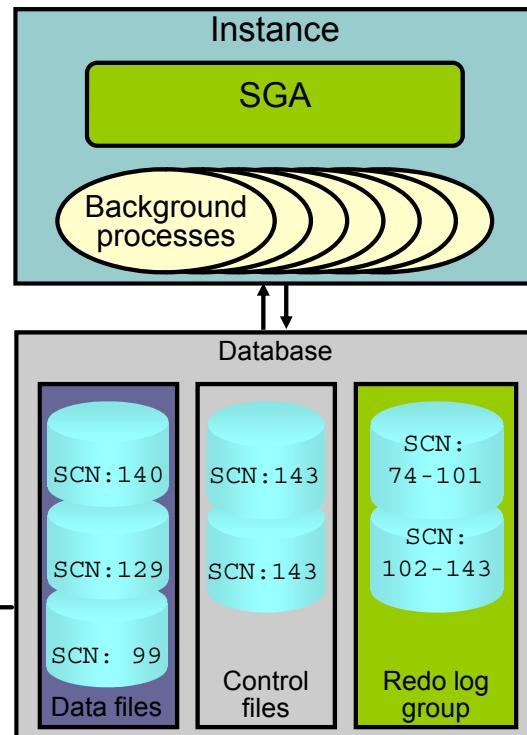


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Oracle database automatically recovers from instance failure. All that needs to happen is for the instance to be started normally. If Oracle Restart is enabled and configured to monitor this database, then this happens automatically. The instance mounts the control files and then attempts to open the data files. When it discovers that the data files have not been synchronized during shutdown, the instance uses information contained in the redo log groups to roll the data files forward to the time of shutdown. Then the database is opened and any uncommitted transactions are rolled back.

Phases of Instance Recovery

1. Instance startup (data files are out of sync)
2. Roll forward (redo)
3. Committed and uncommitted data in files
4. Database opened
5. Roll back (undo)
6. Committed data in files



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

For an instance to open a data file, the system change number (SCN) contained in the data file's header must match the current SCN that is stored in the database's control files.

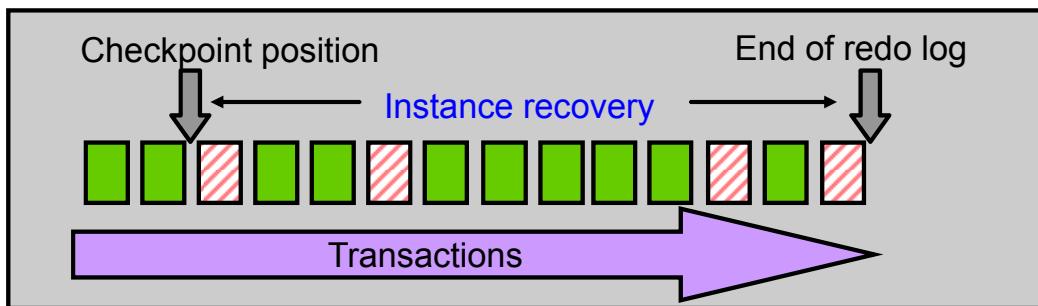
If the numbers do not match, the instance applies redo data from the online redo logs, sequentially "redoing" transactions until the data files are up-to-date. After all data files have been synchronized with the control files, the database is opened and users can log in.

When redo logs are applied, *all* transactions are applied to bring the database up to the state as of the time of failure. This usually includes transactions that are in progress but have not yet been committed. After the database has been opened, those uncommitted transactions are rolled back.

At the end of the rollback phase of instance recovery, the data files contain only committed data.

Tuning Instance Recovery

- During instance recovery, the transactions between the checkpoint position and the end of redo log must be applied to data files.
- You tune instance recovery by controlling the difference between the checkpoint position and the end of redo log.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Transaction information is recorded in the redo log groups before the instance returns commit complete for a transaction. The information in the redo log groups guarantees that the transaction can be recovered in case of a failure. The transaction information must also be written to the data file. The data file write usually happens at some time after the information is recorded in redo log groups because the data file write process is much slower than the redo writes. (Random writes for data files are slower than serial writes for redo log files.)

Every three seconds, the checkpoint process records information in the control file about the checkpoint position in the redo log. Therefore, the Oracle Database server knows that all redo log entries recorded before this point are not necessary for database recovery. In the graphic in the slide, the striped blocks have not yet been written to the disk.

The time required for instance recovery is the time required to bring data files from their last checkpoint to the latest SCN recorded in the control file. The administrator controls that time by setting an MTTR target (in seconds) and through the sizing of redo log groups. For example, for two redo groups, the distance between the checkpoint position and the end of the redo log group cannot be more than 90% of the smallest redo log group.

Using the MTTR Advisor

- Specify the desired time in seconds or minutes.
- The default value is 0 (disabled).
- The maximum value is 3,600 seconds (one hour).

The screenshot shows the 'Recovery Settings' page in Oracle Enterprise Manager. At the top right, it says 'Logged in as DBA1'. Below that are three buttons: 'Show SQL', 'Revert', and 'Apply'. The main section is titled 'Instance Recovery'. It contains a detailed description of the fast-start checkpointing feature and its purpose. Below the description, there are two input fields: 'Current Estimated Mean Time To Recover (seconds)' with a value of '19', and 'Desired Mean Time To Recover' with a value of '0' and a dropdown menu set to 'Minutes'.

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The FAST_START_MTTR_TARGET initialization parameter simplifies the configuration of recovery time from instance or system failure. The MTTR Advisor converts the FAST_START_MTTR_TARGET value into several parameters to enable instance recovery in the desired time (or as close to it as possible). Please note, explicitly setting the FAST_START_MTTR_TARGET parameter to 0 disables the MTTR Advisor.

The FAST_START_MTTR_TARGET parameter must be set to a value that supports the service level agreement for your system. A small value for the MTTR target increases I/O overhead because of additional data file writes (affecting the performance). However, if you set the MTTR target too large, the instance takes longer to recover after a crash.

For assistance in setting the MTTR target by using Enterprise Manager Cloud Control, navigate as follows:

- Performance > Advisors Home > MTTR Advisor
- Availability > Backup & Recovery > Recovery Settings

Media Failure

Typical Causes	Possible Solutions
Failure of disk drive	<ol style="list-style-type: none">1. Restore the affected file from backup.
Failure of disk controller	<ol style="list-style-type: none">2. Inform the database about a new file location (if necessary).
Deletion or corruption of a file needed for database operation	<ol style="list-style-type: none">3. Recover the file by applying redo information (if necessary).



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

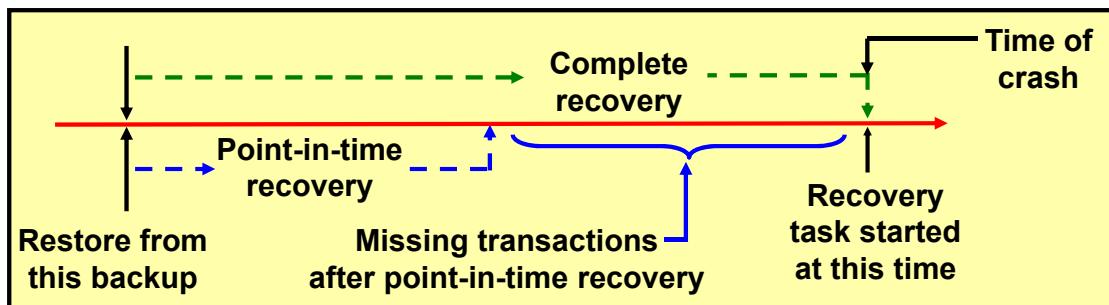
Oracle Corporation defines media failure as any failure that results in the loss or corruption of one or more database files (data, control, or redo log file).

Recovering from media failure requires that you restore and recover the missing files.

Comparing Complete and Incomplete Recovery

Recovery can have two kinds of scope:

- Complete recovery: Brings the database or tablespace up to the present, including all committed data changes made to the point in time when the recovery was requested
- Incomplete or point-in-time recovery (PITR): Brings the database or tablespace up to a specified point in time in the past, before the recovery operation was requested

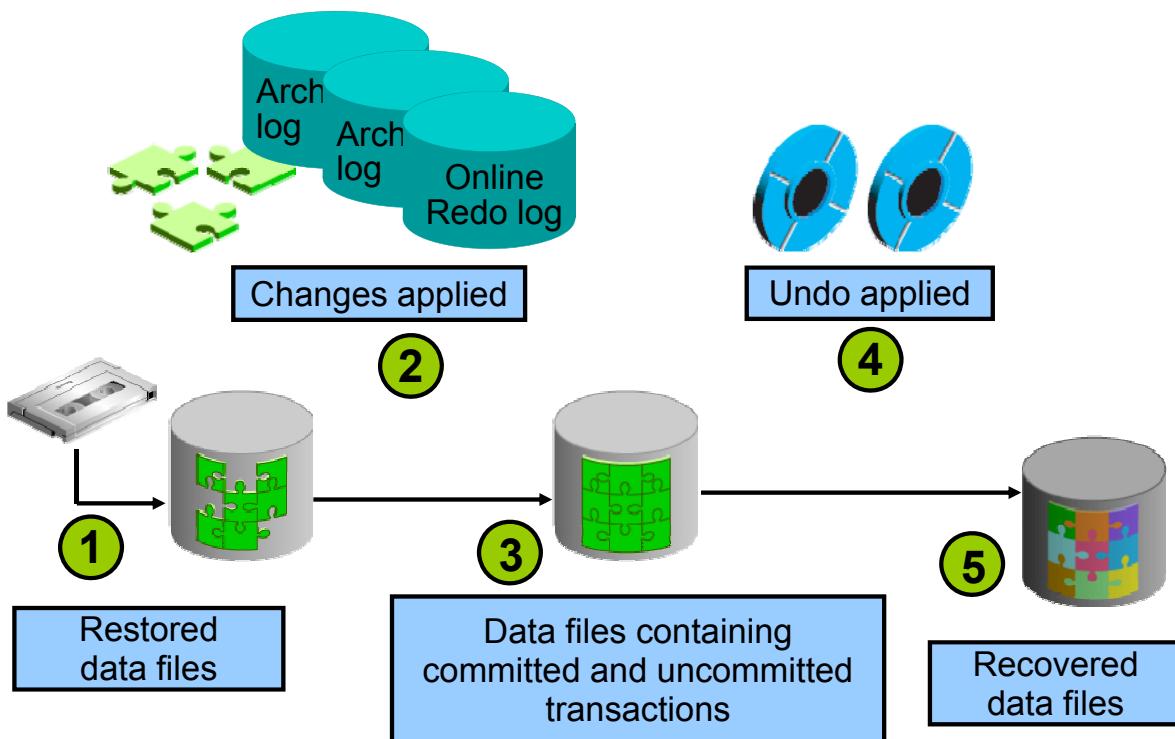


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

When you perform complete recovery, you bring the database to the state where it is fully up-to-date, including all committed data modifications to the present time.

Incomplete recovery, however, brings the database or tablespace to some point of time in the past. This is also known as “point-in-time recovery (PITR).” It 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 changes made to the database that need to be undone. Recovering to a point in the past is a way to remove the unwanted changes.

Complete Recovery Process



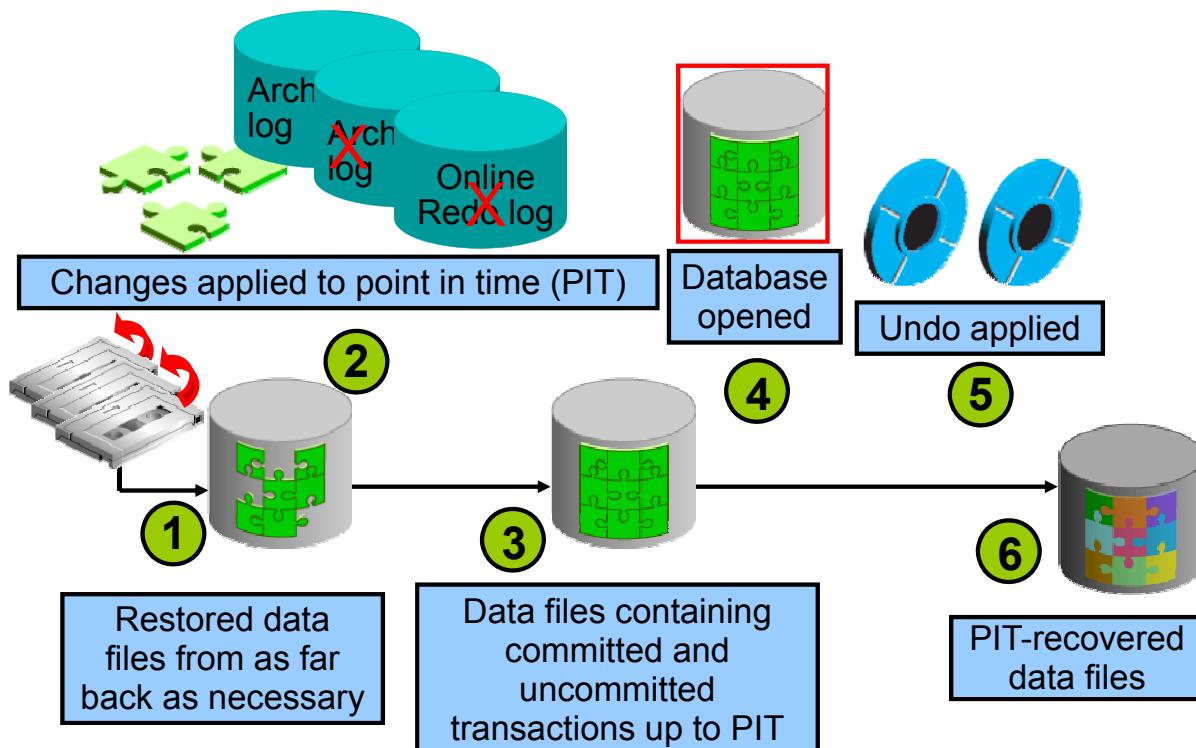
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The following steps describe what takes 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 re-entered. 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 undo blocks are used to roll back any uncommitted changes. This is sometimes referred to as transaction recovery.
5. The data files are now in a recovered state and are consistent with the other data files in the database.

Point-in-Time Recovery Process



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Incomplete recovery, or database point-in-time recovery (DBPITR), 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 point-in-time 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 steps to perform a point-in-time recovery are as follows:

1. **Restore the data files from backup:** The backup that is used must be from before your target recovery point. This entails either copying files using OS commands or using the RMAN RESTORE command.
2. **Use the RECOVER command:** Apply redo from the archived redo log files, including as many as necessary to reach the restore point destination.
3. **State of over-recovery:** Now the data files contain some committed and some uncommitted transactions because the redo can contain uncommitted data.
4. **Use the ALTER DATABASE OPEN command:** The database is opened before undo is applied. This is to provide higher availability.

5. **Apply undo data:** 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. **Process complete:** The data files are now recovered to the point in time that you chose.

Oracle Flashback Database is the most efficient alternative to DBPITR. Unlike the other flashback features, it operates at a physical level and reverts the current data files to their contents at a past time. The result is like the result of a DBPITR, including the OPEN RESETLOGS, but Flashback Database is typically faster because it does not require you to restore data files and requires only limited application of redo compared to media recovery.

Oracle Data Protection Solutions

Backup and Recovery Objective	Recovery Time Objective (RTO)	Oracle Solution
Physical data protection	Hours/Days	Recovery Manager Oracle Secure Backup
Logical data protection	Minutes/Hours	Flashback Technologies
Recovery analysis	Minimize time for problem identification and recovery planning	Data Recovery Advisor

Disaster Recovery Objective	Recovery Time Objective (RTO)	Oracle Solution
Physical data protection	Seconds/Minutes	Data Guard Active Data Guard



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle provides an appropriate data protection solution depending on your backup and recovery objective and RTO:

- Oracle Recovery Manager (RMAN) is the core Oracle Database software component that manages database backup, restore, and recovery processes.
- Oracle Secure Backup (OSB) is Oracle's enterprise-grade tape backup management solution for both database and file system data.
- Oracle Database Flashback technologies are a set of data recovery solutions that enable human errors to be reversed by selectively and efficiently undoing the effects of a mistake.
- The Data Recovery Advisor provides intelligent database problem identification and recovery capabilities.
- Data Guard and Active Data Guard enable physical standby databases to be open for read access while being kept synchronized with the production database through media recovery.

Quiz

Statement failure is never by design and always requires the DBA to address the issue.

- a. True
- b. False



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b

Summary

In this lesson, you should have learned how to:

- Identify the types of failure that can occur in an Oracle database
- Describe instance recovery
- Describe complete and incomplete recovery



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

13

Backup and Recovery: Configuration

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

- Configure the fast recovery area
- Multiplex the control file
- Multiplex redo log files
- Configure ARCHIVELOG mode

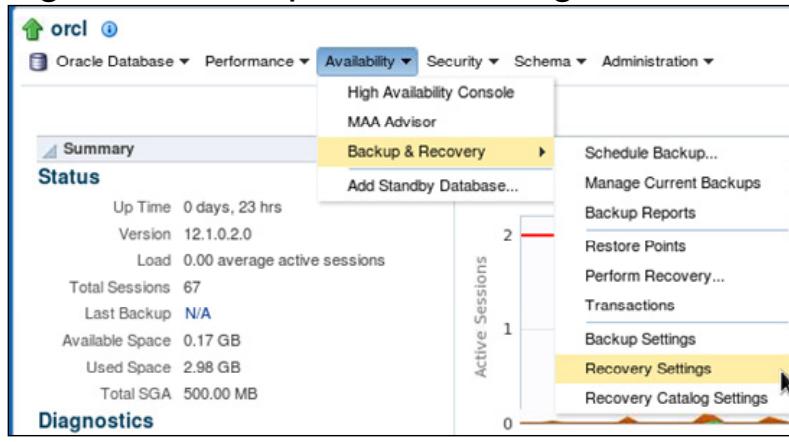


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Configuring for Recoverability

Configure your database for maximum recoverability by:

- Scheduling regular backups
- Multiplexing control files
- Multiplexing redo log groups
- Retaining archived copies of redo logs



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To provide the best protection for your data, you must:

- **Schedule regular backups:** Most media failures require that you restore the lost or damaged file from backup.
- **Multiplex control files:** All control files associated with a database are identical. Recovering from the loss of a single control file is not difficult; recovering from the loss of *all* control files is much more challenging. Guard against losing all control files by having at least two copies.
- **Multiplex redo log groups:** To recover from instance or media failure, redo log information is used to roll data files forward to the last committed transaction. If your redo log groups rely on a single redo log file, the loss of that file means that data is likely to be lost. Ensure that there are at least two copies of each redo log group; if possible, each copy should be under different disk controllers.
- **Retain archived copies of redo logs:** If a file is lost and restored from backup, the instance must apply redo information to bring that file up to the latest SCN contained in the control file. With the default setting, the database can overwrite redo information after it has been written to the data files. Your database can be configured to retain redo information in archived copies of the redo logs. This is known as placing the database in ARCHIVELOG mode.

You can perform configuration tasks in Enterprise Manager Cloud Control or by using the SQL command line.

Configuring the Fast Recovery Area

- Fast recovery area:
 - Strongly recommended for simplified backup storage management
 - Storage space (separate from working database files)
 - Location specified by the `DB_RECOVERY_FILE_DEST` parameter
 - Size specified by the `DB_RECOVERY_FILE_DEST_SIZE` parameter
 - Large enough for backups, archived logs, flashback logs, multiplexed control files, and multiplexed redo logs
 - Automatically managed according to your retention policy
- Configuration of the fast recovery area includes specifying the location, size, and retention policy.



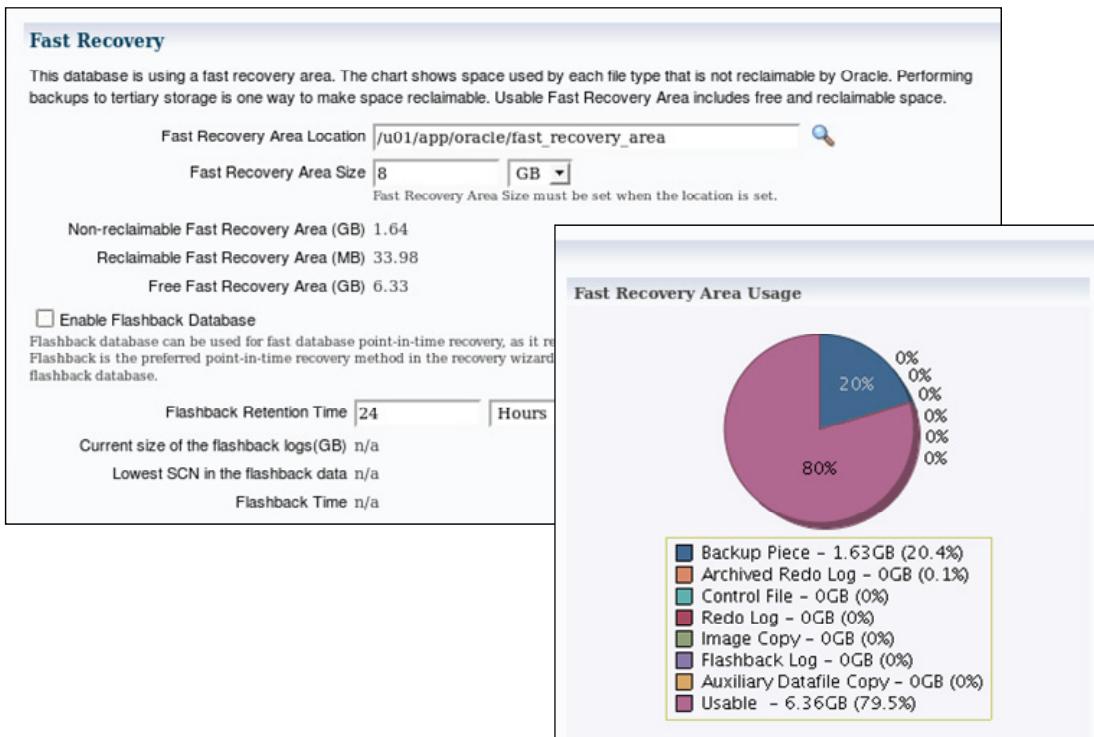
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The fast recovery area is space that is set aside on disk to contain archived logs, backups, flashback logs, multiplexed control files, and multiplexed redo logs. A fast recovery area simplifies backup storage management and is strongly recommended. You should place the fast recovery area on storage space that is separate from the location of your database data files and primary online log files and control file.

The amount of disk space to allocate for the fast recovery area depends on the size and activity levels of your database. As a general rule, the larger the fast recovery area, the more useful it is. Ideally, the fast recovery area should be large enough for copies of your data and control files and for flashback, online redo, and archived logs needed to recover the database with the backups kept based on the retention policy. (In short, the fast recovery area should be at least twice the size of the database so that it can hold one backup and several archived logs.)

Space management in the fast recovery area is governed by a backup retention policy. A retention policy determines when files are obsolete, which means that they are no longer needed to meet your data recovery objectives. The Oracle Database server automatically manages this storage by deleting files that are no longer needed.

Monitoring the Fast Recovery Area



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

If you have configured your archived logs to be written to this location, it is important to monitor this space to ensure that it does not reach its capacity. If the instance is unable to create an archived log because of lack of space, it pauses until the administrator corrects the situation.

In Enterprise Manager Cloud Control, select Availability > Backup & Recovery > Recovery Settings. On this page, you can:

- Verify how much of the fast recovery area has been consumed
- Specify the location of the fast recovery area
- Specify the size of the fast recovery area
- Configure Flashback Database
- Specify the retention time

The retention time determines when files are obsolete (that is, when they are no longer needed to meet your data recovery objectives). The Oracle Database server automatically manages this storage, deleting files that are no longer needed. You can back up the recovery area so that Oracle Recovery Manager (RMAN) can fail over to other archived redo log destinations if the archived redo log in the fast recovery area is inaccessible or corrupted.

Periodically copying backups to tape frees space in the fast recovery area for other files, but retrieving files from tape causes longer database restoration and recovery times.

Multiplexing Control Files

To protect against database failure, your database should have multiple copies of the control file.

	ASM Storage	File System Storage
Best Practice	One copy on each disk group (such as +DATA and +FRA)	At least two copies, each on separate disk (at least one on separate disk controller)
Steps to create additional control files	No additional control file copies required	<ol style="list-style-type: none">1. Alter the SPFILE with the ALTER SYSTEM SET control_files command.2. Shut down the database.3. Copy control file to a new location.4. Open the database and verify the addition of the new control file.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

A control file is a small binary file that describes the structure of the database. It must be available for writing by the Oracle server whenever the database is mounted or opened. Without this file, the database cannot be mounted, and recovery or re-creation of the control file is required. Your database should have a minimum of two control files on different storage devices to minimize the impact of a loss of one control file.

The loss of a single control file causes the instance to fail because all control files must be available at all times. However, recovery can be a simple matter of copying one of the other control files. The loss of all control files is slightly more difficult to recover from but is not usually catastrophic.

Adding a Control File

If you are using ASM as your storage technique, then as long as you have two control files, one in each disk group (such as +DATA and +FRA), then you should not require further multiplexing. In a database using Oracle Managed Files (OMF)—such as a database using ASM storage—all additional control files must be created as part of a recovery process using RMAN (or through Enterprise Manager).

In a database using regular file system storage, adding a control file is a manual operation:

1. Alter the SPFILE with the following command specifying the appropriate location of your files:

```
ALTER SYSTEM SET control_files =
  '/u01/app/oracle/oradata/orcl/control01.ctl' ,
  '/u02/app/oracle/oradata/orcl/control02.ctl' ,
  '/u03/app/oracle/oradata/orcl/control03.ctl' SCOPE=SPFILE;
```

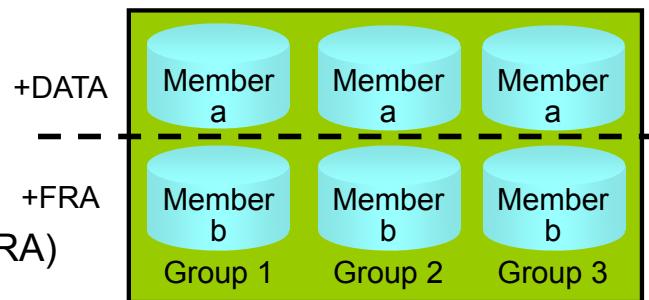
2. Shut down the database instance.
3. Use an operating system command to copy an existing control file to the location you select for your new file.
4. Open the database.

Note: More information about using RMAN is provided in the *Oracle Database 12c: Backup and Recovery Workshop* and in product documentation.

Redo Log Files

Multiplex redo log groups to protect against media failure and loss of data. This increases database I/O. It is suggested that redo log groups have:

- At least two members (files) per group
- Each member:
 - On a separate disk or controller if using file system storage
 - In a separate disk group (such as +DATA and +FRA) if using ASM



Note: Multiplexing redo logs may impact overall database performance.

ORACLE

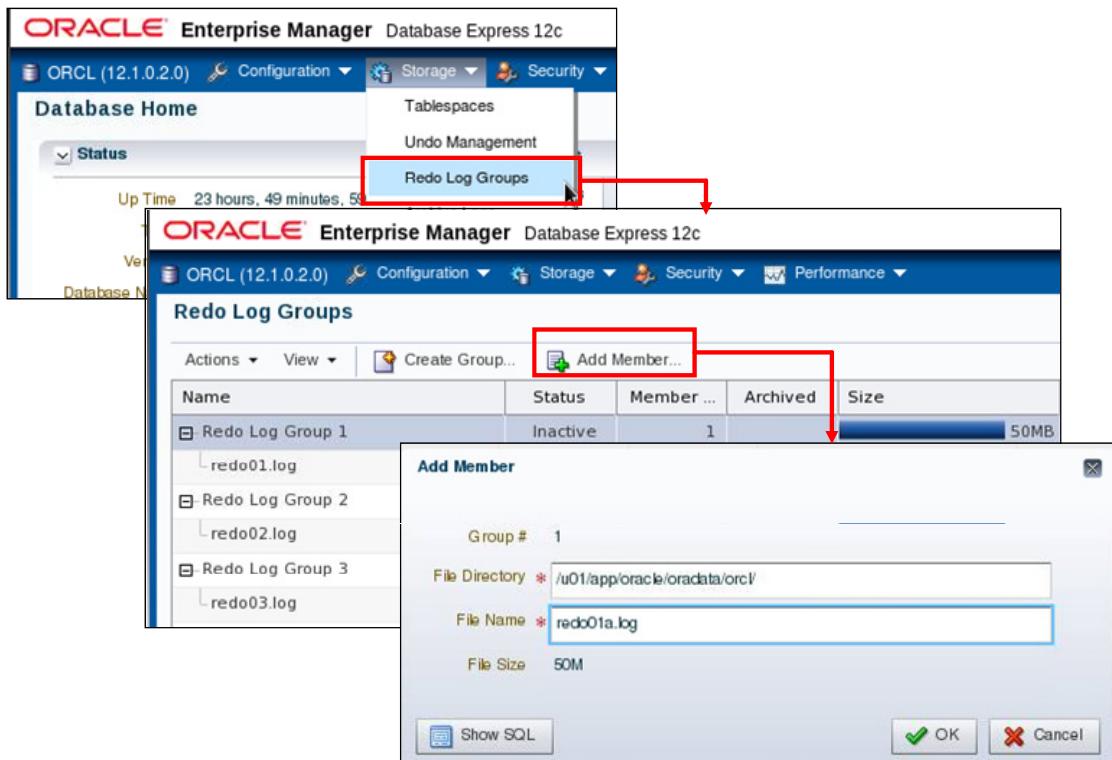
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Redo log groups are made up of one or more redo log files. Each log file in a group is a duplicate of the others. Oracle Corporation recommends that redo log groups have at least two files per group. If using file system storage, then each member should be distributed on separate disks or controllers so that no single equipment failure impacts an entire log group. If you are using ASM storage, then each member should be in a separate disk group, such as +DATA and +FRA.

The loss of an entire current log group is one of the most serious media failures because it can result in loss of data. The loss of a single member of a multiple-member log group is trivial and does not affect database operation (other than causing an alert to be published in the alert log). Recovery from the loss of an entire log group requires advanced recovery techniques and is discussed in the course titled *Oracle Database 12c: Backup and Recovery Workshop*.

Remember that multiplexing redo logs may heavily influence database performance because a commit cannot complete until the transaction information has been written to the logs. You must place your redo log files on your fastest disks served by your fastest controllers. If possible, do not place any other database files on the same disks as your redo log files (unless you are using ASM). Because only one group is written to at a given time, there is no performance impact in having members from several groups on the same disk.

Multiplexing the Redo Log



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can multiplex your redo log by adding a member to an existing log group. To add a member to a redo log group (with open database and no impact on user performance), perform the following steps in Enterprise Manager Database Express:

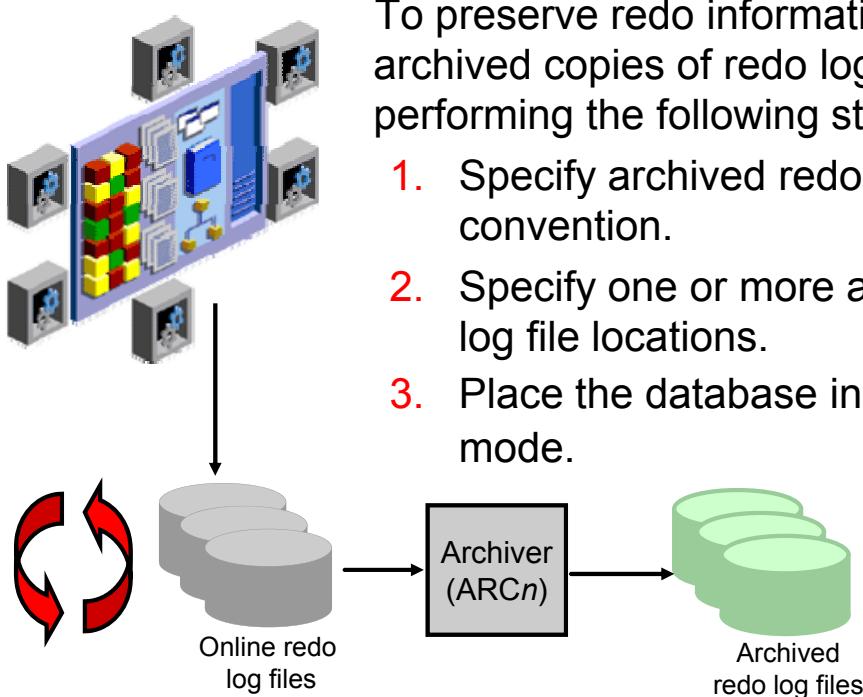
1. Select Storage > Redo Log Groups.
2. Select a group and click Add Member.
3. The Add Member page appears.
4. For File System storage, enter the file name and the file directory. Click OK.

Repeat these steps for every existing group that you want to multiplex. An example showing the SQL syntax of adding a redo log member to redo log group 1 (using ASM) is shown here:

```
SQL> ALTER DATABASE ADD LOGFILE MEMBER '+DATA' TO GROUP 1;
```

When you add the redo log member to a group, the member's status is marked as INVALID (as can be seen in the V\$LOGFILE view). This is the expected state because the new member of the group has not yet been written to. When a log switch occurs and the group containing the new member becomes CURRENT, the member's status changes to null.

Creating Archived Redo Log Files



To preserve redo information, create archived copies of redo log files by performing the following steps:

1. Specify archived redo log file-naming convention.
2. Specify one or more archived redo log file locations.
3. Place the database in ARCHIVELOG mode.

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Oracle Database server treats the online redo log groups as a circular buffer in which to store transaction information, filling one group and then moving on to the next. After all groups have been written to, the Oracle Database server begins overwriting information in the first log group.

To configure your database for maximum recoverability, you must instruct the Oracle Database server to make a copy of the online redo log group before allowing it to be overwritten. These copies are known as *archived redo log files*.

To facilitate the creation of archived redo log files:

1. Specify a naming convention for your archived redo log files.
2. Specify a destination or destinations for storing your archived redo log files.
3. Place the database in ARCHIVELOG mode.

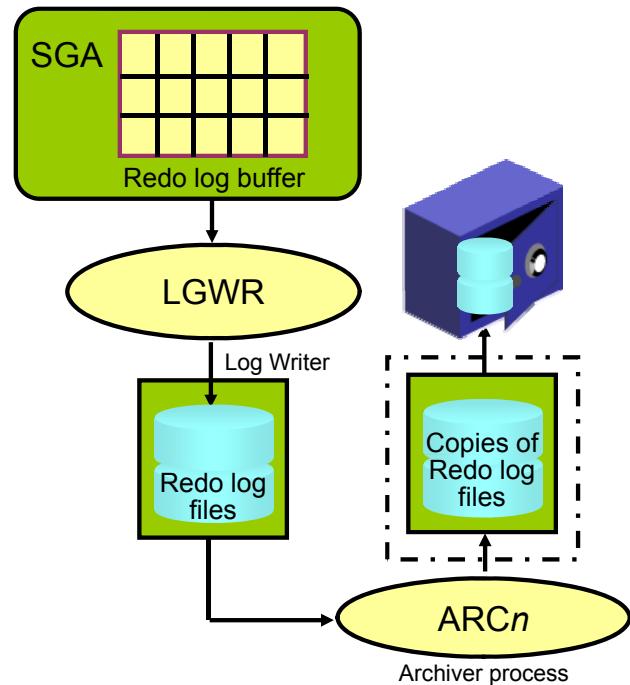
Note: Steps 1 and 2 are not necessary if you are using a fast recovery area.

The destination should exist before placing the database in ARCHIVELOG mode. When a directory is specified as a destination, there should be a slash at the end of the directory name.

Archiver (ARCn) Process

Archiver (ARCn):

- Is an optional background process
- Automatically archives online redo log files when the database is in ARCHIVELOG mode
- Preserves a record of all changes made to the database



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

ARCn is an optional background process. However, it is crucial to the recovery of a database after the loss of a disk. When an online redo log group gets filled, the Oracle Database server begins writing to the next online redo log group. The process of switching from one online redo log group to another is called a *log switch*. The ARCn process initiates archiving of the filled log group at every log switch. It automatically archives the online redo log group before the log group can be reused so that all the changes made to the database are preserved. This enables recovery of the database to the point of failure even if a disk drive is damaged.

One of the important decisions that a DBA must make is whether to configure the database to operate in ARCHIVELOG mode or in NOARCHIVELOG mode.

- In NOARCHIVELOG mode, the online redo log files are overwritten each time a log switch occurs.
- In ARCHIVELOG mode, inactive groups of filled online redo log files must be archived before they can be used again.

Note

- ARCHIVELOG mode is essential for most backup strategies.
- If the archived redo log file destination fills up or cannot be written to, the database will eventually come to a halt. Remove archived redo log files from the archived redo log file destination and the database will resume operations.

Archived Redo Log Files: Naming and Destinations

Specify naming and archive destination information on the Recovery Settings page. If you are using file system storage, it is recommended that you add multiple locations across different disks.

Media Recovery

The database is currently in NOARCHIVELOG mode. In ARCHIVELOG mode, hot backups and recovery to the latest time are possible, but you must provide space for archived redo log files. If you change the database to ARCHIVELOG mode, you should perform a backup immediately. In NOARCHIVELOG mode, only cold backups are possible and data may be lost in the event of database corruption.

ARCHIVELOG Mode*

Log Archive Filename Format*

Number	Archived Redo Log Destination	Status	Type
1	USE_DB_RECOVERY_FILE_DEST	VALID	Local

[Add Another Row](#)

TIP It is recommended that archived redo log files be written to multiple locations spread across the different disks.
 TIP You can specify up to 10 archived redo log destinations.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To configure archived redo log file names and destinations by using Enterprise Manager Cloud Control, select Availability > Backup & Recovery > Recovery Settings.

Each archived redo log file must have a unique name to avoid overwriting older log files. Specify the naming format as shown in the slide. To help create unique file names, Oracle Database allows several wildcard characters in the name format:

- **%s**: Includes the log sequence number as part of the file name
- **%t**: Includes the thread number as part of the file name
- **%r**: Includes the resetlogs ID to ensure that the archive log file name remains unique (even after certain advanced recovery techniques that reset log sequence numbers)
- **%d**: Includes the database ID as part of the file name

The format should include **%s**, **%t**, and **%r** as best practice (**%d** can also be included if multiple databases share the same archive log destination).

By default, if the fast recovery area is enabled, `USE_DB_RECOVERY_FILE_DEST` is specified as an archived redo log file destination. Archived redo log files can be written to as many as 10 different destinations. Destinations may be local (a directory) or remote (an Oracle Net alias for a standby database).

Click Add Another Row to add further destinations. To change recovery settings, you must be connected as SYSDBA, SYSOPER, or SYSBACKUP.

Note: If you do not want archives sent to this location, delete
`USE_DB_RECOVERY_FILE_DEST`.

Configuring ARCHIVELOG Mode

To place the database in ARCHIVELOG mode, perform the following steps:

- Using Enterprise Manager Cloud Control:
 1. On the Recovery Settings page, select “ARCHIVELOG Mode” and click Apply. The database can be set to ARCHIVELOG mode only from the MOUNT state.
 2. Restart the database instance by clicking “Yes” when prompted.
- Using SQL commands:
 - Mount the database.
 - Issue the ALTER DATABASE ARCHIVELOG command.
 - Open the database.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Placing the database in ARCHIVELOG mode prevents redo logs from being overwritten until they have been archived.

In Enterprise Manager Cloud Control, select Availability > Backup & Recovery > Recovery Settings. Select “ARCHIVELOG Mode” and click Apply. The database instance must be restarted after making this change.

To issue the SQL command to put the database in ARCHIVELOG mode, the database must be in MOUNT mode. If the database is currently open, you must shut it down cleanly (not abort), and then mount it as shown in the following example:

```
shutdown immediate
startup mount
alter database archivelog;
alter database open;
```

With the database in NOARCHIVELOG mode (the default), recovery is possible only until the time of the last backup. All transactions made after that backup are lost.

In ARCHIVELOG mode, recovery is possible until the time of the last commit. Most production databases are operated in ARCHIVELOG mode.

Note: Back up your database after switching to ARCHIVELOG mode because your database is recoverable only from the first backup taken in that mode.

Quiz

Which parameters configure the fast recovery area?

- a. FLASH_RECOVERY_AREA_SIZE
- b. DB_RECOVERY_FILE_DEST
- c. FLASH_RECOVERY_AREA_LOC
- d. DB_RECOVERY_FILE_DEST_SIZE



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b, d

Summary

In this lesson, you should have learned how to:

- Configure the fast recovery area
- Multiplex the control file
- Multiplex redo log files
- Configure ARCHIVELOG mode



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice: Overview

This practice covers the following topics:

- Verifying control files
- Configuring a default fast recovery area
- Multiplexing redo log groups
- Placing your database in ARCHIVELOG mode
- Ensuring that redundant archive logs are created



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

14

Performing Database Backups

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

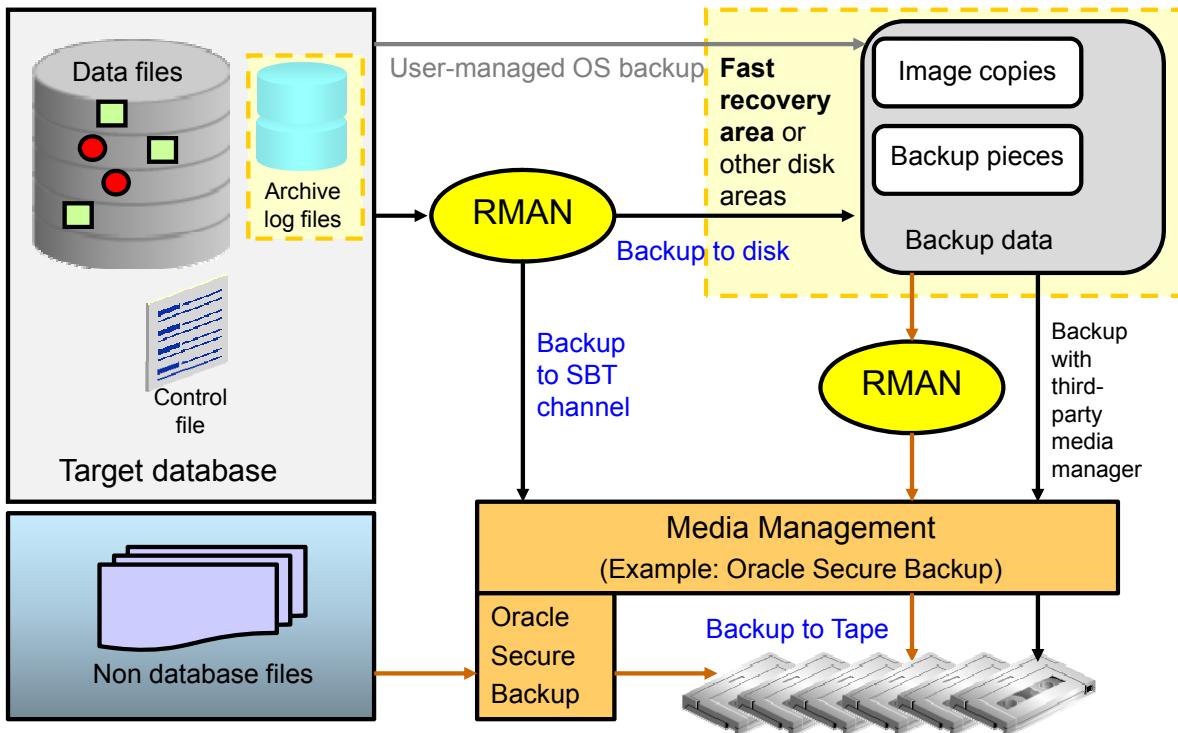
After completing this lesson, you should be able to:

- Create consistent database backups
- Back up your database without shutting it down
- Create incremental backups
- Automate database backups
- Manage backups



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Backup Solutions: Overview



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Recovery Manager (RMAN) is the recommended method of backing up your Oracle database. You can use it to back up to disk or to a system backup to tape (SBT) channel. Oracle recommends that disk backups be stored in the fast recovery area (FRA).

Oracle Secure Backup complements existing functionality by adding backup to tape and backup of file system data. It interacts transparently with RMAN. Third-party media managers can also be used to back up to tape.

User-managed backups are non-RMAN backups, for example, using an OS utility. They are often based on scripts that a DBA must write. This option is being phased out because it is more labor intensive.

Oracle Secure Backup

- Oracle Secure Backup and RMAN provide an end-to-end backup solution for Oracle environments:
 - Centralized tape backup management for file system data and the Oracle database
 - Most well-integrated media management layer for RMAN backups
 - Backup of any data anywhere on the network
- A single technical support resource for the entire backup solution expedites problem resolution.
- This ensures reliable data protection at lower cost and complexity.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle's current backup and recovery product for the database is Recovery Manager. Oracle Secure Backup complements existing functionality in the following ways:

- **Complete backup solution:** Oracle Secure Backup provides data protection for the database and nondatabase data to protect the entire Oracle environment.
- **Media management:** Oracle Secure Backup provides the media management layer for RMAN database backups to tape. Before Oracle Secure Backup, customers had to purchase expensive third-party media management products offering integration with RMAN tape backups.
- **Backup anywhere on the network:** Oracle Secure Backup backs up data from multiple network-attached computer systems to tertiary storage resources on the network. Oracle Secure Backup supports diverse configurations of servers, clients, Network Attached Storage (NAS) servers, and tertiary storage devices and protects network storage environments.

The combination of RMAN and Oracle Secure Backup provides an end-to-end backup solution that is entirely within the Oracle product stack. This solution makes better customer support possible because Oracle Corporation is responsible for the entire backup solution.

User-Managed Backup

A user-managed scenario:

- Is a manual process of tracking backup needs and status
- Typically uses your own written scripts
- Requires that database files be put in the correct mode for backup
- Relies on operating system commands to make backups of files



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

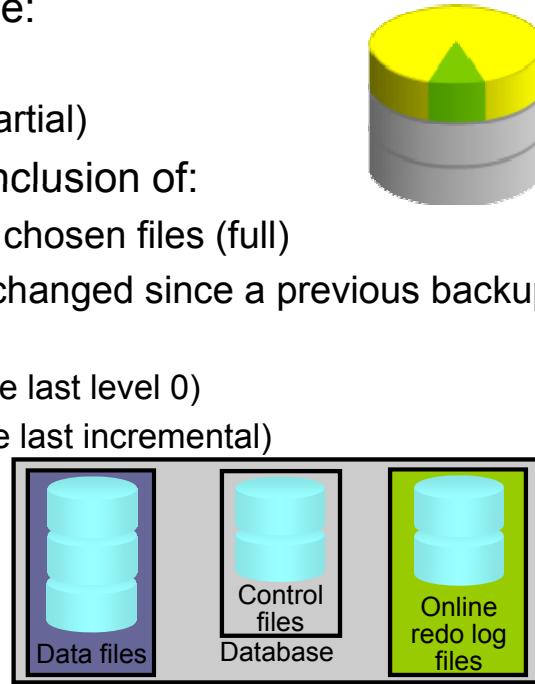
A user-managed backup can be performed interactively. However, most often it entails the writing of scripts to perform the backup. There are several scenarios that can be run, and scripts must be written to handle them.

Some of the actions that scripts must take:

- Querying V\$DATAFILE to determine the data files that need to be backed up and their current state
- Querying V\$LOGFILE to identify the online redo log files
- Querying V\$CONTROLFILE to identify the control file to back up
- Placing each tablespace in online backup mode
- Querying V\$BACKUP to see what data files are part of a tablespace that has been placed in online backup mode
- Issuing operating system copy commands to copy the data files to the backup location
- Bringing each tablespace out of online backup mode

Understanding Backup Terminology

- *Backup strategy* may include:
 - Entire database (whole)
 - Portion of the database (partial)
- *Backup type* may indicate inclusion of:
 - All data blocks within your chosen files (full)
 - Only information that has changed since a previous backup (incremental)
 - Cumulative (changes since last level 0)
 - Differential (changes since last incremental)
- *Backup mode* may be:
 - Offline (consistent, cold)
 - Online (inconsistent, hot)



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Whole database backup: Includes all data files and at least one control file (Remember that all control files in a database are identical.)

Partial database backup: May include zero or more tablespaces and zero or more data files; may or may not include a control file

Full backup: Makes a copy of each data block that contains data and that is within the files being backed up

Incremental backup: Makes a copy of all data blocks that have changed since a previous backup. Oracle Database supports two levels of incremental backup (0 and 1). A level 1 incremental backup can be one of two types: *cumulative* or *differential*. A cumulative backup backs up all changes since the last level 0 backup. A differential backup backs up all changes since the last incremental backup (which could be either a level 0 or level 1 backup). Change Tracking with RMAN supports incremental backups.

Offline backups (also known as “cold” or *consistent* backup): Are taken while the database is not open. They are consistent because, at the time of the backup, the system change number (SCN) in data file headers matches the SCN in the control files.

Online backups (also known as “hot” or *inconsistent* backup): Are taken while the database is open. They are inconsistent because, with the database open, there is no guarantee that the data files are synchronized with the control files.

Understanding Types of Backups

Backups may be stored as:

- Image copies
- Backup sets

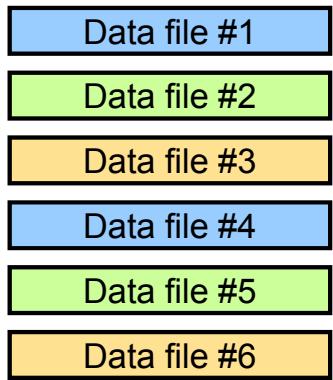
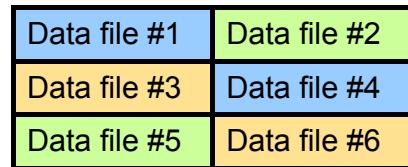


Image copies
(Duplicate data and log files in OS format)



Backup set
(Binary, compressed files in Oracle proprietary format)

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Image copies: Are duplicates of data or archived log files (similar to simply copying the files by using operating system commands)

Backup sets: Are collections of one or more binary files that contain one or more data files, control files, server parameter files, or archived log files. With backup sets, empty data blocks are not stored, thereby causing backup sets to use less space on the disk or tape. Backup sets can be compressed to further reduce the space requirements of the backup.

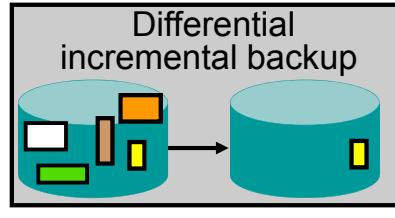
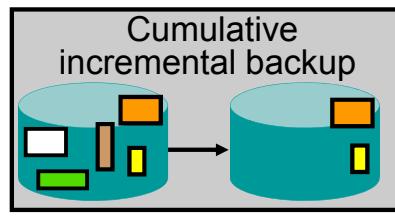
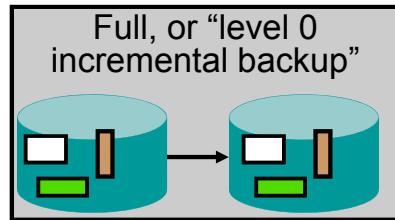
Image copies must be backed up to the disk. Backup sets can be sent to the disk or directly to the tape.

The advantage of creating a backup as an image copy is improved granularity of the restore operation. With an image copy, only the file or files need to be retrieved from your backup location. With backup sets, the entire backup set must be retrieved from your backup location before you extract the file or files that are needed.

The advantage of creating backups as backup sets is better space usage. In most databases, 20% or more of the data blocks are empty blocks. Image copies back up every data block, even if the data block is empty. Backup sets significantly reduce the space required by the backup. In most systems, the advantages of backup sets outweigh the advantages of image copies.

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 © 2014, Oracle and/or its affiliates. All rights reserved.

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 those data file blocks that are not part of an existing segment. 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 (as well as an image copy) 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 using the `INCREMENTAL` keyword of the `BACKUP` command. You specify `INCREMENTAL LEVEL [0 | 1]`.

RMAN can create multilevel incremental backups as follows:

- **Differential:** Is 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.

Note: It is possible to perform any type of backup (full or incremental) of a database that is in NOARCHIVELOG mode—if, of course, the database is not open. Note also that recovery is limited to the time of the last backup. The database can be recovered to the last committed transaction only when the database is in ARCHIVELOG mode.

Using Recovery Manager (RMAN)

- Provides a powerful control and scripting language
- Includes a published API that enables interface with most popular backup software
- Backs up data, control, archived redo log, and server parameter files
- Backs up files to disk or tape
- Is integrated with Enterprise Manager Cloud Control



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

RMAN is the component of the Oracle Database server that is used to perform backup and recovery operations. It can be used to make consistent and inconsistent backups, perform incremental and full backups, and back up either the whole database or a portion of it.

RMAN uses its own powerful job control and scripting language, as well as a published API that interfaces RMAN with many popular backup software solutions.

RMAN can store backups on the disk for quick recovery or place them on the tape for long-term storage. For RMAN to store backups on the tape, you must either use Oracle Secure Backup or configure an interface to the tape device known as a media management library (MML).

Enterprise Manager Cloud Control provides a graphical interface to the most commonly used RMAN functionality. Advanced backup and recovery operations are accessible through RMAN's command-line client. For more information about advanced RMAN capabilities, see the course titled *Oracle Database 12c: Backup and Recovery Workshop* or consult the *Oracle Backup and Recovery User's Guide*.

Configuring Backup Settings

The screenshot shows two stacked configuration panels for backup settings.

Top Panel (Disk Settings):

- Device Tab:** Selected tab.
- Parallelism:** Set to 1. Description: Concurrent streams to disk drives.
- Disk Backup Location:** A note states: "The fast recovery area is the current disk backup location. If you would like to override the disk backup location, specify an existing directory or diskgroup."
- Disk Backup Type:** Radio buttons for:
 - Backup Set:** An Oracle backup file format that allows for more efficient backups by interleaving multiple backup files into one output file.
 - Compressed Backup Set:** An Oracle backup set in which the data is compressed to reduce its size.
 - Image Copy:** A bit-by-bit copy of database files that can be used as-is to perform restores.

Bottom Panel (Backup Set):

- Backup Set Tab:** Selected tab.
- Maximum Backup Piece (File) Size:** Input field with dropdown menu set to MB. Description: "Specify a value to restrict the size of each backup piece."
- Compression Algorithm:** A note: "Specify the compression algorithm that will be used for both disk and tape compressed backup sets." Includes:
 - Algorithm Name:** BASIC (selected).
 - Release:** DEFAULT (selected). Description: "The algorithm will be configured as of the specified release. The choices are DEFAULT and a specific Selecting DEFAULT will accommodate changes in algorithm definitions after a database upgrade, without changing the definition as of that particular database version."
 - Optimize For Load:** A checked checkbox. Description: "Controls pre-compression processing. Enabling optimizes CPU usage and avoids pre-compression block processing. Disabling uses additional CPU resources."

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

By using RMAN commands or the Enterprise Manager Cloud Control interface, you can manage the persistent backup settings that are used for creating backups. In Enterprise Manager Cloud Control, select Availability > Backup & Recovery > Backup Settings. There are separate settings for disk and tape. Tape settings depend on the media management library capabilities. Disk settings include:

- **Parallelism:** How many separate streams of backup information do you want to create? The best setting for parallelism depends on your hardware. As hardware resources increase, the appropriate degree of parallelism also increases. Generally, you want to set your parallelism to the number of disks that your disk backup location is striped over. For tape backup, you want to set your parallelism to the same number of tape drives that you have.
- **Disk backup location:** Where should backups be stored? The default is the fast recovery area. If you change this, click Test Disk Backup to verify that RMAN can write to the new location.
- **Disk backup type:** Select Backup Set, Compressed Backup Set, or Image Copy.

Click the Backup Set tab to set the maximum file size of backup pieces, specify the compression algorithm to be used for compressed backup sets, and specify redundancy for tape backups. Host credentials are required to save changes to the backup settings.

Configuring Backup Settings

Backup Settings

Policy

Backup Policy

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

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 backed up to the fast 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

Tablespaces Excluded From Whole Database Backup
Add the tablespaces you want to exclude from a whole database backup.

Select	Tablespace Name	Tablespace Number	Status	Contents
	No Items Selected			

TIP These tablespaces can be backed up separately using tablespace backup.

Retention Policy

Retain All Backups You must manually delete any backups

Retain backups that are necessary for a recovery to any time within the specified number of days (point-in-time recovery)
Days: 31 Recovery Window

Retain at least the specified number of full backups for each datafile
Backups: 1 Redundancy

Archived Redo Log Deletion Policy
Specify the deletion policy for archived redo log files. The archived redo log files will be eligible for deletion if the fast recovery area becomes full.

None

ORACLE

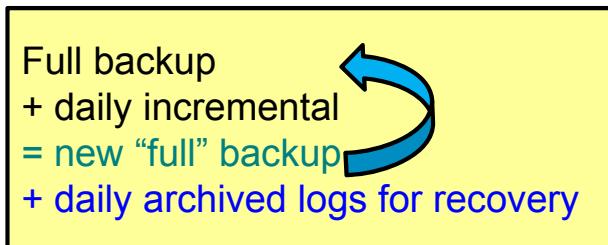
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Click the Policy tab to:

- Automatically back up the control file and server parameter file (SPFILE) with each backup. You can also specify a location for these backups if you do not want them to go to the fast recovery area.
- Optimize backups by not backing up files that exactly match a file that is already part of the retained backups. This setting enables you to skip read-only and offline data files.
- Enable block change tracking and specify a location for the tracking file. If you intend to create incremental backups, this setting can decrease the time required to choose which blocks to include in the incremental backup.
- Exclude tablespaces from a whole database backup. Some administrators choose not to back up tablespaces containing data or objects that can be easily re-created (such as indexes or data that is batch-loaded frequently).
- Specify a retention policy: How long should RMAN keep your backups? If you are using the fast recovery area to store backups, RMAN automatically deletes old backups to make room for new ones (if the retention policy allows it). By default, only the last backup is retained. The retention policy can be specified as a number of backups or a number of days.

Oracle-Suggested Backup

- Provides an out-of-the-box backup strategy based on the backup destination
- Sets up a recovery window for backup management
- Schedules recurring and immediate backups:



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Enterprise Manager Cloud Control makes it easy for you to set up an Oracle-suggested backup strategy that protects your data and provides efficient recoverability to any point in the preceding 24 hours, and possibly as far back as 48 hours, depending on when the last backup was created. The Oracle-suggested strategy uses the incremental backup and incrementally updated backup features, providing faster recoverability than is possible when applying database changes from the archived redo log files.

Because these backups on disk are retained, you can always perform a full database recovery or a point-in-time recovery to any time within the past 24 hours, at the minimum. The recovery time could reach back as far as 48 hours. This is because just before a backup is taken on a given day, the backup from the beginning of day n-1 still exists.

Selecting a Backup Strategy

The screenshot shows the Oracle Enterprise Manager Cloud Control 12c interface. In the top left, it says "ORACLE Enterprise Manager Cloud Control 12c". Below that, a "Schedule Backup" section is shown. It contains a sub-section titled "Oracle-Suggested Backup" which says: "Schedule a backup using Oracle's automated backup strategy." To the right of this is a button labeled "Schedule Oracle-Suggested Backup". Below this, it says: "This option will back up the entire database. The database will be backed up on daily and weekly intervals." To the right of this, a callout box titled "Backup Strategies" provides details about both Oracle-suggested and customized backup strategies.

Oracle-Suggested Backup

Schedule a backup using Oracle's automated backup strategy.

This option will back up the entire database. The database will be backed up on daily and weekly intervals.

Customized Backup

Select the object(s) you want to back up.

Whole Database
 Tablespaces
 Datafiles
 Archived Logs
 All Recovery Files on Disk

Includes all archived logs and disk backups that are not already backed up.

Backup Strategies

Oracle-suggested:

- Provides an out-of-the-box backup strategy based on the backup destination
- Sets up recovery window for backup management
- Schedules recurring and immediate backups
- Automates backup management

Customized:

- Specify the objects to be backed up
- Choose disk or tape backup destination
- Override the default backup settings
- Schedule the backup

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

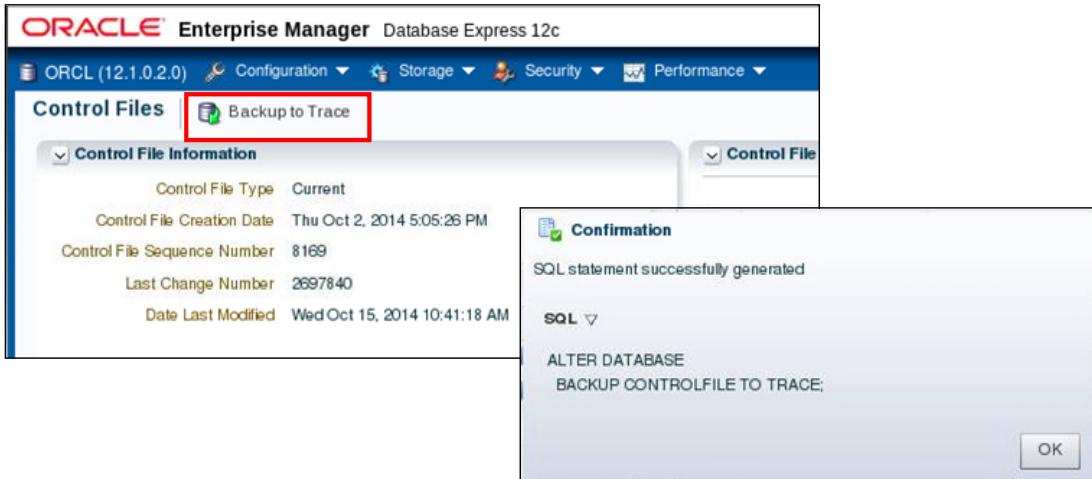
To establish an Oracle-suggested strategy, navigate to the database home page > Availability > Backup & Recovery > Schedule Backup. The Backup Strategies section enables you to select from the Oracle-suggested backup and Customized backup strategies. The Oracle-suggested strategy takes a full database copy as the first backup. Because it is a whole database backup, you might want to consider taking this at the period of least activity. After that, an incremental backup to disk is taken every day. Optionally, a weekly tape backup can be made, which backs up all recovery-related files.

By clicking Schedule Customized Backup, you gain access to a wider range of configuration options. Select the objects that you want to back up—the whole database (the default) or individual tablespaces, data files, archived logs, or any Oracle backups currently residing on the disk (to move them to the tape).

Both strategies enable you to set up encrypted backups.

Backing Up the Control File to a Trace File

- Control files can be backed up to a trace file, generating a SQL command to re-create the control file.
- Control file trace backups may be used to recover from loss of all control files.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

ORACLE

In Enterprise Manager Database Express, select Storage > Control Files to manage your database's control files. Control files have an additional backup option; they may be backed up to a trace file. A control file trace backup contains the SQL statement that is required to re-create the control files in the event that all control files are lost.

Although it is very unlikely that a properly configured database (with multiple copies of the control file placed on separate disks and separate controllers) would lose all control files at the same time, it is possible. Therefore, you should back up the control file to a trace file after each change to the physical structure of the database (adding tablespaces or data files, or adding additional redo log groups).

Trace copies of the control file can be created by using Enterprise Manager Database Express, Enterprise Manager Cloud Control, or by using the following SQL command:

```
ALTER DATABASE BACKUP CONTROLFILE TO TRACE
```

The trace backup is created in the location specified by the `DIAGNOSTIC_DEST` initialization parameter. For example, in this course, the trace file for the `orcl` database is found in the `/u01/app/oracle/diag/rdbms/orcl/orcl/trace` directory and will have a file name such as `orcl_ora_924.trc`.

Managing Backups

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

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"/>	5	ORA\$OEM_LEVEL_0	Jan 4, 2013 2:04:06 AM	SPFILE, CONTROLFILE	DISK	AVAILABLE	NO	1
<input type="checkbox"/>	4	BACKUP_ORCL_000021_010313095405	Jan 3, 2013 9:57:35 AM	CONTROLFILE	DISK	AVAILABLE	NO	1
<input type="checkbox"/>	3	BACKUP_ORCL_000021_010313095405	Jan 3, 2013 9:57:30 AM	ARCHIVED LOG	DISK	AVAILABLE	NO	1
<input type="checkbox"/>	2	BACKUP_ORCL_000021_010313095405	Jan 3, 2013 9:57:20 AM	SPFILE, CONTROLFILE	DISK	AVAILABLE	NO	1
<input type="checkbox"/>	1	BACKUP_ORCL_000021_010313095405	Jan 3, 2013 9:57:05 AM	DATAFILE	DISK	AVAILABLE	NO	1

ORACLE®

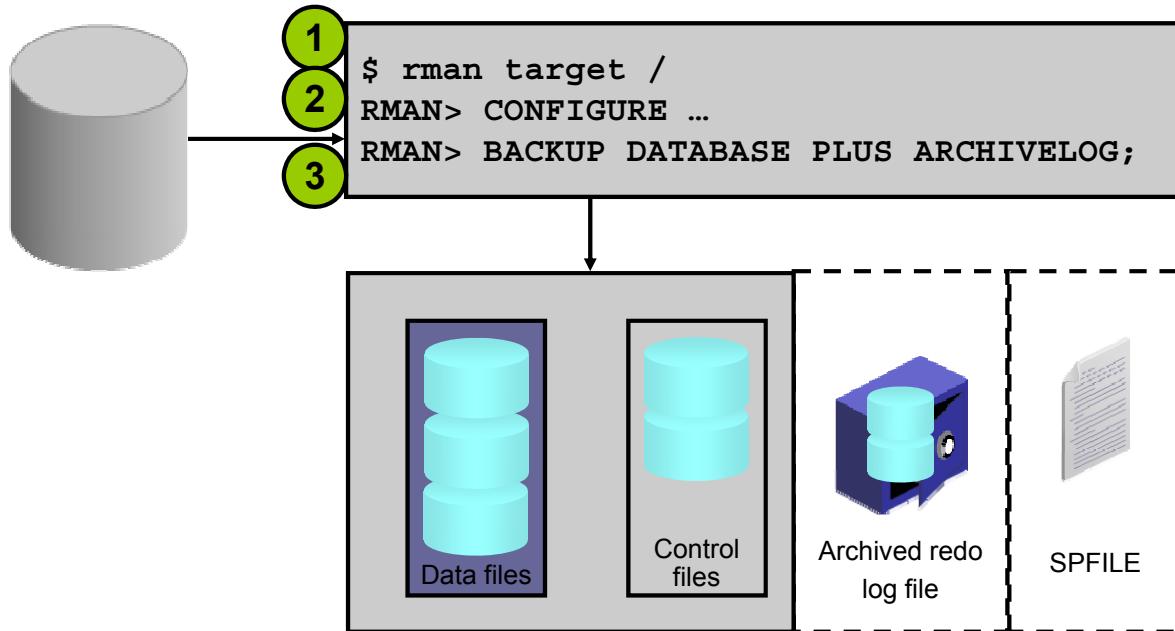
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In Enterprise Manager Cloud Control, select Availability > Backup & Recovery > Manage Current Backups to manage your existing backups. On this page, you can see when a backup was completed, where it was created (disk or tape), and whether it is still available.

At the top of the Manage Current Backups page, four buttons enable you to work with existing backups:

- **Catalog Additional Files:** Although RMAN (working through Enterprise Manager) is the recommended way to create backups, you might have image copies or backup sets that were created by some other means or in some other environment with the result that RMAN is not aware of them. This task identifies those files and adds them to the catalog.
- **Crosscheck All:** RMAN can automatically delete obsolete backups, but you can also delete them by using operating system commands. If you delete a backup without using RMAN, the catalog does not know whether the backup is missing until you perform a cross-check between the catalog and what is really there.
- **Delete All Obsolete:** This deletes backups older than the retention policy.
- **Delete All Expired:** This deletes the catalog listing for any backups that are not found when the cross-check is performed as described previously.

Using RMAN Commands to Create Backups



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

1. In a terminal session, start RMAN and connect to the target database.
2. Execute configuration commands:
 - CONFIGURE DEFAULT DEVICE TYPE TO disk;
 - CONFIGURE DEVICE TYPE DISK BACKUP TYPE TO COPY;
 - CONFIGURE CONTROLFILE AUTOBACKUP ON;
3. A whole database backup is a copy of all data files and the control file. You can optionally include the server parameter file (SPFILE) and archived redo log files. Using RMAN to make an image copy of all the database files simply requires mounting or opening the database, starting RMAN, and entering the BACKUP command shown in the slide.

Optionally, you can supply the `DELETE INPUT` option when backing up archive log files. That causes RMAN to remove the archive log files after backing them up. This is useful especially if you are not using a fast recovery area, which would perform space management for you, deleting files when space pressure grows. In that case, the command in the slide would look like the following:

```
RMAN> BACKUP DATABASE PLUS ARCHIVELOG DELETE INPUT;
```

You can also create a backup (either a backup set 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;
```

Quiz

Using the change-tracking feature, an image copy backup performed by RMAN can skip blocks that have not changed since the last backup.

- a. True
- b. False



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b

Summary

In this lesson, you should have learned how to:

- Create consistent database backups
- Back up your database without shutting it down
- Create incremental backups
- Automate database backups
- Manage backups and view backup reports



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice: Overview

This practice covers the following topics:

- Backing up your database while the database is open for user activity
- Scheduling automatic nightly incremental backups for your database



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

15

Performing Database Recovery

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

- Determine the need for performing recovery
- Describe and use available options, such as Recovery Manager (RMAN) and the Data Recovery Advisor
- Perform recovery:
 - Control file
 - Redo log file
 - Data file

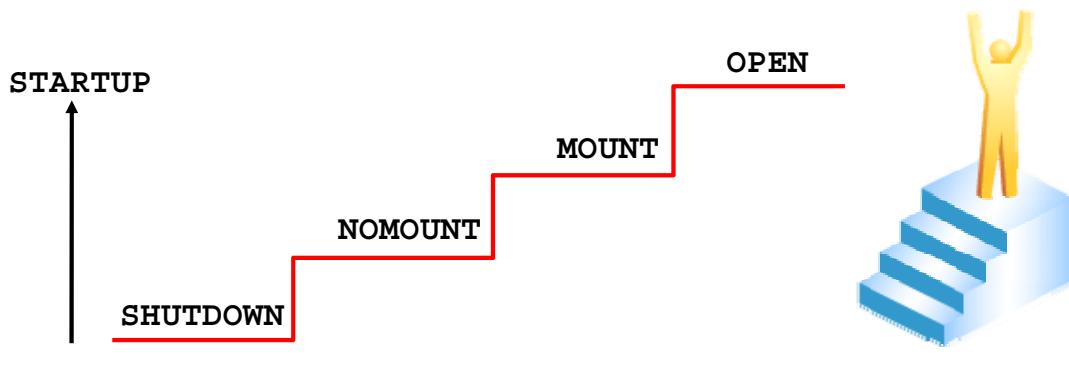


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Opening a Database

To open a database:

- All control files must be present and synchronized
- All online data files must be present and synchronized
- At least one member of each redo log group must be present



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

As a database moves from the shutdown stage to being fully open, it performs internal consistency checks with the following stages:

- **NOMOUNT:** For an instance to reach the NOMOUNT (also known as STARTED) status, the instance must read the initialization parameter file. No database files are checked while the instance enters the NOMOUNT state.
- **MOUNT:** As the instance moves to the MOUNT status, it checks whether all control files listed in the initialization parameter file are present and synchronized. If even one control file is missing or corrupt, the instance returns an error (noting the missing control file) to the administrator and remains in the NOMOUNT state.
- **OPEN:** When the instance moves from the MOUNT state to the OPEN state, it does the following:
 - Checks whether all redo log groups known to the control file have at least one member present. Any missing members are noted in the alert log.

- Verifies that all data files known to the control file are present unless they have been taken offline. Offline files are not checked until the administrator tries to bring them online. The administrator may take a data file offline and open the instance if the data file does not belong to the SYSTEM or UNDO tablespaces. If any files are missing, an error noting the first missing file is returned to the administrator and the instance remains in the MOUNT state. When the instance finds files that are missing, only the first file causing a problem appears in the error message. To find all files that need recovery, the administrator can check the v\$recover_file dynamic performance view to get a complete list of the files that need attention:

```

SQL> startup
ORACLE instance started.

Total System Global Area  171966464 bytes
Fixed Size                  775608 bytes
Variable Size                145762888 bytes
Database Buffers            25165824 bytes
Redo Buffers                 262144 bytes

Database mounted.

ORA-01157: cannot identify/lock data file 4 - see DBWR trace file
ORA-01110: data file 4: '/oracle/oradata/orcl/users01.dbf'

SQL> SELECT name, error
      2  FROM v$datafile
      3  JOIN v$recover_file
      4  USING (file#);

NAME                      ERROR
-----
/oracle/oradata/orcl/users01.dbf    FILE NOT FOUND
/oracle/oradata/orcl/example01.dbf  FILE NOT FOUND

```

- Verifies that all data files that are not offline or read-only are synchronized with the control file. If necessary, instance recovery is automatically performed. However, if a file is out of synchronization to the extent that it cannot be recovered by using the online redo log groups, then the administrator must perform media recovery. If any files require media recovery, an error message noting the first file requiring recovery is returned to the administrator and the instance remains in the MOUNT state:

```

ORA-01113: file 4 needs media recovery
ORA-01110: data file 4: '/oracle/oradata/orcl/users01.dbf'

```

Again, v\$recover_file gives a complete list of files that need attention. Files that are present and require media recovery are listed, but no error message is displayed.

Keeping a Database Open

After the database is open, it fails in case of the loss of:

- Any control file
- A data file belonging to the system or undo tablespaces
- An entire redo log group (As long as at least one member of the group is available, the instance remains open.)



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

After a database is open, instance failure can be caused by media failure: for example, by the loss of a control file, the loss of an entire redo log group, or the loss of a data file belonging to the SYSTEM or UNDO tablespaces. Even if an inactive redo log group is lost, the database would eventually fail due to log switches.

In many cases, the failed instance does not completely shut down but is unable to continue to perform work. Recovering from these types of media failure must be done with the database down. As a result, the administrator must use the SHUTDOWN ABORT command before beginning recovery efforts.

The loss of data files belonging to other tablespaces does not cause instance failure, and the database can be recovered while open, with work continuing in other tablespaces.

These errors can be detected by inspecting the alert log file or by using the Data Recovery Advisor.

Data Recovery Advisor

- Fast detection, analysis, and repair of failures
- Handling of down-time and runtime failures
- Minimizing disruptions for users
- User interfaces:
 - Enterprise Manager Cloud Control
 - RMAN command line



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Data Recovery Advisor automatically gathers data failure information when an error is encountered. In addition, it can proactively check for failures. In this mode, it can potentially detect and analyze data failures before a database process discovers the corruption and signals an error. (Note that repairs are always under human control.)

Data failures can be very serious. For example, if your current log files are missing, you cannot open your database. Some data failures (like block corruptions in data files) are not catastrophic because they do not take the database down or prevent you from opening the Oracle database. The Data Recovery Advisor handles both cases: the one when you cannot start up the database (because required database files are missing, inconsistent, or corrupted) and the one when file corruptions are discovered during run time.

The preferred way to address serious data failures is as follows:

1. Fail over to a standby database if you are in a Data Guard configuration. This allows users to come back online as soon as possible.
2. Repair the primary cause of the data failure.

User Interfaces

The Data Recovery Advisor is available in Enterprise Manager Cloud Control. When failures exist, there are several ways to access the Data Recovery Advisor.

You can also use the Data Recovery Advisor by using the RMAN command line:

```
rman target /  
rman> list failure all;
```

Supported Database Configurations

In the current release, the Data Recovery Advisor supports single-instance databases. Oracle Real Application Clusters databases are not supported.

The Data Recovery Advisor cannot use blocks or files transferred from a standby database to repair failures on a primary database. Furthermore, you cannot use the Data Recovery Advisor to diagnose and repair failures on a standby database. However, the Data Recovery Advisor does support failover to a standby database as a repair option (as mentioned previously).

Loss of a Control File

If a control file is lost or corrupted, the instance normally aborts.

- If control files are stored in ASM disk groups, recovery options are as follows:
 - Perform guided recovery using Enterprise Manager.
 - Put database in NOMOUNT mode and use an RMAN command to restore control file from existing control file.

```
RMAN> restore controlfile from
      '+DATA/orcl/controlfile/current.260.695209463' ;
```

- If control files are stored as regular file system files, then:
 - Shut down the database
 - Copy existing control file to replace lost control file

After control file is successfully restored, open the database.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The options for recovery from the loss of a control file depend on the storage configuration of the control files and on whether at least one control file remains or have all been lost.

If using ASM storage, and at least one control file copy remains, you can perform guided recovery using Enterprise Manager or perform manual recovery using RMAN as follows:

1. Put the database in NOMOUNT mode.
2. Connect to RMAN and issue the RESTORE CONTROLFILE command to restore the control file from an existing control file, for example:

```
restore controlfile from
      '+DATA/orcl/controlfile/current.260.695209463' ;
```
3. After the control file is successfully restored, open the database.

If your control files are stored as regular file system files and at least one control file copy remains, then, while the database is down, you can just copy one of the remaining control files to the missing file's location. If the media failure is due to the loss of a disk drive or controller, copy one of the remaining control files to some other location and update the instance's parameter file to point to the new location. Alternatively, you can delete the reference to the missing control file from the initialization parameter file. Remember that Oracle recommends having at least two control files at all times.

Note: Recovering from the loss of all control files is covered in the course titled *Oracle Database 12c: Backup and Recovery Workshop*.

Loss of a Redo Log File

If a member of a redo log file group is lost and if the group still has at least one member, note the following results:

- Normal operation of the instance is not affected.
- You receive a message in the alert log notifying you that a member cannot be found.
- You can restore the missing log file by dropping the lost redo log member and adding a new member.
- If the group with the missing log file has been archived, you can clear the log group to re-create the missing file.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Recovering from the loss of a single redo log group member should not affect the running instance.

To perform this recovery by using SQL commands:

1. Determine whether there is a missing log file by examining the alert log.
2. Restore the missing file by first dropping the lost redo log member:

```
ALTER DATABASE DROP LOGFILE MEMBER '<filename>'
```

Then add a new member to replace the lost redo log member:

```
ALTER DATABASE ADD LOGFILE MEMBER '<filename>'  
TO GROUP <integer>
```

Note: If you are using Oracle Managed Files (OMF) for your redo log files and you use the preceding syntax to add a new redo log member to an existing group, that new redo log member file will not be an OMF file. If you want to ensure that the new redo log member is an OMF file, then the easiest recovery option would be to create a new redo log group and then drop the redo log group that had the missing redo log member.

3. If the media failure is due to the loss of a disk drive or controller, rename the missing file.
4. If the group has already been archived, or if you are in NOARCHIVELOG mode, you may choose to solve the problem by clearing the log group to re-create the missing file or files. You can clear the affected group manually with the following command:

```
ALTER DATABASE CLEAR LOGFILE GROUP <integer>;
```

Note: Enterprise Manager does not allow you to clear a log group that has not been archived. Doing so breaks the chain of redo information. If you must clear an unarchived log group, you should *immediately* take a full backup of the whole database. Failure to do so may result in a loss of data if another failure occurs. To clear an unarchived log group, use the following command:

```
ALTER DATABASE CLEAR UNARCHIVED LOGFILE GROUP <integer>
```

Loss of a Data File in NOARCHIVELOG Mode

If the database is in NOARCHIVELOG mode and if any data file is lost, perform the following tasks:

1. Shut down the instance if it is not already down.
2. Restore the entire database—including all data and control files—from the backup.
3. Open the database.
4. Have users re-enter all changes that were made since the last backup.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The loss of *any* data file from a database in NOARCHIVELOG mode requires complete restoration of the database, including control files and all data files.

With the database in NOARCHIVELOG mode, recovery is possible only up to the time of the last backup. So users must re-enter all changes made since that backup.

To perform this type of recovery by using Enterprise Manager Cloud Control:

1. Shut down the instance if it is not already down.
2. Select Availability > Backup & Recovery > Perform Recovery.
3. Select Whole Database as the type of recovery.

If you have a database in NOARCHIVELOG mode that has an incremental backup strategy, RMAN first restores the most recent level 0 and then RMAN recovery applies the incremental backups.

Loss of a Noncritical Data File in ARCHIVELOG Mode

If a data file is lost or corrupted, and if that file does not belong to the SYSTEM or UNDO tablespace, you restore and recover the missing data file.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

With the database in ARCHIVELOG mode, the loss of any data file not belonging to the SYSTEM or UNDO tablespaces affects only the objects that are in the missing file. The rest of the database remains available for users to continue work.

To restore and recover the missing data file by using Enterprise Manager Cloud Control, use the Data Recovery Advisor or perform the following steps:

1. Select Availability > Backup & Recovery > Perform Recovery.
2. In the User Directed Recovery section, select Datafiles in the Recovery Scope menu and then select “Recover to current time.”
3. Click Recover to begin the guided restore and recovery process.
4. Click Add to select the data files to restore and recover.
5. Specify whether you want to restore the files to the default location or (if a disk or controller is missing) to a new location.
6. Submit the RMAN job to restore and recover the missing files.

Because the database is in ARCHIVELOG mode, recovery is possible up to the time of the last commit and users are not required to re-enter any data.

Loss of a System-Critical Data File in ARCHIVELOG Mode

If a data file is lost or corrupted, and if that file belongs to the SYSTEM or UNDO tablespace, perform the following tasks:

- 1.** The instance may or may not shut down automatically. If it does not, use SHUTDOWN ABORT to bring the instance down.
- 2.** Mount the database.
- 3.** Restore and recover the missing data file.
- 4.** Open the database.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Data files belonging to the SYSTEM tablespace or containing UNDO data are considered system critical. A loss of one of these files requires the database to be restored from the MOUNT state (unlike other data files that may be restored with the database open).

To perform this recovery by using Enterprise Manager Cloud Control:

1. If the instance is not already shut down, shut it down.
2. Mount the database.
3. Select Availability > Backup & Recovery > Perform Recovery.
4. In the User Directed Recovery Section, select Datafiles in the Recovery Scope menu and then select “Recover to current time.”
5. Click Add to select all data files that need recovery.
6. Specify whether you want to restore the files to the default location or (if a disk or controller is missing) to a new location.
7. Submit the RMAN job to restore and recover the missing files.
8. Open the database. Users are not required to re-enter data because the recovery is up to the time of the last commit.

Quiz

An Oracle Database instance will not fail if the following event occurs:

- a. Loss of a control file if there is a remaining multiplexed control file
- b. Loss of the SYSTEM tablespace
- c. Loss of one redo log member if there is a remaining multiplexed redo log member from the same group of the lost member
- d. Loss of the active undo tablespace



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: c

Summary

In this lesson, you should have learned how to:

- Determine the need for performing recovery
- Describe and use available options, such as Recovery Manager (RMAN) and the Data Recovery Advisor
- Perform recovery:
 - Control file
 - Redo log file
 - Data file



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice: Overview

This practice covers recovering from the loss of a:

- Control file
- Noncritical data file
- System-critical data file



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

16

Moving Data

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

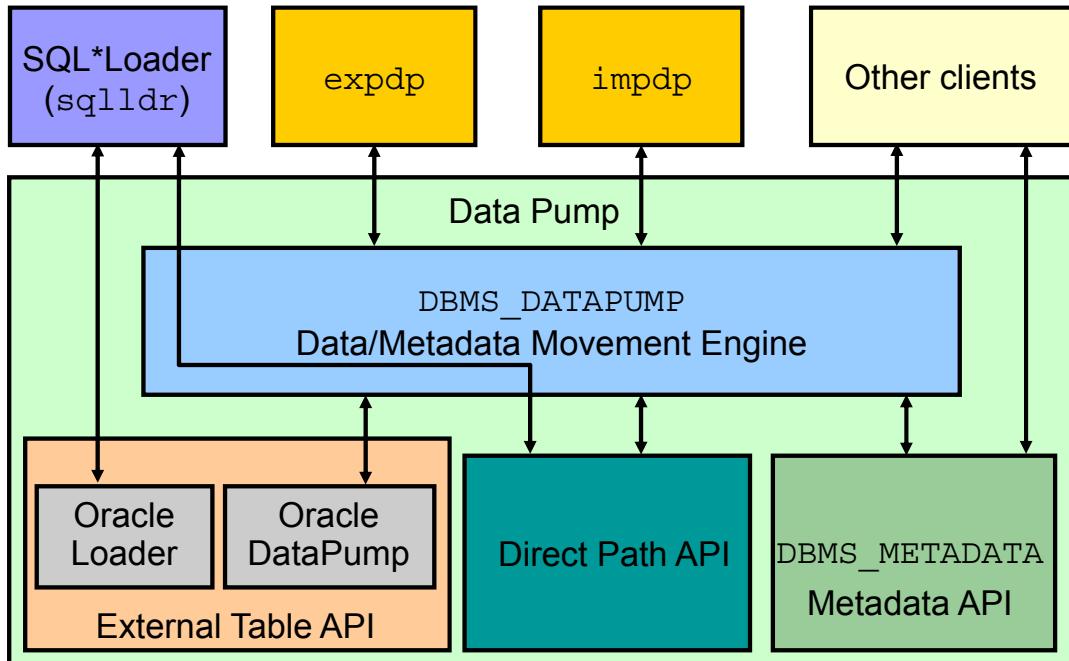
After completing this lesson, you should be able to:

- Describe ways to move data
- Explain the general architecture of Oracle Data Pump
- Create and use directory objects
- Use Data Pump Export and Import to move data between Oracle databases
- Use SQL*Loader to load data from a non-Oracle database (or user files)
- Use external tables to move data via platform-independent files



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Moving Data: General Architecture



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Major functional components:

- **DBMS_DATAPUMP:** Contains the API for high-speed export and import utilities for bulk data and metadata movement
- **Direct Path API (DPAPI):** Oracle Database supports a Direct Path API interface that minimizes data conversion and parsing at both unload and load time.
- **DBMS_METADATA:** Used by worker processes for all metadata unloading and loading. Database object definitions are stored using XML rather than SQL.
- **External Table API:** With the ORACLE_DATAPUMP and ORACLE_LOADER access drivers, you can store data in external tables (that is, in platform-independent files). The SELECT statement reads external tables as though they were stored in an Oracle database.
- **SQL*Loader:** Has been integrated with external tables, providing automatic migration of loader control files to external table access parameters
- **expdp and impdp:** Thin layers that make calls to the DBMS_DATAPUMP package to initiate and monitor Data Pump operations
- **Other clients:** Applications (such as replication, transportable tablespaces, and user applications) that benefit from this infrastructure. SQL*Plus may also be used as a client of DBMS_DATAPUMP for simple status queries against ongoing operations.

Oracle Data Pump: Overview

As a server-based facility for high-speed data and metadata movement, Oracle Data Pump:

- Is callable via DBMS_DATAPUMP
- Provides the following tools:
 - expdp
 - impdp
 - GUI interface in Enterprise Manager Cloud Control
- Provides four data movement methods:
 - Data file copying
 - Direct path
 - External tables
 - Network link support
- Detaches from and re-attaches to long-running jobs
- Restarts Data Pump jobs



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle Data Pump enables very high-speed data and metadata loading and unloading of Oracle databases. The Data Pump infrastructure is callable via the DBMS_DATAPUMP PL/SQL package. Thus, custom data movement utilities can be built by using Data Pump.

Oracle Database provides the following tools:

- Command-line export and import clients called expdp and impdp, respectively
- Export and import interface in Enterprise Manager Cloud Control

Data Pump automatically decides the data access methods to use; these can be either direct path or external tables. Data Pump uses direct path load and unload when a table's structure allows it and when maximum single-stream performance is desired. However, if there are clustered tables, referential integrity constraints, encrypted columns, or several other items, Data Pump uses external tables rather than direct path to move the data.

The ability to detach from and re-attach to long-running jobs without affecting the job itself enables you to monitor jobs from multiple locations while they are running. All stopped Data Pump jobs can be restarted without loss of data as long as the metainformation remains undisturbed. It does not matter whether the job is stopped voluntarily or involuntarily due to a crash.

Oracle Data Pump: Benefits

Data Pump offers many benefits and many features, such as:

- Fine-grained object and data selection
- Explicit specification of database version
- Parallel execution
- Estimation of export job space consumption
- Network mode in a distributed environment
- Remapping capabilities
- Data sampling and metadata compression
- Compression of data during a Data Pump export
- Security through encryption
- Ability to export XMLType data as CLOBs
- Legacy mode to support old import and export files



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The EXCLUDE, INCLUDE, and CONTENT parameters are used for fine-grained object and data selection.

You can specify the database version for objects to be moved (using the VERSION parameter) to create a dump file set that is compatible with a previous release of Oracle Database that supports Data Pump.

You can use the PARALLEL parameter to specify the maximum number of threads of active execution servers operating on behalf of the export job.

You can estimate how much space an export job would consume (without actually performing the export) by using the ESTIMATE_ONLY parameter.

Network mode enables you to export from a remote database directly to a dump file set. This can be done by using a database link to the source system.

During import, you can change the target data file names, schemas, and tablespaces.

In addition you can specify a percentage of data to be sampled and unloaded from the source database when performing a Data Pump export. This can be done by specifying the SAMPLE parameter.

You can use the COMPRESSION parameter to indicate whether the metadata should be compressed in the export dump file so that it consumes less disk space. If you compress the metadata, it is automatically uncompressed during import.

Features of Data Pump enable you to:

- Compress both data and metadata, only data, only metadata, or no data during an export
- Specify additional encryption options in the following areas:
 - You can choose to encrypt both data and metadata, only data, only metadata, no data, or only encrypted columns during an export.
 - You can specify a particular encryption algorithm to use during an export.
 - You can specify the type of security to use for performing encryption and decryption during an export. For example, perhaps the dump file set will be imported into a different or remote database and it must remain secure in transit. Or perhaps the dump file set will be imported on-site using the Oracle Encryption Wallet but it may also need to be imported off-site where the Oracle Encryption Wallet is not available.
- Perform table mode exports and imports using the transportable method; specify how partitioned tables should be handled during import operations
- Overwrite existing dump files during an export operation
- Rename tables during an import operation
- Specify that a data load should proceed even if nondeferred constraint violations are encountered (This is valid only for import operations that use the external tables access method.)
- Specify that XMLType columns are to be exported in uncompressed CLOB format regardless of the XMLType storage format that was defined for them
- During an export, specify a remap function that takes as a source the original value of the designated column and returns a remapped value that will replace the original value in the dump file
- Remap data as it is being imported into a new database
- Use legacy mode to support the use of original Export (`exp`) and Import (`imp`) scripts

Directory Objects for Data Pump

Directory Objects

Search

Object Name Go

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.

Selection Mode Single

Edit View Delete Actions Create Like Go

Select	Name	Path
<input checked="" type="radio"/>	DATA_FILE_DIR	/u01/app/oracle/product/12.1.0/dbhome_1/demo/schema/sales_history/
<input type="radio"/>	DATA_PUMP_DIR	/u01/app/oracle/admin/orcl/dpdump/
<input type="radio"/>	LOG_FILE_DIR	/u01/app/oracle/product/12.1.0/dbhome_1/demo/schema/log/
<input type="radio"/>	MEDIA_DIR	/u01/app/oracle/product/12.1.0/dbhome_1/demo/schema/product_media/
<input type="radio"/>	OPATCH_LOG_DIR	/u01/app/oracle/product/12.1.0/dbhome_1/Qpatch
<input type="radio"/>	OPATCH_SCRIPT_DIR	/u01/app/oracle/product/12.1.0/dbhome_1/Qpatch
<input type="radio"/>	ORACLE_OCM_CONFIG_DIR	/u01/app/oracle/product/12.1.0/dbhome_1/ccrhosts/EDRSR11P1/state
<input type="radio"/>	ORACLE_OCM_CONFIG_DIR2	/u01/app/oracle/product/12.1.0/dbhome_1/ccr/state
<input type="radio"/>	SS_OE_XMLDIR	/u01/app/oracle/product/12.1.0/dbhome_1/demo/schema/order_entry/
<input type="radio"/>	SUBDIR	/u01/app/oracle/product/12.1.0/dbhome_1/demo/schema/order_entry//2002/Sep
<input type="radio"/>	XMLDIR	/u01/app/oracle/product/12.1.0/dbhome_1/rdbms/xml
<input type="radio"/>	XSDDIR	/ade/b/4061281185/oracle/rdbms/xml/schema

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Directory objects are logical structures that represent a physical directory on the server's file system. They contain the location of a specific operating system directory. This directory object name can be used in Enterprise Manager so that you do not need to hard-code directory path specifications. This provides greater flexibility for file management. Directory objects are owned by the `SYS` user. Directory names are unique across the database because all the directories are located in a single name space (that is, `SYS`).

Directory objects are required when you specify file locations for Data Pump because it accesses files on the server rather than on the client.

In Enterprise Manager Cloud Control, select Schema > Database Objects > Directory Objects. To edit or delete a directory object, select the object and click the appropriate button.

Creating Directory Objects

The screenshot illustrates the process of creating a directory object in Oracle Database 12c. It consists of three main panels:

- Create Directory Object (General Tab):** Shows the directory object being created with the name "ext_tab_logdir" and the path "/home/oracle/labs/extab1". A "Test File System" button is visible.
- Privileges:** Shows the list of users with privileges for this directory. User "HR" is selected, granting both Read Access and Write Access.
- Show SQL:** Displays the SQL statements used to create the directory:

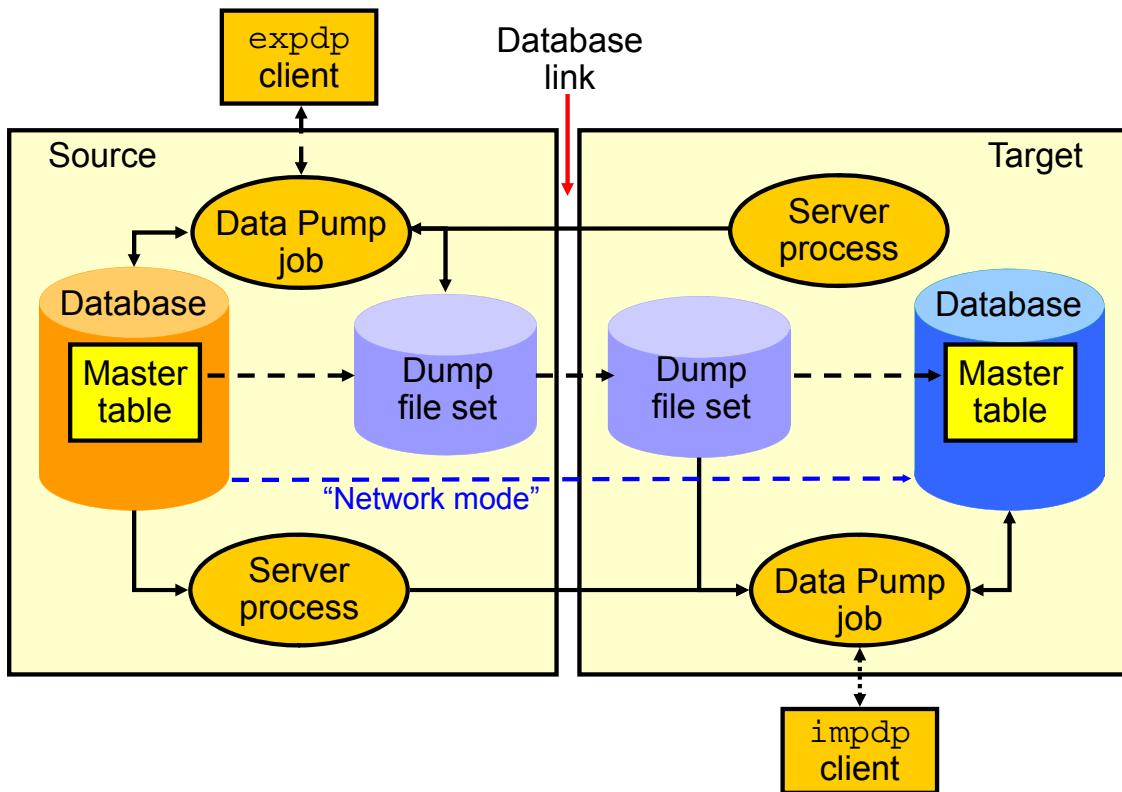
```
CREATE DIRECTORY "EXT_TAB_LOGDIR" AS '/home/oracle/labs/extab1'
GRANT READ ON DIRECTORY "EXT_TAB_LOGDIR" TO "HR"
GRANT WRITE ON DIRECTORY "EXT_TAB_LOGDIR" TO "HR"
```

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

1. On the Directory Objects page, click Create.
2. Enter the name of the directory object and the OS path to which it maps. OS directories should be created before they are used. You can test this by clicking Test File System. For the test, provide the host login credentials (the OS user who has privileges on this OS directory).
3. Permissions for directory objects are not the same as OS permissions on the physical directory on the server file system. You can manage user privileges on individual directory objects. This increases the level of security and gives you granular control over these objects. On the Privileges page, click Add to select the user to which you give read or write privileges (or both).
4. Click Show SQL to view the underlying statements. Click Return when finished.
5. Click OK to create the object.

Data Pump Export and Import Clients: Overview



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Data Pump Export is a utility for unloading data and metadata into a set of operating system files called *dump file sets*. Data Pump Import is used to load metadata and data stored in an export dump file set into a target system.

The Data Pump API accesses its files on the server rather than on the client.

These utilities can also be used to export from a remote database directly to a dump file set, or to load the target database directly from a source database with no intervening files. This is known as *network mode*. This mode is particularly useful to export data from a read-only source database.

At the center of every Data Pump operation is the master table (MT), which is a table created in the schema of the user running the Data Pump job. The MT maintains all aspects of the job. The MT is built during a file-based export job and is written to the dump file set as the last step. Conversely, loading the MT into the current user's schema is the first step of a file-based import operation and is used to sequence the creation of all objects imported.

Note: The MT is the key to Data Pump's restart capability in the event of a planned or unplanned stopping of the job. The MT is dropped when the Data Pump job finishes normally.

Data Pump Utility: Interfaces and Modes

- Data Pump Export and Import interfaces:
 - Command line
 - Parameter file
 - Interactive command line
 - Enterprise Manager Cloud Control
- Data Pump Export and Import modes:
 - Full
 - Schema
 - Table
 - Tablespace
 - Transportable tablespace
 - Transportable database



ORACLE®

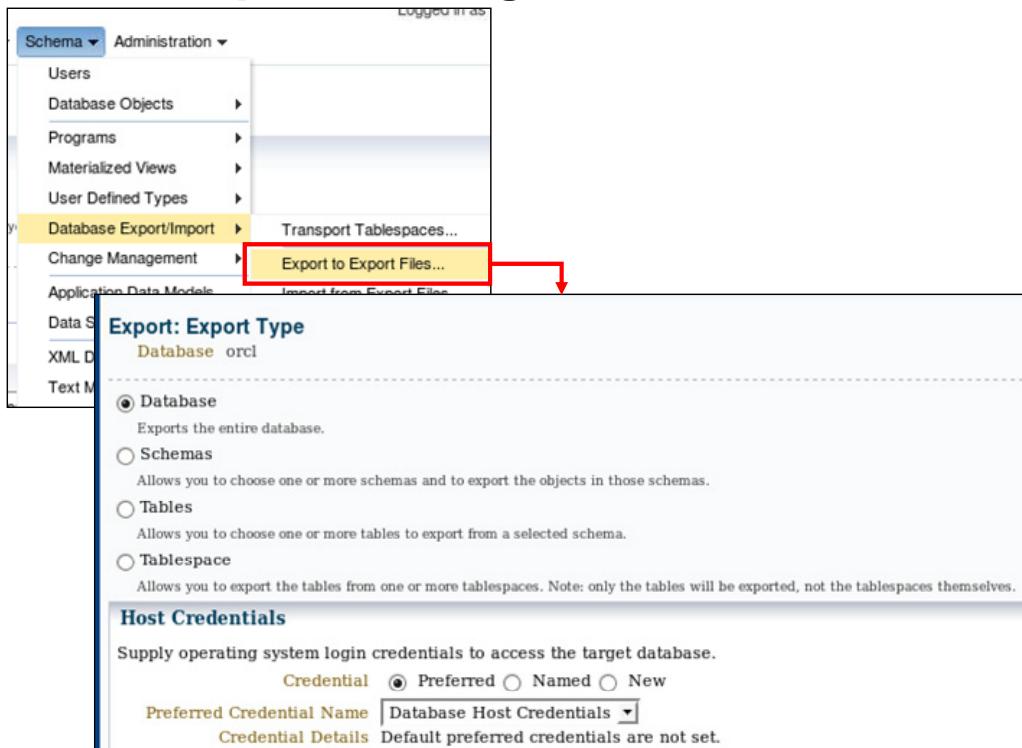
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can interact with Data Pump Export and Import by using one of the following interfaces:

- **Command-line interface:** Enables you to specify most of the export parameters directly on the command line
- **Parameter file interface:** Enables you to specify all command-line parameters in a parameter file. The only exception is the PARFILE parameter.
- **Interactive-command interface:** Stops logging to the terminal and displays the export or import prompts, where you can enter various commands. This mode is enabled by pressing Ctrl + C during an export operation that is started with the command-line interface or the parameter file interface. Interactive-command mode is also enabled when you attach to an executing or stopped job.
- **GUI interface:** Select Schema > Database Export/Import. In the menu, select the export or import operation you want to execute.

Data Pump Export and Import provide different modes for unloading and loading different portions of the database. The mode is specified on the command line by using the appropriate parameter. The available modes are listed in the slide and are the same as in the original export and import utilities.

Performing a Data Pump Export by Using Enterprise Manager Cloud Control



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Enterprise Manager Cloud Control provides a wizard to guide you through the process of performing a Data Pump export and import procedure. The example in the slide shows a Data Pump export.

From the Database home page, expand the Schema menu, select Database Export/Import, and then select Export to Export Files to begin a Data Pump export session.

The next window that appears enables the selection of export type. If a privileged user is connected to the database instance, the export types include the following:

- Database
- Schemas
- Tables
- Tablespace

If a non-administrative account is used, the export type list is limited to the following:

- Schemas
- Tables

Click Continue to proceed with the export.

Performing a Data Pump Import

Data Pump can be invoked on the command line:

```
$ impdp hr DIRECTORY=DATA_PUMP_DIR \
DUMPFILE=HR_SCHEMA.DMP \
PARALLEL=1 \
CONTENT=ALL \
TABLES="EMPLOYEES" \
LOGFILE=DATA_PUMP_DIR:import_hr_employees.log \
JOB_NAME=importHR \
TRANSFORM=STORAGE:n
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle Database also includes Data Pump command-line clients for import and export operations. The example in the slide illustrates a Data Pump import using the `impdp` utility. There are some parameters that are available only by using the command-line interface. For a complete list of options, refer to *Oracle Database Utilities*.

Data Pump Import: Transformations

You can remap:

- Data files by using REMAP_DATAFILE
- Tablespaces by using REMAP_TABLESPACE
- Schemas by using REMAP_SCHEMA
- Tables by using REMAP_TABLE
- Data by using REMAP_DATA

```
REMAP_TABLE = 'EMPLOYEES' : 'EMP'
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Because object metadata is stored as XML in the dump file set, it is easy to apply transformations when DDL is being formed during import. Data Pump Import supports several transformations:

- REMAP_DATAFILE is useful when moving databases across platforms that have different file-system semantics.
- REMAP_TABLESPACE enables objects to be moved from one tablespace to another.
- REMAP_SCHEMA provides the old FROMUSER /TOUSER capability to change object ownership.
- REMAP_TABLE provides the ability to rename entire tables.
- REMAP_DATA provides the ability to remap data as it is being inserted.

Using Enterprise Manager Cloud Control to Monitor Data Pump Jobs

The screenshot shows two pages from Oracle Enterprise Manager Cloud Control. The top page is a navigation menu with a sidebar on the left containing links like 'Users', 'Database Objects', 'Programs', etc., and a main panel on the right showing a list of items. A red box highlights the 'Database Export/Import' menu item, which has a submenu with options: 'Transport Tablespaces...', 'Export to Export Files...', 'Import from Export Files...', 'Import from Database...', 'Load Data from User Files...', and 'View Export & Import Jobs'. The 'View Export & Import Jobs' option is also highlighted with a red box. A red arrow points from this highlighted option down to the second page. The bottom page is titled 'Export and Import Jobs' and displays a table of current jobs. The table has columns for 'Data Pump Job', 'Owner', and 'Job Status'. It shows one job named 'HR_EXPORT' owned by 'SYSTEM' with a status of 'EXECUTING'. The page also includes a note about monitoring data pump jobs through the interface.

Data Pump Job	Owner	Job Status
HR_EXPORT	SYSTEM	EXECUTING

ORACLE®

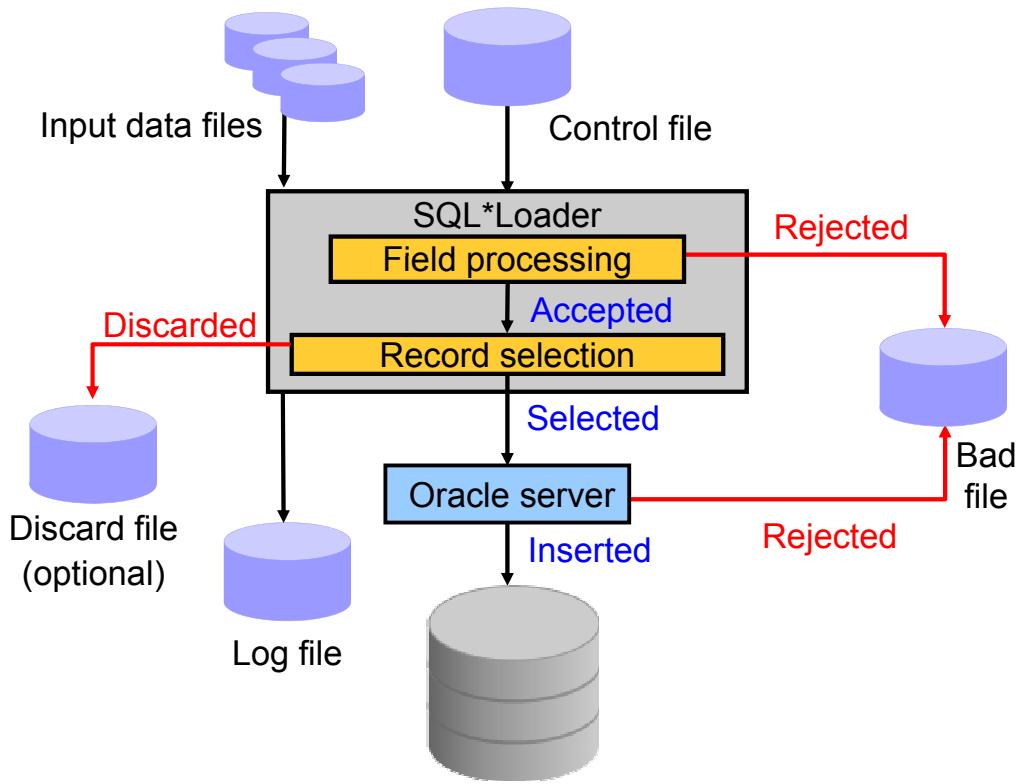
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can use the Enterprise Manager graphical user interface (GUI) to monitor all Data Pump jobs, including those created by using the `expdp` or `impdp` command-line interfaces or by using the `DBMS_DATAPUMP` package.

You can view the current status of the job and change the status to EXECUTE, STOP, or SUSPEND.

To access the “Export and Import Jobs” page, select View Export & Import Jobs in the Database Export/Import menu.

SQL*Loader: Overview



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

SQL*Loader loads data from external files into tables of an Oracle database. It has a powerful data parsing engine that puts little limitation on the format of the data in the data file.

SQL*Loader uses the following files:

Input data files: SQL*Loader reads data from one or more files (or operating system equivalents of files) that are specified in the control file. From SQL*Loader's perspective, the data in the data file is organized as records. A particular data file can be in fixed record format, variable record format, or stream record format. The record format can be specified in the control file with the `INFILE` parameter. If no record format is specified, the default is stream record format.

Control file: The control file is a text file that is written in a language that SQL*Loader understands. The control file indicates to SQL*Loader where to find the data, how to parse and interpret the data, where to insert the data, and so on. Although not precisely defined, a control file can be said to have three sections.

- The first section contains such session-wide information as the following:
 - Global options, such as the input data file name and records to be skipped
 - `INFILE` clauses to specify where the input data is located
 - Data to be loaded

- The second section consists of one or more `INTO TABLE` blocks. Each of these blocks contains information about the table (such as the table name and the columns of the table) into which the data is to be loaded.
- The third section is optional and, if present, contains input data.

Log file: When SQL*Loader begins execution, it creates a log file. If it cannot create a log file, execution terminates. The log file contains a detailed summary of the load, including a description of any errors that occurred during the load.

Bad file: The bad file contains records that are rejected, either by SQL*Loader or by the Oracle database. Data file records are rejected by SQL*Loader when the input format is invalid. After a data file record is accepted for processing by SQL*Loader, it is sent to the Oracle database for insertion into a table as a row. If the Oracle database determines that the row is valid, the row is inserted into the table. If the row is determined to be invalid, the record is rejected and SQL*Loader puts it in the bad file.

Discard file: This file is created only when it is needed and only if you have specified that a discard file should be enabled. The discard file contains records that are filtered out of the load because they do not match any record-selection criteria specified in the control file.

For more information about SQL*Loader, see the *Oracle Database Utilities* guide.

SQL*Loader Control File

The SQL*Loader control file instructs SQL*Loader about:

- Location of the data to be loaded
- Data format
- Configuration details:
 - Memory management
 - Record rejection
 - Interrupted load handling details
- Data manipulation details



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The SQL*Loader control file is a text file that contains data definition language (DDL) instructions. DDL is used to control the following aspects of a SQL*Loader session:

- Where SQL*Loader finds the data to load
- How SQL*Loader expects that data to be formatted
- How SQL*Loader is being configured (including memory management, selection and rejection criteria, interrupted load handling, and so on) as it loads the data
- How SQL*Loader manipulates the data being loaded

```

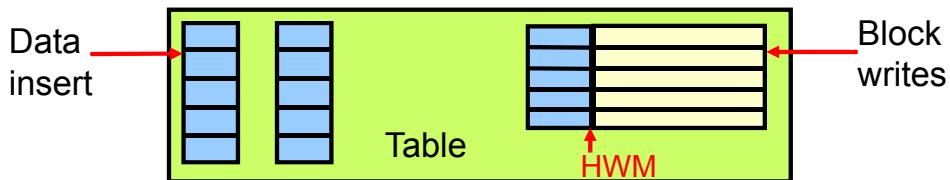
1 -- This is a sample control file
2 LOAD DATA
3 INFILE 'SAMPLE.DAT'
4 BADFILE 'sample.bad'
5 DISCARDFILE 'sample.dsc'
6 APPEND
7 INTO TABLE emp
8 WHEN (57) = '.'
9 TRAILING NULLCOLS
10 (hiredate SYSDATE,
    deptno POSITION(1:2) INTEGER EXTERNAL (3)
    NULLIF deptno=BLANKS,
    job POSITION(7:14) CHAR TERMINATED BY WHITESPACE
    NULLIF job=BLANKS "UPPER(:job)",
    mgr POSITION(28:31) INTEGER EXTERNAL
    TERMINATED BY WHITESPACE, NULLIF mgr=BLANKS,
    ename POSITION(34:41) CHAR
    TERMINATED BY WHITESPACE "UPPER(:ename)",
    empno POSITION(45) INTEGER EXTERNAL
    TERMINATED BY WHITESPACE,
    sal POSITION(51) CHAR TERMINATED BY WHITESPACE
    "TO_NUMBER(:sal,'$99,999.99')",
    comm INTEGER EXTERNAL ENCLOSED BY '(' AND '%'
    ":comm * 100"
)

```

The explanation of this sample control file (by line numbers) is as follows:

1. Comments can appear anywhere in the command section of the file, but they must not appear in the data. Precede any comment with two hyphens. All text to the right of the double hyphen is ignored until the end of the line.
2. The `LOAD DATA` statement indicates to SQL*Loader that this is the beginning of a new data load. If you are continuing a load that has been interrupted in progress, use the `CONTINUE LOAD DATA` statement.
3. The `INFILE` keyword specifies the name of a data file containing data that you want to load.
4. The `BADFILE` keyword specifies the name of a file into which rejected records are placed.
5. The `DISCARDFILE` keyword specifies the name of a file into which discarded records are placed.
6. The `APPEND` keyword is one of the options that you can use when loading data into a table that is not empty. To load data into a table that is empty, use the `INSERT` keyword.
7. The `INTO TABLE` keyword enables you to identify tables, fields, and data types. It defines the relationship between records in the data file and tables in the database.
8. The `WHEN` clause specifies one or more field conditions that each record must match before SQL*Loader loads the data. In this example, SQL*Loader loads the record only if the 57th character is a decimal point. That decimal point delimits dollars and cents in the field and causes records to be rejected if `SAL` has no value.
9. The `TRAILING NULLCOLS` clause prompts SQL*Loader to treat any relatively positioned columns that are not present in the record as null columns.
10. The remainder of the control file contains the field list, which provides information about column formats in the table that is being loaded.

Loading Methods



Conventional Load	Direct Path Load
Uses COMMIT	Uses data saves (faster operation)
Always generates redo entries	Generates redo only under specific conditions
Enforces all constraints	Enforces only PRIMARY KEY, UNIQUE, and NOT NULL
Fires INSERT triggers	Does not fire INSERT triggers
Can load into clustered tables	Does not load into clusters
Allows other users to modify tables during load operation	Prevents other users from making changes to tables during load operation
Maintains index entries on each insert	Merges new index entries at the end of the load

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Method of Saving Data

A conventional path load executes SQL `INSERT` statements to populate tables in an Oracle database. A direct path load eliminates much of the Oracle database overhead by formatting Oracle data blocks and writing the data blocks directly to the database files. A direct load does not compete with other users for database resources, so it can usually load data at close to disk speed. Conventional path loads use SQL processing and a database `COMMIT` operation for saving data. The insertion of an array of records is followed by a `COMMIT` operation. Each data load may involve several transactions.

Direct path loads use data saves to write blocks of data to Oracle data files. This is why the direct path loads are faster than the conventional ones. The following features differentiate a data save from `COMMIT`:

- During a data save, only full database blocks are written to the database.
- The blocks are written after the high-water mark (HWM) of the table.
- After a data save, the HWM is moved.
- Internal resources are not released after a data save.
- A data save does not end the transaction.
- Indexes are not updated at each data save.

Loading Data by Using Enterprise Manager Cloud Control

Load Data: Generate Or Use Existing Control File
Database orcl

Automatically Generate Control File
A control file will be generated after you define the structure of the data file.

Use Existing Control File
Allows you to use an existing control file that defines the structure of the data file.

Host Credentials

Credential Preferred Named New

* UserName

* Password

* Confirm Password

Save As NC_ORCL_2012-11-28-130014

Set As Preferred Credentials

Control File Data File Load Method Options Schedule Review

Load Data: Control File
Database orcl

A control file is used to describe what will be loaded and how. Specify the full path and name of the control file on the database server machine.
/u01/app/oracle/oradata/orcl/LOAD.CTL

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

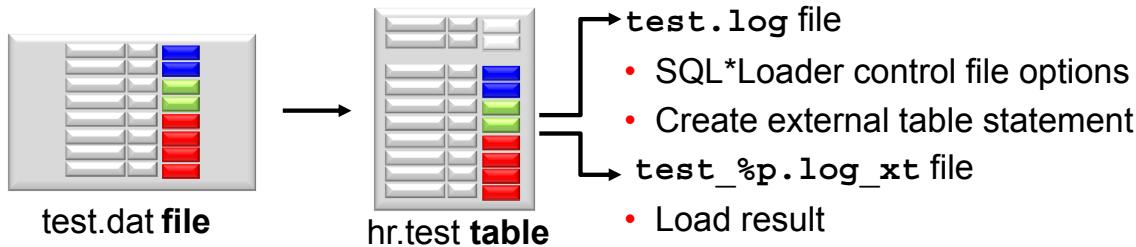
You can load data by using Enterprise Manager Cloud Control. In the Schema menu, select Database Export/Import. Then select Load Data From User Files.

On the first page you indicate whether you are using an existing control file or want a new control file to be generated. Click Continue on this page to invoke the wizard.

SQL*Loader Express Mode

- Specify a table name to initiate an Express Mode load.
- Table columns must be scalar data types (character, number, or datetime).
- A data file can contain only delimited character data.
- SQL*Loader uses table column definitions to determine input data types.
- There is no need to create a control file.

```
$ sqldr hr TABLE=test
```



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

If you activate SQL*Loader Express Mode, specifying only the username and the TABLE parameter, it uses default settings for several other parameters. You can override most of the defaults by specifying additional parameters on the command line.

SQL*Loader Express Mode generates two files. The names of the log files come from the name of the table (by default).

- A log file that includes:
 - The control file output
 - A SQL script for creating the external table and performing the load using a SQL INSERT AS SELECT statement

Neither the control file nor the SQL script are used by SQL*Loader Express Mode. They are made available to you in case you want to use them as a starting point to perform operations using regular SQL*Loader or stand-alone external tables.

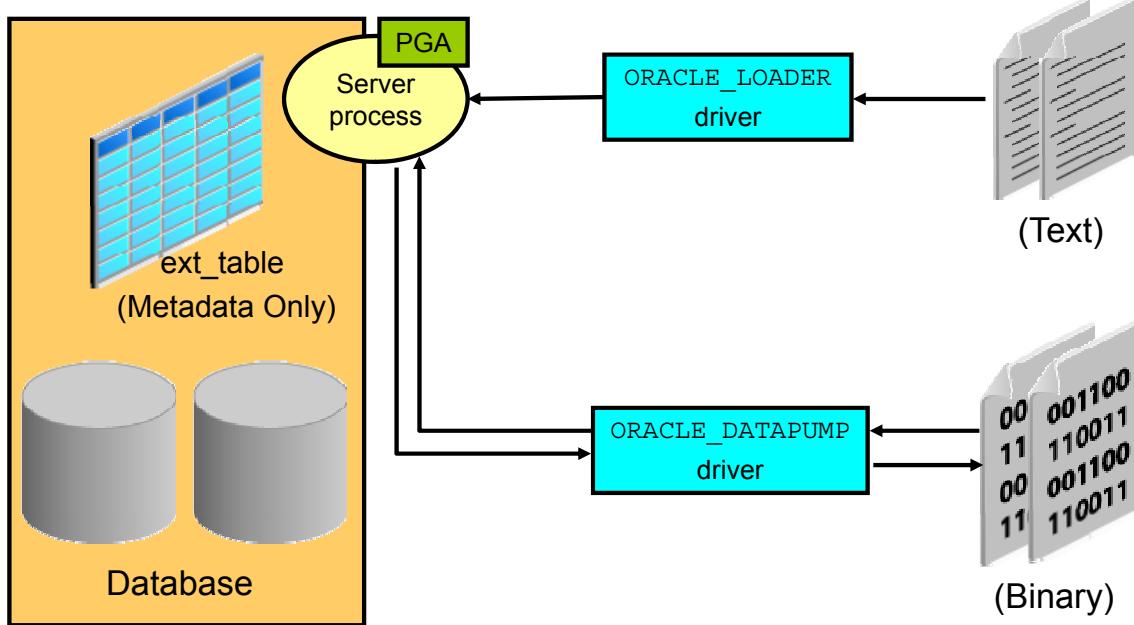
- A log file similar to a SQL*Loader log file that describes the result of the operation
The "%p" represents the process ID of the SQL*Loader process.

The following is an example of the input data file named `test.dat` containing two records to load and excerpts of the two log files generated after the load completed. The table `HR.TEST` was created with three columns, `C1 NUMBER`, `C2 VARCHAR2 (10)`, and `C3 VARCHAR2 (10)`.

```
$ more test.dat
3 C WWW
4 D UUU
$ sqlldr hr TABLE=test
$ more test.log
...
Express Mode Load, Table: TEST
Data File:      test.dat
Bad File:      test_%p.bad      ...
Table TEST, loaded from every logical record.
Insert option in effect for this table: APPEND
Column Name      Position   Len   Term Encl Datatype
-----
C1              FIRST          *      , CHARACTER
C2              NEXT           *      , CHARACTER
C3              NEXT           *      , CHARACTER
Generated control file for possible reuse:
OPTIONS(EXTERNAL_TABLE=EXECUTE, TRIM=LRTRIM)
LOAD DATA INFILE 'test' APPEND INTO TABLE TEST
FIELDS TERMINATED BY "," ( C1, C2, C3)
End of generated control file for possible reuse.
...
creating external table "SYS_SQLLDR_X_EXT_TEST"
CREATE TABLE "SYS_SQLLDR_X_EXT_TEST"
("C1" NUMBER,"C2" CHAR(1),"C3" VARCHAR2(20)) ORGANIZATION external(
  TYPE oracle_loader DEFAULT DIRECTORY SYS_SQLLDR_XT_TMPDIR_00000
  ACCESS PARAMETERS
    ( RECORDS DELIMITED BY NEWLINE CHARACTERSET US7ASCII
      BADFILE 'SYS_SQLLDR_XT_TMPDIR_00000':'test_%p.bad'
      LOGFILE 'test_%p.log_xt' READSIZE 1048576
      FIELDS TERMINATED BY "," LRTRIM REJECT ROWS WITH ALL NULL FIELDS ("C1"
      CHAR(255), "C2" CHAR(255), "C3" CHAR(255)))
    location ('test.dat')) REJECT LIMIT UNLIMITED
executing INSERT statement to load database table TEST
INSERT /*+ append parallel(auto) */ INTO TEST (C1, C2, C3)
SELECT   "C1",   "C2",   "C3" FROM "SYS_SQLLDR_X_EXT_TEST" ...
Table TEST: 2 Rows successfully loaded.
```

External Tables

External tables are read-only tables stored as files on the operating system outside of the Oracle database.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

External tables access data in external sources as if it were in a table in the database. You can connect to the database and create metadata for the external table using DDL. The DDL for an external table consist of two parts: one part that describes the Oracle Database column types, and another part that describes the mapping of the external data to the Oracle Database data columns.

An external table does not describe any data that is stored in the database. Nor does it describe how data is stored in the external source. Instead, it describes how the external table layer must present the data to the server. It is the responsibility of the access driver and the external table layer to do the necessary transformations required on the data in the external file so that it matches the external table definition. External tables are read only; therefore, no DML operations are possible, and no index can be created on them.

There are two access drivers used with external tables. The `ORACLE_LOADER` access driver can be used only to read table data from an external table and load it into the database. It uses text files as the data source. The `ORACLE_DATAPUMP` access driver can both load table data from an external file into the database and also unload data from the database into an external file. It uses binary files as the external files. The binary files have the same format as the files used by the Data Pump Import and Export utilities and can be interchanged with them.

External Table: Benefits

- Data can be used directly from the external file or loaded into another database.
- External data can be queried and joined directly in parallel with tables residing in the database, without requiring it to be loaded first.
- The results of a complex query can be unloaded to an external file.
- You can combine generated files from different sources for loading purposes.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The data files created for the external table can be moved and used as the data files for another external table in the same database or different database. External data can be queried and joined directly in parallel to tables residing in the database, without requiring the data to be loaded first. You can choose to have your applications directly access external tables with the `SELECT` command, or you can choose to have data loaded first into a target database.

The results of a complex query can be unloaded to an external file using the `ORACLE_DATAPUMP` access driver.

Data files that are populated by different external tables can all be specified in the `LOCATION` clause of another external table. This provides an easy way of aggregating data from multiple sources. The only restriction is that the metadata for all the external tables must be exactly the same.

Defining an External Tables with ORACLE_LOADER

```
CREATE TABLE extab_employees
    (employee_id          NUMBER(4),
     first_name           VARCHAR2(20),
     last_name            VARCHAR2(25),
     hire_date             DATE)
ORGANIZATION EXTERNAL
  ( TYPE ORACLE_LOADER DEFAULT DIRECTORY extab_dat_dir
    ACCESS PARAMETERS
      ( records delimited by newline
        badfile extab_bad_dir:'empxt%a_%p.bad'
        logfile extab_log_dir:'empxt%a_%p.log'
        fields terminated by ','
        missing field values are null
      ( employee_id, first_name, last_name,
        hire_date char date_format date mask "dd-mon-yyyy"))
    LOCATION ('empxt1.dat', 'empxt2.dat') )
PARALLEL REJECT LIMIT UNLIMITED;
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The metadata for an external table is created using the SQL language in the database. The ORACLE_LOADER access driver uses the SQL*Loader syntax to define the external table. This command does not create the external text files.

The example in the slide shows three directory objects (EXTAB_DAT_DIR, EXTAB_BAD_DIR, and EXTAB_LOG_DIR) that are created and mapped to existing OS directories to which the user is granted access.

When the EXTAB_EMPLOYEES table is accessed, SQL*Loader functionality is used to load the table, and at that instance the log file and bad file are created.

Best-practice tip: If you have a lot of data to load, enable PARALLEL for the load operation:

```
ALTER SESSION ENABLE PARALLEL DML;
```

External Table Population with ORACLE_DATAPUMP

```
CREATE TABLE ext_emp_query_results
  (first_name, last_name, department_name)
ORGANIZATION EXTERNAL
(
  TYPE ORACLE_DATAPUMP
  DEFAULT DIRECTORY ext_dir
  LOCATION ('emp1.exp', 'emp2.exp', 'emp3.exp')
)
PARALLEL
AS
SELECT e.first_name, e.last_name, d.department_name
FROM employees e, departments d
WHERE e.department_id = d.department_id AND
      d.department_name in
        ('Marketing', 'Purchasing');
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

This example shows you how the external table population operation can help to export a selective set of records resulting from the join of the EMPLOYEES and DEPARTMENTS tables.

Because the external table can be large, you can use a parallel populate operation to unload your data to an external table. As opposed to a parallel query from an external table, the degree of parallelism of a parallel populate operation is constrained by the number of concurrent files that can be written to by the access driver. There is never more than one parallel execution server writing into one file at a particular point in time.

The number of files in the LOCATION clause must match the specified degree of parallelism because each input/output (I/O) server process requires its own file. Any extra files that are specified are ignored. If there are not enough files for the specified degree of parallelism, the degree of parallelization is lowered to match the number of files in the LOCATION clause.

The external table is read-only after it has been populated. The SELECT command can be very complex, allowing specific information to be populated in the external table. The external table, having the same file structure as binary Data Pump files, can then be migrated to another system, and imported with the impdp utility or read as an external table.

Note: For more information about the ORACLE_DATAPUMP access driver parameters, see the *Oracle Database Utilities* guide.

Using External Tables

- Querying an external table:

```
SQL> SELECT * FROM extab_employees;
```

- Querying and joining an external table with an internal table:

```
SQL> SELECT e.employee_id, e.first_name, e.last_name,
d.department_name FROM departments d, extab_employees e
WHERE d.department_id = e.department_id;
```

- Appending data to an internal table from an external table:

```
SQL> INSERT /*+ APPEND */ INTO hr.employees SELECT *
FROM extab_employees;
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

External tables are queried just like internal database tables. The first example illustrates querying an external table named `EXTAB_EMPLOYEES` and only displaying the results. The results are not stored in the database.

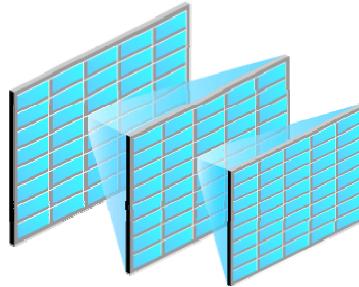
The second example shows the joining of an internal table named `DEPARTMENTS` with an external table named `EXTAB_EMPLOYEES` and only displaying the results.

The third example in the slide illustrates the direct appending of an internal table data with the query and load of data from an external table.

Data Dictionary

View information about external tables in:

- [DBA | ALL | USER] _EXTERNAL_TABLES
- [DBA | ALL | USER] _EXTERNAL_LOCATIONS
- [DBA | ALL | USER] _TABLES
- [DBA | ALL | USER] _TAB_COLUMNS
- [DBA | ALL] _DIRECTORIES



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The data dictionary views in the slide provide information about external tables:

[DBA | ALL | USER] _EXTERNAL_TABLES: Specific attributes of external tables in the database

[DBA | ALL | USER] _EXTERNAL_LOCATIONS: Data sources for external tables

[DBA | ALL | USER] _TABLES: Descriptions of the relational tables in the database

[DBA | ALL | USER] _TAB_COLUMNS: Descriptions of the columns of tables, views, and clusters in the database

[DBA | ALL] _DIRECTORIES: Descriptions of the directory objects in the database

Quiz

Like other database objects, directory objects are owned by the user that creates them unless another schema is specified during creation.

- a. True
- b. False



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b

Quiz

An index can be created on an external table.

- a. True
- b. False



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b

Summary

In this lesson, you should have learned how to:

- Describe ways to move data
- Explain the general architecture of Oracle Data Pump
- Create and use directory objects
- Use Data Pump Export and Import to move data between Oracle databases
- Use SQL*Loader to load data from a non-Oracle database (or user files)
- Use external tables to move data via platform-independent files



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice: Overview

This practice covers the following topics:

- Using the Data Pump Export wizard to select database objects to be exported
- Monitoring a Data Pump Export job
- Using the Data Pump Import wizard to import tables to your database
- Using the Load Data wizard to load data into your database
- Loading data by using the command line



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

17

Database Maintenance

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

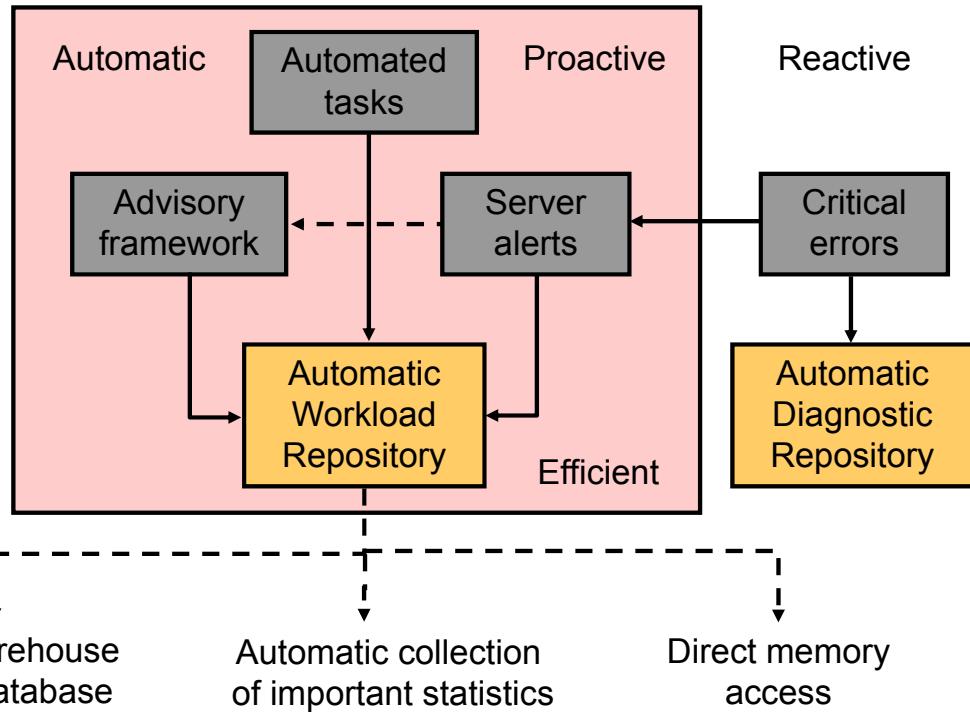
After completing this lesson, you should be able to:

- Manage the Automatic Workload Repository (AWR)
- Use the Automatic Database Diagnostic Monitor (ADDM)
- Describe and use the advisory framework
- Set alert thresholds
- Use server-generated alerts
- Use automated tasks



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Database Maintenance



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

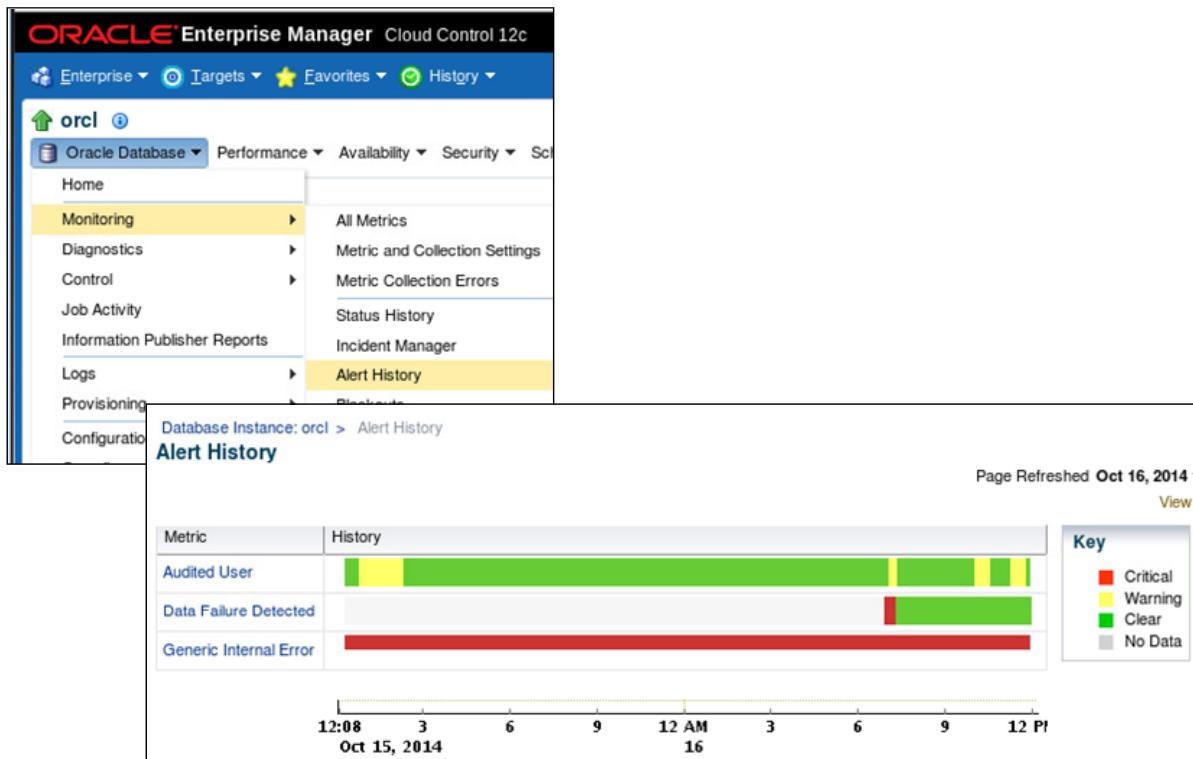
Proactive database maintenance is made easy by the sophisticated infrastructure of the Oracle database, including the following main elements:

- The Automatic Workload Repository (AWR) is a built-in repository in each Oracle database. At regular intervals, the Oracle database server makes a snapshot of all its vital statistics and workload information and stores this data in the AWR. The captured data can be analyzed by you, by the database server itself, or by both.
- Using automated tasks, the database server performs routine maintenance operations, such as regular backups, refreshing optimizer statistics, and database health checks.

Reactive database maintenance includes critical errors and conditions discovered by database health checkers:

- For problems that cannot be resolved automatically and require administrators to be notified (such as running out of space), the Oracle database server provides server-generated alerts. The Oracle database server, by default, monitors itself and sends out alerts to notify you of problems. The alerts notify you and often also provide recommendations on how to resolve the reported problem.
- Recommendations are generated from several advisors, each of which is responsible for a subsystem. For example, there are memory, segment, and SQL advisors.

Viewing the Alert History



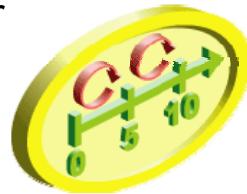
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

ORACLE®

The Alert History page displays a chart that shows the alert history of the current database in segments of time that you designate. An alert indicates a potential problem: either a warning or critical threshold for a monitored metric, or an indication that a target is no longer available. Click the metric name listed on the Alert History page to get detailed statistics, graphs, and actual time stamps for each alert.

Terminology

- Statistics: Data collections providing database and object detail
 - Optimizer statistics: Used by query optimizer
 - Database statistics: Used for performance
- Metric: Rate of change in a cumulative statistic
- Threshold: A boundary value against which metric values are compared
- Automatic Workload Repository (AWR): Infrastructure for data gathering, analysis, and solutions recommendations
- AWR Baseline: A set of AWR snapshots for performance comparison



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Statistics are collections of data that provide more details about the database and the objects in it. Optimizer statistics are used by the query optimizer to choose the best execution plan for each SQL statement. Database statistics provide information for performance monitoring.

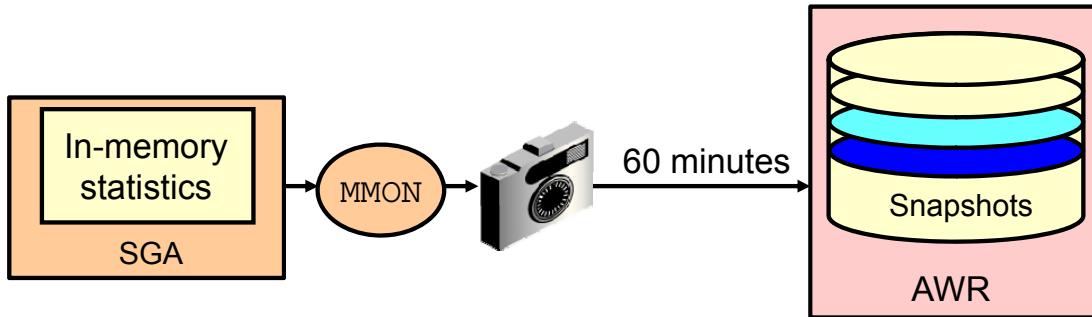
The *Automatic Workload Repository* (AWR) provides services to internal Oracle server components to collect, process, maintain, and use performance statistics for problem detection and self-tuning purposes. *Active Session History* (ASH) is the history of recent session activity stored in the AWR.

AWR snapshots include database statistics and metrics, application statistics (transaction volumes, response time), operating system statistics, and other measures. An *AWR baseline* is a set of AWR snapshots collected over a period of time. The baseline is used for performance comparison, either current performance versus the baseline or one baseline compared to another.

The *System Moving Window* baseline is collected by default. The System Moving Window baseline is a changing set of snapshots that include the last eight days of snapshots by default. This baseline becomes valid after sufficient data has been collected and the statistics calculation occurs. The statistics calculation is scheduled for every Saturday at midnight by default.

Automatic Workload Repository (AWR): Overview

- Built-in repository of performance information
- Snapshots of database metrics taken every 60 minutes and retained for eight days
- Foundation for all self-management functions



ORACLE®

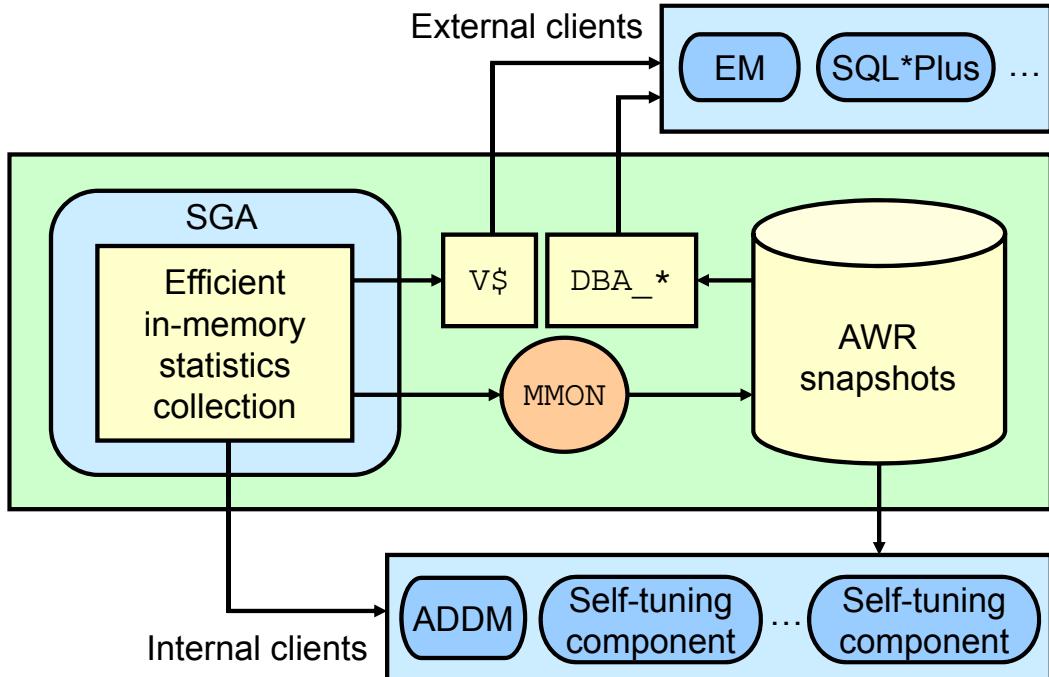
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The AWR is the infrastructure that provides services to Oracle Database components to collect, maintain, and use statistics for problem detection and self-tuning purposes. You can view it as a data warehouse for database statistics, metrics, and so on.

Every 60 minutes (by default), the database automatically captures statistical information from the SGA and stores it in the AWR in the form of snapshots. These snapshots are stored on disk by a background process called Manageability Monitor (MMON). By default, snapshots are retained for eight days. You can modify both the snapshot interval and the retention intervals.

The AWR contains hundreds of tables, all belonging to the `SYS` schema and stored in the `SYSAUX` tablespace. Oracle recommends that the repository be accessed only through Enterprise Manager or the `DBMS_WORKLOAD_REPOSITORY` package to work with the AWR. Direct data manipulation language (DML) commands against the repository tables are not supported.

AWR Infrastructure



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The AWR infrastructure has two major parts:

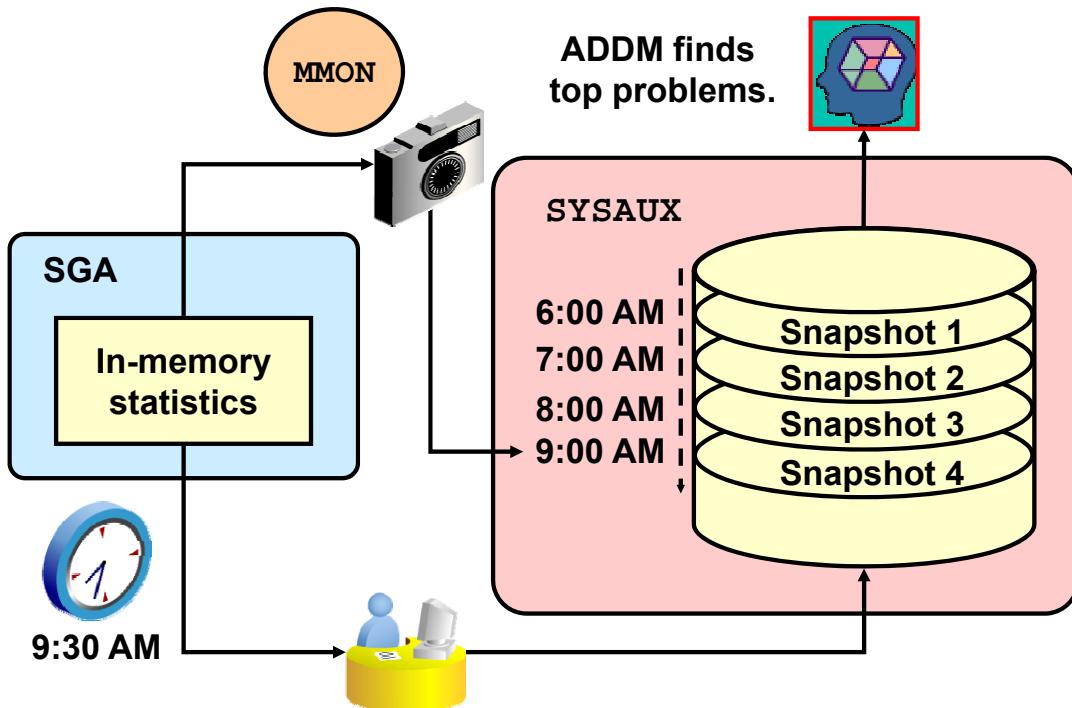
- An in-memory statistics collection facility that is used by Oracle Database components to collect statistics. These statistics are stored in memory for performance reasons. Statistics stored in memory are accessible through dynamic performance (V\$) views.
- The AWR snapshots that represent the persistent portion of the facility. AWR snapshots are accessible through data dictionary views and Enterprise Manager.

Statistics are stored in persistent storage for several reasons:

- The statistics need to survive instance crashes.
- Some analyses need historical data for baseline comparisons.
- A memory overflow can occur. When old statistics are replaced by new ones because of memory shortage, the replaced data can be stored for later use.

The memory version of the statistics is transferred to disk on a regular basis by the MMON background process. With the AWR, the Oracle database server provides a way to capture historical statistics data automatically without DBA intervention.

Automatic Workload Repository



ORACLE

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. Each snapshot is identified by a snapshot sequence number (`SNAP_ID`) that is unique in the AWR. 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 four hours, you may miss transient events that would be noticeable in 60-minute intervals.

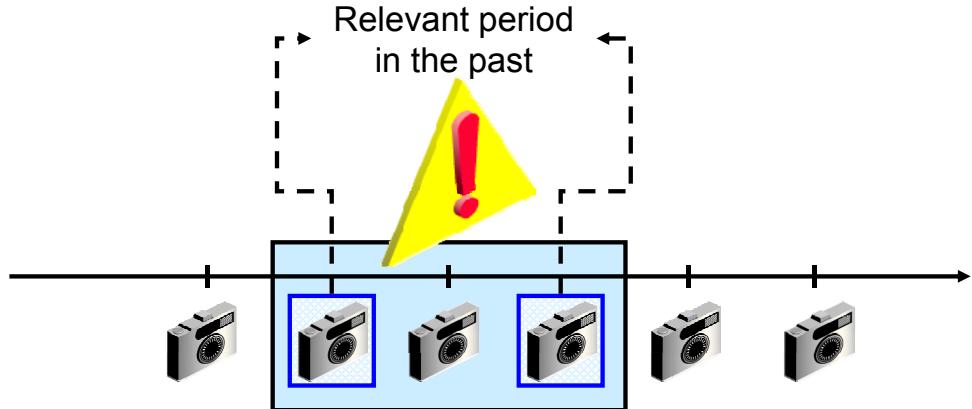
You can use the `DBMS_WORKLOAD_REPOSITORY.MODIFY_SNAPSHOT_SETTINGS` stored procedure or Enterprise Manager to change the settings that control snapshot collection.

You can take manual snapshots by using Enterprise Manager 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.

Statspack is a bundled utility that provides a subset of the collection and reporting capability of the AWR. However, there is no supported path to migrate Statspack data into the workload repository. Also, the workload repository is not compatible with the Statspack schema. Statspack is not accessible through Enterprise Manager; it requires setup, and does not have automatic retention settings, or automatic purge. The Statspack utility does provide scripts for setup, automatic snapshot collection, and reporting. Statspack snapshots can be marked for retention, as part of a Statspack baseline, or purged with provided scripts.

Statspack is documented in the `$ORACLE_HOME/rdbms/admin/spdoc.txt` file.

AWR Baselines



```
DBMS_WORKLOAD_REPOSITORY.CREATE_BASELINE ( -  
    start_snap_id IN NUMBER,  
    end_snap_id   IN NUMBER,  
    baseline_name IN VARCHAR2) ;
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

An AWR baseline is a set of AWR snapshots. This is usually a set of snapshot data for an important period that you tag and retain in the AWR. A baseline is defined on a pair of snapshots; the snapshots are identified by their snapshot sequence numbers (SNAP_IDS) or a start and end time. Each snapshot set has starting and ending snapshots and includes all the snapshots in between. Snapshot sets are used to retain snapshot data. Therefore, by default, snapshots belonging to snapshot sets are retained until the snapshot sets are dropped. You can specify an expiration value to indicate the number of days that the snapshot will be retained.

A baseline is identified by a user-supplied name. Execute the CREATE_BASELINE procedure to create a baseline from a set of snapshots, and specify a name and a pair of snapshot identifiers. A baseline identifier that is unique for the life of a database is assigned to the newly created baseline. Usually, you set up baselines from 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 Enterprise Manager. You can set the expiration time (in number of days) with the expiration parameter of this procedure. The default is NULL, meaning “never expire.”

You can get the SNAP_IDS directly from DBA_HIST_SNAPSHOT, or from Enterprise Manager.

Note: For more information about the DBMS_WORKLOAD_REPOSITORY package, see the *Oracle Database PL/SQL Packages and Types Reference* guide.

Accessing the AWR Page

The screenshot shows the Oracle Enterprise Manager Cloud Control interface. In the top-left corner, the database name 'orcl' is displayed. The 'Performance' menu is selected, which is highlighted in blue. Under the 'Performance' menu, the 'AWR Administration' option is also highlighted in yellow. On the left side, there is a summary of database status, including 'Up Time 1 d', 'Version 12.', 'Load 0.0', 'Total 79 Sessions', and 'Real-Time ADDM'. The main content area is titled 'Automatic Workload Repository'. It contains a general section with settings like 'Snapshot Retention (days) 8', 'Snapshot Interval (minutes) 60', 'Collection Level TYPICAL', and 'Next Snapshot Capture Time Oct 16, 2014 1:00:19 PM'. Below this is a 'Manage Snapshots and Baselines' section with buttons for 'Run AWR Report' and 'Run Compare Periods Report'. It shows statistics: 'Snapshots 197' and 'Baselines 1'. The latest snapshot time is listed as 'Oct 16, 2014 12:00:19 PM' and the earliest as 'Oct 8, 2014 7:44:16 AM'.

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In Enterprise Manager Cloud Control, navigate to the Automatic Workload Repository page by expanding the Performance menu, selecting AWR, and then selecting AWR Administration.

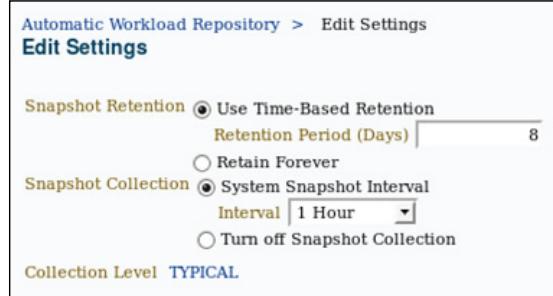
On the Automatic Workload Repository page, click Edit to change the settings.

On the Automatic Workload Repository page, you can:

- Edit the workload repository settings
- Look at detailed information about created snapshots and manually create new ones
- Create AWR baselines
- Generate an AWR report

Managing the AWR

- Retention period
 - Default: Eight days
 - Consider storage needs
- Collection interval
 - Default: 60 minutes
 - Consider storage needs and performance impact
- Collection level
 - Basic (disables most ADDM functionality)
 - Typical (recommended)
 - All (adds additional SQL tuning information to snapshots)



ORACLE®

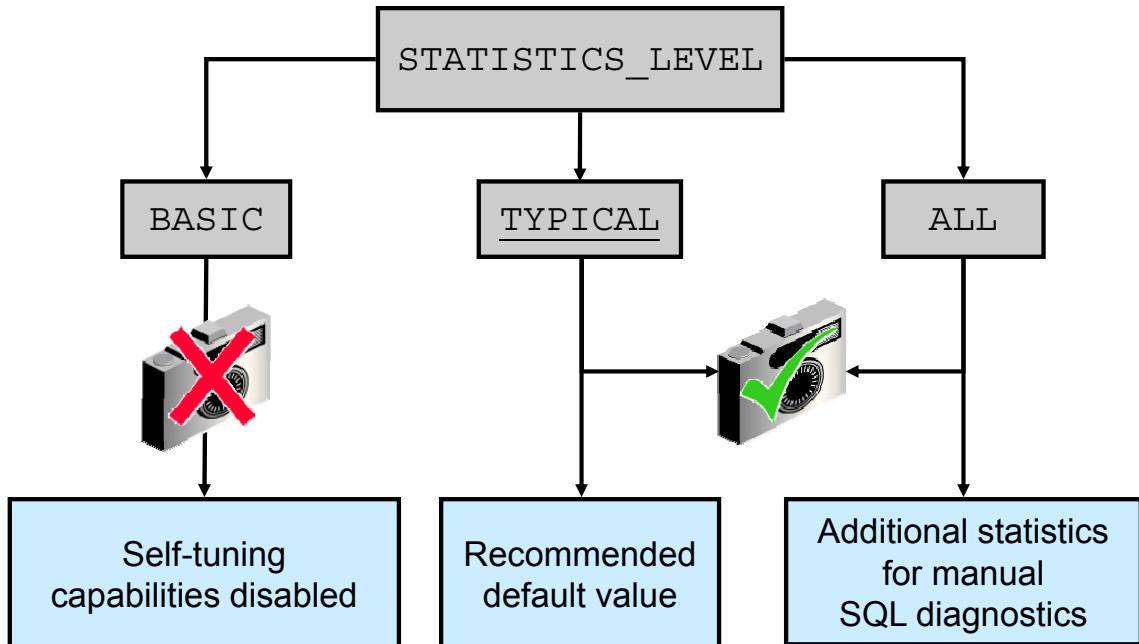
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

AWR settings include retention period, collection interval, and collection level. Remember that decreasing any of these settings affects the functionality of components that depend on the AWR, including the advisors.

Increasing the settings can provide improved advisor recommendations—but at the cost of the space that is required to store the snapshots and the performance expended in collecting the snapshot information.

Consider setting collection level to `ALL` when tuning a new application. The `ALL` setting collects SQL execution plans and timing statistics that enhance the recommendations of the SQL advisors. When tuning is complete, this setting should be returned to the `TYPICAL` setting.

Statistic Levels



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

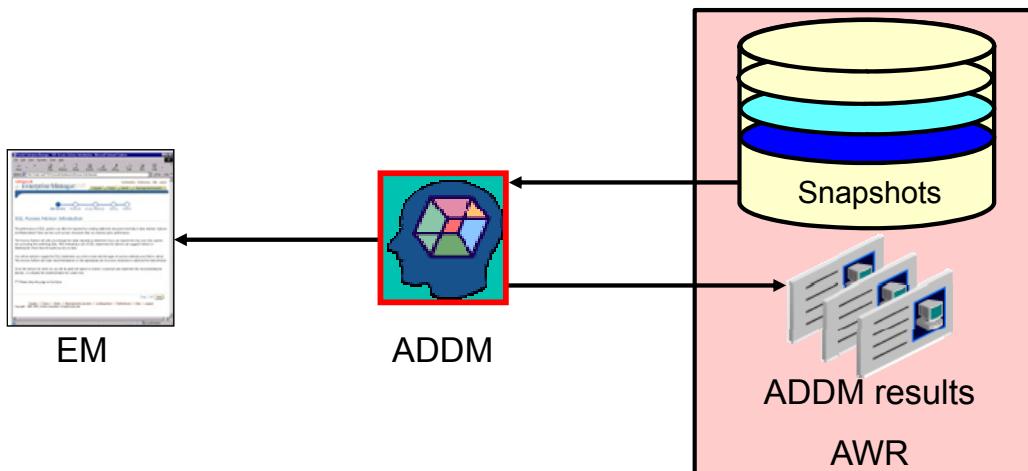
The STATISTICS_LEVEL initialization parameter controls the capture of a variety of statistics and various advisors, including the automatic maintenance tasks. The automatic maintenance tasks include gathering optimizer statistics. The STATISTICS_LEVEL parameter can be set to the following levels:

- **BASIC:** The computation of AWR statistics and metrics is turned off. The automatic optimizer statistics task is disabled, as are all advisors and server-generated alerts.
- **TYPICAL:** Major statistics that are required for database self-management are collected. They represent what is typically needed to monitor Oracle database behavior. This includes automatic gathering of statistics to reduce the likelihood of poorly performing SQL statements due to stale or invalid statistics.
- **ALL:** All possible statistics are captured. This level of capture adds timed OS statistics and plan execution statistics. These statistics are not needed in most cases and should not be enabled for best performance; they are sometimes needed for specific diagnostics tests.

Oracle recommends that the default value of TYPICAL be set for the STATISTICS_LEVEL initialization parameter. Setting the value to BASIC disables the automatic gathering of optimizer statistics.

Automatic Database Diagnostic Monitor (ADDM)

- Runs after each AWR snapshot
- Monitors the instance; detects bottlenecks
- Stores results in the AWR



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Unlike the other advisors, the ADDM runs automatically after each AWR snapshot. Each time a snapshot is taken, the ADDM performs an analysis of the period corresponding to the last two snapshots. The ADDM proactively monitors the instance and detects most bottlenecks before they become a significant problem.

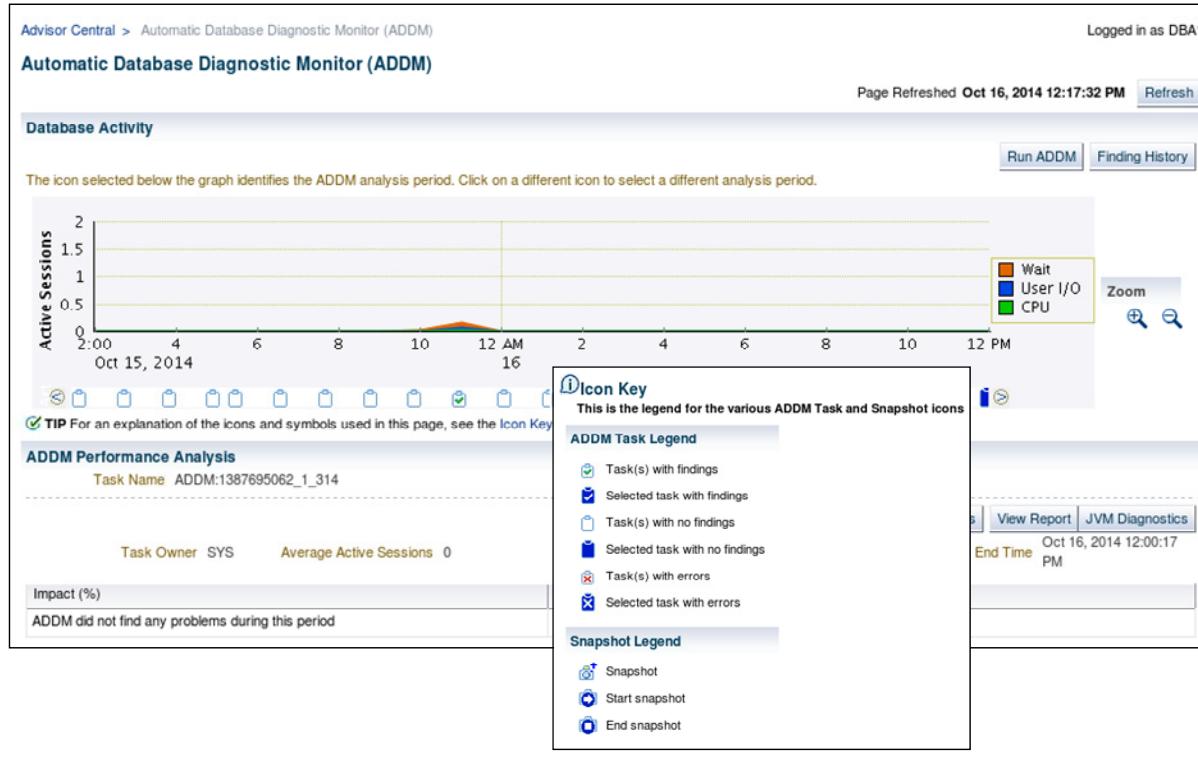
In many cases, the ADDM recommends solutions for detected problems and even quantifies the benefits for the recommendations.

Some common problems that are detected by the ADDM:

- CPU bottlenecks
- Poor Oracle Net connection management
- Lock contention
- Input/output (I/O) capacity
- Undersizing of database instance memory structures
- High-load SQL statements
- High PL/SQL and Java time
- High checkpoint load and cause (for example, small log files)

The results of each ADDM analysis are stored in the AWR and are also accessible through Enterprise Manager.

ADDM Findings in Enterprise Manager Cloud Control



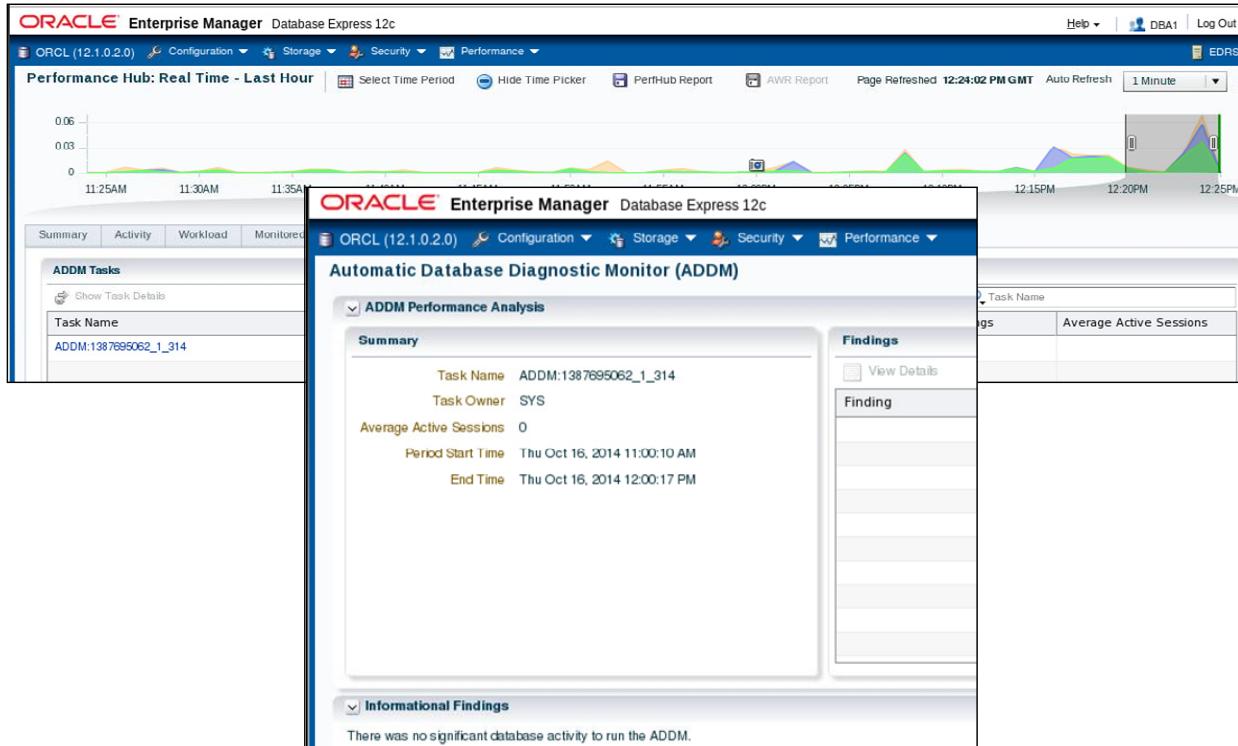
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can use Enterprise Manager Cloud Control to view ADDM findings. Select Advisors Home in the Performance menu. Select an ADDM autorun on the Advisors Central page.

On the Automatic Database Diagnostic Monitor (ADDM) page, you see detailed findings for the latest ADDM run. Database Time represents the sum of the non-idle time spent by sessions in the database for the analysis period. A specific impact percentage is given for each finding. The impact represents the time consumed by the corresponding issue compared with the database time for the analysis period.

Click the View Report button to get details about the performance analysis in the form of a text report.

ADDM Findings in Enterprise Manager Database Express

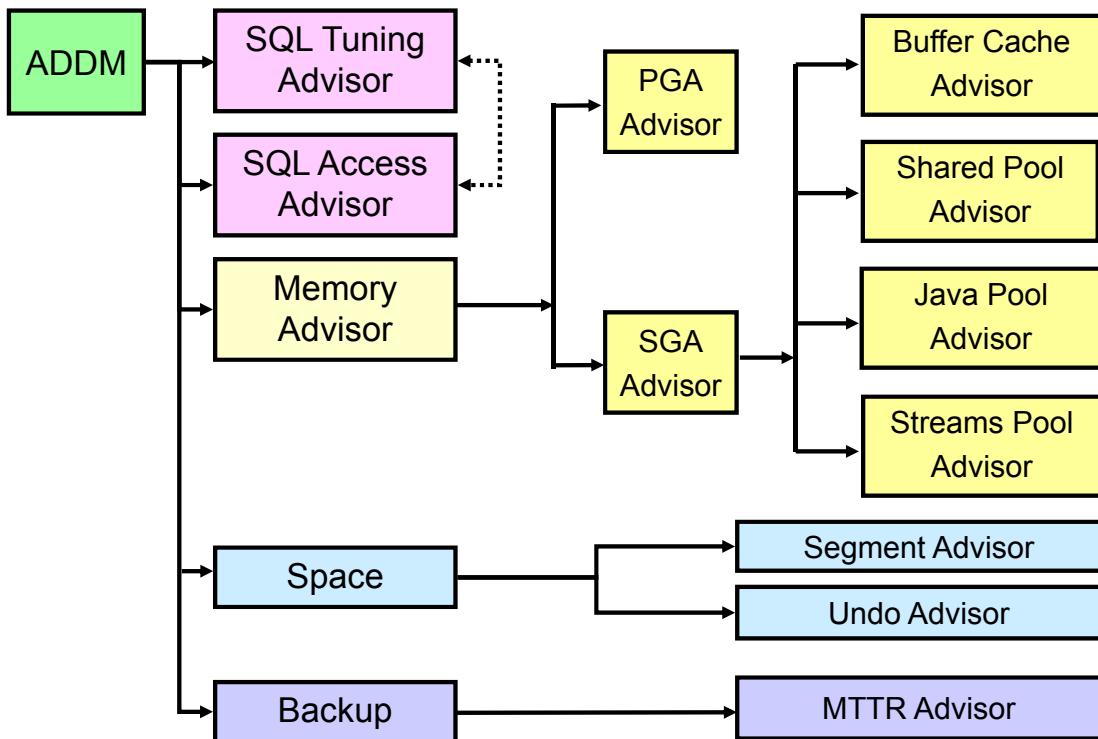


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can also use Enterprise Manager Database Express to view ADDM information. Select Performance Hub from the Performance menu. Click the ADDM tab. Select a task to view detailed information.

Advisory Framework



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Advisors provide you with useful feedback about resource utilization and performance for their respective server components. For example, the Memory Advisor provides a recommended value for the `MEMORY_TARGET` initialization parameter, which controls the total amount of memory used by the Oracle database instance.

By building on the data captured in the AWR, the ADDM enables the Oracle Database server to diagnose its own performance and determine how identified problems can be resolved. ADDM runs automatically after each AWR statistics capture. It can potentially call other advisors.

Here are the major benefits that are provided by the advisor infrastructure:

- All advisors use a uniform interface.
- All advisors have a common data source and results storage by using the workload repository.

Not all advisors are shown in the slide (for example, the Data Recovery Advisor and the SQL Repair Advisor are not listed).

Automatic Database Diagnostic Monitor (ADDM)

The ADDM is a server-based expert that reviews database performance every 60 minutes. Its goal is to detect possible system bottlenecks early and recommend fixes before system performance degrades noticeably.

Memory Advisors

The Memory Advisor is actually a collection of several advisory functions that help determine the best settings for the total memory used by the database instance. The System Global Area (SGA) has a set of advisors for the shared pool, database buffer cache, Java pool, and streams pool. The Java pool and streams pool advisors are not exposed on the Enterprise Manager Memory Advisor page. There is an advisor for the Program Global Area (PGA). In addition to the advisory functions, this advisor provides a central point of control for the large pool and the Java pool.

Mean-Time-To-Recover (MTTR) Advisor

Using the MTTR Advisor, you set the length of time required for the database to recover after an instance crash.

Segment Advisor

This advisor looks for tables and indexes that consume more space than they require. The advisor checks for inefficient space consumption at the tablespace or schema level and produces scripts to reduce space consumption where possible.

SQL Access Advisor

This advisor analyzes all SQL statements that are issued in a given period and suggests the creation of additional indexes or materialized views that will improve performance.

SQL Tuning Advisor

This advisor analyzes an individual SQL statement and makes recommendations for improving its performance. Recommendations may include actions, such as rewriting the statement, changing the instance configuration, or adding indexes.

Undo Management Advisor

With the Undo Management Advisor, you can determine the undo tablespace size that is required to support a given retention period. Undo management and the use of the advisor is covered in the lesson titled “Managing Undo Data.”

Data Recovery Advisor

This advisor automatically diagnoses persistent data failures, presents repair options to the user, and executes repairs at the user’s request. The purpose of the Data Recovery Advisor is to reduce the mean time to recover (MTTR) and provide a centralized tool for automated data repair.

SQL Repair Advisor

You run the SQL Repair Advisor after a SQL statement fails with a critical error that generates a problem in the Automatic Diagnostic Repository. The advisor analyzes the statement and, in many cases, recommends a patch to repair the statement. If you implement the recommendation, the applied SQL patch circumvents the failure by causing the query optimizer to choose an alternative execution plan for future executions. This is done without changing the SQL statement itself.

Viewing the Advisor Central Page in Enterprise Manager Cloud Control

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Advisor Central page is the main page of all advisors. You can reach this page by selecting Advisors Home in the Performance menu. It is also possible to have access to advisors in certain contexts.

On the Advisors tab of the Advisor Central page, you can see all the advisor tasks that are registered in the workload repository. You can also filter this list by advisor type and for predefined time periods.

The Checkers tab of the Advisor Central page enables you to schedule various database integrity checkers. You can see all the checker runs by name, type, or time period.

Some advisors are described in greater detail in the lessons titled “Managing Undo Data,” “Managing Performance,” “Managing Performance: SQL Tuning,” and “Backup and Recovery Concepts.”

Note: Use the Change Default Parameters page to change the default expiration (in days) for all future tasks. You can also use this page to change the parameters of some important advisors.

Using Packages to Invoke the Advisors

Package Name	Advisor Name
DBMS_ADDM	Automatic Database Diagnostic Monitor (DBMS_ADDM)
DBMS_ADVISOR	SQL Access Advisor and Segment Advisor
DBMS_COMPRESSION	Compression Advisor
DBMS_SQLDIAG	SQL Repair Advisor
DBMS_SQLPA	SQL Performance Analyzer
DBMS_SQLTUNE	SQL Tuning Advisor



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The DBMS_ADVISOR package contains all constants and procedure declarations for all advisor modules. However, some of the advisors have their own separate packages. Oracle recommends that you use the advisor-specific package rather than DBMS_ADVISOR for the following advisors:

- Automatic Database Diagnostic Monitor (DBMS_ADDM)
- SQL Performance Analyzer (DBMS_SQLPA)
- SQL Repair Advisor (DBMS_SQLDIAG)
- SQL Tuning Advisor (DBMS_SQLTUNE)
- Compression Advisor (DBMS_COMPRESSION.GET_COMPRESSION_RATIO)

You can use DBMS_ADVISOR and the other advisor packages to execute tasks from the command line.

To execute advisor procedures, you must be granted the ADVISOR privilege. The ADVISOR privilege permits full access to the advisor procedures and views.

Note: For more information about all the procedures found in the advisor packages, refer to the *Oracle Database PL/SQL Packages and Types Reference*.

Automated Maintenance Tasks

Autotask maintenance process:

1. Maintenance Window opens.
2. Autotask background process schedules jobs.
3. Scheduler initiates jobs.
4. Resource Manager limits impact of Autotask jobs.

Default Autotask maintenance jobs:

- Gathering optimizer statistics
- Automatic Segment Advisor
- Automatic SQL Advisor



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

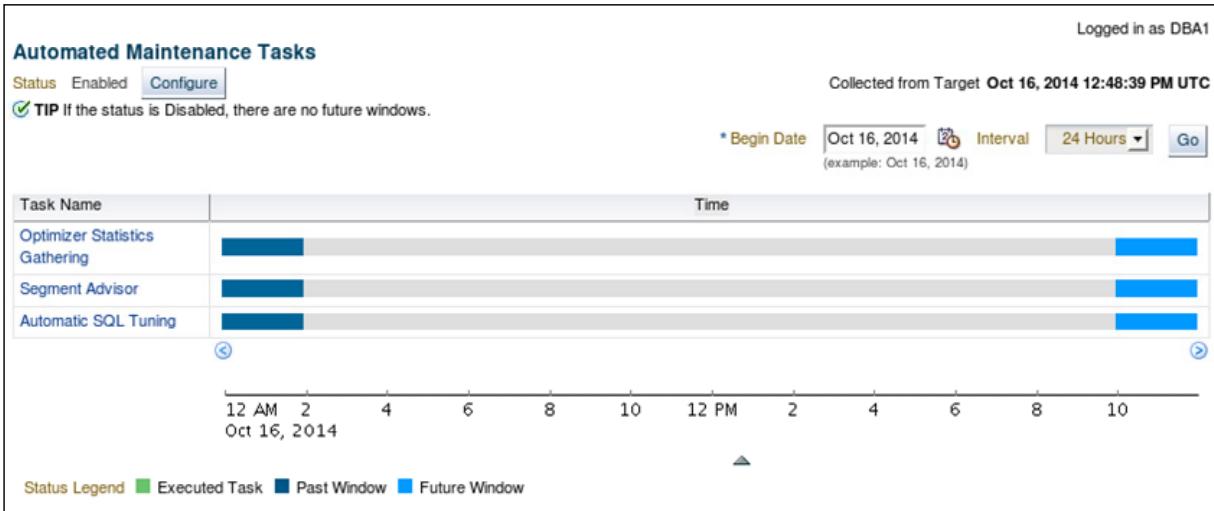
By analyzing the information stored in the AWR, the database server can identify the need to perform routine maintenance tasks, such as optimizer statistics refresh. The automated maintenance tasks infrastructure enables the Oracle Database server to automatically perform such operations. It uses the Scheduler to run such tasks in predefined maintenance windows.

By default, the weekday maintenance windows start at 10:00 PM and last four hours. On Saturday and Sunday, the maintenance window starts at 6:00 AM and lasts for 20 hours. All attributes of the maintenance windows are customizable, including the start and end times, frequency, days of the week, and so on. In addition, the impact of automated maintenance tasks on normal database operations can be limited by associating a Database Resource Manager resource plan to the maintenance window.

Examples of maintenance:

- Optimizer statistics are automatically refreshed by using the automatic maintenance task infrastructure.
- The Automatic Segment Advisor has default jobs, which run in the maintenance window.
- When creating a database with the DBCA, you can initiate regular database backups.

Automated Maintenance Tasks



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To access the Automated Maintenance Task page, expand the Administration menu and select Automated Maintenance Tasks in the Oracle Scheduler submenu. On this page, you can view the automated maintenance task schedule and recent history. From here, you can drill down to details on some tasks. Click Configure to go to the Automated Maintenance Tasks Configuration page. A task executes in a window. The graph shows the last window in which a task was executed and the next window in which the task is scheduled to be executed.

Note: The default windows for tasks are shown in the example. When the maintenance window closes, the Scheduler terminates the optimizer statistics-gathering job by default. The remaining objects are then processed in the next maintenance window.

Automated Maintenance Tasks Configuration

Window	Optimizer Statistics Gathering	Segment Advisor	Automatic SQL Tuning
THURSDAY_WINDOW	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
FRIDAY_WINDOW	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
SATURDAY_WINDOW	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
SUNDAY_WINDOW	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
MONDAY_WINDOW	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
TUESDAY_WINDOW	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
WEDNESDAY_WINDOW	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

ORACLE®

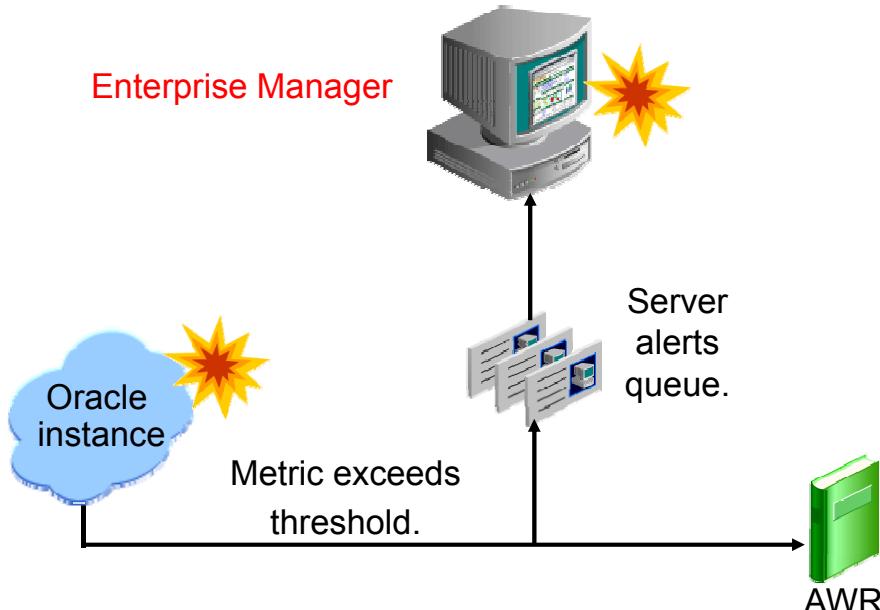
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

On the Automated Maintenance Tasks Configuration page, you can enable and disable automatic maintenance tasks—all at once, by individual tasks, or by particular windows. You can also configure the settings that are used for optimizer statistics gathering and the job control parameters for the automatic SQL Tuning Advisor.

Select the window name to view or edit the window schedule.

Click Edit Window Group to add and remove windows in the window group.

Server-Generated Alerts



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Alerts are notifications of when a database is in an undesirable state and needs your attention. By default, the Oracle Database server provides alerts via Enterprise Manager. Optionally, Enterprise Manager can be configured to send an email message to the administrator about problem conditions as well as display alert information on the console.

You can also set thresholds on many of the pertinent metrics for your system. Oracle Database proactively notifies you if the database deviates sufficiently from normal readings to reach those thresholds. An early notification of potential problems enables you to respond quickly and, in many cases, resolve issues before users even notice them.

Approximately, 60 metrics are monitored by default, among which are:

- Broken Job Count
- Database Time Spent Waiting (%)
- Dump Area Used (%)
- SQL Response Time (%) compared to baseline
- Tablespace Used (%)
- Generic Incident

A few additional key metrics can provide early problem notification:

- Average File Read Time (centiseconds)
- Response Time (per transaction)
- Wait Time (%)

Setting Metrics Thresholds

The screenshot shows the 'Metric and Collection Settings' page for the 'orcl' database instance. The 'Metrics' tab is selected. A dropdown menu 'View' is set to 'Metrics with thresholds'. The main area displays a table of metrics with their corresponding threshold settings. The table columns are: Metric, Comparison Operator, Warning Threshold, Critical Threshold, Corrective Actions, Collection Schedule, and Edit. The 'Edit' column contains icons for modifying each row. The metrics listed under 'Alert Log' include 'Archiver Hung Alert Log Error', 'Data Block Corruption Alert Log Error', 'Generic Alert Log Error', 'Media Failure Alert Log Error', and 'Session Terminated Alert Log Error'. The metrics listed under 'Alert Log Error Status' include 'Archiver Hung Alert Log Error Status' and 'Data Block Corruption Alert Log Error Status'. The 'Collection Schedule' column for all rows is currently 'Disabled'.

Metric	Comparison Operator	Warning Threshold	Critical Threshold	Corrective Actions	Collection Schedule	Edit
orcl						
Alert Log					Disabled	
Archiver Hung Alert Log Error	Contains		ORA-	None		
Data Block Corruption Alert Log Error	Contains		ORA-	None		
Generic Alert Log Error	Matches	ORA-0*(600?)		None		
Media Failure Alert Log Error	Contains		ORA-	None		
Session Terminated Alert Log Error	Contains	ORA-		None		
Alert Log Error Status					Disabled	
Archiver Hung Alert Log Error Status	>	0		None		
Data Block Corruption Alert Log Error Status	>	0		None		

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To access the Metric and Collection Settings page, expand the Oracle Database menu and select Metric and Collection Settings from the Monitoring submenu.

Enter your desired warning and critical threshold values for the metric. The appropriate alerts appear when the database reaches your specified values.

The thresholds that are already set appear in the “Metrics with thresholds” list. By default, approximately 60 metrics have preset thresholds; you may change these as needed. The “All metrics” list shows the metrics that do not have thresholds set.

Click the Edit icon to access a page where you can specify additional corrective actions for either warning or critical thresholds.

Click a Collection Schedule link to change the scheduled collection interval. Be aware that each schedule affects a group of metrics.

Reacting to Alerts

- If necessary, you should gather more input (for example, by running ADDM or another advisor).
- Investigate critical errors.
- Take corrective measures.
- Acknowledge alerts that are not automatically cleared.



ORACLE®

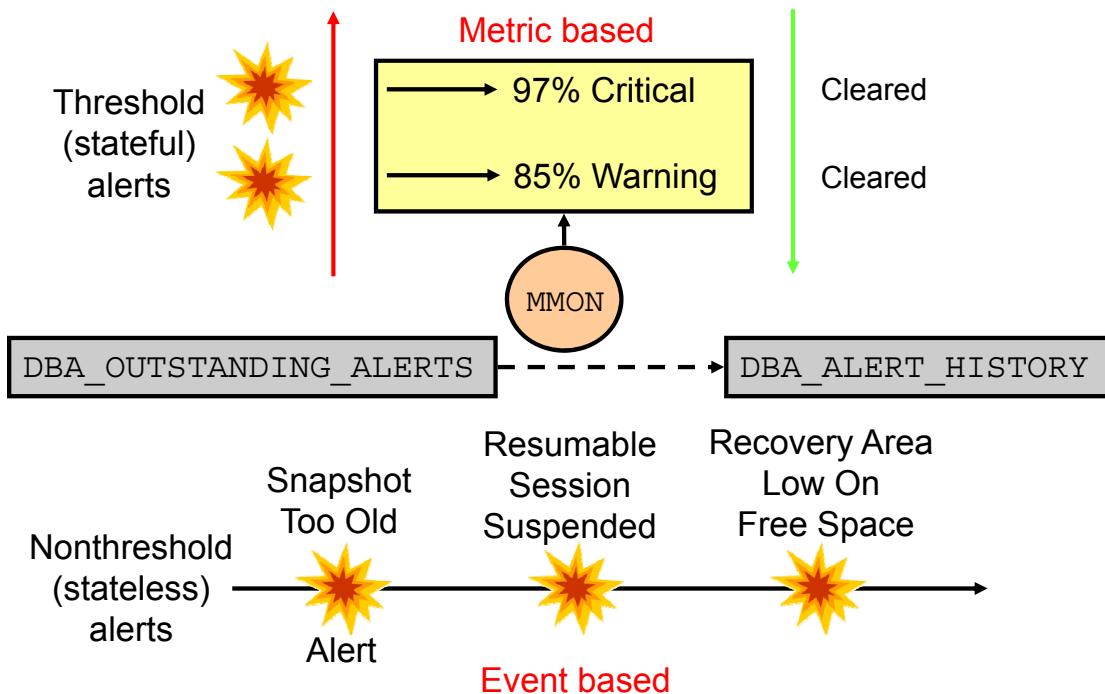
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

When you receive an alert, follow the recommendations that it provides. Or you can consider running the ADDM (or another advisor as appropriate) to obtain more detailed diagnostics of system or object behavior.

Alerts and incidents are generated for critical errors. Critical errors usually generate incidents that are collected into problems. You use the Support Workbench to investigate and possibly report the problem to Oracle Support.

Most alerts (such as “Out of Space”) are cleared automatically when the cause of the problem disappears. However, other alerts (such as Generic Alert Log Error) are sent to you for notification and must be acknowledged by you. After taking the necessary corrective measures, you acknowledge an alert by clearing or purging it. Clearing an alert sends the alert to the Alert History, which is accessible from the Monitoring submenu. Purging an alert removes it from the Alert History.

Alert Types and Clearing Alerts



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

There are two kinds of server-generated alerts: threshold and nonthreshold.

Most server-generated alerts are configured by setting a warning and critical threshold values on database metrics. You can define thresholds for more than 120 metrics, including the following:

- Physical Reads Per Sec
- User Commits Per Sec
- SQL Service Response Time

Except for the Tablespace Space Usage metric, which is database related, the other metrics are instance related. Threshold alerts are also referred to as *stateful alerts*, which are automatically cleared when an alert condition clears. Stateful alerts appear in DBA_OUTSTANDING_ALERTS and, when cleared, go to DBA_ALERT_HISTORY.

Other server-generated alerts correspond to specific database events, such as ORA-* errors, "Snapshot too old" errors, Recovery Area Low On Free Space, and Resumable Session Suspended. These are non-threshold-based alerts, also referred to as *stateless alerts*. Stateless alerts go directly to the history table.

Quiz

Stateless alerts, such as SNAPSHOT TOO OLD can be found in the dictionary view DBA_OUTSTANDING_ALERTS.

- a. True
- b. False



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b

Summary

In this lesson, you should have learned how to:

- Manage the Automatic Workload Repository (AWR)
- Use the Automatic Database Diagnostic Monitor (ADDM)
- Describe and use the advisory framework
- Set alert thresholds
- Use server-generated alerts
- Use automated tasks



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice: Overview

This practice covers proactively managing your database with ADDM, including:

- Setting up an issue for analysis
- Reviewing your database performance
- Implementing a solution



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

18

Managing Performance

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

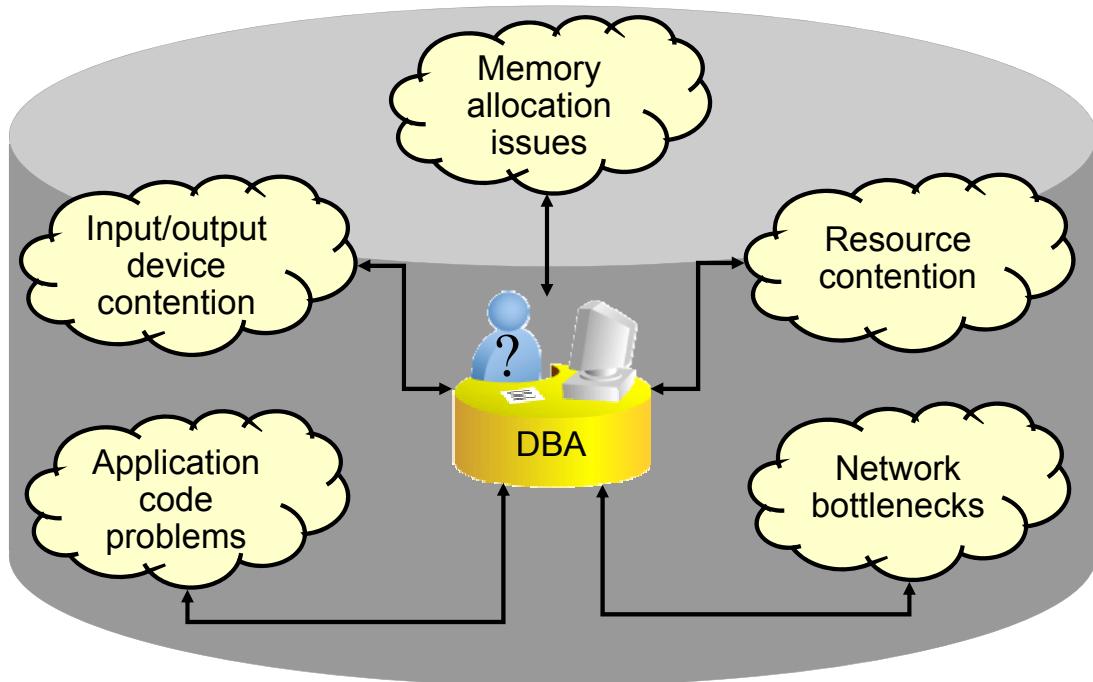
After completing this lesson, you should be able to use:

- Enterprise Manager to monitor performance
- Automatic Memory Management (AMM)
- The Memory Advisor to size memory buffers



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Performance Monitoring



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To administer Oracle Database and keep it running smoothly, the database administrator (DBA) must regularly monitor its performance to locate bottlenecks and to correct problem areas.

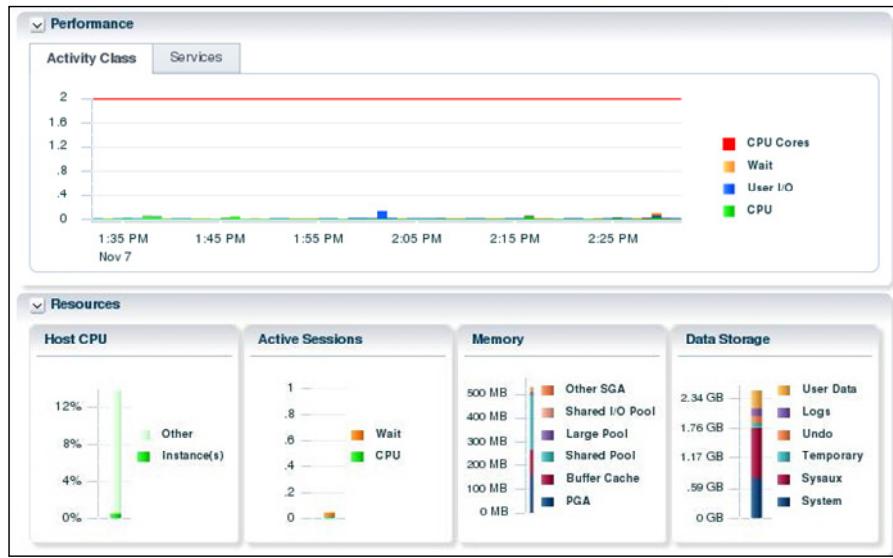
A DBA can look at hundreds of performance measurements, covering everything from network performance and disk input/output (I/O) speed to the time spent working on individual application operations. These performance measurements are commonly referred to as *database metrics*.

Note: For more information about Oracle database performance, see the *Oracle Database 12c: Performance Tuning* course.

Performance Monitoring

Use the Enterprise Manager Database Express home page for:

- Performance overview
- Graphs of metrics and details



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can respond to changes in performance only if you know the performance has changed. Oracle Database provides several ways to monitor the current performance of the database instance. The database home page of Enterprise Manager Database Express provides a quick check of the health of the instance and the server, with graphs showing CPU usage, active sessions, memory and data storage usage. The home page also shows any alerts that have been triggered.

Additional detail is available on the Performance Hub page. This page will be reviewed later in the lesson.

The information displayed in Enterprise Manager is based on performance views that exist in the database. You can access these views directly with SQL*Plus. Occasionally, you may need to access these views for some detail about the raw statistics.

Tuning Activities

The three activities in performance management are:

- Performance planning
- Instance tuning
- SQL tuning



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

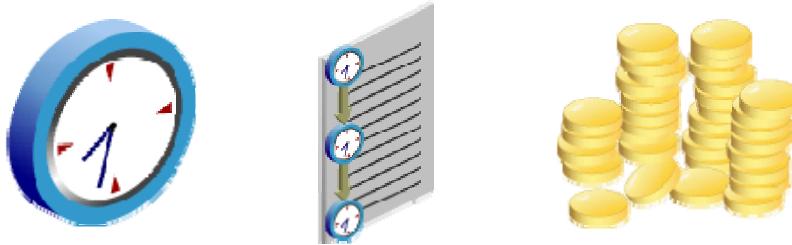
The three facets of tuning involve performance planning, instance tuning, and SQL tuning.

- 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 (OS) parameters to gain better performance of the Oracle database.
- SQL tuning involves making your application submit efficient SQL statements. SQL tuning is performed for the application as a whole, as well as for individual statements. At the application level, 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.

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 © 2014, Oracle and/or its affiliates. All rights reserved.

There are many facets to performance planning. Planning must include a balance between performance (speed), cost, and reliability. You must consider the investment in your system architecture: the hardware and software infrastructure needed to meet your requirements. This, of course, requires analysis to determine the value for your given environment, application, and performance requirements. For example, the number of hard drives and controllers has an impact on the speed of data access.

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 of scalability is serializing operations among users. If all users 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.

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.

A difficult and necessary task is testing the existing applications when changing the infrastructure—for example, upgrading the database to a newer version, or changing the operating system or server hardware. Before the application is deployed for production in the new configuration, you want to know the impact. The application will almost certainly require additional tuning. You need to know that the critical functionality will perform, without errors.

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 © 2014, Oracle and/or its affiliates. All rights reserved.

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 server 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:
 1. Design
 2. Application code
 3. Instance
- Tune the area with the greatest potential benefit. Identify and tune:
 - SQL using the greatest resources
 - The longest waits
 - The largest service times
- Stop tuning when the goal is met.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle has developed a tuning methodology based on years of experience. The basic steps are:

1. Check the OS statistics and general machine health before tuning the instance to be sure that the problem is in the database. Use the Enterprise Manager database home page.
2. Tune from the top down. Start with the design, then the application, and then the instance. For example, try to eliminate the full table scans that cause the I/O contention before tuning the tablespace layout on disk. This activity often requires access to the application code.
3. 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. The Advisors available in Oracle Database use this methodology.
4. 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.

Performance Tuning Data

Type of data gathered:

- Cumulative statistics:
 - Wait events with time information
 - Time model
- Metrics: Statistic rates
- Sampled statistics: Active session history
 - Statistics by session
 - Statistics by SQL
 - Statistics by service
 - Other dimensions



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

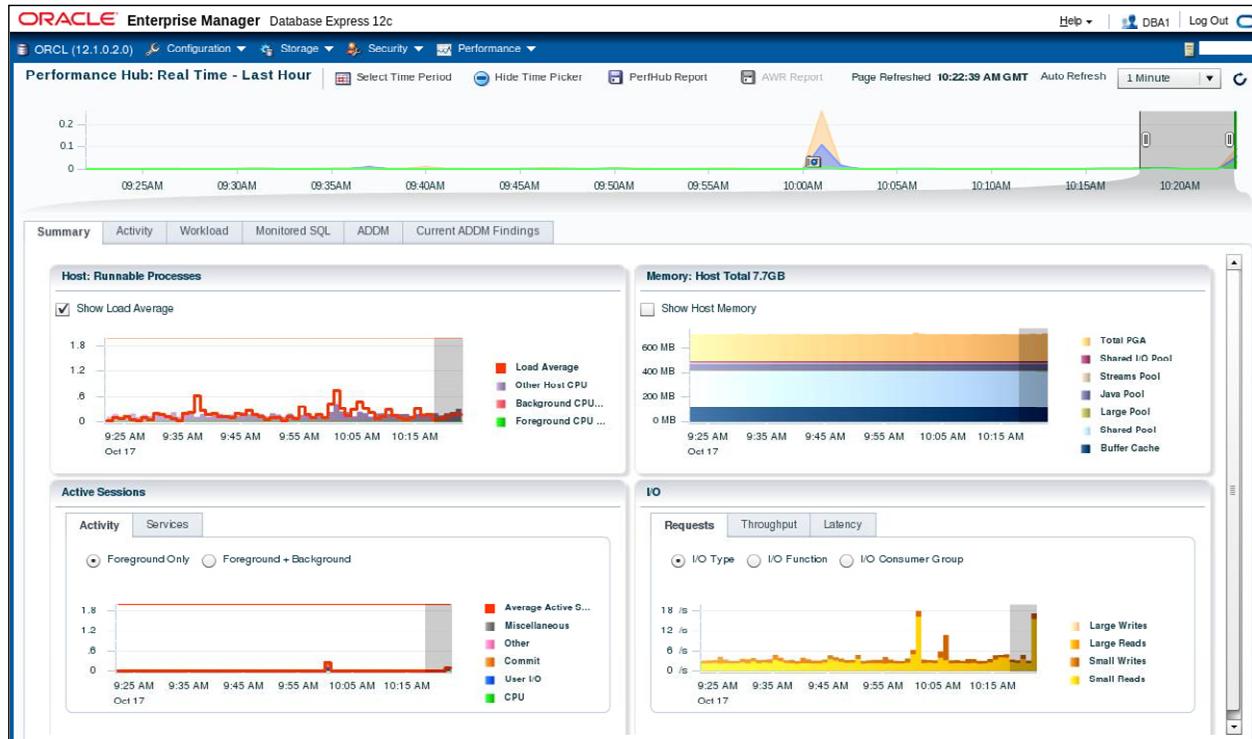
The Oracle database server software captures information about its own operation. Three major types of data are collected: cumulative statistics, metrics, and sampled statistics.

Cumulative statistics are counts and timing information of a variety of events that occur in the database server. Some are quite important, such as buffer busy waits. Others have little impact on tuning, such as index block split. The most important events for tuning are usually the ones showing the greatest cumulative time values. The statistics in Oracle Database are correlated by the use of a time model. The time model statistics are based on a percentage of DB time, giving them a common basis for comparison.

Metrics are statistic counts per unit. The unit could be time (such as seconds), transaction, or session. Metrics provide a base to proactively monitor performance. You can set thresholds on a metric, causing an alert to be generated. For example, you can set thresholds for when the reads per millisecond exceed a previously recorded peak value or when the archive log area is 95% full.

Sampled statistics are gathered automatically when `STATISTICS_LEVEL` is set to `TYPICAL` or `ALL`. Sampled statistics allow you to look back in time. You can view session and system statistics that were gathered in the past, in various dimensions, even if you had not thought of specifying data collection for these beforehand.

Using the Enterprise Manager Database Express Performance Hub Page



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can access the Performance Hub page from the Performance menu. On the Performance Hub page, you can view all the performance data available for a specified time period.

You can choose to view real-time data or historical data. More granular data is shown when you choose real-time data. When you view historical data, the data points are averaged out to the Automatic Workload Repository (AWR) interval (usually an hour).

Different tabs are available on the Performance Hub page based on whether you choose real-time or historical data.

The following tabs are available for both real-time and historical data:

- **Summary:** Provides an overall view of the performance of the system for the specified time period
- **RAC:** Appears only when Enterprise Manager Database Express is being used with an Oracle Real Application Clusters (RAC) database or cluster database
- **Activity:** Shows Active Session History (ASH) analytics
- **Workload:** Profile charts show the pattern of user calls, parse calls, redo size, and SQL*Net over the last 60 minutes in real-time mode. The Sessions chart shows the logon rate, current logons, and open cursors.

- **Monitored SQL:** Shows information about monitored SQL statements that were executing or that completed during the selected time period
- **ADDM:** Shows performance findings and recommendations from the Automatic Database Diagnostics Monitor (ADDM) for tasks performed in the database during the selected time period

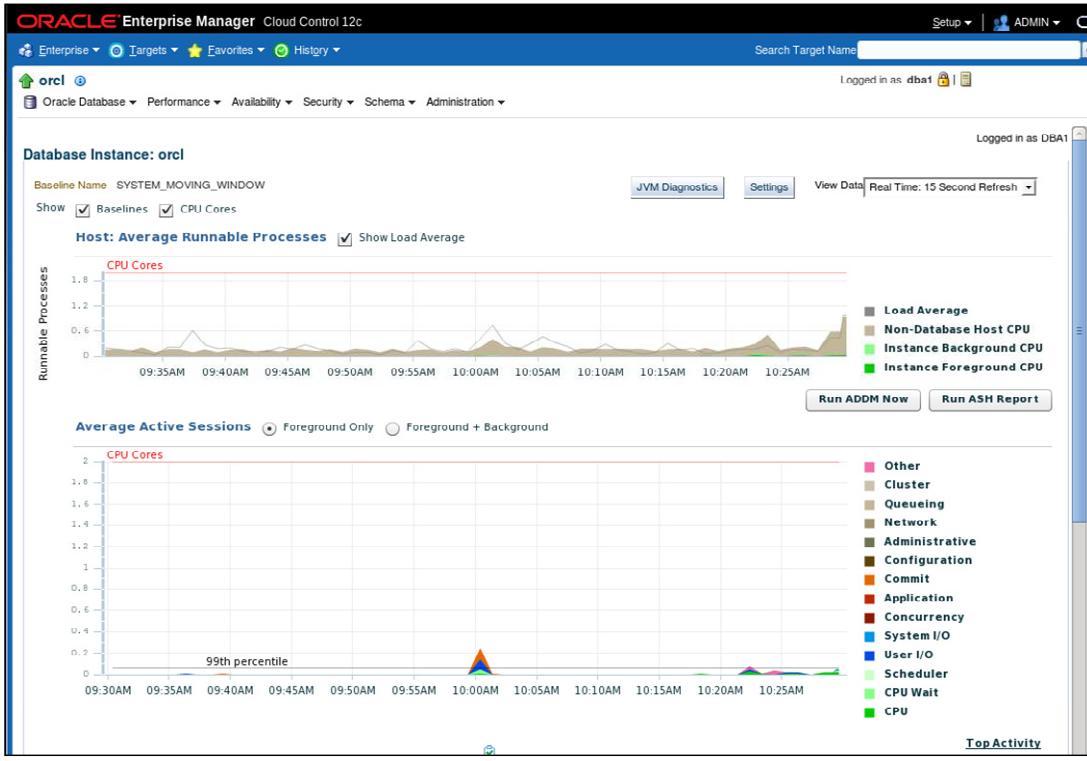
The following tab is only available if you select real-time data:

- **Current Findings:** Shows ADDM findings for the past five minutes

The following tabs are only available if you select historical data:

- **Database Time:** Shows wait events by category for various metrics, and to view time statistics for various metrics for the selected time period
- **Resources:** Shows operating system resource usage statistics, I/O resource usage statistics, and memory usage statistics for the selected time period
- **System Statistics:** Shows database statistics by value, per transaction, or per second for the selected time period

Using the Enterprise Manager Cloud Control Performance Home Page



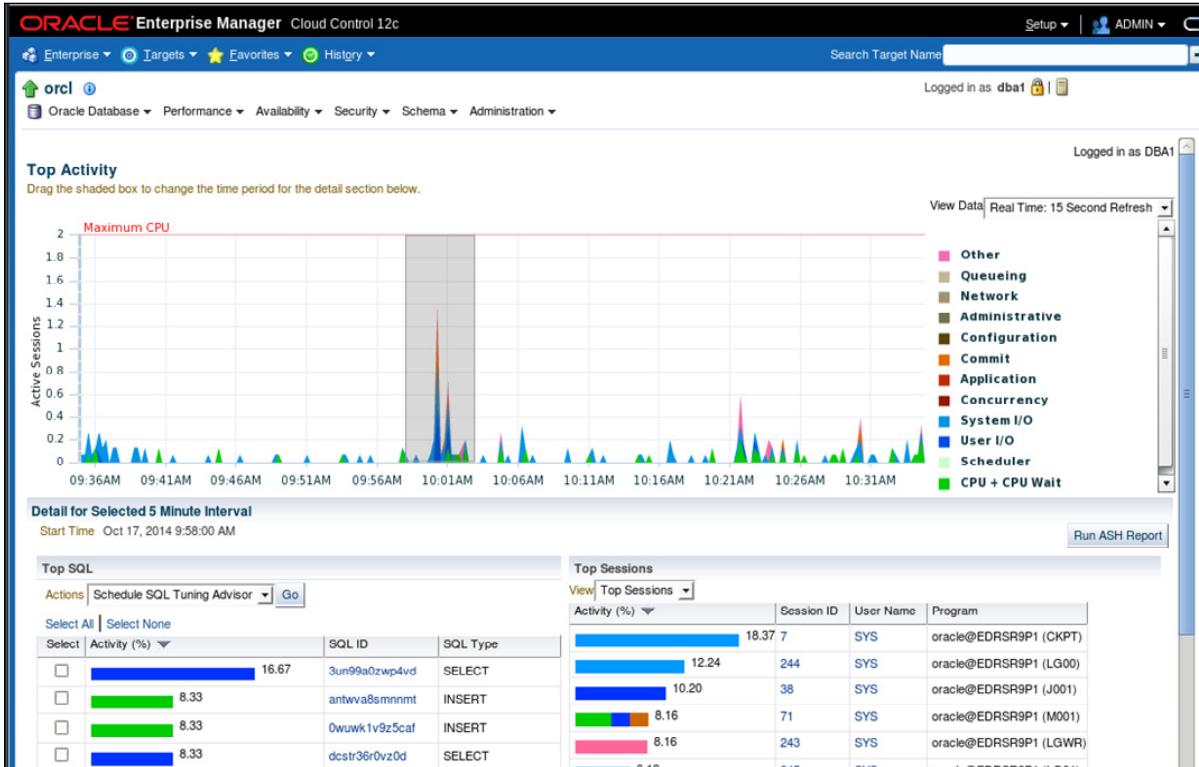
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

ORACLE

You can also use Enterprise Manager Cloud Control to manage the performance of your database. Access the Performance Home page by selecting Performance Home from the Performance menu.

The Performance Home page provides an overall view of the performance statistics for your database. You can use the information on this page to determine whether resources need to be added or redistributed.

Monitoring Session Performance



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Enterprise Manager Cloud Control provides session detail pages so that you can view the wait events occurring in individual sessions. Select Top Activity in the Performance menu to view the summary of all sessions. On the lower-right corner of the Top Activity page is a list of the Top Sessions. Click the session identifier to view the Session Details page.

The Top Activity page and Session Details page are based on performance views in the database.

Performance Monitoring: Top Sessions

Top Consumers												
Collected From Oct 17, 2014 10:38:36 AM U												
Overview Top Services Top Modules Top Actions Top Clients Top Sessions												
Kill Session View Disable SQL Trace Enable SQL Trace												
Select	SID	DB User	CPU (1/100 sec)	PGA Memory (bytes)	Physical Reads	Logical Reads	Hard Parses	Total Parses	Disk Sorts	Status	Program	Module
<input checked="" type="radio"/>	47	DBSNMP	22	10006632	1	459	7	50	0	ACTIVE	OMS	EM Realtime Connection
<input type="radio"/>	247	MMON	1	12243256	0	0	0	0	0	ACTIVE	oracle@EDRSR9P1 (MMON)	
<input type="radio"/>	32	DBSNMP	1	8678856	0	9	0	6	0	ACTIVE	JDBC Thin Client	emagent_SQL_oracle_database
<input type="radio"/>	240	QM00	0	2207848	0	0	0	0	0	ACTIVE	oracle@EDRSR9P1 (QM00)	Streams
<input type="radio"/>	241	DBRM	0	1888568	0	0	0	0	0	ACTIVE	oracle@EDRSR9P1 (DBRM)	
<input type="radio"/>	280	Q003	0	1290344	0	0	0	0	0	ACTIVE	oracle@EDRSR9P1 (Q003)	Streams



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

At the bottom of the Top Activity page, click Top Consumers in the Additional Monitoring Links section to access the Top Consumers page.

The Top Consumers Overview page shows in graphical format:

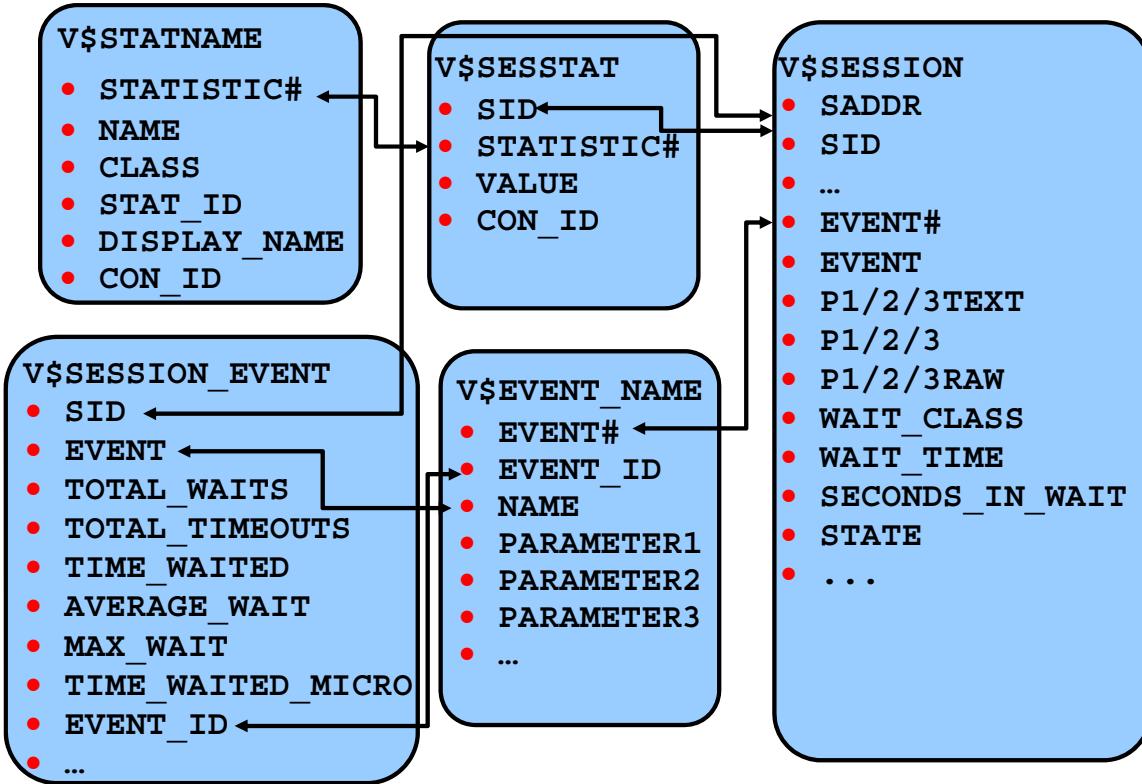
- Top services
- Top modules (by service)
- Top actions (by service and module)
- Top clients

On the Top Consumers page, click the Top Sessions tab to see critical statistics of the sessions that are using the most resources:

- CPU
- PGA Memory
- Logical Reads
- Physical Read
- Hard Parse count
- Sort count

Click a column name to have the results sorted by the value in that column.

Displaying Session-Related Statistics



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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.

You can view user session statistics in V\$SESSTAT. The V\$SESSION_EVENT view lists information about waits for an event by a session.

Cumulative values for instance statistics are generally available through dynamic performance views, such as V\$SESSTAT and V\$SYSSSTAT. 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 V\$SESSMETRIC 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.

Performance Monitoring: Top Services

The screenshot shows the 'Top Consumers' section of the Oracle Database Performance Monitoring interface. The top navigation bar includes tabs for Overview, Top Services (which is selected), Top Modules, Top Actions, Top Clients, and Top Sessions. A timestamp at the top right indicates 'Latest Data Collected From Target Oct 17, 2014 10:41:20 AM UTC' with a 'Refresh' button. Below the tabs, there's a dropdown menu set to 'Active Services' and buttons for 'Enable SQL Trace', 'Disable SQL Trace', and 'View SQL Trace File'. A checkbox row allows selecting all or none of the listed services. The main table displays two rows of service data:

Select	Service	Activity (%) for the last 5 minutes	SQL Trace Enabled	Delta Elapsed Time (seconds)	Cumulative Elapsed Time (seconds)	Delta CPU Time (seconds)	Cumulative CPU Time (seconds)
<input type="checkbox"/>	SYS\$BACKGROUND	61.9	FALSE	0	0	0	
<input type="checkbox"/>	SYS\$USERS	38.1	FALSE	0	9825	0	74

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In multitier systems where there is an application server that is pooling database connections, viewing sessions may not provide the information that you need to analyze performance. Grouping sessions into service names enables you to monitor performance more accurately. Regardless of the session that was used for a particular request, if it connected via one of these services, the performance data of the session is captured under that service name.

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



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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.

You can define a service in the database by using the `DBMS_SERVICE` package and use the net service name to assign applications to a service.

Viewing Wait Events



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can access detailed information on waits from the Average Active Sessions graph on the Performance Home page.

When you drill down to a particular wait category, you can view details of specific five-minute intervals and also see the Top Working SQL and the Top Working Sessions associated with that particular wait event during that time. This enables you to perform after-the-fact analysis of system slowdowns and determine potential causes.

Oracle Wait Events

- 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.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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.

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 © 2014, Oracle and/or its affiliates. All rights reserved.

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, other areas may not function properly because of lack of memory. 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.

Managing Memory Components

- Automatic Memory Management (AMM) enables you to specify total memory allocated to instance (including both SGA and PGA)
- Automatic Shared Memory Management (ASMM):
 - Enables you to specify total SGA memory through one initialization parameter
 - Enables the Oracle server to manage the amount of memory allocated to the shared pool, Java pool, buffer cache, streams pool, and large pool
- Manually setting shared memory management:
 - Sizes the components through multiple individual initialization parameters
 - Uses the appropriate Memory Advisor to make recommendations

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle Database enables you to specify the total memory allocated to the instance. Memory will be dynamically reallocated between the System Global Area (SGA) and Program Global Area (PGA) as needed. This method is called Automatic Memory Management (AMM) and is available on only those platforms that support dynamic release of memory. This simplifies your memory management tasks.

Memory advisors are available to help you set the initialization parameters on various levels. The advisor available depends on the level on which you are specifying the memory parameters. If you enable AMM, only the Memory Size Advisor is available.

Automatic Shared Memory Management (ASMM) enables you to manage the SGA as a whole. The SGA comprises several components. The sizes of many of these components are dynamically adjusted for best performance within the limits of the initialization parameters. When the AMM is enabled, the ASMM is automatically enabled. If the ASMM is enabled but not the AMM, the SGA Size Advisor is available.

You can manage the size of individual components manually by setting the initialization parameter for each component. If the Oracle server notifies you of a performance problem that is related to the size of an SGA or PGA component, you can use the Memory Advisor for the component to determine appropriate new settings. The Memory Advisor can model the effect of parameter changes.

Efficient Memory Usage: Guidelines

- Fit the SGA into physical memory.
- Use the Memory Advisors.
- Tune for the most efficient use of memory
 - Reduce overall physical I/O
 - Reduce the total memory needs



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

If possible, it is best to fit the SGA into physical memory, which provides the fastest access. Even though the OS may provide additional virtual memory, this memory, by its nature, can often be swapped out to disk. On some platforms, you can use the `LOCK_SGA` initialization parameter to lock the SGA into physical memory. This parameter cannot be used in conjunction with AMM or ASMM.

When a SQL statement executes, data blocks are requested for reading or writing, or both. This is considered a logical I/O. As the block is requested, the block is checked to see whether it already exists in memory. If it is not in memory, it is read from disk, which is called a physical I/O. When the block is found in memory, the cost is several orders of magnitude less than the cost of reading the block from disk. The size of the SGA components in combination with the workload has a large affect on the number of physical reads. A simple view of this implies that you should increase the memory for the SGA components as much as possible. A larger SGA is not always better. There is a point where adding more memory yields diminishing returns. This principle applies to the buffer cache, the shared pool, and other SGA components. In practice, you find that shifting memory from one SGA component to another may increase overall performance, without changing the total amount of memory given to the SGA depending on the characteristics of the workload.

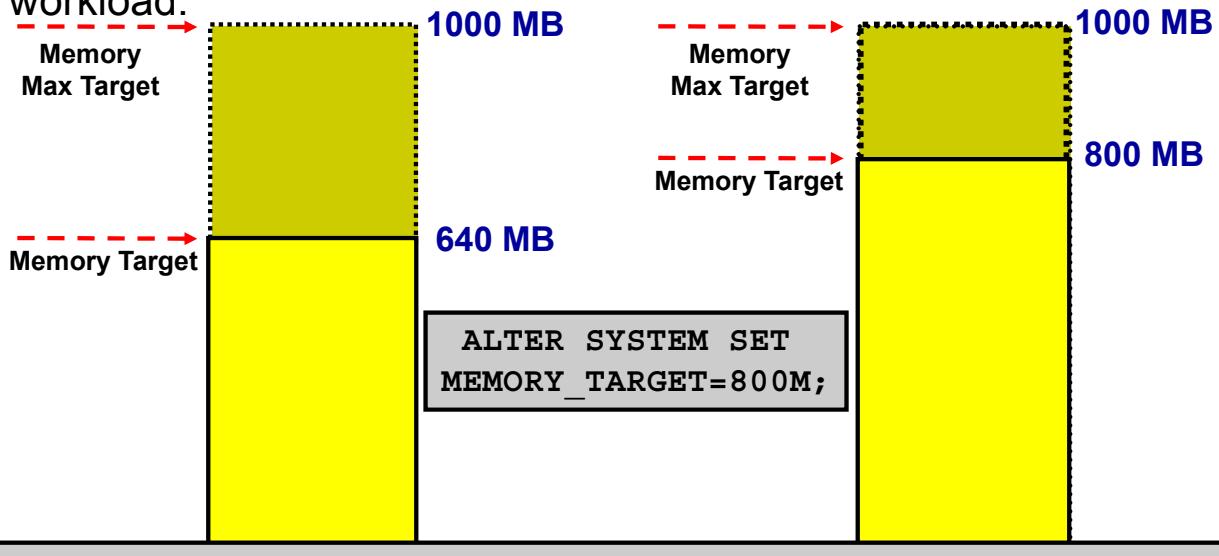
Memory has an upper limit in all current machines. That limit may be imposed by the hardware, operating system, or the cost of the memory. The goal of memory tuning is to produce the most efficient use of existing memory. When the workload changes often, the most efficient division of memory between the SGA components will change. There is also the amount of memory allocated to SGA and PGA. Online transaction processing (OLTP) systems typically use very little PGA memory compared to data warehouse (DW) or decision support systems (DSS).

Enterprise Manager Cloud Control and Enterprise Manager Database Express both provide Memory Advisors. These tools monitor the memory usage by the SGA components, and PGA, and project the differences in terms of efficiency for increased and decreased memory allocations. These projections use the current workload. They can help you size the SGA on the basis of the activity in your particular database. The advisors will make sizing recommendations for manual settings, when the automatic memory management is disabled. These same advisors provide input to the automatic memory management, to determine the most efficient component sizes.

Using the existing memory efficiently also includes tuning the applications. A poorly tuned application can use large quantities of memory. For example, an application that uses frequent full table scans because indexes do not exist or are unusable can cause a large amount of I/O, reducing performance. The first and most effective tuning technique is to tune high cost SQL statements. Tuning SQL statements is covered in more detail in the lesson titled “Managing Performance: SQL Tuning.”

Automatic Memory Management: Overview

With Automatic Memory Management, the database server can size the SGA and PGA automatically according to your workload.



Oracle recommends the use of AMM unless you have special requirements.

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

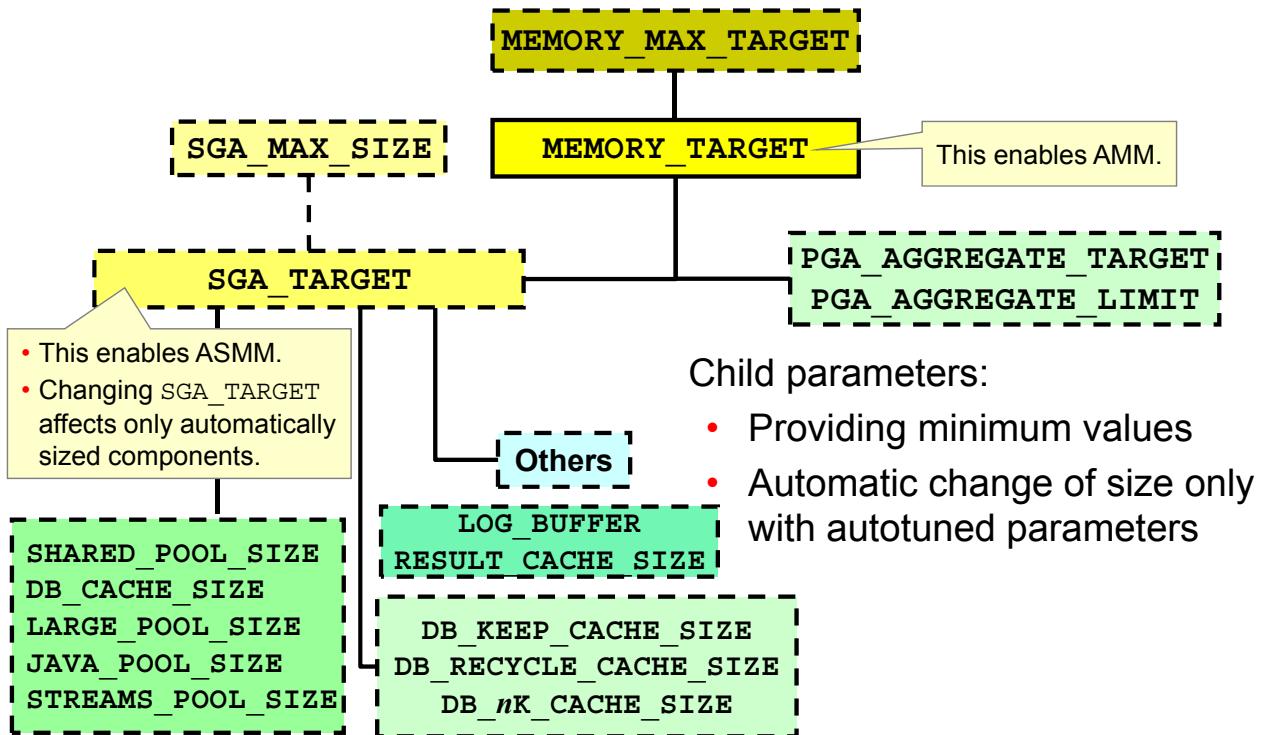
Automatic Memory Management (AMM) allows the Oracle Database server to manage SGA memory and instance PGA memory sizing automatically. To do so (on most platforms), you set only a target memory size initialization parameter (`MEMORY_TARGET`) and a maximum memory size initialization parameter (`MEMORY_MAX_TARGET`), and the database server dynamically exchanges memory between the SGA and the instance PGA as needed to meet processing demands.

With this memory management method, the database server also dynamically tunes the sizes of the individual SGA components and the sizes of the individual PGAs.

Because the target memory initialization parameter is dynamic, you can change the target memory size at any time without restarting the database instance. The maximum memory size serves as an upper limit so that you cannot accidentally set the target memory size too high. Because certain SGA components either cannot easily shrink or must remain at a minimum size, the database server also prevents you from setting the target memory size too low.

This indirect memory transfer relies on the operating system (OS) mechanism of freeing shared memory. After memory is released to the OS, the other components can allocate memory by requesting memory from the OS. Currently, Automatic Memory Management is implemented on Linux, Solaris, HPUX, AIX, and Windows.

Oracle Database Memory Parameters



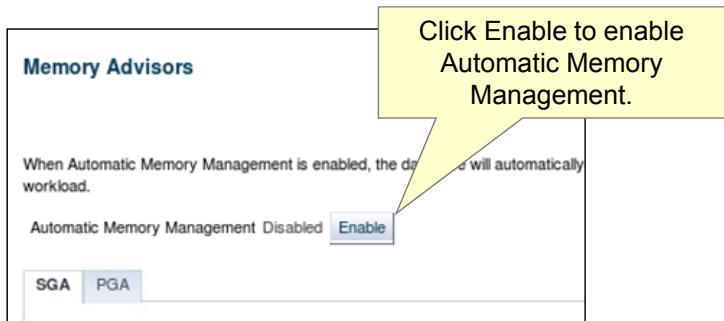
ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The graphic in the slide shows you the memory initialization parameters hierarchy. Although you have to set only **MEMORY_TARGET** to trigger Automatic Memory Management, you still have the ability to set lower bound values for various caches. Therefore, if the child parameters are user-set, they are the minimum values below which the Oracle database server will not autotune that component.

- If **SGA_TARGET** and **PGA_AGGREGATE_TARGET** are set to a nonzero value, they are considered to be the minimum values for the sizes of the SGA and the PGA, respectively. **MEMORY_TARGET** can take values from **SGA_TARGET + PGA_AGGREGATE_TARGET** to **MEMORY_MAX_SIZE**.
- If **SGA_TARGET** is set, the database server autotunes only the sizes of the subcomponents of the SGA. PGA is autotuned independent of whether it is explicitly set or not. However, the whole SGA (**SGA_TARGET**) and the PGA (**PGA_AGGREGATE_TARGET**) are not autotuned—that is, do not grow or shrink automatically.

Enabling Automatic Memory Management (AMM) by Using Enterprise Manager Cloud Control



Enable Automatic Memory Management
When Automatic Memory Management is enabled, the database will automatically set the optimal distribution of memory in the workload. The Maximum Memory Size specifies the maximum memory that the database may allocate and must be larger than the Current Memory Usage.

Current Memory Usage (PGA+SGA) (MB) 700
Maximum Memory Size 700 MB
Total Memory Size for Automatic Memory Management 700 MB

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

If you did not enable Automatic Memory Management (AMM) when you configured your database, you can enable it by performing the following steps:

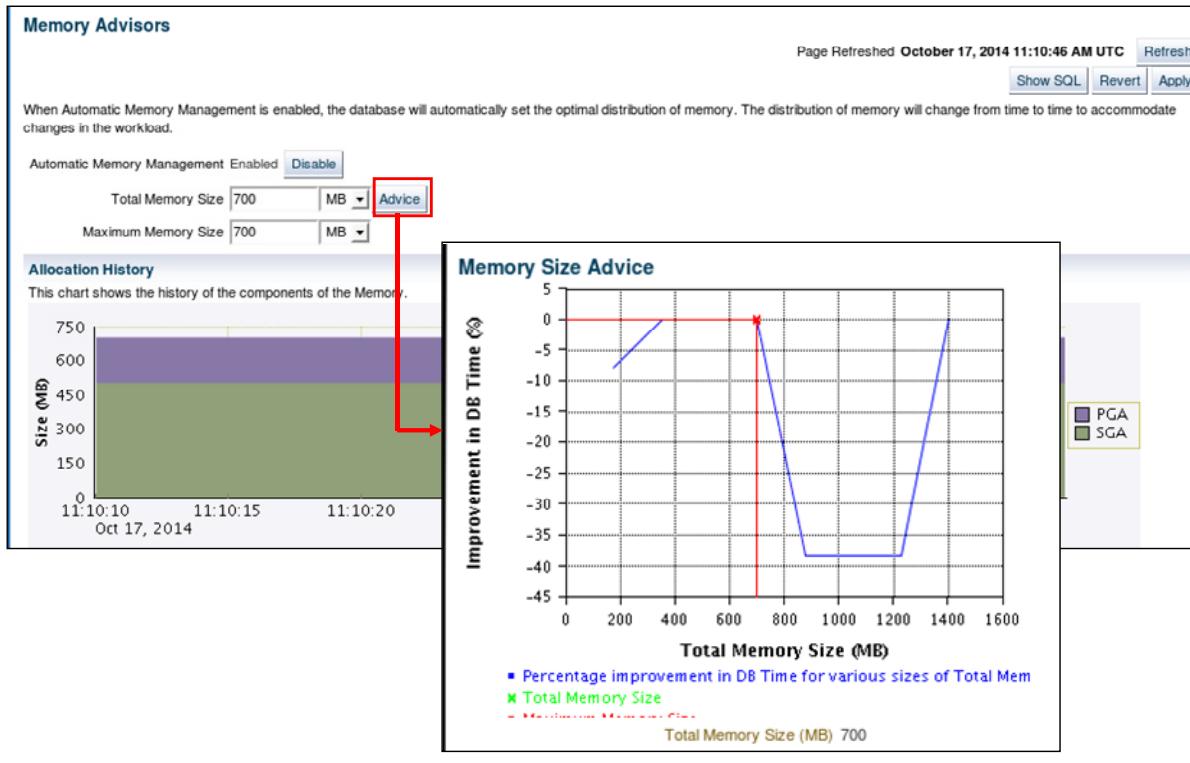
1. Click Advisors Home in the Performance menu.
2. Select Memory Advisors in the Advisors region.
3. Click Enable for Automatic Memory Management.
Note: If you change the Maximum Memory Size, the database instance must be restarted.
4. Click OK.

You can increase the size at a later time by increasing the value of the Total Memory Size field or the `MEMORY_TARGET` initialization parameter. However, you cannot set it higher than the value specified by the Maximum Memory Size field or the `MEMORY_MAX_TARGET` parameter. For more information, see the *Oracle Database Administrator's Guide*.

After AMM is enabled, the Memory Size Advisor is available to help you adjust the maximum and target memory sizes.

Note: Oracle recommends that you use Automatic Memory Management to simplify memory management tasks.

Monitoring Automatic Memory Management



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

After Automatic Memory Management is enabled, you can see the graphical representation of the history of your memory size components in the Allocation History section of the Memory Advisors page. The top portion in the first histogram is tunable PGA only and the lower portion is all of SGA.

Additional data is available on the page showing the SGA components history and a graphical representation of the SGA allocation.

On this page, you can also access the memory target advisor by clicking the Advice button. This advisor gives you the possible DB time improvement for various total memory sizes.

Note: You can also look at the memory target advisor by using the V\$MEMORY_TARGET_ADVISOR view.

Monitoring Automatic Memory Management

Use the following views to monitor Automatic Memory Management:

- V\$MEMORY_DYNAMIC_COMPONENTS: Current status of all memory components
- V\$MEMORY_RESIZE_OPS: Circular history buffer of the last 800 memory resize requests
- V\$MEMORY_TARGET_ADVICE: Tuning advice for the MEMORY_TARGET initialization parameter



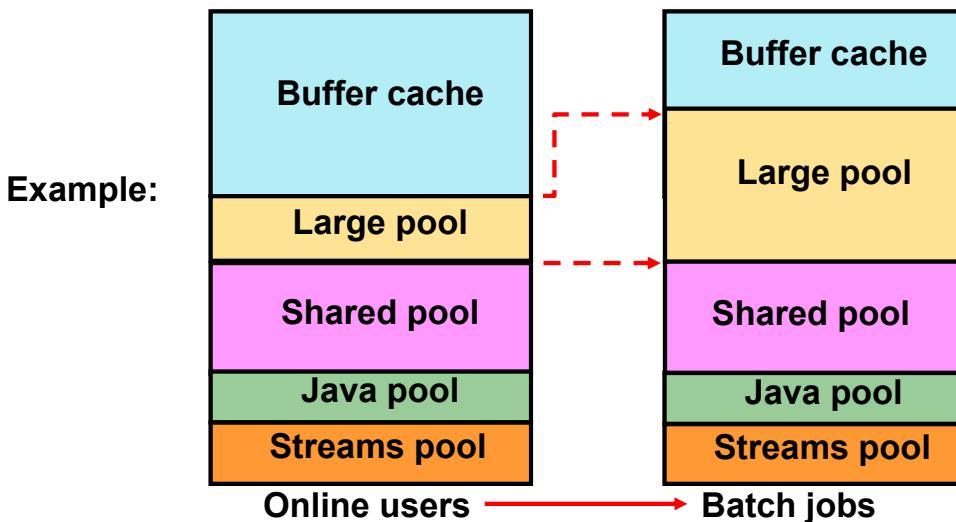
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The V\$MEMORY_DYNAMIC_COMPONENTS dynamic performance view shows the current sizes of all dynamically tuned memory components, including the total sizes of the SGA and instance PGA. The V\$MEMORY_TARGET_ADVICE view provides tuning advice for the MEMORY_TARGET initialization parameter.

In the V\$MEMORY_TARGET_ADVICE view, the row with the MEMORY_SIZE_FACTOR of 1 shows the current size of memory, as set by the MEMORY_TARGET initialization parameter, and the amount of DB time required to complete the current workload. In previous and subsequent rows, the results show several alternative MEMORY_TARGET sizes. For each alternative size, the database server shows the size factor (the multiple of the current size), and the estimated DB time to complete the current workload if the MEMORY_TARGET parameter were changed to the alternative size. Notice that for a total memory size smaller than the current MEMORY_TARGET size, the estimated DB time increases.

Automatic Shared Memory Management: Overview

- Automatically adapts to workload changes
- Maximizes memory utilization
- Helps eliminate out-of-memory errors



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

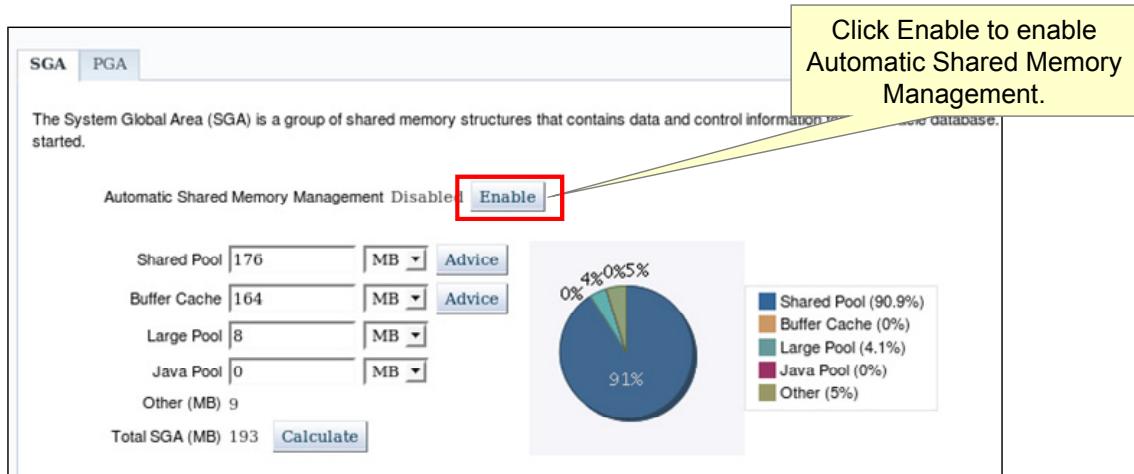
If AMM does not work for you, because you need a fixed PGA, consider the use of Automatic Shared Memory Management (ASMM) which simplifies SGA memory management. You specify the total amount of SGA memory available to an instance by using the `SGA_TARGET` initialization parameter and Oracle Database server automatically distributes this memory among the various SGA components to ensure the most effective memory utilization.

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 uses 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.

The Oracle Database server remembers the sizes of the automatically tuned components across instance shutdowns if you are using a server parameter file (SPFILE). As a result, the system does need to learn the characteristics of the workload again each time an instance is started. It can begin with information from the past instance and continue evaluating workload where it left off at the last shutdown.

Enabling Automatic Shared Memory Management (ASMM)



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Automatic Shared Memory Management is automatically enabled if you have enabled AMM. If you have not enabled AMM or did not enable ASMM when you configured your database, you can enable Automatic Shared Memory Management by performing the following steps:

1. Select Memory Advisors in the Performance menu.
2. The Memory Advisors page appears. Scroll down to the SGA section. Click Enable for Automatic Shared Memory Management.
3. The Enable Automatic Shared Memory Management page appears. Specify the total SGA size. Click OK.

You can increase the total SGA size at a later time by increasing the value of the Total SGA Size field or the `SGA_TARGET` initialization parameter. However, you cannot set it higher than the value specified by the Maximum SGA Size field or the `SGA_MAX_SIZE` parameter. For more information, see the *Oracle Database Administrator's Guide*.

When AMM is disabled, the PGA advisor is accessible. The PGA advisor is recommended for setting the PGA memory value. Click the PGA tab to access the PGA property page. Click Advice to invoke the PGA Advisor.

Note: Oracle recommends that you use Automatic Shared Memory Management to simplify your memory management tasks.

Understanding Automatic Shared Memory Management

- ASMM is based on workload information that MMON captures in the background.
- MMON uses memory advisors.
- Memory is moved to where it is needed the most by MMAN.
- 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



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

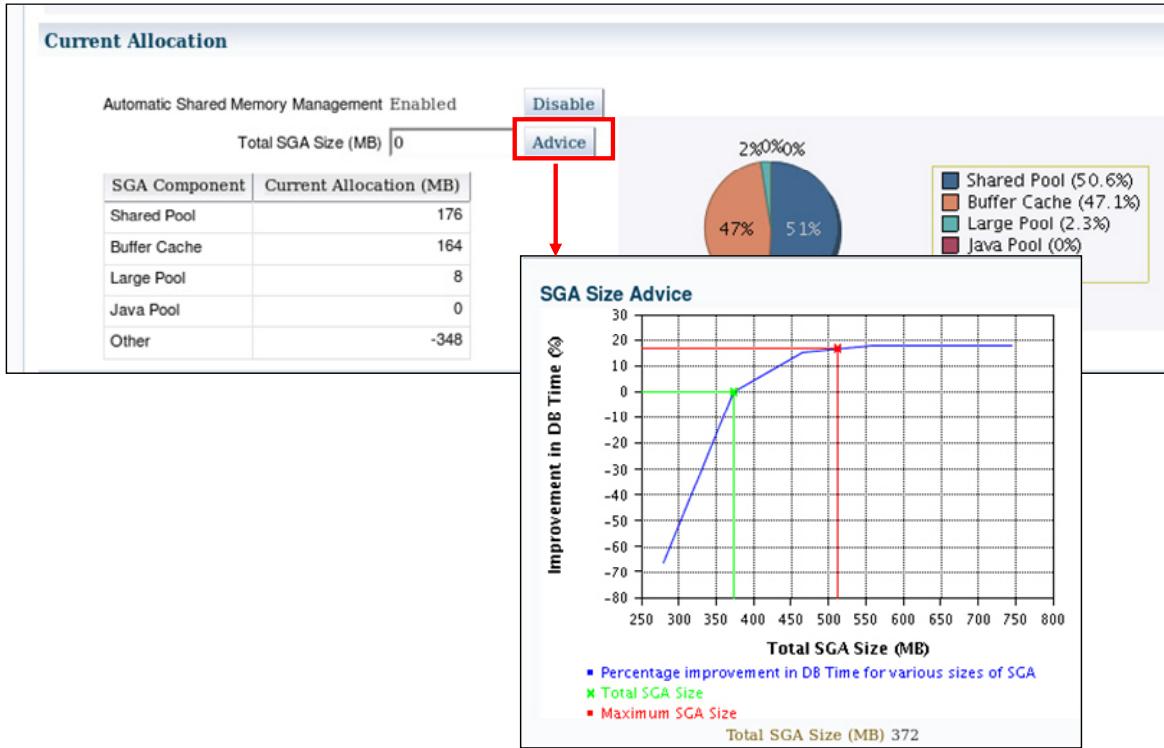
The Automatic Shared Memory Management feature uses the SGA memory broker that is implemented by two background processes: Manageability Monitor (MMON) and Memory Manager (MMAN). Statistics and memory advisory data are periodically captured in memory by MMON. MMAN coordinates the sizing of the memory components according to MMON decisions. 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 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)

Automatic Shared Memory Advisor



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Before enabling ASMM, you should remove the individual memory area parameters from the SPFILE because having them set can impose restrictions on ASMM. After ASMM is enabled, the SGA Size Advisor is available to help you choose the best value for total SGA size.

If after seeing the effects of the ASMM allocations you decide that you want to adjust certain component allocations, you can specify values for those components. If the values you are specifying are lower than the current values, those values are treated as minimum memory sizes for their respective components. If the values you are specifying are larger than the current values, the sizes of the memory components are resized upward to the values you provided as long as free memory is available. Setting limits reduces the amount of memory available for automatic adjustment, but the capability is available and can help overall performance.

The initialization parameters of concern are the following:

- SHARED_POOL_SIZE
- LARGE_POOL_SIZE
- JAVA_POOL_SIZE
- DB_CACHE_SIZE
- STREAMS_POOL_SIZE

To adjust these parameters while ASMM is enabled, you must use the `ALTER SYSTEM` command.

Enabling Automatic Shared Memory Management

To enable ASMM from manual shared memory management:

1. Get a value for **SGA_TARGET**:

```
SELECT ((SELECT SUM(value) FROM V$SGA) - (SELECT CURRENT_SIZE  
FROM V$SGA_DYNAMIC_FREE_MEMORY)) "SGA_TARGET" FROM DUAL;
```

2. Use that value to set **SGA_TARGET**.
3. Set the values of the automatically sized SGA components to 0.

To switch to ASMM from Automatic Memory Management:

1. Set the **MEMORY_TARGET** initialization parameter to 0.
2. Set the values of the automatically sized SGA components to 0.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The procedure for enabling ASMM differs depending on whether you are changing to ASMM from manual shared memory management or from automatic memory management. To change to ASMM from manual shared memory management:

1. Execute the following query to obtain a value for **SGA_TARGET**:

```
SELECT ((SELECT SUM(value) FROM V$SGA) - (SELECT CURRENT_SIZE  
FROM V$SGA_DYNAMIC_FREE_MEMORY)) "SGA_TARGET" FROM DUAL;
```
2. Set the value of **SGA_TARGET**:

```
ALTER SYSTEM SET SGA_TARGET=value [SCOPE={SPFILE|MEMORY|BOTH}]
```

where **value** is the value computed in step 1 or is some value between the sum of all SGA component sizes and **SGA_MAX_SIZE**.
3. Set the values of the automatically sized SGA components to 0. Do this by editing the text initialization parameter file or by issuing **ALTER SYSTEM** statements. Restart the instance if required.

To change to ASMM from automatic memory management:

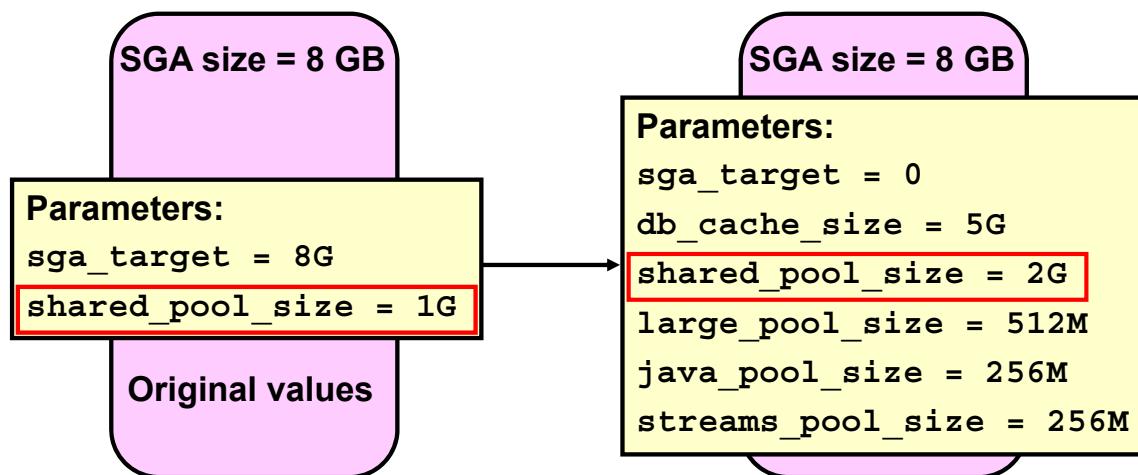
1. Set the **MEMORY_TARGET** initialization parameter to 0.

```
ALTER SYSTEM SET MEMORY_TARGET = 0;
```

The database sets **SGA_TARGET** based on current SGA memory allocation.
2. Set the values of the automatically sized SGA components to 0. Restart the instance when finished.

Disabling Automatic Shared Memory Management

- Setting SGA_TARGET to 0 disables autotuning.
- Autotuned parameters are set to their current sizes.
- The SGA size as a whole is unaffected.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can dynamically choose to disable Automatic Shared Memory Management by setting SGA_TARGET to 0. 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, setting SGA_TARGET to 0 results in SHARED_POOL_SIZE being set to 2 GB, thereby overriding the original user-specified value.

Using V\$PARAMETER to View Memory Component Sizes

```
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';
```

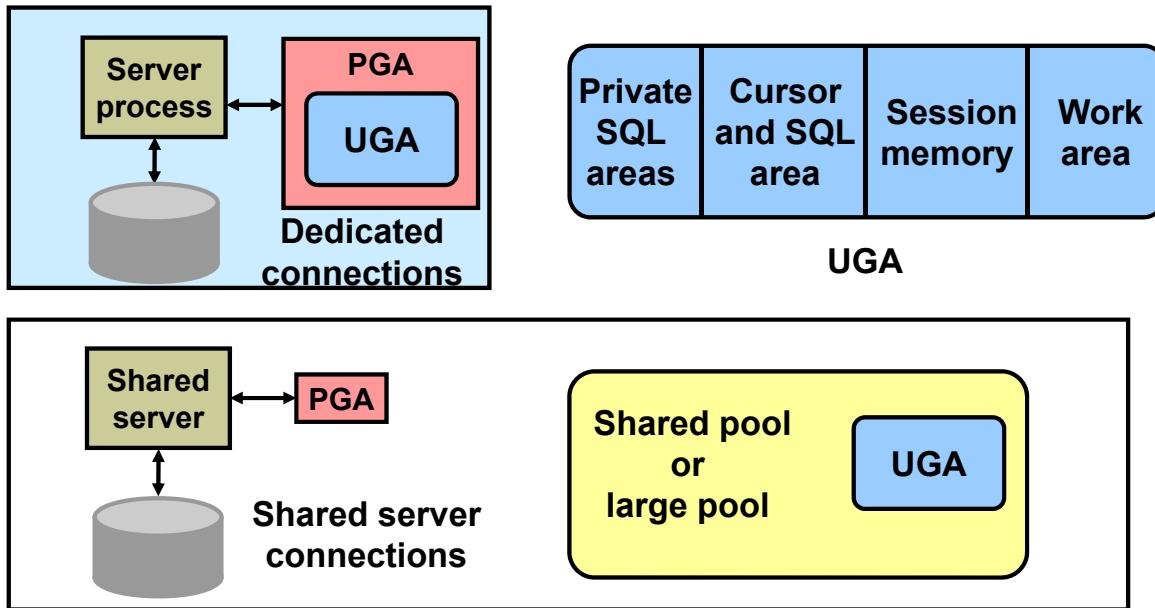


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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.

Managing the Program Global Area (PGA)



Automatic PGA memory management is enabled by default.

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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 also called the user global area (UGA) contains data, such as bind information and runtime 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.

Cursor and SQL Areas

The application developer of an Oracle Pro*C program or Oracle Call Interface (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

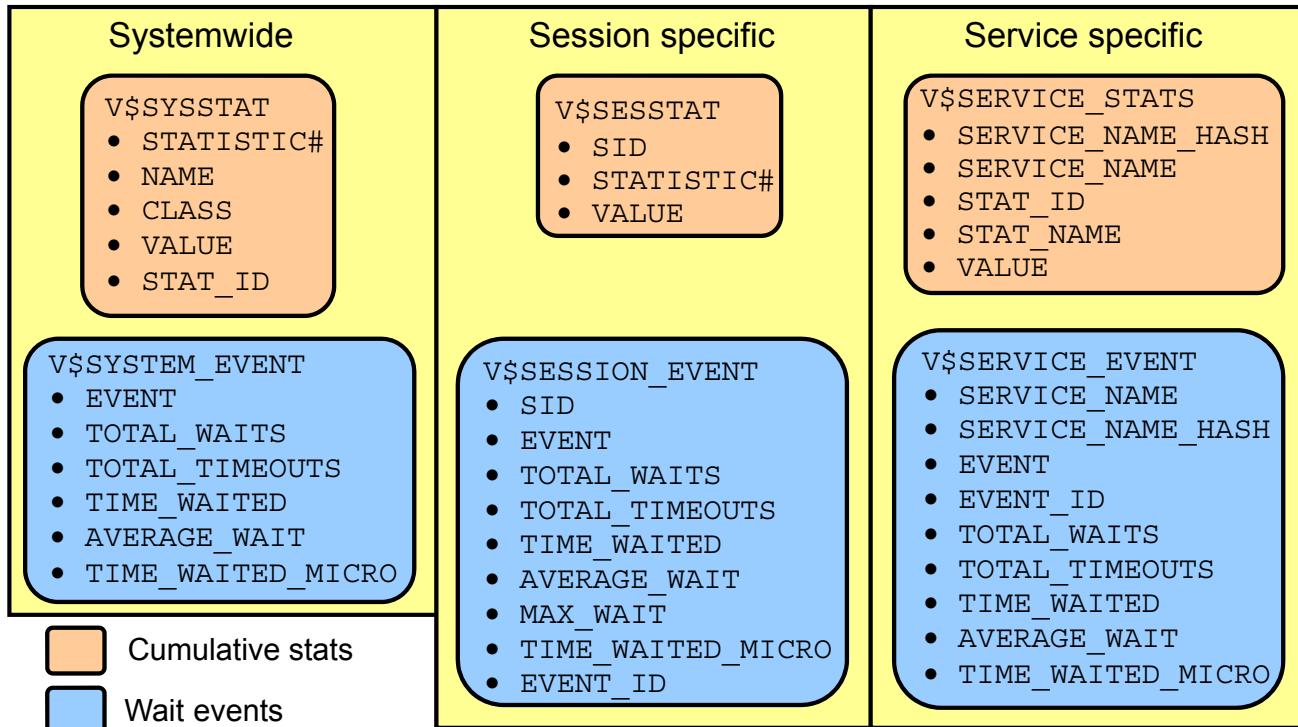
Session memory is the memory allocated to hold a session's variables (logon information) and other information related to the session. For a shared server, the session memory is shared and not private.

Automatic PGA Memory Management

By default, Oracle Database automatically and globally manages the total amount of memory dedicated to the instance PGA. You can control this amount by setting the initialization parameter `PGA_AGGREGATE_TARGET`. Oracle Database then tries to ensure that the total amount of PGA memory allocated across all database server processes and background processes never exceeds this target. But this is target value and not a hard limit.

`PGA_AGGREGATE_LIMIT` sets a hard limit for the amount of PGA that can be used. The minimum value is 1024 MB and the maximum is 120% of physical memory minus the total SGA, and it must be at least as large as `PGA_AGGREGATE_TARGET`. If `PGA_AGGREGATE_LIMIT` is not set, it defaults to 200% of `PGA_AGGREGATE_TARGET` within the same minimum and maximum as stated. When `PGA_AGGREGATE_LIMIT` is exceeded, the sessions using the most memory will have their calls aborted. Parallel queries will be treated as a unit. If the total PGA memory usage is still over the limit, sessions using the most memory will be terminated. `SYS` processes and fatal background processes are exempt from this limit.

Dynamic Performance Statistics



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Statistics must be available for the effective diagnosis of performance problems. The Oracle server generates many types of statistics for different levels of granularity.

At the systemwide, session, and service levels, both wait events and accumulated statistics are computed. In the slide, the top row of views shows the cumulative statistics. The bottom row shows the wait event views.

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. All the possible wait events are cataloged in the V\$EVENT_NAME view. All statistics are cataloged in the V\$STATNAME view; approximately 480 statistics are available in Oracle Database.

Displaying Systemwide Statistics

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.

Troubleshooting and Tuning Views

Instance/Database

V\$DATABASE
V\$INSTANCE
V\$PARAMETER
V\$SPPARAMETER
V\$SYSTEM_PARAMETER
V\$PROCESS
V\$BGPROCESS
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



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In the slide, some of the views that can help you determine the cause of performance problems or analyze the current status of your database are listed.

For a complete description of these views, see the *Oracle Database Reference*.

Quiz

Automatic Memory Management allows the Oracle instance to reallocate memory from the _____ to the SGA .

- a. Large Pool
- b. Log Buffer
- c. PGA
- d. Streams Pool



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: c

Quiz

SGA_TARGET may not be sized greater than _____.

- a. LOG_BUFFER
- b. SGA_MAX_SIZE
- c. STREAMS_POOL_SIZE
- d. PGA_AGGREGATE_TARGET



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b

Summary

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

- Enterprise Manager to monitor performance
- Automatic Memory Management (AMM)
- The Memory Advisor to size memory buffers



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice: Overview

This practice covers the following topics:

- Using the Performance page in Enterprise Manager
- Diagnosing a memory allocation problem
- Enabling and implementing Automatic Memory Management
- Monitoring Top Services and Sessions



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

19

Managing Performance: SQL Tuning

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

- Manage optimizer statistics
- Use the SQL Tuning Advisor to:
 - Identify SQL statements that are using the most resources
 - Tune SQL statements that are using the most resources
- Use the SQL Access Advisor to tune a workload



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

SQL Tuning

SQL tuning process

- Identify poorly tuned SQL statements.
- Tune the individual statements.
- Tune the application as a whole.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Generally, the tuning effort that yields the most benefit is SQL tuning. Poorly tuned SQL uses more resources than required. This inefficiency prevents scalability, uses more OS and database resources, and increases response time. To tune poorly tuned SQL statements, they must be identified, and then tuned. SQL statements can be tuned individually, but often the solution that optimizes one statement can hurt the performance of several others.

The SQL statements that use the most resources are by definition the statements in need of tuning. These are statements that have the longest elapsed time, use the most CPU, or do the most physical or logical reads.

Tune the individual statements by checking the optimizer statistics, check the explain plan for the most efficient access path, test alternate SQL constructions, and test possible new indexes, materialized views, and partitioning.

Test the application as a whole, using the tuned SQL statements. Is the overall performance better?

The methodology is sound, but tedious. Tuning an individual statement is not difficult. Testing the overall impact of the individual statement tuning on an application can be very difficult.

In Oracle Database, a set of SQL advisors are available to identify and tune statements, individually, or as a set.

Oracle Optimizer: Overview

The Oracle optimizer determines the most efficient execution plan and is the most important step in the processing of any SQL statement.

The optimizer:

- Evaluates expressions and conditions
- Uses object and system statistics
- Decides how to access the data
- Decides how to join tables
- Determines the most efficient path



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The optimizer is the part of the Oracle Database server that creates the execution plan for a SQL statement. The determination of the execution plan is an important step in the processing of any SQL statement and can greatly affect execution time.

The execution plan is a series of operations that are performed in sequence to execute the statement. The optimizer considers many factors related to the referenced objects and the conditions specified in the query. The information necessary to the optimizer includes:

- Statistics gathered for the system (I/O, CPU, and so on) as well as schema objects (number of rows, index, and so on)
- Information in the dictionary
- WHERE clause qualifiers
- Hints supplied by the developer

When you use diagnostic tools, such as Enterprise Manager, EXPLAIN PLAN, and SQL*Plus AUTOTRACE, you can see the execution plan that the optimizer chooses.

Note: The Oracle optimizer has two names based on its functionality: the *query optimizer* and the *Automatic Tuning Optimizer*.

Optimizer Statistics

Optimizer statistics are:

- A snapshot at a point in time
- Persistent across instance restarts
- Collected automatically

```
SQL> SELECT COUNT(*) FROM hr.employees;
      COUNT(*)
-----
214
SQL> SELECT num_rows FROM dba_tables
  2 WHERE owner='HR' AND table_name = 'EMPLOYEES';
      NUM_ROWS
-----
107
```

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Optimizer statistics include table, column, index, and system statistics. Statistics for tables and indexes are stored in the data dictionary. These statistics are not intended to provide real-time data. They provide the optimizer a *statistically* correct snapshot of data storage and distribution, which the optimizer uses to make decisions on how to access data.

The statistics that are collected include:

- Size of the table or index in database blocks
- Number of rows
- Average row size and chain count (tables only)
- Height and number of deleted leaf rows (indexes only)

As data is inserted, deleted, and modified, these facts change. Because the performance impact of maintaining real-time data distribution statistics is prohibitive, these statistics are updated by periodically gathering statistics on tables and indexes.

Optimizer statistics are collected automatically by an automatic maintenance job that runs during predefined maintenance windows once daily by default. System statistics are operating system characteristics that are used by the optimizer. These statistics are not collected automatically. For details about collecting system statistics, see the *Oracle Database Performance Tuning Guide*.

Optimizer statistics are not the same as the database performance statistics that are gathered in the AWR snapshot.

Optimizer Statistics Collection

- SQL performance tuning: Depends on collection of accurate statistics
- Optimizer statistics:
 - Object statistics
 - Operating system statistics
- Ways to collect statistics:
 - Automatically: Automatic Maintenance Tasks
 - Manually: DBMS_STATS package
 - By setting database initialization parameters
 - By importing statistics from another database



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Optimizer statistics are collections of data that are specific details about database objects. These statistics are essential for the query optimizer to choose the best execution plan for each SQL statement. These statistics are gathered periodically and do not change between gatherings.

The recommended approach to gathering optimizer statistics is to allow the Oracle Database server to automatically gather the statistics. The Automatic Maintenance Tasks can be 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 by default. You can change the default configuration through the Automatic Maintenance Tasks page.

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 server 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.

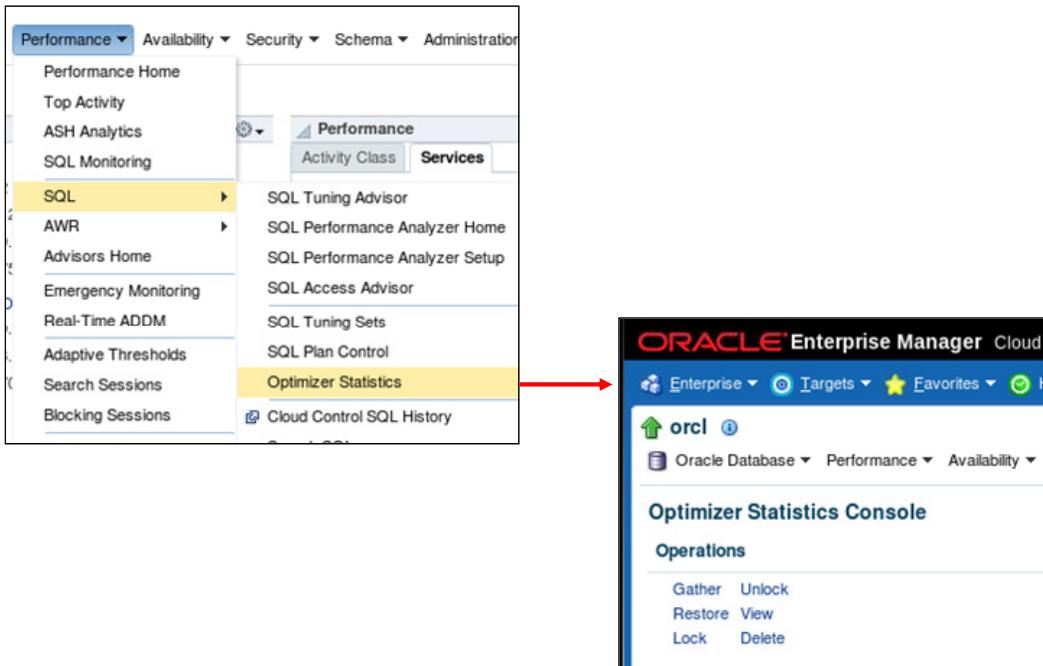
If you choose not to use automatic statistics gathering, you must manually collect statistics in all schemas, including system schemas. If the data in your database changes regularly, you also need to gather statistics regularly to ensure that the statistics accurately represent characteristics of your 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 selectivity 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`. You can query the `V$STATISTICS_LEVEL` view to determine which parameters are affected by the `STATISTICAL_LEVEL` parameter.

Note: Setting `STATISTICS_LEVEL` to `BASIC` disables many automatic features and is not recommended.

Using the Optimizer Statistics Console



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

ORACLE

To manage optimizer statistics in Enterprise Manager, select Optimizer Statistics in the SQL submenu of the Performance menu. By clicking the links under the Operations heading on the Optimizer Statistics Console, you can perform the following tasks:

- Gather optimizer statistics manually.
- Restore optimizer statistics to a point in the past. The chosen point in time must be within the optimizer statistics retention period, which defaults to 30 days.
- Lock optimizer statistics to guarantee that the statistics for certain objects are never overwritten. This is useful if statistics have been calculated for a certain table at a time when well-representative data is present, and you want to always have those statistics. No fluctuations in the table affect the statistics if they are locked.
- Unlock optimizer statistics to undo the previously done lock.
- View optimizer statistics for a specified object.
- Delete optimizer statistics.

Best-Practice Tip

Use the automatic maintenance tasks to gather optimizer statistics. To enable the task for gathering optimizer statistics, you must ensure that the STATISTICS_LEVEL initialization parameter is set to TYPICAL or ALL.

Setting Global Preferences by Using Enterprise Manager Cloud Control

Manage Optimizer Statistics > Global Statistics Gathering Options
Global Statistics Gathering Options
Database orcl

Statistics History

Retention Period (days) The number of days for which optimizer statistics history will be retained.

Gather Optimizer Statistics Default Options

Oracle recommends that you use the Gather Auto choice for the Gather Objects options when you use the Gather Optimizer Statistics process for use Gather Auto, the defaults for the other options are set here. Changing the options will impact the automated Optimizer Statistics Gathering task

Estimate Percentage Auto (Oracle recommended) 100% Percentage

Degree of Parallelism Table default Auto System default Degree

Granularity

Cursor Invalidation Auto (Oracle recommended) Immediate None

Cascade Auto (Oracle recommended) True False

Target Object Class (Auto Job) Auto (Oracle recommended) All Oracle

Stale Percentage

Incremental True False

Publish True False

Histograms

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can manage global preference settings by using Enterprise Manager. Click the Global Statistics Gathering Options link on the Optimizer Statistics Console page.

On the Global Statistics Gathering Options page, change the global preferences in the Gather Optimizer Statistics Default Options section. When you have finished, click the Apply button.

Note: To change the statistics gathering options at the object level or schema level, click the Object Level Statistics Gathering Preferences link on the Optimizer Statistics Console page.

Gathering Optimizer Statistics Manually

The screenshot shows the 'Gather Optimizer Statistics: Scope' page of the Oracle Database 12c Enterprise Manager. At the top, it displays 'Database orcl' and 'Task Status Enabled'. On the right, it shows 'Logged In As dba1' and 'Scope Database'. Below this, a message says 'Select the type of object for which you want to gather optimizer statistics.' A radio button for 'Object Type' is selected next to 'Database'. Other options include 'Schema', 'Tables', 'Indexes', 'Fixed Objects' (with a note about dynamic performance tables), and 'Dictionary Objects' (with a note about objects in 'SYS', 'SYSTEM', and 'all non-user defined schemas'). A tip message at the bottom left states: 'TIP The Objects step will be skipped when Database, Fixed Objects or Dictionary Objects is selected.' Under 'Options for Scope: Database', there are two radio button options: 'Use Oracle-recommended option settings' (selected) and 'Customize Options' (with a note about customizing options on the 'Customize Options' step). At the bottom, there is a section titled 'Validate With SQL Performance Analyzer' with a checkbox for validating impact on SQL performance prior to publishing.

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You may need to gather statistics manually at certain times, such as when the contents of a table have changed so much between automatic gathering jobs that the statistics no longer represent the table accurately. This is common for large tables that experience more than a 10 percent change in size in a 24-hour period.

Best-practice tip: Collect statistics often enough that the table never changes by more than about 10 percent between collection periods. This may require manual statistics collection or additional maintenance windows.

Statistics can be manually collected by using either Enterprise Manager or the DBMS_STATS package. System statistics can be gathered by using only the DBMS_STATS package. System statistics describe the system's hardware characteristics, such as I/O and CPU performance and utilization, to the query optimizer.

The Gather Optimizer Statistics menu selection starts a wizard that enables you to select the scope, objects, options, and schedule for a job that will gather optimizer statistics. The wizard submits a DBMS_STATS.GATHER_*_STATS job at the scope you specify: table, schema, or database. In this wizard, you set preferences for the default values used by the DBMS_STATS package and you schedule this job to run at a time that you determine.

Gathering statistics manually for routine statistics collection is not recommended because the statistics are gathered more efficiently and with less impact on users during the maintenance windows. A manual job can also be submitted if the automatic job has failed or been disabled. You can also gather optimizer statistics with the DBMS_STATS package directly:

```
SQL> EXEC dbms_stats.gather_table_stats('HR','EMPLOYEES') ;
SQL> SELECT num_rows FROM dba_tables
      2 WHERE owner='HR' AND table_name = 'EMPLOYEES';
      NUM_ROWS
      -----
      214
```

Notice that the number of rows now correctly reflects what was in the table at the time that the statistics were gathered. DBMS_STATS also enables manual collection of statistics for an entire schema or even for the whole database.

System statistics do not change unless the workload significantly changes. As a result, system statistics do not need frequent adjustment. The DBMS_STATS.GATHER_SYSTEM_STATS procedure will collect system statistics over a specified period, or you can start the gathering of system statistics and make another call to stop gathering.

Best-practice tip: Use the following command when you create a database:

```
SQL> EXEC dbms_stats.gather_system_stats('NOWORKLOAD');
```

The NOWORKLOAD option takes a few minutes (depending on the size of the database) and captures estimates of I/O characteristics such as average read seek time and I/O transfer rate.

Setting Optimizer Statistics Preferences

- Optimizer statistics preferences set the default values of parameters used by:
 - Automatic statistics collection
 - DBMS_STATS.GATHER_*_STATS procedures
- Set preferences at levels:
 - Table, schema, database, or global
- Preferences: CASCADE, DEGREE, ESTIMATE_PERCENT, NO_INVALIDATE, METHOD_OPT, GRANULARITY, INCREMENTAL, PUBLISH, STALE_PERCENT
- Use DBMS_STATS.SET | GET | DELETE | EXPORT | IMPORT_*_PREFS to manage preferences



Optimizer statistics gathering task



```
EXEC DBMS_STATS.SET_TABLE_PREFS('SH', 'SALES', 'STALE_PERCENT', '13');
```

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The DBMS_STATS.GATHER_*_STATS procedures can be called at various levels to gather statistics for an entire database or for individual objects, such as tables. When the GATHER_*_STATS procedures are called, several of the parameters are often allowed to default. The supplied defaults work well for most of the objects in the database, but for some objects or schemas the defaults need to be changed. Instead of running manual jobs for each of these objects, Oracle Database enables you to set values (called preferences) for individual objects, schemas, or databases, or to change the default values with the global-level command.

The preferences specify the parameters that are given to the gather procedures. The SET_*_PREFS procedures create preference values for any object that is not owned by SYS or SYSTEM. The expected use is that the DBA will set the global preferences for any parameters that should be database-wide. These will be applied for any parameter that is allowed to default.

The SET_DATABASE_PREFS procedure iterates over all the tables and schemas in the database setting the specified preference. SET_SCHEMA_PREFS iterates over the tables in the specified schema. SET_TABLE_PREFS sets the preference value for a single table.

All object preferences—whether set at the database, schema, or table level—are held in a single table. Changing the preferences at the schema level overwrites the preferences that were previously set at the table level.

When the various gather procedures execute, they retrieve the object-level preferences that were set for each object. You can view the object-level preferences in the DBA_TAB_STAT_PREFS view. Any preferences that are not set at the object level will be set to the global-level preferences. You can see the global preferences by calling the DBMS_STATS.GET_PREFS procedure for each preference.

You can set, get, delete, export, and import those preferences at the table, schema, database, and global levels. The preference values are expected to be set from global to table levels, applying the preferences to the smallest group last.

Preferences include:

- CASCADE – Determines whether index statistics are collected as part of gathering table statistics
- DEGREE – Sets the degree of parallelism that is used for gathering statistics
- PUBLISH – Is used to decide whether to publish the statistics to the dictionary or store them in a private area. This enables the DBA to validate the statistics before publishing them to the data dictionary with the PUBLISH_PENDING_STATS procedure.
- STALE_PERCENT – Is used to determine the threshold level at which an object is considered to have stale statistics. The value is a percentage of the rows modified since the last statistics gathering. The example changes the 10 percent default to 13 percent for SH.SALES only.
- INCREMENTAL – Is used to gather global statistics on partitioned tables in an incremental way
- METHOD_OPT – Determines the columns and histogram parameters that are used to gather column statistics
- GRANULARITY – Determines the granularity of statistics to collect (which is pertinent only if the table is partitioned)
- NO_INVALIDATE – Is used to determine whether to invalidate cursors
- ESTIMATE_PERCENT – Is used to determine the number of rows to sample to obtain good statistics. It is a percentage of the number of rows in the table

Note: For details about these preferences, see the DBMS_STATS documentation in the *Oracle Database PL/SQL Packages and Types Reference*.

Preferences may be deleted with the DBMS_STATS.DELETE_*_PREFS procedures at the table, schema, and database levels. You can reset the global preferences to the recommended values with the DBMS_STATS.RESET_PARAM_DEFAULTS procedure.

Concurrent Statistics Gathering

- The auto statistics gather job uses concurrency.
- Prevent concurrent statistics gathering to make the system overwhelmed through careful resource usage capping.
- Allow gathering index statistics concurrently.
- Allow gathering of statistics for multiple partitioned tables concurrently.



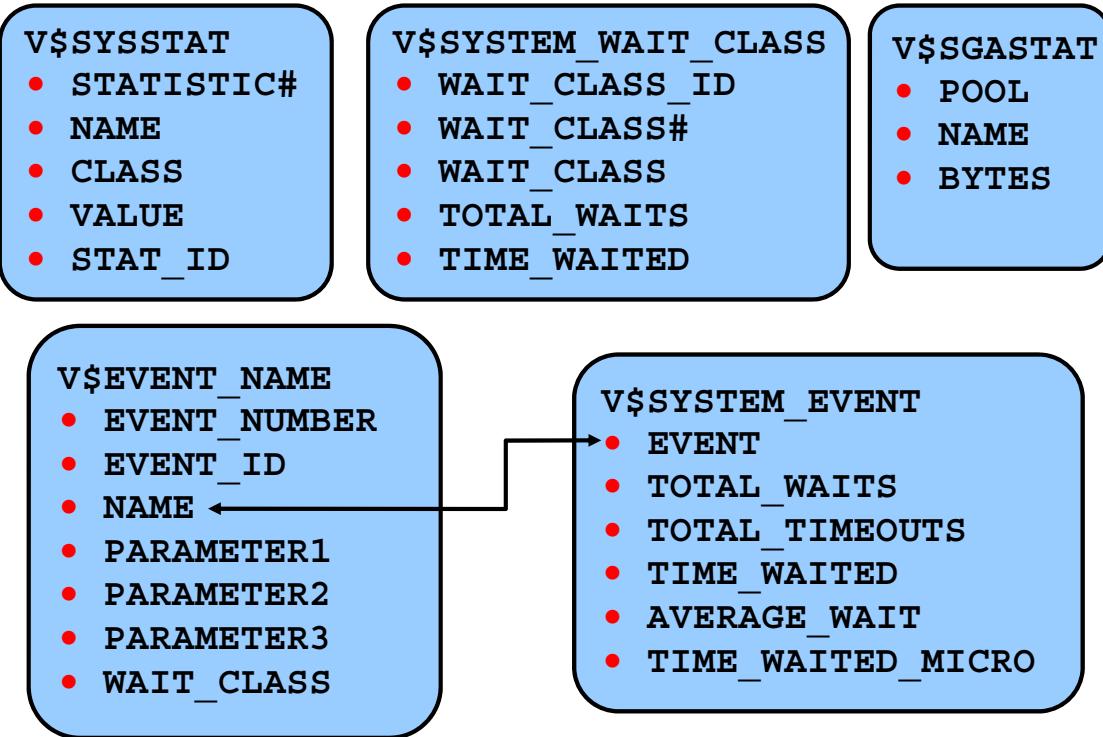
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Before Oracle Database 12c, the automatic statistics gather job gathered statistics one segment at a time. It now uses concurrency.

In Oracle Database 12c, you can run multiple statistics gathering tasks concurrently which may provide considerable improvements in statistics gathering time by increasing the resource utilization rate, in particular, on multi-CPU systems.

Employ smart scheduling mechanisms to have maximum degree of concurrency (to the extent that resources are available), for example, allow gathering of statistics for multiple partitioned tables concurrently.

Viewing Statistics Information



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To effectively diagnose performance problems, statistics must be available. The Oracle Database instance generates many types of cumulative statistics for the system, sessions, and individual SQL statements at the instance level. The Oracle Database server 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.

Note: Instance statistics are dynamic and are reset at every instance startup. These statistics can be captured at a point in time and held in the database in the form of snapshots.

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: Over 400 statistics are available in Oracle Database.

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.

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. Each of the system statistics can belong to more than one class, so you cannot do a simple join on V\$SYSSTATS.CLASS and V\$SYSTEM_WAIT_CLASS.WAIT_CLASS#.

You can also view all wait events for a particular wait class by querying V\$SYSTEM_WAIT_CLASS, as in this 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, as in the following 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.

When the STATISTICS_LEVEL parameter is set to BASIC, the value of the TIMED_STATISTICS parameter defaults to FALSE. Timing information is not collected for wait events and much of the performance-monitoring capability of the database is disabled. The explicit setting of TIMED_STATISTICS overrides the value derived from STATISTICS_LEVEL.

SQL Plan Directives

- A SQL plan directive is additional information and instructions that the optimizer can use to generate a better plan:
 - Collect missing statistics
 - Create column group statistics
 - Perform dynamic sampling
- Directives can be used for multiple statements:
 - Directives are collected on query expressions
- They are persisted to disk in the SYSAUX tablespace.
- Directives are automatically maintained:
 - Created as needed during compilation or execution:
 - Missing statistics, cardinality misestimates
 - Purged if not used after a year



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In Oracle Database 12c, the database server can use a SQL plan directive, which is additional information and instructions that the optimizer can use to generate a more optimal plan. For example, a SQL plan directive might instruct the optimizer to collect missing statistics, create column group statistics, or perform dynamic sampling. During SQL compilation or execution, the database analyzes the query expressions that are missing statistics or that misestimate optimizer cardinality to create SQL plan directives. When the optimizer generates an execution plan, the directives give the optimizer additional information about objects that are referenced in the plan.

SQL plan directives are not tied to a specific SQL statement or `SQL_ID`. The optimizer can use SQL plan directives for SQL statements that are nearly identical because SQL plan directives are defined on a query expression. For example, directives can help the optimizer with queries that use similar patterns, such as web-based queries that are the same except for a select list item. The database stores SQL plan directives persistently in the SYSAUX tablespace. When generating an execution plan, the optimizer can use SQL plan directives to obtain more information about the objects that are accessed in the plan.

Directives are automatically maintained, created as needed, purged if not used after a year.

Directives can be monitored in `DBA_SQL_PLAN_DIR_OBJECTS`. SQL plan directives improve plan accuracy by persisting both compilation and execution statistics in the SYSAUX tablespace, allowing them to be used by multiple SQL statements.

Adaptive Execution Plans

- A query plan changes during execution because runtime conditions indicate that optimizer estimates are inaccurate.
- All adaptive execution plans rely on statistics that are collected during query execution.
- The two adaptive plan techniques are:
 - Dynamic plans
 - Re-optimization
- The database uses adaptive execution plans when `OPTIMIZER_FEATURES_ENABLE` is set to 12.1.0.1 or higher, and `OPTIMIZER_ADAPTIVE_REPORTING_ONLY` is set to the default value of FALSE.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Adaptive Execution Plans feature enables the optimizer to automatically adapt a poorly performing execution plan at run time and prevent a poor plan from being chosen on subsequent executions. The optimizer instruments its chosen plan so that at run time, it can be detected if the optimizer's estimates are not optimal. Then the plan can be automatically adapted to the actual conditions. An adaptive plan is a plan that changes after optimization when optimizer estimates prove inaccurate.

The optimizer can adapt plans based on statistics that are collected during statement execution. All adaptive mechanisms can execute a plan that differs from the plan which was originally determined during hard parse. This improves the ability of the query-processing engine (compilation and execution) to generate better execution plans.

The two Adaptive Plan techniques are:

- **Dynamic plans:** A dynamic plan chooses among subplans during statement execution. For dynamic plans, the optimizer must decide which subplans to include in a dynamic plan, which statistics to collect to choose a subplan, and thresholds for this choice.
- **Re-optimization:** In contrast, re-optimization changes a plan for executions after the current execution. For re-optimization, the optimizer must decide which statistics to collect at which points in a plan and when re-optimization is feasible.

Note: `OPTIMIZER_ADAPTIVE_REPORTING_ONLY` controls reporting-only mode for adaptive optimizations. When set to TRUE, adaptive optimizations run in reporting-only mode where the information required for an adaptive optimization is gathered, but no action is taken to change the plan.

Using the SQL Advisors

The screenshot shows the Oracle Advisor Central interface. In the top navigation bar, the 'Advisors' tab is selected. Below the navigation bar, there's a section titled 'Advisors' containing several advisor names: ADDM, Maximum Availability Architecture (MAA) Advisor, Segment Advisor, Streams Performance Advisor, Automatic Undo Management, Memory Advisors, SQL Advisors, Data Recovery Advisor, MTTR Advisor, and SQL Performance Analyzer Home. The 'SQL Advisors' link is highlighted with a red box and a red arrow points down to the detailed description below.

Advisor Central > SQL Advisors

SQL Advisors

The SQL Advisors address several important use cases having to do with SQL: identify physical structures optimizing a SQL workload, tune individual statements with heavy execution set divergence, build test cases for failed SQL.

SQL Access Advisor

SQL Access Advisor Evaluate an entire workload of SQL and recommend indexes, partitioning, materialized views that will improve the collective performance of the workload.

SQL Tuning Advisor

SQL Tuning Advisor Analyze individual SQL statements, and recommend SQL profiles, statistics, indexes, and restructured SQL to SQL performance.

Automatic SQL Tuning Results View the results of automated execution of SQL Tuning Advisor on observed high-load SQL.

SQL Repair Advisor

The SQL Repair Advisor can analyze and potentially patch failing SQL statements.

SQL Incident Analysis SQL incident analysis is initiated from the Support Workbench for SQL failures that are generating Support Workbench incidents. [Click here to go to Support Workbench.](#)

SQL Failure Analysis SQL failure analysis is used for non-incident SQL failures and can be accessed through either SQL Details or SQL Worksheet. [Click here to go to SQL Worksheet.](#)

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle Database provides a set of advisors for SQL. The AWR identifies and records statistics about the recent high-load SQL statements.

The SQL Access Advisor considers the changes applied to a set of SQL statements and looks for a net gain in performance. This set can be a hypothetical set of SQL, a historical set, or a manually created set.

The SQL Tuning Advisor analyzes one or more SQL statements one at a time. It examines statistics, SQL profiles, indexes, materialized views, and restructured SQL. The SQL Tuning Advisor can be run manually at any time, but it is run during every maintenance window against the recent high-load SQL statements. Click Automatic SQL Tuning Results to view and implement the recommendations. This automatic job can be configured to automatically implement recommended SQL profiles for the high-load statements.

The SQL Repair Advisor is run from the Support Workbench when a SQL statement fails with a critical error. A critical error also produces an incident. The repair advisor attempts to find and recommend a SQL patch. If no patch is found, you can continue in the Support Workbench to package the incident and submit the package to Oracle Support as a Service Request (SR).

The SQL Performance Analyzer can be used to predict and prevent potential performance problems for any database environment change that affects the structure of the SQL execution plans.

Automatic SQL Tuning Results

Advisor Central > Logged in as DBA1

Automatic SQL Tuning Result Summary

The Automatic SQL Tuning runs during system maintenance windows as an automated maintenance task, searching for ways to improve the execution plans of high-load SQL statements.

Task Status

Automatic SQL Tuning (SYS_AUTO_SQL_TUNING_TASK) is currently Enabled [Configure](#)
Automatic Implementation of SQL Profiles is currently Disabled [Configure](#)
Key SQL Profiles 4 [Implement All](#)
 TIP Key SQL Profiles were verified to yield at least a 3X performance improvement and would have been implemented automatically had auto-implementation been enabled.
SPA Validation Results [Navigate to SPA home](#)

Summary Time Period

Choose a time period to focus the graphs and statistics below on a specific range of tuning results. Drill down to view focused results or see the results for all SQLs by clicking the "View Report" button.

Time Period [All](#) [Go](#) [View Report](#)
Begin Date Oct 2, 2014 10:00:03 PM GMT+00:00 End Date Oct 20, 2014 8:17:05 AM GMT+00:00

Overall Task Statistics

Executions 17 Candidate SQL 326 Distinct SQL Examined 43

SQL Examined Status

Status	Count
SQL Examined With Findings	27
SQL Examined Without Findings	14
SQL Skipped Due To Errors	2

Breakdown by Finding Type

Finding Type	Number of SQL
SQL Profile	15
Index	0
Statistics	5
Restructure SQL	8
Alternative Plan	2

■ Implemented ■ Not Implemented

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Automatic SQL Tuning Task runs by default every night.

You can access the Automatic SQL Tuning Result Summary page by selecting Automated Maintenance Tasks in the Oracle Scheduler submenu of the Administration menu. Select the Automatic SQL Tuning link to view the result summary page.

Click Configure to display a page where you can change the defaults of the Automatic Tuning Task and enable automatic implementation of SQL profiles.

Implementing Automatic Tuning Recommendations

Advisor Central > SQL Tuning Summary:SYS.SYS_AUTO_SQL_TUNING_TASK >
Automatic SQL Tuning Result Details: All Analyzed SQLs

Begin Date Oct 2, 2014 10:00:03 PM GMT+00:00 End Date Oct 20, 2014 8:18:39 AM GMT+00:00

Recommendations
 Only profiles that significantly improve SQL performance were implemented.

[View Recommendations](#) [Implement All SQL Profiles](#) [Validate All Profiles with SPA](#)

Select	SQL Text	Parsing Schema	SQL ID	Weekly DB Time Benefit(sec)	Per-Execution % Benefit	Statistics	SQL Profile	Index	Restructure SQL
SELECT /*+first rows */									

Only one recommendation should be implemented.

SQL Information
 SQL Text `SELECT /*+first rows */ d.tablespace_name, NVL(a.bytes / 1024 / 1024, 0) allocated_space, DECODE(d.contents,'UNDO', NVL(u.bytes, 0)/1024/1024, NVL(a.bytes - NVL(t.bytes, 0), 0)/1024) used_space ...`

Select Recommendation
[Original Explain Plan \(Annotated\)](#)

[Implement](#) [Validate with SPA](#)

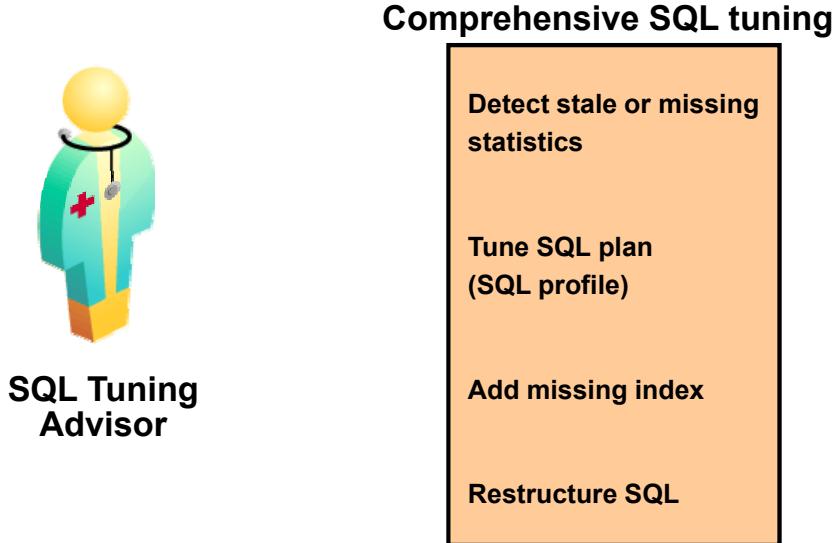
Select	Type	Findings	Recommendations	Rationale	Benefit (%)	Other Statistics	New Explain Plan	Compare Explain Plans
<input checked="" type="radio"/>	Statistics	Optimizer statistics for table "SYS"."TSS" and its indices are stale.	Consider collecting optimizer statistics for this table.	The optimizer requires up-to-date statistics for the table in order to select a good execution plan.				
<input type="radio"/>	SQL Profile	A potentially better execution plan was found for this statement.	Consider accepting the recommended SQL profile. No SQL profile currently exists for this recommendation.	The SQL profile was not automatically created because its benefit could not be verified.	62.93			

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

If you click View Report on the Automatic Tuning Results Summary page, you will see the Automatic SQL Tuning Result Details. You can implement all the recommendations or drill down to view or implement individual recommendations. On the Recommendations page, you can click the eyeglass icon on the right to see the differences that implementing a SQL profile will make in the explain plan.

SQL Tuning Advisor: Overview



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The SQL Tuning Advisor is the primary driver of the tuning process. It performs several types of analyses:

- **Statistics Analysis:** Checks each query object for missing or stale statistics, and makes recommendations to gather relevant statistics.
- **SQL Profiling:** The optimizer verifies its own estimates and collects auxiliary information to remove estimation errors. It builds a SQL profile using the auxiliary information and makes a recommendation to create it. When a SQL profile is created, it enables the query optimizer to generate a well-tuned plan.
- **Access Path Analysis:** New indexes are considered if they significantly improve access to each table in the query. When appropriate, recommendations to create such objects are made.
- **SQL Structure Analysis:** SQL statements that use bad plans are identified and relevant suggestions are made to restructure them. The suggested changes can be syntactic as well as semantic.

The SQL Tuning Advisor considers each SQL statement included in the advisor task independently. Creating a new index may help a query, but may hurt the response time of DML. So, a recommended index or other object should be checked with the SQL Access Advisor over a workload (a set of SQL statements) to determine whether there is a net gain in performance.

Using the SQL Tuning Advisor

- Use the SQL Tuning Advisor to analyze SQL statements and obtain performance recommendations.
- Sources for SQL Tuning Advisor to analyze:
 - Top Activity: Analyzes the top SQL statements currently active
 - SQL Tuning Sets: Analyzes a set of SQL statements you provide
 - Historical SQL (AWR): Analyzes SQL statements from statements collected by AWR snapshots



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The SQL Tuning Advisor runs automatically every night as the Automatic SQL Tuning Task. There may be times when a SQL statement needs immediate tuning action. You can use the SQL Tuning Advisor to analyze SQL statements and obtain performance recommendations at any time. 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.

Even though you can submit multiple statements to be analyzed in a single task, each statement is analyzed independently. To obtain tuning recommendations that consider overall performance of a set of SQL, use the SQL Access Advisor.

Using the SQL Tuning Advisor

The screenshot shows the 'Schedule SQL Tuning Advisor' page. At the top, it says 'Specify the following parameters to schedule a job to run the SQL Tuning Advisor.' Below this, there are several input fields:

- * Name: SQL_TUNING_1352461752065
- Description: (empty)
- * SQL Tuning Set: SYS.TOP_SQL_1352461750826 (with a magnifying glass icon)
- SQL Tuning Set Description: Automatically generated by Top SQL.
- SQL Statements Counts: 10
- SQL Statements section: A collapsible panel showing 'SQL Statements'.
- Scope section:
 - Total Time Limit (minutes): 30
 - Scope of Analysis:
 - Limited: The analysis is done without SQL Profile recommendation and takes about 1 second per statement.
 - Comprehensive: This analysis includes SQL Profile recommendation, but may take a long time.
 - Time Limit per Statement (minutes): 5
- Schedule section:
 - Time Zone: UTC
 - Immediately

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

On the Schedule SQL Tuning Advisor page, you can choose the SQL statements to include and change the automatic defaults for a tuning task. You can set the source of the SQL statements, and if you have been granted the ADVISOR system privilege, you can submit the task. Enterprise Manager then creates a tuning task for the SQL Tuning Advisor.

The SQL statement options allow you to choose one or more SQL statements from recent Top Activity, choose Historical SQL stored in the AWR, or choose from a SQL Tuning Set that you have already created.

It is important to choose an appropriate scope for the tuning task. If you choose the Limited option, the SQL Tuning Advisor produces recommendations based on statistics check, access path analysis, and SQL structure analysis. The Limited option will not make a SQL profile recommendation. If you choose the Comprehensive option, the SQL Tuning Advisor produces all recommendations that the Limited option produces, but it also invokes the optimizer under the SQL profiling mode to build a SQL profile. With the Comprehensive option, you can also specify a time limit for the tuning task, which, by default, is 30 minutes. After you select Run SQL Tuning Advisor, configure your tuning task by using the SQL Tuning Options page.

SQL Tuning Advisor Recommendations

Advisor Central > SQL Tuning Summary:SYS.SQL_TUNING_1352461752065 > Logged in As SYS

SQL Tuning Result Details: All Analyzed SQLs

Status	Completed	Tuning Set Owner	SYS
Started	11/09/2012 11:51:01	Tuning Set Name	TOP_SQL_1352461750826
Completed	11/09/2012 11:51:21	Time Limit (seconds)	1800
Running Time (minutes)	0		

Recommendations
Only profiles that significantly improve SQL performance were implemented.

[View Recommendations](#) [Implement All SQL Profiles](#)

Select	SQL Text	Parsing Schema	SQL ID	Cumulative DB Time Benefit(sec) ▼	Per-Execution % Benefit	Statistics	SQL Profile	Index	Restructure SQL
<input checked="" type="radio"/>	WITH MONITOR_DATA AS (SELECT INST_ID, KE...	SYS	7r24h5ucyjggz	2.28	98	(98%) ✓			
<input type="radio"/>	SELECT ash.sql_id, decode(ash.sql_plan_h...	SYS	dzw9p1af48pgn						
<input type="radio"/>	SELECT ash.sql_id, decode(ash.sql_plan_h...	SYS	8k5pb5szj3gtd						
<input type="radio"/>	SELECT XMLTYPE(DBMS_REPORT.GET_REPORT_WL...	SYS	0w26sk6t6gq98						



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The SQL Tuning Results for a task are displayed as soon as the task completes and can also be accessed later from the Advisor Central page. You can review and implement individual recommendations. Select a statement and click View Recommendations.

Duplicate SQL

Duplicate SQL

Logged in as DBA1

Page Refresh

Applications can cause database to consume excessive CPU by parsing SQL statements that are similar and that can share the same SQL text. Such applications can also cause slow performance by creating contention.

CPU Consumption Since Instance Started

CPU Used as percentage of Total CPU (%) 0.40
CPU Used for Parsing as percentage of CPU Used(%) 11.87

Duplicate SQL Statements

This report identifies similar SQL statements that could be shared by a single SQL statement if the database application used bind variables to replace literals and SQL coding conventions to remove differences based on:

Note: Only the first 2000 SQL statements that are executed only once are examined. The actual number of SQL statements that are duplicates can be more than 2000.

Expand All | Collapse All

Duplicates	Plan Hash Value	SQL Text
▼ Duplicates		
▷ 4	120363598	SELECT /* STN_REPT_STAT with_stats */ count(distinct sql_id) FROM (SELECT *
▷ 4	346815704	select /* STN_REPT_STAT with_results */ count(distinct object_id) from (SELECT /*+ cardinality(o)
▷ 4	3021663871	SELECT /* STN_REPT_STAT dist_in_report */ count(distinct sql_id) FROM (SELECT *
▷ 4	1555381724	SELECT /* STN_REPT_STAT with_errors */ count(distinct sql_id) FROM (SELECT *
▷ 4	3800876181	SELECT /* STN_REPT_STAT prof_times */ profile_impl, round(sum(last_week
▷ 4	120363598	SELECT /* STN_REPT_STAT with_index */ count(distinct sql_id) FROM (SELECT *
▷ 4	1033394119	SELECT DECODE(COUNT(*), 0, 'DISABLED', 'ENABLED') STATUS FROM DBA_AUTOTASK_WINDOW_CLIENTS WHERE SQL_

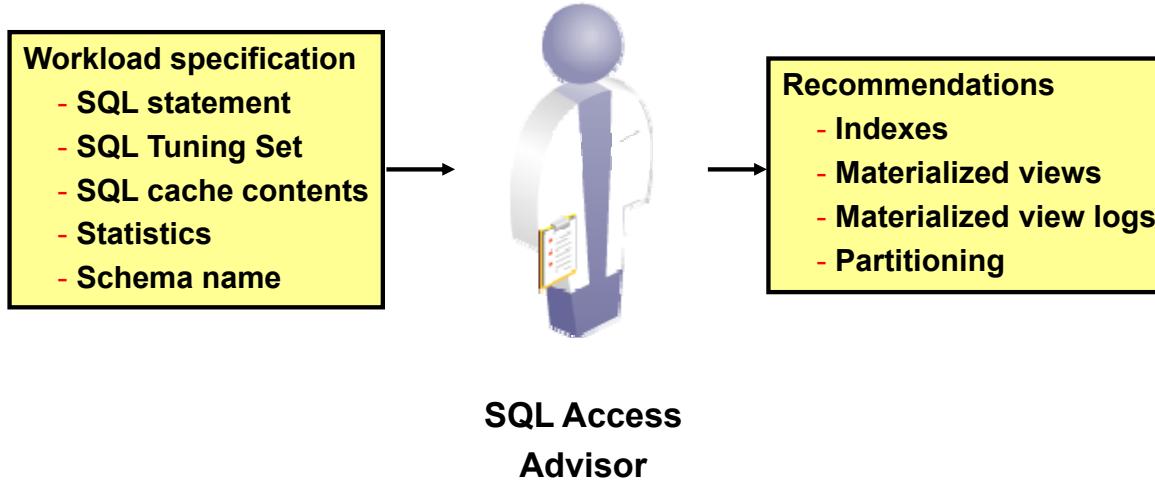


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Duplicate SQL statements are statements that are different only in the literal values they use or in their formatting. Statements that are different get a separate cursor in the Library Cache. Statements that are duplicates can use the same cursor if the literals are replaced with bind variables, and the formatting is made to be the same.

Duplicate SQL statements can be identified by clicking Duplicate SQL in the Additional Monitoring Links section of the Performance Home page. SQL that is determined to be duplicate, except for formatting or literal differences, is listed together. This helps you to determine which SQL in your application can be consolidated, thus lowering the requirements on the Library Cache and speeding up the execution of a statement.

SQL Access Advisor: Overview



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

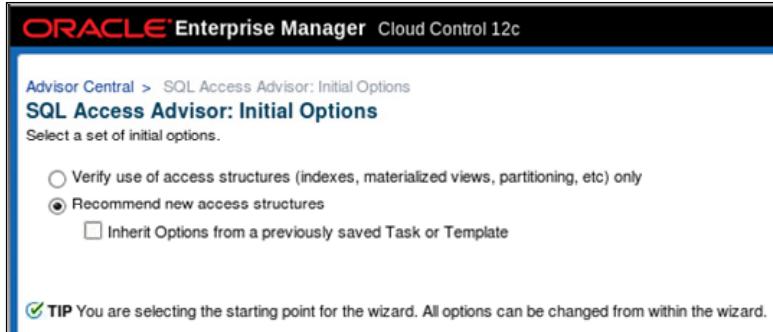
The SQL Access Advisor can recommend the proper set of materialized views, materialized view logs, partitioning, and indexes for a given workload. Understanding and using these structures is essential when optimizing SQL because they can result in significant performance improvements in data retrieval.

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*.

Using the SQL Access Advisor



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Access the SQL Access Advisor by selecting SQL Access Advisor in the SQL submenu of the Performance menu.

When starting a SQL Access Advisor session, you can 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 previously saved 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.

Workload Source

Workload Source Recommendation Options Schedule Review

SQL Access Advisor: Workload Source

Database orcl
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.

Current and Recent SQL Activity
SQL will be selected from the cache.

Use an existing SQL Tuning Set

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.

Schemas and Tables

Add

Comma-separated list

TIP Enter a schema name to specify all the tables belonging to that schema.

Filter Options

TIP For workloads containing a large number of SQL statements, Oracle recommends using filtering to reduce analysis time.

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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
- **Use an existing SQL Tuning Set:** Enables you to specify a previously created SQL Tuning Set as the workload source
- **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 orcl
Logged In As SYS

Access Structures to Recommend

Indexes
 Materialized Views
 Partitioning

Scope

The advisor can run in one of two modes, Limited or Comprehensive. Limited Mode is meant to return certain threshold. Comprehensive Mode will perform an exhaustive analysis.

Limited
Analysis will focus on highest cost statements

Comprehensive
Analysis will be exhaustive

▷ Advanced Options

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Use the Recommendations Options page to choose whether to limit the advisor to recommendations based on a single access method. Choose Indexes, Materialized Views, Partitioning, or a combination of those from the “Access Structures to Recommend” 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 Scope section to specify a mode of Limited or Comprehensive. These modes affect the quality of recommendations as well as the length of time required for processing. In the Comprehensive mode, the advisor searches a large pool of candidates resulting in recommendations of the highest quality. In the Limited mode, the advisor performs quickly, limiting the candidate recommendations.

Recommendation Options

The screenshot shows the 'Advanced Options' section of the Oracle SQL Access Advisor. It includes sections for Workload Categorization, Space Restrictions, Tuning Prioritization, and Default Storage Locations. In the Tuning Prioritization section, there is a dropdown menu set to 'Buffer Gets'. The Default Storage Locations section contains dropdown menus for Index Tablespace, Index Schema, Materialized View Tablespace, Materialized View Schema, Materialized View Log Tablespace, and Partitioning Tablespaces, each with a 'Comma-separated list' placeholder.

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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.

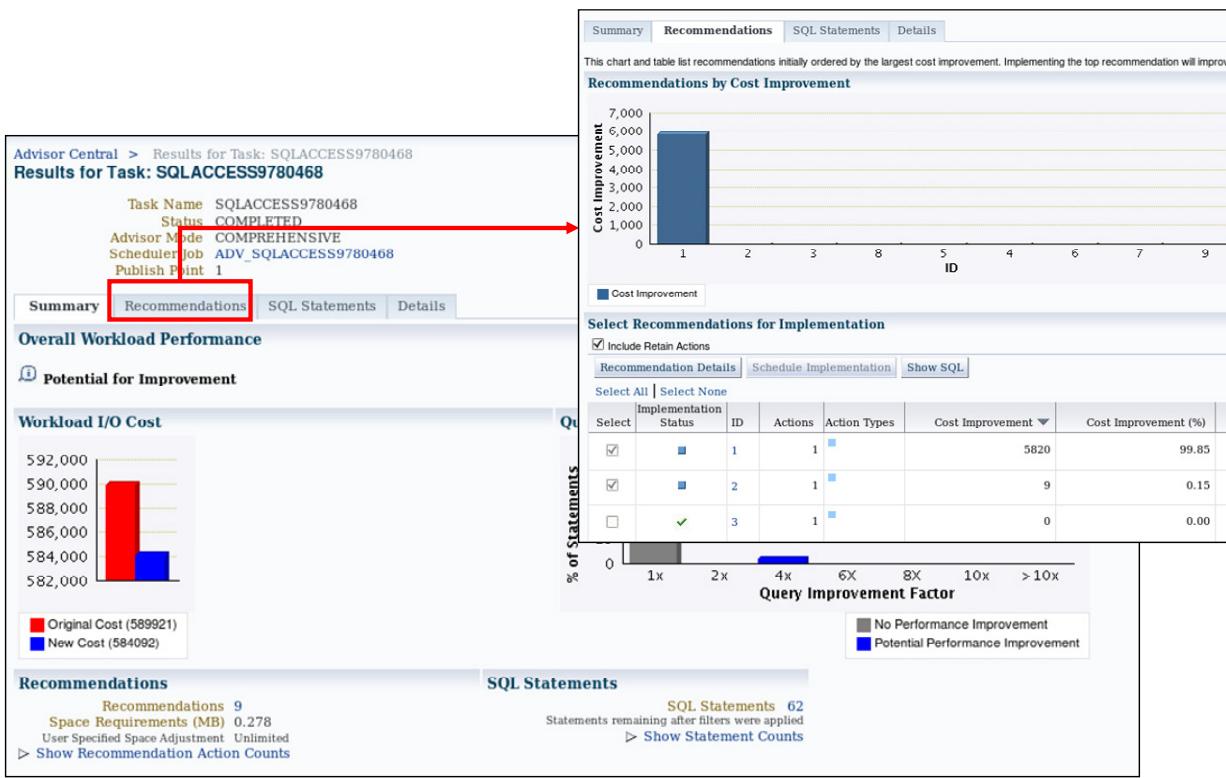
Use the Space Restrictions section to specify a hard space limit, which forces the advisor to produce recommendations only with total space requirements that do not exceed the specified limit.

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



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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.

SQL Performance Analyzer: Overview

- Targeted users: DBAs, QAs, application developers
- Helps predict the impact of system changes on SQL workload response time
- Builds different versions of SQL workload performance (that is, SQL execution plans and execution statistics)
- Executes SQL serially (concurrency not honored)
- Analyzes performance differences
- Offers fine-grained performance analysis on individual SQL
- Is integrated with SQL Tuning Advisor to tune regressions



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle Database includes SQL Performance Analyzer, which gives you an exact and accurate assessment of the impact of change on the SQL statements that make up the workload. SQL Performance Analyzer helps you forecast the impact of a potential change on the performance of a SQL query workload. This capability provides you with detailed information about the performance of SQL statements, such as before-and-after execution statistics, and statements with performance improvement or degradation. This enables you (for example) to make changes in a test environment to determine whether the workload performance will be improved through a database upgrade.

SQL Performance Analyzer: Use Cases

SQL Performance Analyzer is beneficial in the following use cases:

- Database upgrades
- Implementation of tuning recommendations
- Schema changes
- Statistics gathering
- Database parameter changes
- OS and hardware changes

It is accessible through Enterprise Manager and the DBMS_SQLPA package.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

SQL Performance Analyzer can be used to predict and prevent potential performance problems for any database environment change that affects the structure of the SQL execution plans. The changes can include (but are not limited to) any of the following:

- Database upgrades
- Implementation of tuning recommendations
- Schema changes
- Statistics gathering
- Database parameter changes
- OS and hardware changes

You can use SQL Performance Analyzer to predict SQL performance changes that result from changes for even the most complex environments. As applications evolve through the development life cycle, database application developers can test changes to schemas, database objects, and rewritten applications to mitigate any potential performance impact.

SQL Performance Analyzer also enables the comparison of SQL performance statistics.

You can access SQL Performance Analyzer through Enterprise Manager or by using the DBMS_SQLPA package.

For details about the DBMS_SQLPA package, see the *Oracle Database PL/SQL Packages and Types Reference Guide*.

Using SQL Performance Analyzer

1. Capture SQL workload on production.
2. Transport the SQL workload to a test system.
3. Build “before-change” performance data.
4. Make changes.
5. Build “after-change” performance data.
6. Compare results from steps 3 and 5.
7. Tune regressed SQL.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

1. **Gather SQL:** In this phase, you collect the set of SQL statements that represent your SQL workload on the production system.
2. **Transport:** You must transport the resultant workload to the test system. The STS is exported from the production system and the STS is imported into the test system.
3. **Compute “before-version” performance:** Before any changes take place, you execute the SQL statements, collecting baseline information that is needed to assess the impact that a future change might have on the performance of the workload.
4. **Make a change:** After you have the before-version data, you can implement your planned change and start viewing the impact on performance.
5. **Compute “after-version” performance:** This step takes place after the change is made in the database environment. Each statement of the SQL workload runs under a mock execution (collecting statistics only), collecting the same information as captured in step 3.
6. **Compare and analyze SQL Performance:** After you have both versions of the SQL workload performance data, you can carry out the performance analysis by comparing the after-version data with the before-version data.
7. **Tune regressed SQL:** At this stage, you have identified exactly which SQL statements may cause performance problems when the database change is made. Here, you can use any of the database tools to tune the system. After implementing any tuning action, you should repeat the process to create a new after-version and analyze the performance differences to ensure that the new performance is acceptable.

Quiz

Even when you enable Automatic Maintenance tasks, the SQL Tuning Advisor always has to be started separately.

- a. True
- b. False



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b

Quiz

You can receive performance recommendations for historical SQL statements that are collected by AWR snapshots.

- a. True
- b. False



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: a

Quiz

The SQL Access Advisor can recommend the proper set of materialized views, materialized view logs, partitioning, and indexes for a given workload.

- a. True
- b. False



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: a

Quiz

The SQL Performance Analyzer provides you with detailed information about the performance of SQL statements, such as before-and-after execution statistics, and statements with performance improvement or degradation.

- a. True
- b. False



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: a

Summary

In this lesson, you should have learned how to:

- Manage optimizer statistics
- Use the SQL Tuning Advisor to:
 - Identify SQL statements that are using the most resources
 - Tune SQL statements that are using the most resources
- Use the SQL Access Advisor to tune a workload



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice: Overview

This practice covers using SQL Tuning Advisor.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Using Database Resource Manager

20

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

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

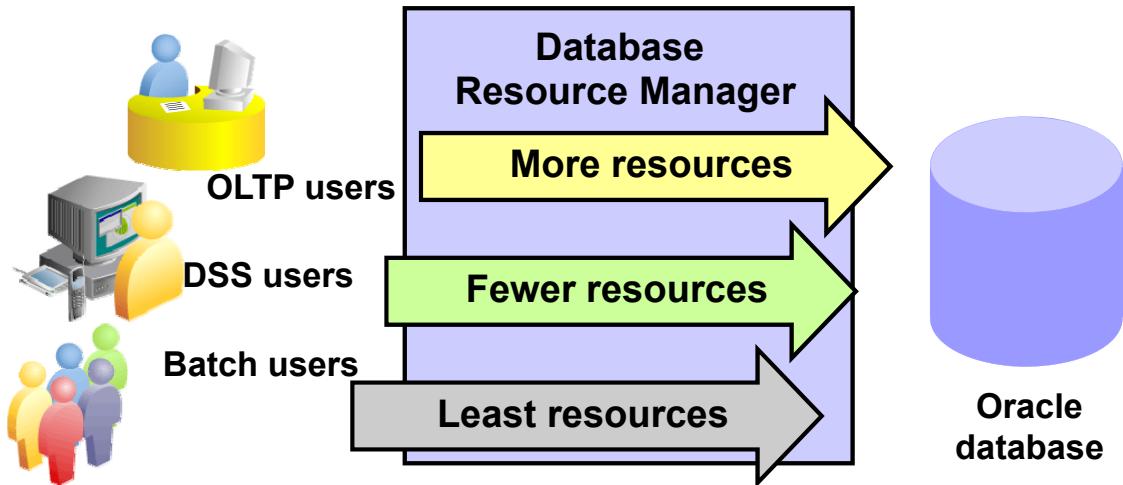


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Database Resource Manager: Overview

Use the Resource Manager to:

- Manage mixed workload
- Control system performance



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

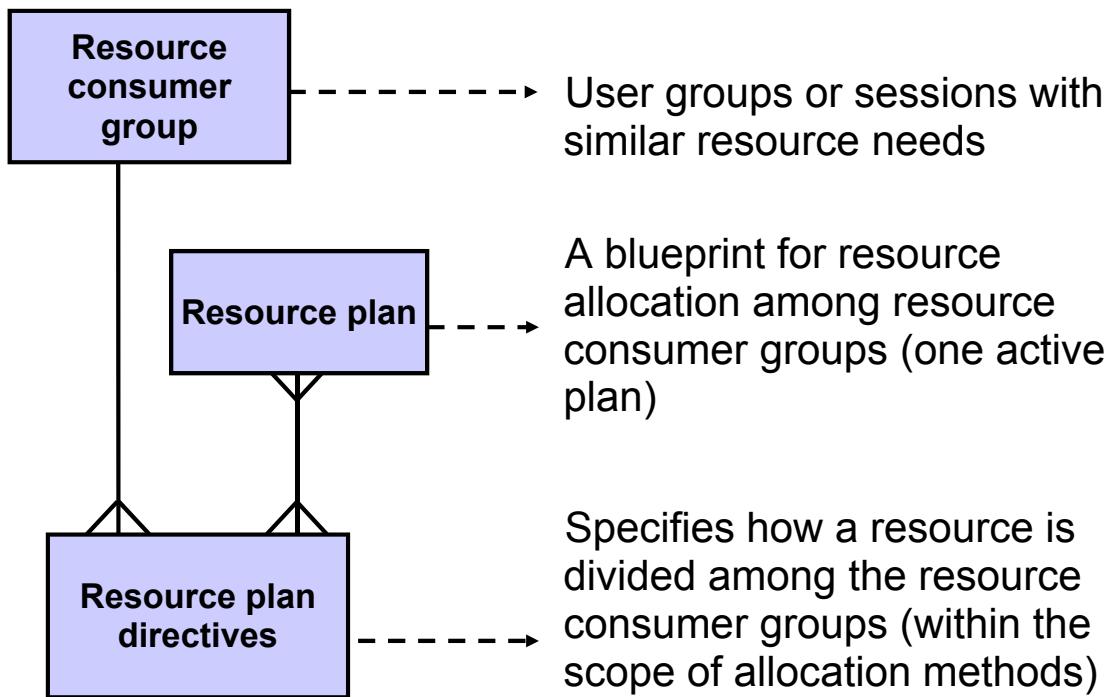
By using the Database Resource Manager (also called the Resource Manager), you have more control over the allocation of machine resources than is normally possible through operating system resource management alone. If resource management decisions are made by the operating system, it can lead to problems 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 process 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.

The DBMS_RESOURCE_MANAGER_PRIVS package contains the procedures to grant and revoke the ADMINISTER_RESOURCE_MANAGER system privilege, which is a prerequisite for invoking the Resource Manager.

Database Resource Manager: Concepts



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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.

Using the 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
 - Database consolidation
 - Server consolidation
- You can also specify criteria that, if met, cause the automatic switching of sessions to another consumer group.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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.

- **Execution Time Limit:** Allows you to specify a maximum execution time allowed for an operation. The Oracle Database server 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 terminate only those sessions that are blocking other sessions (`MAX_IDLE_TIME_BLOCKER`).
- **Consumer Group Switching:** Specifies conditions that will cause a session to switch consumer groups. Typically, overuse of a resource is specified and a session is switched to a more restrictive consumer group. The session remains in the switched consumer group until it becomes idle, or if directed after the top-level call is completed. Then it will return to the initial consumer group. The initial consumer group is the group that a session is assigned to at login. The top is the current PL/SQL block or each SQL statement that is issued outside a PL/SQL block by the client. You can create a plan directive, so that the Resource Manager automatically switches the user back to the initial consumer group at the end of the top call.
- **Database Consolidation:** The Resource Manager enables you to optimize resource allocation among concurrent database sessions. Database consolidation requires that applications are isolated from each other. If one application experiences an increase in workload, that increase should not affect other applications. In addition, the performance of each application should be consistent. Good candidate applications for database consolidation are automated maintenance tasks because currently, these applications can take up to 100% of the server CPU resources.
- **Server Consolidation:** Because many test, development and small production databases are unable to fully utilize the servers that they are on, *server consolidation* provides a possible alternative. With server consolidation, resources are more fully utilized by running multiple database instances on the server. The method for managing CPU allocations on a multi-CPU server with multiple database instances is called Instance Caging. Because Instance Caging is simple to configure and does not require any new software to be licensed or installed, it is an excellent alternative to other server consolidation tools, such as virtualization and O/S workload managers.

You can access resource plans with the graphical interface of Enterprise Manager Cloud Control or the command line of the `DBMS_RESOURCE_MANAGER` package.

Default Plan for Maintenance Windows

Resource Plans > View Resource Plan: DEFAULT_MAINTENANCE_PLAN

View Resource Plan: DEFAULT_MAINTENANCE_PLAN

Logged in as DBA1

Actions Create Like Go Edit Return

General Parallelism Runaway Query Idle Time

A Resource Plan contains directives that specify how resources are allocated to Consumer Groups. For each Consumer Group, a directive specifies the amount of CPU resources are allocated. It also specifies limits, such as the maximum degree of parallelism, execution time, and amount of I/O, that each session in the Consumer Group can consume. You can enable a Resource Plan manually or automatically, using Scheduler Windows. The maximum number of consumer groups in a plan can not exceed 28.

Plan: DEFAULT_MAINTENANCE_PLAN

Description: Default plan for maintenance windows that prioritizes SYS_GROUP operations, leaving 5% for automated maintenance operations and 20% for other tasks.

Activate this plan
 Automatic Plan Switching Enabled

Resource Allocations

Group/Subplan	Shares	Percentage	Utilization Limit %
ORA\$AUTOTASK	5	5	90
SYS_GROUP	75	75	
OTHER_GROUPS	20	20	
Total Shares:	100		



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The automated maintenance tasks rely on the Resource Manager being enabled during the maintenance windows. When a maintenance window opens, the DEFAULT_MAINTENANCE_PLAN resource manager plan is automatically set to control the amount of CPU that is used by the automated maintenance tasks. To be able to give different priorities to each possible task during a maintenance window, various consumer groups are assigned to DEFAULT_MAINTENANCE_PLAN.

Default Plan

The screenshot shows the Oracle Database Resource Plans interface. At the top, it says "Resource Plans > Edit Resource Plan: DEFAULT_MAINTENANCE_PLAN". On the right, it says "Logged in as DBA1". Below that is a toolbar with "Actions", "Create Like", "Go", "Execute On Multiple Databases", "Show SQL", "Revert", and "Apply". There are tabs for "General", "Parallelism", "Runaway Query", and "Idle Time". The "General" tab is selected.

A descriptive text block states: "A Resource Plan contains directives that specify how resources are allocated to Consumer Groups. For each Consumer Group, a directive specifies the amount of CPU resources are allocated. It also specifies limits, such as the maximum degree of parallelism, execution time, and amount of I/O, that each session in the Consumer Group can consume. You can enable a Resource Plan manually or automatically, using Scheduler Windows. The maximum number of consumer groups in a plan can not exceed 28."

The "Plan" section shows "DEFAULT_MAINTENANCE_PLAN". The "Description" field contains: "Default plan for maintenance windows that prioritizes SYS_GROUP operations, leaving 5% for automated maintenance operat". There are two checkboxes: "Activate this plan" (unchecked) and "Automatic Plan Switching Enabled" (unchecked).

The "Resource Allocations" section has a table:

Group/Subplan	Shares	Percentage	Utilization Limit %
ORA\$AUTOTASK	5	5	90
SYS_GROUP	75	75	
OTHER_GROUPS	20	20	
Total Shares:	100		

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The DEFAULT_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.
- **ORA\$AUTOTASK:** A group with lower priority than SYS_GROUP and OTHER_GROUPS in this plan

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 DEFAULT_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 Simple Resource Plan

Create consumer groups and allocate resources to them by executing a single procedure call:

```
BEGIN  
DBMS_RESOURCE_MANAGER.CREATE_SIMPLE_PLAN(SIMPLE_PLAN =>  
  'SIMPLE_RESPLAN1',  
  CONSUMER_GROUP1 => 'CONSGROUP1', GROUP1_PERCENT => 80,  
  CONSUMER_GROUP2 => 'CONSGROUP2', GROUP2_PERCENT => 20);  
END;
```

Consumer Group	Level 1	Level 2	Level 3
SYSGROUP	100%		
CONSGROUP1		80%	
CONSGROUP2		20%	
OTHER_GROUPS			100%



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can create a simple resource plan by using the DBMS_RESOURCE_MANAGER.CREATE_SIMPLE_PLAN procedure. You can create consumer groups and allocate resources to them by using this single procedure.

The CREATE_SIMPLE_PLAN procedure accepts the following arguments:

- SIMPLE_PLAN: Name of the plan
- CONSUMER_GROUP1 (through CONSUMER_GROUP8): Consumer group name(s)
- GROUP1_PERCENT (through GROUP8_PERCENT): CPU resource allocated to the group(s)

The plan uses the default CPU allocation policy (**EMPHASIS**). Each consumer group uses the default scheduling policy (**ROUND_ROBIN**).

The consumer groups specified in the plan are allocated CPU percentage at level 2 as shown in the table in the slide. The plan also includes the **SYS_GROUP** consumer group and the **OTHER_GROUPS** consumer group. As shown in the table in the slide, **SYSGROUP** is allocated 100% of the CPU at level 1 and **OTHER_GROUPS** is allocated 100% of the CPU at level 3.

Creating a Complex Resource Plan

1. Create a pending area.
2. Create, modify, or delete consumer groups.
3. Map sessions to consumer groups.
4. Create the resource plan.
5. Create resource plan directives.
6. Validate the pending area.
7. Submit the pending area.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can perform the following steps by using the named procedures in the DBMS_RESOURCE_MANAGER package or by using Enterprise Manager Cloud Control.

1. Create a pending area: To create a new resource plan, or update or delete an existing resource plan, you must create a pending area. The pending area is a staging area where you can create and modify plans without affecting executing applications. After you create the pending area, the Oracle Database server copies existing plans into it so they can be updated if required. Create a pending area by using the CREATE_PENDING_AREA procedure.
2. Create, modify, or delete consumer groups: Create a resource consumer group and specify a resource allocation method (ROUND-ROBIN or RUN-TO-COMPLETION) for distributing CPU among the sessions in the consumer group. Use the CREATE_CONSUMER_GROUP procedure.
3. Map sessions to consumer groups: Use the SET_CONSUMER_GROUP_MAPPING procedure to map a session attribute type and attribute value to a consumer group. The session attribute types include Oracle Database username, database service name, operating system username, client program name, and module name. Refer to the *Oracle Database Administrator's Guide* for a complete list of accepted session attributes.

Note: Unassigned sessions are part of the OTHER_GROUPS consumer group.

4. Create the resource plan: When you create the resource plan, you provide a name and optionally specify the resource allocation method for CPU (`EMPHASIS` or `RATIO`), active session pool resource allocation method (`ACTIVE_SESS_POOL_ABSOLUTE`), resource allocation method for degree of parallelism (`PARALLEL_DEGREE_LIMIT_ABSOLUTE`), and queuing resource allocation method (`FIFO_TIMEOUT`). Use `CREATE_PLAN` to create a resource plan. Additional information follows in this lesson.
 5. Create resource plan directives: Resources are allocated to consumer groups based on the resource plan directives. Through resource plan directives, you can specify:
 - The maximum number of concurrently active sessions for a consumer group
 - A limit on the degree of parallelism for any operation
 - The time (in CPU seconds) that a call can execute before an action is taken
 - A maximum in kilobytes (K) on the total amount of undo for uncommitted transactions that can be generated by a consumer group
- Use `CREATE_PLAN_DIRECTIVE` to specify resource plan directives.
6. Validate the pending area: Use the `VALIDATE_PENDING_AREA` procedure to validate the pending area at any time. The validate procedure checks for the following:
 - Plans do not contain loops
 - All plans and resource consumer groups referred to by plan directives exist
 - All plans have plan directives that point to either plans or resource consumer groups
 - All percentages in any given level do not add up to greater than 100
 - A plan that is currently being used as a top plan by an active instance is not being deleted
 - That certain parameters appear only in plan directives that refer to resource consumer groups
 - No more than 28 resource consumer groups appear in any active plan
 - Plans and resource consumer groups do not have the same name
 - A plan directive for `OTHER_GROUPS` appears somewhere in any active plan
 7. Submit the pending area: After validating the pending area, submit it by using `SUBMIT_PENDING_AREA`. When you submit the pending area, new and updated plan information is stored in the data dictionary. New plans are not activated when the pending area is submitted. Modified plans are reactivated with their new plan definition.

Specifying Resource Plan Directives

The image displays three separate property pages from Oracle Enterprise Manager Cloud Control, each with tabs for General, Parallelism, Runaway Query, and Idle Time. The Parallelism page shows settings for consumer groups (ORASAUTOTASK, SYS_GROUP, OTHER_GROUPS) regarding parallelism limits. The Runaway Query page shows resource limits (Elapsed Time, CPU Time, I/O, Logical I/O, Request Limit) for the same groups. The Idle Time page shows the maximum idle time allowed for sessions in these groups, including a column for sessions blocking others.

Group	Bypass Queue	Max Degree of Parallelism	Parallel Server Limit	Parallel Statement Queue Timeout
ORASAUTOTASK	<input type="checkbox"/>			
SYS_GROUP	<input type="checkbox"/>			
OTHER_GROUPS	<input type="checkbox"/>			

Group	Elapsed Time Limit (Sec)	CPU Time Limit (Sec)	I/O Limit (MB)	Logical I/O Limit	I/O Request Limit (Requests)	Action
ORASAUTOTASK						
SYS_GROUP						
OTHER_GROUPS						

Group	Max Idle Time (sec)	Max Idle Time if Blocking Another Session (sec)
ORASAUTOTASK		
SYS_GROUP		
OTHER_GROUPS		

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In Enterprise Manager Cloud Control, there are several property pages that you can use to specify plan directives:

1. On the General page, associate consumer groups with plans and specify how much CPU each consumer group or subplan is allocated. See page 8 in this lesson for a screenshot of the General page.
2. On the Parallelism page, specify a limit on the degree of parallelism for any operation issued by this consumer group, a limit on the total number of parallel server processes that can be used by all sessions in this consumer group, and the maximum time a parallel statement can be queued.
3. On the Thresholds page, specify the time duration or the resource limits under which a session can execute in a consumer group.
4. On the Idle Time page, 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 those sessions that are blocking other sessions.

Resource Allocation Methods for Resource Plans

Parameter	Possible Values
MGMT_MTH: Allocating CPU usage	EMPHASIS, RATIO
PARALLEL_DEGREE_LIMIT_MTH: Limiting degree of parallelism of any operation	PARALLEL_DEGREE_LIMIT_ABSOLUTE
ACTIVE_SESS_POOL_MTH: Limiting number of active sessions, queuing inactive ones	ACTIVE_SESS_POOL_ABSOLUTE
QUEUING_MTH: Controlling queues, how inactive sessions enter active session pool	FIFO_TIMEOUT



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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 MGMT_MTH parameter when you create a resource plan:

- EMPHASIS is the default method for single-level plans. It is also used 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 PARALLEL_DEGREE_LIMIT_ABSOLUTE method is the only 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.

The ACTIVE_SESS_POOL_MTH parameter limits the number of active sessions. All other sessions are inactive and wait in a queue to be activated. The only value (the only available method) for this parameter is ACTIVE_SESS_POOL_ABSOLUTE, which is its default value.

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 eight 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 © 2014, Oracle and/or its affiliates. All rights reserved.

The CPU allocation method **EMPHASIS** 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 the 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.

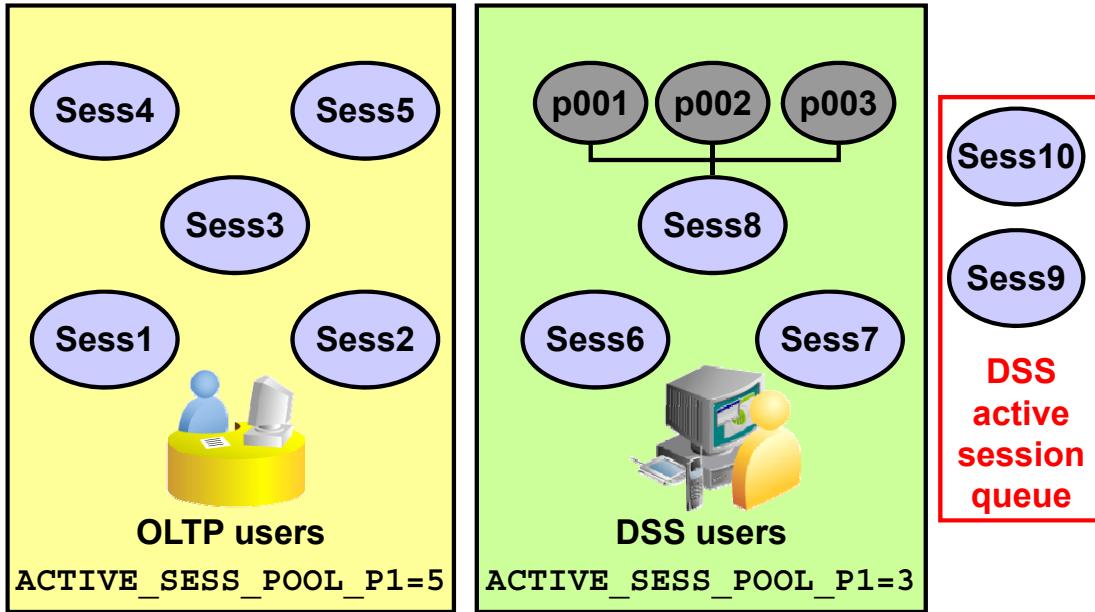
The RATIO policy is a single-level CPU allocation method. Instead of percentages, you specify numbers corresponding to the ratio of CPU you want to give to the consumer group. For example, given three consumer groups OLTP_USERS, DSS_USERS, and BATCH_USERS, you can specify the following ratios:

- 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, the OLTP_USERS consumer group would get two-thirds (4 shares out of 6 shares) of the available resources and the BATCH_USERS consumer group would get the other third (2 shares out of 6). If all groups had sessions running, the division would be based on 10 shares.

Active Session Pool Mechanism



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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.

Specifying Thresholds

Specifying execution time limit:

- Proactive estimation of the execution time for an operation (via cost-based optimizer statistics), default: UNLIMITED
- Specifying maximum estimated execution time at the resource consumer group level
- Huge jobs will not be allowed to start if the estimate is longer than `MAX_EST_EXEC_TIME`: (ORA-07455)

Specifying other thresholds:

- Limiting session I/O with `SWITCH_IO_MEGABYTES` (in MB)
- Limiting session I/O requests with `SWITCH_IO_REQS`

Returning to original consumer group with `SWITCH_FOR_CALL`
(Default: FALSE, consumer group is not restored)



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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, which is calculated using the statistics from the cost-based optimizer.
- 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.
- If the operation's estimate is more than `MAX_EST_EXEC_TIME`, 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.
- The `SWITCH_IO_MEGABYTES` directive specifies the amount of I/O (in MB) that a session can issue before an action is taken. The default is `NULL`, which means unlimited.
- The `SWITCH_IO_REQS` directive specifies the number of I/O requests that a session can issue before an action is taken. The default is `NULL`, which means unlimited.
- The `SWITCH_FOR_CALL` directive specifies that if an action is taken because of the `SWITCH_TIME`, `SWITCH_IO_MEGABYTES`, or `SWITCH_IO_REQS` parameters, the consumer group is restored to its original consumer group at the end of the top call. Default is FALSE, which means that the original consumer group is not restored at the end of the top call.

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.

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, whereas the `SWITCH_TIME_IN_CALL` parameter is for three-tier applications.

Setting Idle Timeouts

```
DBMS_RESOURCE_MANAGER.UPDATE_PLAN_DIRECTIVE
(PLAN => 'DAY_PLAN',
 GROUP_OR_SUBPLAN => 'APPUSER',
 COMMENT => 'Limit Idle Time Example',
 NEW_MAX_IDLE_TIME => 600,
 NEW_MAX_IDLE_BLOCKER_TIME => 300);
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In Enterprise Manager Cloud Control, use the 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 NEW_MAX_IDLE_TIME and NEW_MAX_IDLE_BLOCKER_TIME resource directives in the DBMS_RESOURCE_MANAGER.UPDATE_PLAN_DIRECTIVE procedure. They are both specified in seconds.

NEW_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 NEW_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.

Limiting CPU Utilization at the Database Level

- Database consolidation requirements:
 - Applications isolated from each other
 - Consistent performance
- CPU directives can be used to:
 - Specify a minimum CPU allocation for each application
 - Designate how unused allocations should be redistributed
 - Specify the `MAX_UTILIZATION_LIMIT` attribute to impose an absolute upper limit on CPU utilization (which overrides any redistribution of CPU within a plan)
 - Good candidate: Auto-maintenance tasks



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

For concurrent database sessions: Database consolidation requires that applications are isolated from each other. If one application experiences an increase in workload, that increase should not affect other applications. In addition, the performance of each application should be consistent.

Fixed Policy CPU Resource Management

The `MAX_UTILIZATION_LIMIT` attribute of resource plan directives enables you to impose an absolute upper limit on CPU utilization for a resource consumer group. This absolute limit overrides any redistribution of CPU within a plan.

Note: Good candidate applications for database consolidation are automated maintenance tasks because currently these applications can take up to 100% of the server CPU resources. You can set a maximum limit for each auto-task consumer group.

Limiting CPU Utilization at the Database Level

Specify minimum and maximum CPU utilization limits.

DB Consolidation Plan #1

	CPU Allocation	Maximum Utilization Limit
App 1	50%	60%
App 2	20%	30%
App 3	20%	30%
App 4	10%	20%

Specify maximum CPU utilization limits only.

DB Consolidation Plan #2

	CPU Allocation	Maximum Utilization Limit
App 1	null	50%
App 2	null	20%
App 3	null	20%
App 4	null	10%

```
EXEC DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE( -
    plan          => 'db_consolidation_plan',
    group_or_subplan => 'App_1',
    mgmt_p1        => 50,
    max_utilization_limit => 60);
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The MAX_UTILIZATION_LIMIT directive limits the CPU consumption of an application. You can set minimum and maximum boundaries, as shown in the slide.

The PL/SQL example in the slide specifies a minimum value percentage (50%) for the CPU allocation resource at level 1 for the APP_1 consumer group. This example also specifies an absolute maximum CPU utilization percentage (60%) permitted for that same consumer group. The example uses the DB_CONSOLIDATION_PLAN plan.

Similar commands can be executed for each consumer group shown in the sample tables.

Limiting CPU Utilization at the Server Level: Instance Caging

- Managing CPU allocations on a multi-CPU server with multiple database instances
- Enabling instance caging:
 - Enable any CPU resource plan.

```
ALTER SYSTEM SET resource_manager_plan = 'default_plan';
```

- Specify the maximum number of CPUs that the instance can use at any time.

```
ALTER SYSTEM SET cpu_count=4;
```

- Two approaches:
 - Over-provisioning: The sum of the CPU limit for each instance exceeds the actual number of CPUs.
 - Partitioning: The sum of the CPU limit for each instance equals the actual number of CPUs.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Because many test, development, and small production databases are unable to fully utilize the servers that they are on, *server consolidation* provides a possible alternative. With server consolidation, resources are more fully utilized by running multiple database instances on the server. However, this may bring about CPU contention and an adverse impact due to workload surges on one instance.

Instance caging is a method that uses the `CPU_COUNT` initialization parameter to limit the number of CPUs that an instance can use. In addition, the Resource Manager is employed to allocate the CPUs for the database sessions based on the instance resource plan.

Configure instance caging in two steps, by enabling:

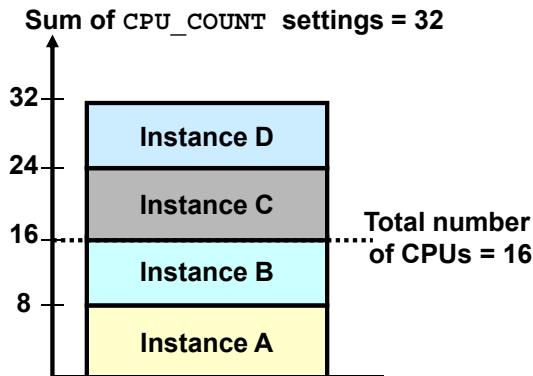
- The Resource Manager, which limits the amount of CPU that the database instance consumes
- The `CPU_COUNT` parameter, which specifies the maximum (the limit), not actual amount of CPU that the database instance can use at any time

By default, the CPU Resource Manager assumes that the database instance can use all CPUs on a server. To enable instance caging, any resource plan with CPU directives can be used.

Instance Caging: Examples

Over-provisioning approach:

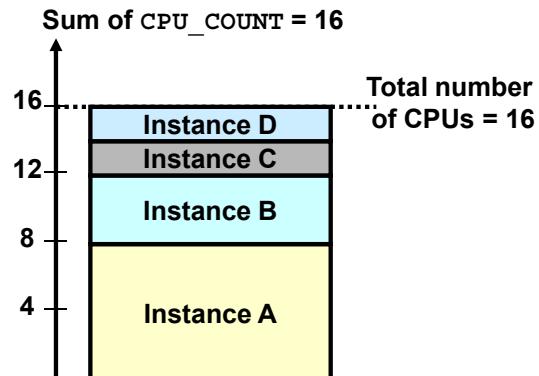
One database instance can still impact the others.



With all four instances active, one instance can get
 $8 / (8 + 8 + 8 + 8) = 25\%$ of CPU.

Partitioning approach:

One database instance cannot impact the others.



Each instance has a dedicated number of CPUs.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Over-provisioning approach: This approach is appropriate for noncritical databases and low-load, noncritical production systems. Although the instances impact each other's performance, at any given time, one or more of the instances may be idle or experiencing a low load.

Although the database instances can impact each other's performance, instance caging limits their impact and helps to provide predictable performance. In the example on the left, where all four instances have CPU_COUNT set to 8, the maximum percentage of CPU that a database instance can consume at any point in time is its own limit divided by the sum of the limits for all active databases. In this example, one instance will be able to consume $8 / (8 + 8 + 8 + 8) = 25\%$ of the CPU. If only two instances are active, one instance will be able to consume $8 / (8 + 8) = 50\%$ of the CPU.

Partitioning approach: This approach is appropriate for critical product systems. It prevents the instances from interfering with each other and provides predictable performance.

Instance caging can partition the CPU resources by ensuring that the sum of all CPU limits does not exceed the total number of CPUs. In the example on the right, if four database instances share a 16-CPU server, their limits can be set to 8, 4, 2, and 2. By dedicating CPU resources to a database instance, partitioning provides two advantages:

- One database instance's CPU load cannot affect another's.
- Each database instance's CPU resources is fixed, leading to more predictable performance.

Monitoring Instance Caging

View value of the CPU_COUNT parameter:

```
SELECT value FROM v$parameter WHERE name = 'cpu_count'  
    AND (isdefault = 'FALSE' OR ismodified != 'FALSE');
```

Determine the Resource Manager status:

```
SELECT name FROM v$rsrc_plan  
WHERE is_top_plan = 'TRUE' AND cpu_managed = 'ON';
```

Manage throttling:

```
SELECT begin_time, consumer_group_name,  
      cpu_consumed_time, cpu_wait_time  
FROM v$rsrcmgrpmetric_history  
ORDER BY begin_time;
```

```
SELECT name, consumed_cpu_time, cpu_wait_time  
FROM v$rsrc_consumer_group;
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

- If the CPU_COUNT parameter is not set, no value is returned by the first query.
- If no rows are returned by the second query in the slide, the Resource Manager is not managing the CPU. If a row is returned, it indicates the active plan.

Instance caging limits the CPU consumption of the foreground processes by throttling them. A foreground process is throttled when it is waiting on the “resmgr:cpu quantum” wait event.

You can monitor the amount of throttling in two ways:

- The V\$RSRCMGRMETRIC_HISTORY view shows the amount of CPU consumption (CPU_CONSUMED_TIME) and throttling (CPU_WAIT_TIME) for each minute in the past hour. Values are displayed in milliseconds.
- The V\$RSRC_CONSUMER_GROUP view shows the amount of CPU consumption (CPU_CONSUMED_TIME) and throttling (CPU_WAIT_TIME) since CPU Resource Management was enabled. The time is displayed in milliseconds.

Note: For case studies, see the Oracle White Paper titled *Database Instance Caging: A Simple Approach to Server Consolidation*.

Runaway Queries and Resource Manager

- Parameters used to trigger consumer group switching:
 - SWITCH_IO_LOGICAL
 - SWITCH_ELAPSED_TIME
- Meta consumer group called LOG_ONLY
- Columns in V\$SQL_MONITOR:
 - RM_LAST_ACTION
 - RM_LAST_ACTION_REASON
 - RM_LAST_ACTION_TIME
 - RM_CONSUMER_GROUP
- V\$RSRCMGRMETRIC and V\$RSRCMGRMETRIC_HISTORY always populated

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To better track runaway queries, Resource Manager included the following:

- Directive parameters:
 - SWITCH_IO_LOGICAL: Number of logical I/Os that will trigger the action specified by SWITCH_GROUP
 - SWITCH_ELAPSED_TIME: Elapsed time that will trigger the action specified by SWITCH_GROUP
- LOG_ONLY meta-consumer group: This can be used as the argument for the SWITCH_GROUP parameter to log the runaway query without changing its consumer group or taking other action.
- Resource Manager integrates the runaway query information with SQL Monitor. To retain important results for Resource Manager, it pins up to five SQL statements per consumer group. SQL Monitor does not purge these SQL executions until they are unpinned. V\$SQL_MONITOR columns:
 - RM_LAST_ACTION: The most recent action that was taken on this SQL operation by Resource Manager. Its value is one of the following: CANCEL_SQL, KILL_SESSION, LOG_ONLY, SWITCH_TO <CG NAME>

- RM_LAST_ACTION_REASON: The reason for the most recent action that was taken on this SQL operation by Resource Manager. Its value is one of the following:
SWITCH_CPU_TIME, SWITCH_IO_REQS, SWITCH_IO_MBS,
SWITCH_ELAPSED_TIME, SWITCH_IO_LOGICAL
- RM_LAST_ACTION_TIME: The time of the most recent action that was taken on this SQL operation by Resource Manager
- RM_CONSUMER_GROUP: The current consumer group for this SQL operation

Note: V\$RSRCMGRMETRIC and V\$RSRCMGRMETRIC_HISTORY will always produce a row every minute regardless of whether there is a Resource Manager plan set.

Resource Consumer Group Mapping

The screenshot shows the 'Consumer Group Mappings' page in Oracle Database Resource Manager. The 'Priorities' tab is selected, indicated by a red box and arrow. The interface includes a table for defining mappings between session attributes and consumer groups, with columns for Select, Priority, Value, and Consumer Group. A modal window titled 'Attribute Mappings' lists various session attributes: Service Module and Action, Service and Module, Module and Action, Module, Service, Oracle User, Client Program, Client OS User, Client Machine, and Client ID. Arrows next to each attribute indicate their priority relative to others.

Select	Priority ▲	Value	Consumer Group
<input checked="" type="radio"/>	1	Service Module and Action	No Mappings Specified
<input type="radio"/>	2	Service and Module	No Mappings Specified
<input type="radio"/>	3	Module and Action	No Mappings Specified
<input type="radio"/>	4	Module	No Mappings Specified
<input type="radio"/>	5	Service	No Mappings Specified
<input type="radio"/>	6	Oracle User	SYS, SYSTEM
<input type="radio"/>	7	Client Program	No Mappings Specified
<input type="radio"/>	8	Client OS User	No Mappings Specified
<input type="radio"/>	9	Client Machine	No Mappings Specified
<input type="radio"/>	10	Client ID	No Mappings Specified

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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 runtime attributes. The login attributes 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 runtime attributes.

Select Consumer Group Mappings on the Getting Started with Database Resource Manager page of Enterprise Manager Cloud Control. For each of the attributes, set up a mapping that consists of a way to identify a session (for example, username), 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. The mappings at the top of the list have the highest priority.

Example to give the Client OS User a higher priority than the Client Program:

```
BEGIN
    dbms_resource_manager.clear_pending_area();
    dbms_resource_manager.create_pending_area();
    dbms_resource_manager.set_consumer_group_mapping(
        dbms_resource_manager.oracle_user,
        'SCOTT',
        'LOW_GROUP'
    );
    dbms_resource_manager.set_consumer_group_mapping_pri(
        EXPLICIT => 1, SERVICE_MODULE_ACTION => 2,
        SERVICE_MODULE => 3,
        MODULE_NAME_ACTION => 4,
        MODULE_NAME => 5,
        SERVICE_NAME => 6,
        ORACLE_USER => 7,
        CLIENT_OS_USER => 8,
        CLIENT_PROGRAM => 9,
        CLIENT_MACHINE => 10
    );
    dbms_resource_manager.submit_pending_area();
END;
```

Activating a Resource Plan

Resource Plans

Select	Plan	Status	Description
<input type="radio"/>	APPQOS_PLAN	ACTIVE	Plan for Application QOS Management that provides a fixed set of allocation.
<input type="radio"/>	DEFAULT_MAINTENANCE_PLAN	DEACTIVATED	Default plan for maintenance windows that prioritizes SYS_GROUP operations.
<input checked="" type="radio"/>	DEFAULT_PLAN	ACTIVE	Default, basic, pre-defined plan that prioritizes SYS_GROUP operations.

SQL> show parameter resource_manager_plan

NAME	TYPE	VALUE
resource_manager_plan	string	DEFAULT_PLAN

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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, 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.

You can use the Resource Plans page of Enterprise Manager Cloud Control 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.

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 INITIAL_RSRC_CONSUMER_GROUP
DBA_RSRC_MANAGER_SYSTEM_PRIVS	Users/roles



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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.

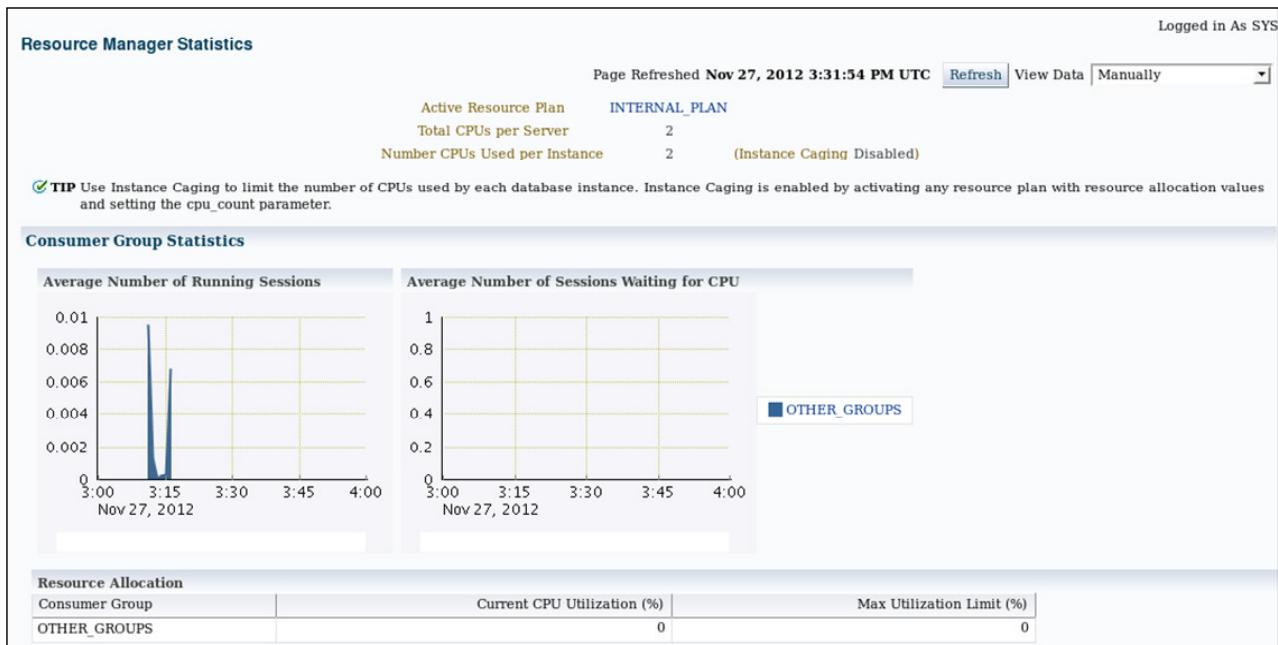
Use the following query to obtain information about resource plans defined in the database:

```
SQL> SELECT plan, num_plan_directives, status, mandatory
  2  FROM dba_rsrc_plans;
    PLAN          NUM_PLAN_DIRECTIVES  STATUS      MAN
    -----
    DEFAULT_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, the plan cannot be deleted.

Viewing Resource Manager Statistics



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can access Resource Manager statistics by selecting Performance Statistics on the Getting Started with Database Resource Manager page in Enterprise Manager Cloud Control. Information is provided so that you can monitor the currently enabled resource 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



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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 contains the following columns:

- **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

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.

Quiz

Select the statements that are true about the Resource Manager and its functionality.

- a. You can set threshold values only for execution time, not for session I/O.
- b. You can limit CPU utilization at the database level to isolate applications for each other.
- c. On a multi-CPU server with multiples database instances, you can limit each server's CPU utilization by enabling instance caging.
- d. When the SWITCH_TIME, SWITCH_IO_MEGABYTES, or SWITCH_IO_REQS parameters cause a switch in consumer groups, you can never return to the original consumer groups.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b, c

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 © 2014, Oracle and/or its affiliates. All rights reserved.

Practice: Overview

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



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

21

Using Oracle Scheduler to Automate Tasks

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

- Simplify management tasks by using Oracle Scheduler
- Create a job, program, and schedule
- Monitor job execution
- Use a time-based or event-based schedule for executing Oracle Scheduler jobs
- Describe the use of windows, window groups, job classes, and consumer groups
- Use email notification
- Use job chains to perform a series of related tasks
- Describe Scheduler jobs on remote systems
- Use advanced Scheduler features to prioritize jobs



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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 provides advanced scheduling capabilities through the Oracle 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 Cloud Control.

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.

Understanding a Simple Job

```
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;
```



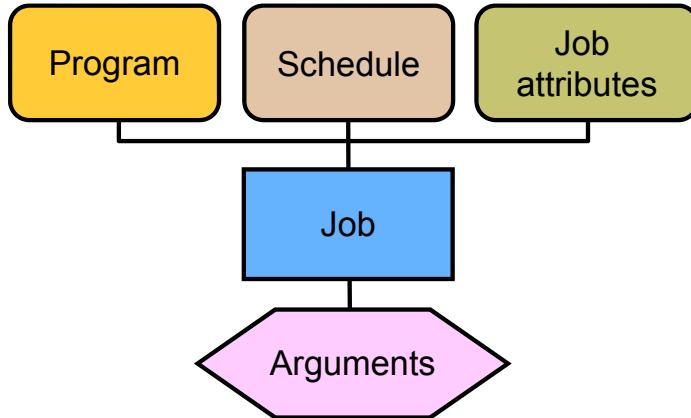
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

A job has two key components:

- *Action*: “What” needs to be done
- *Schedule*: “When” the action occurs

In the example in the slide, the “what” is enclosed in the top box and the “when” portion is shown in the second box. The “what” is expressed in the `JOB_TYPE` and `JOB_ACTION` parameters. The “when” is expressed in a “schedule,” which can be based on time (as shown by the `START_DATE` parameter) or events, or be dependent on the outcome of other jobs.

Oracle Scheduler Core Components



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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 the following basic components:

- A **job** specifies what needs to be executed. It could be as example 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 runtime 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 runtime behavior of the task.

Using Oracle Scheduler

To simplify management tasks with the Scheduler:

1. Create a program (enabled or disabled)—optional
 - To reuse this action within multiple jobs
 - To change the schedule for a job without having to re-create the PL/SQL block
2. Create and use a schedule.
3. Create and submit a job.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can perform all steps either by using Enterprise Manager Cloud Control or by using the DBMS_SCHEDULER PL/SQL package.

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.

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.

2. Creating and Using Schedules

The schedule for a job can be a predefined schedule (created with the 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

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, each job that uses that schedule automatically uses the new schedule.

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. There are many attributes that you can set for a job. Attributes control how the job executes.

Quiz

Select the statements that are true about the Scheduler.

- a. Creating a program is a mandatory part of using the Scheduler.
- b. When the job action is in a program (rather than directly in the job), you can change the job schedule without having to re-create the PL/SQL block.
- c. Creating a job is an optional part of using the Scheduler.
- d. Each job must have a schedule. It can be a predefined one or defined as part of the job creation.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b, d

Persistent Lightweight Jobs

Persistent lightweight jobs:

- Reduce the overhead and time required to start a job
- Have a small footprint on disk for the job metadata and for storing runtime data
- Are created from a job template (in the command line)

```
BEGIN
DBMS_SCHEDULER.CREATE_JOB (
    job_name          => 'my_lightweight_job2',
    program_name      => 'MY_PROG',
    schedule_name     => 'MY_SCHED',
    job_style         => 'LIGHTWEIGHT');
END;
/
```

Choosing the right job:

- Use regular jobs for maximum flexibility.
- Use persistent lightweight jobs when you need to create a large number of jobs in a very short time.

ORACLE

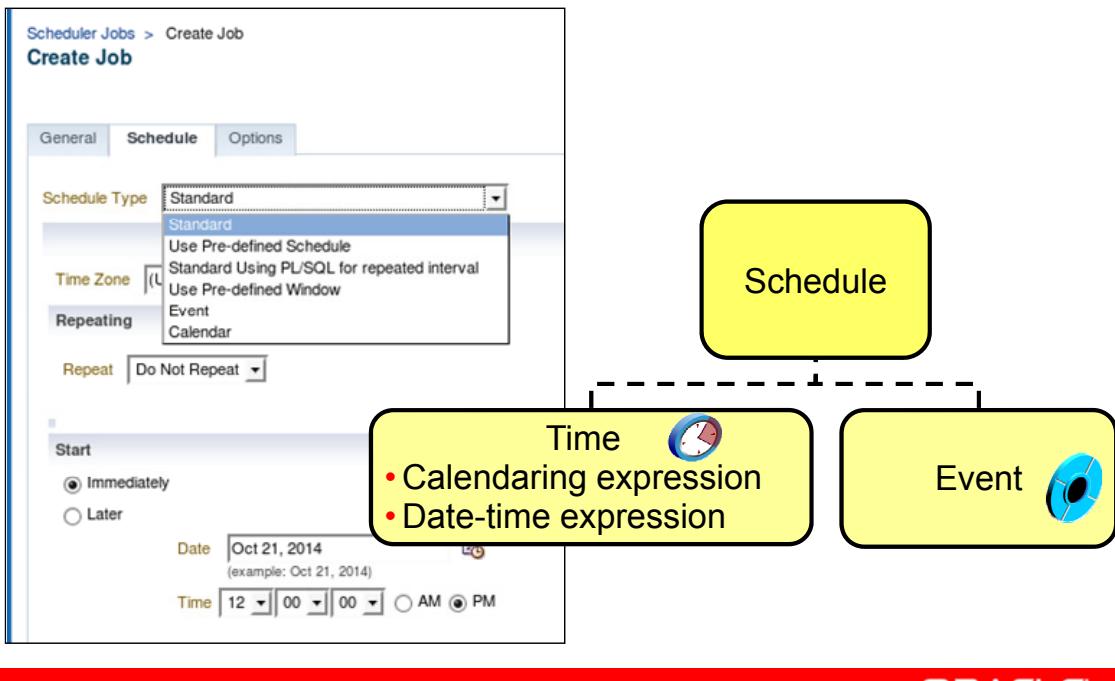
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

A lightweight job:

- Is for customers who need to create hundreds of jobs a second. With regular jobs, each job creates a database object describing the job, modifying several tables, and creating redo in the process. The overhead associated with this type of job need is substantial. In the Oracle Database Scheduler, there is a *persistent lightweight job*. The goal of a lightweight job is to reduce the overhead and time required to start a job. A minimal amount of metadata is created for the job. This reduces the time required and redo created when the job starts.
- Has a small footprint on disk for the job metadata and for storing runtime data. The small footprint on disk also makes load-balancing possible in RAC environments.
- Is always created from a job template. The job template must be a stored procedure or a program. The stored procedure holds all the information needed for the job, including privileges. A few job attributes can be specified: job arguments and schedule.
- Must be created in command line. The `JOB_STYLE` argument is not available in Enterprise Manager Cloud Control.

In the example, `MY_PROG` is the job template and the schedule is applied from a named schedule.

Using a Time-Based or Event-Based Schedule



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To specify a time-based schedule for a job, you can specify either a calendaring expression or a date-time expression. When you use a calendaring expression, the next start time for a job is calculated by using the repeat interval and the start date of the job. When you use date-time 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 © 2014, Oracle and/or its affiliates. All rights reserved.

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 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 (OS) command line

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.

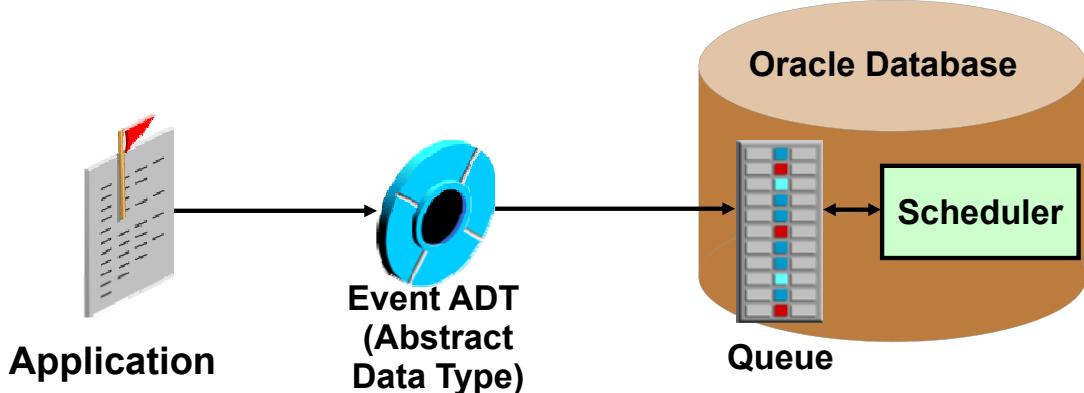
An external job is a job that runs outside the database. All external jobs run as a low-privileged guest user, as has been determined by the database administrator while configuring external job support. Because the executable is run as a low-privileged guest account, you should verify that it has access 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.

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

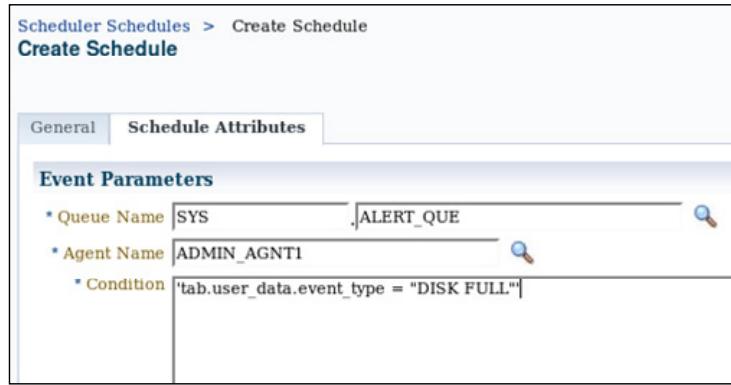
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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 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. 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 inline job attributes, or create an event-based schedule with these attributes and then create a job that references this schedule.

Creating Event-Based Schedules by Using Enterprise Manager Cloud Control



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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, 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 *Oracle Database PL/SQL Packages and Types Reference*.

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 AM.

```
BEGIN
  DBMS_SCHEDULER.CREATE_JOB(
    job_name=>'ADMIN.PERFORM_DATA_LOAD',
    job_type => 'EXECUTABLE',
    job_action => '/loaddir/start_my_load.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 © 2014, Oracle and/or its affiliates. All rights reserved.

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 inline as job attributes or can specify event information by pointing to an event schedule. The example shown in the slide uses an inline, 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 AM. 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-generated or application-generated events
- Scheduler-generated events

Events raised by Scheduler jobs:

- JOB_STARTED
- JOB_SUCCEEDED
- JOB_FAILED
- JOB_BROKEN
- JOB_COMPLETED
- JOB_STOPPED
- JOB_SCH_LIM_REACHED
- JOB_DISABLED
- JOB_CHAIN_STALLED
- JOB_ALL_EVENTS
- JOB_RUN_COMPLETED
- JOB_OVER_MAX_DUR

Example of raising an event:

```
DBMS_SCHEDULER.SET_ATTRIBUTE('hr.do_backup',
  'raise_events', DBMS_SCHEDULER.JOB_FAILED);
```

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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-generated 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.

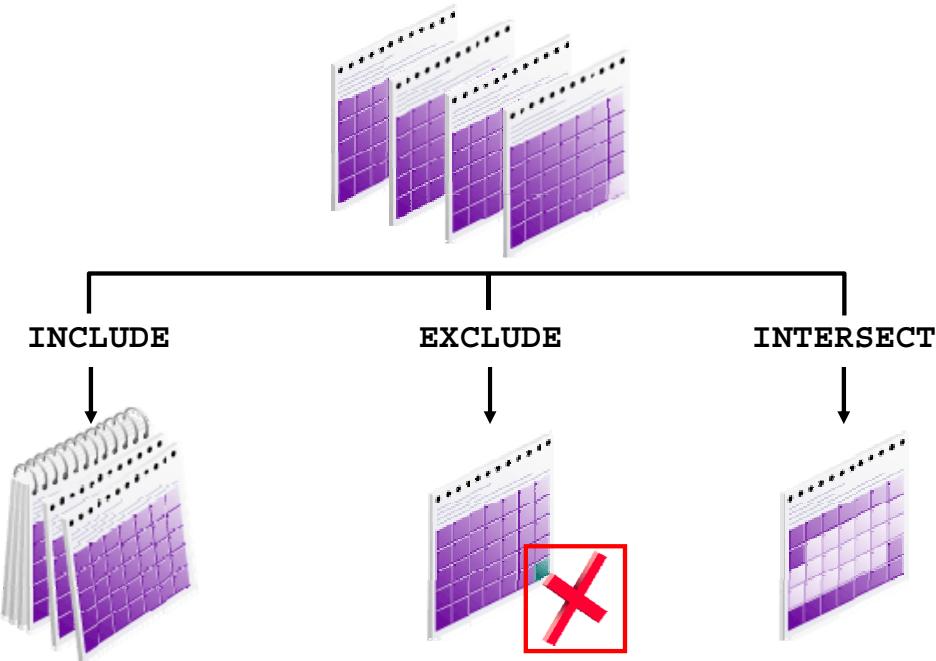
After you enable job state change events for a job, the Scheduler raises these events by enqueueing messages onto the default event queue `SYS.SCHEDULER$_EVENT_QUEUE`.

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_OVER_MAX_DUR</code>	The job has run over the maximum time it was set to be allowed to run.
<code>JOB_RUN_COMPLETED</code>	A job run is completed. It either failed, succeeded, or is stopped.

Creating Complex Schedules



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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.

Quiz

Select the statements that are true about persistent lightweight jobs:

- a. Persistent lightweight jobs have a small footprint on disk for the job metadata and also for storing runtime data.
- b. Use persistent lightweight jobs for maximum flexibility.
- c. Persistent lightweight jobs are created from a job template.
- d. Persistent lightweight jobs can be created in Enterprise Manager and via command line.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: a, c

Using Email Notification

- Email notifications for change of job state
- Triggered by job state events
- Multiple notifications, multiple recipients
- *_SCHEDULER_NOTIFICATIONS views

Using Scheduler Email Notification:

1. Specify the address of the SMTP server you will use to send email messages:

```
DBMS_SCHEDULER.SET_SCHEDULER_ATTRIBUTE  
('email_server','host[:port]');
```

2. Optionally, set a default sender email address:

```
DBMS_SCHEDULER.SET_SCHEDULER_ATTRIBUTE  
('email_sender','valid_email_address');
```

3. Add email notifications for a specified job. *(continued)*

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The job email notification feature enables you to add email notifications to existing jobs so that events of interest that happen to the job are emailed to specified email addresses. For each job, you can add notifications for different events. You can send the email notification to more than one recipient.

To enable the email notification feature, you must:

1. Set the EMAIL_SERVER Scheduler attribute.
2. Optionally, use the EMAIL_SENDER Scheduler attribute to specify a default sender email address for the email notifications.
3. After creating a job, execute the DBMS_SCHEDULER.ADD_JOB_EMAIL_NOTIFICATION procedure to add one or more notifications for the job.

The data dictionary supports email notifications with the *_SCHEDULER_NOTIFICATIONS views.

Adding and Removing Email Notifications

```
DBMS_SCHEDULER.ADD_JOB_EMAIL_NOTIFICATION (
    job_name          IN VARCHAR2,
    recipients        IN VARCHAR2,           Comma-separated list of
    sender            IN VARCHAR2 DEFAULT NULL,
    subject           IN VARCHAR2
        DEFAULT dbms_scheduler.default_notification_subject,
    body              IN VARCHAR2
        DEFAULT dbms_scheduler.default_notification_body,
    events            IN VARCHAR2           Mandatory comma-separated list
        DEFAULT 'JOB FAILED,JOB BROKEN,JOB SCH_LIM_REACHED,
                  JOB_CHAIN_STALLED,JOB_OVER_MAX_DUR',
    filter_condition IN VARCHAR2 DEFAULT NULL);
```

```
DBMS_SCHEDULER.REMOVE_JOB_EMAIL_NOTIFICATION (
    job_name          IN VARCHAR2,
    recipients        IN VARCHAR2 DEFAULT NULL,
    events            IN VARCHAR2 DEFAULT NULL);
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

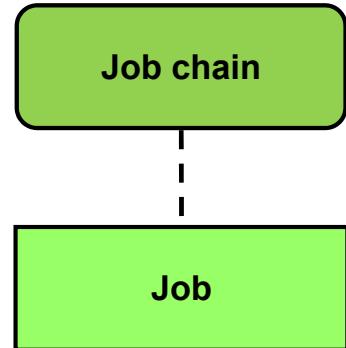
Add one or more job email notifications with the DBMS_SCHEDULER.ADD_JOB_EMAIL_NOTIFICATION procedure. Email messages will be sent to the specified recipient addresses whenever any of the listed events are generated by the job. The job is automatically modified to raise these events. If a filter condition is specified, only events that match the specification in FILTER_CONDITION will generate an email message. This procedure will fail if the EMAIL_SERVER scheduler attribute is not set or if the specified job does not exist. The user calling this procedure must be the owner of the job, have the CREATE ANY JOB system privilege, or have been granted the ALTER privilege for the job.

- The **subject** of notification emails can contain the following variables for which values will be substituted: %job_owner%, %job_name%, %event_type%, %event_timestamp%, %log_id%, %error_code%, %error_message%, %run_count%, %failure_count%, %retry_count%, %job_subname%, %job_class_name%.
- The notification email message **body** can contain any of the variables that are valid in the SUBJECT.
- The comma-separated list of **events** cannot be NULL. Refer to the list of events for the RAISE_EVENTS attribute of JOBS for valid events.
- If **filter_condition** is NULL (the default), all occurrences of the specified events will be emailed to all specified recipient addresses.

Remove one or more email notifications for a specified job with the DBMS_SCHEDULER.REMOVE_JOB_EMAIL_NOTIFICATION procedure.

Creating Job Chains

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 © 2014, Oracle and/or its affiliates. All rights reserved.

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. Typically, after an initial set of chain steps has started, the execution of successive steps depends on the completion of one or more previous steps. 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.

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 inline 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.

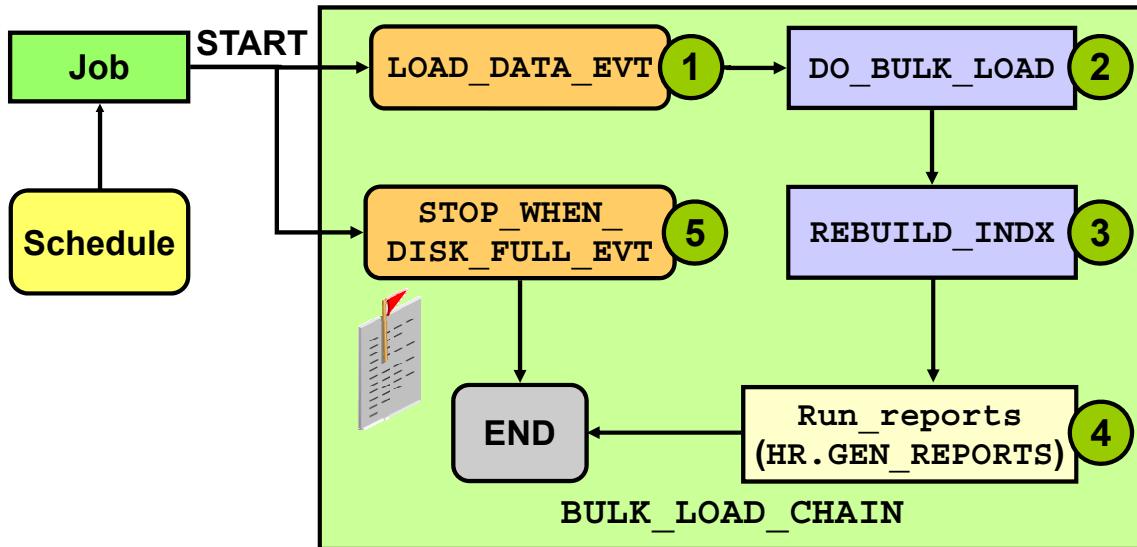
3. After creating the chain object, you **define chain rules**. Chain rules define when steps run, and define dependencies between steps. Each rule has a *condition* and an *action*:
 - If the condition evaluates to `TRUE`, the action is performed. The condition can contain any syntax that is valid in 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

Dependency scheduling

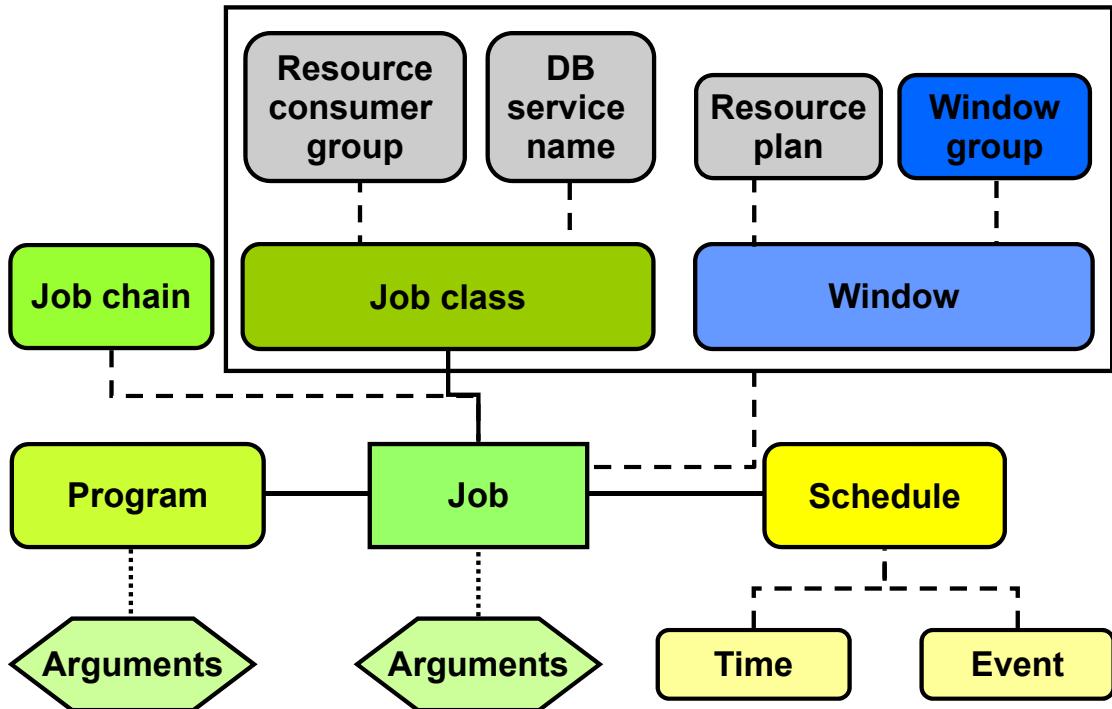


ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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. The slide shows an example of dependency scheduling.

Using Advanced Scheduler Features



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

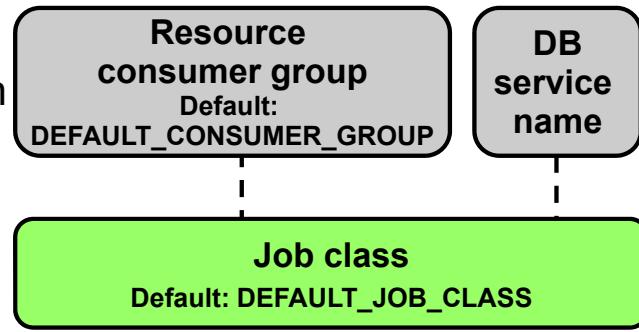
Using advanced Scheduler features, you can exercise more control over various aspects of scheduling, such as job windows and prioritizing jobs. These advanced features are summarized below, and are discussed in detail in the following slides.

- 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 plans 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: The gray objects are not Scheduler objects.

Job Classes

- Assign the same set of attribute values to member jobs
- Are created by the `CREATE_JOB_CLASS` procedure
- Specify jobs in a job class (with the `SET_ATTRIBUTE` procedure)
- Belong to the `SYS` schema
- Set resource allocation for member jobs
- Set the service attribute to a desired database service name
- Group jobs for prioritization



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Job classes are policies for their participating member jobs. Each job class specifies a set of attributes, such as the logging level. When you assign a job to a job class, the job inherits those attributes. For example, you can specify the same policy for purging log entries for all payroll jobs.

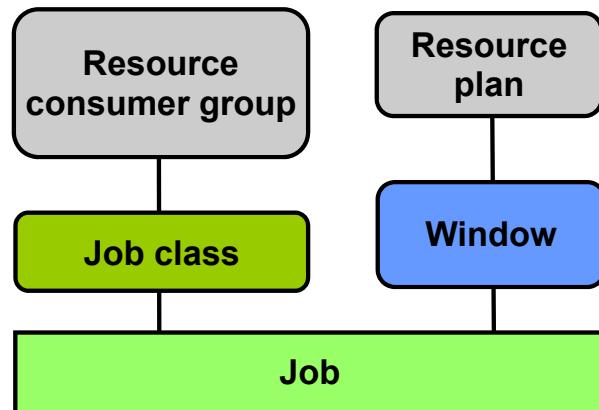
- You can use the `CREATE_JOB_CLASS` procedure to create a job class. A class always belongs to the `sys` schema. To create a class, you must have the `MANAGE SCHEDULER` privilege. There is a default job class named `DEFAULT_JOB_CLASS` that is created with the database.
- After a job class has been created, 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 a job is not associated with a job class, the job belongs to this default job class.
- Set the service attribute of a job class to a desired database service name. This determines the instances in a Real Application Clusters environment that run the member jobs, and optionally, the system resources that are assigned to the member jobs.

- Set resource allocation for member jobs. Job classes provide the link between the Database Resource Manager and the Scheduler because each job class can specify a resource consumer group as an attribute. Member jobs then belong to the specified consumer group and are assigned resources according to settings in the current resource plan. Alternatively, you can leave the `RESOURCE_CONSUMER_GROUP` attribute as `NULL` and set the service attribute of a job class to a desired database service name. That service can in turn be mapped to a resource consumer group. If both the `RESOURCE_CONSUMER_GROUP` and service attributes are set, and the designated service maps to a resource consumer group, the resource consumer group named in the `RESOURCE_CONSUMER_GROUP` attribute takes precedence. 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 may not be allocated enough resources to complete their tasks when the Resource Manager is enabled.
- Group jobs for prioritization. Within the same job class, you can assign priority values of 1–5 to individual jobs so that if two jobs in the class are scheduled to start at the same time, the one with the higher priority takes precedence. This ensures that you do not have a less important job preventing the timely completion of a more important one. If two jobs have the same assigned priority value, the job with the earlier start date takes precedence. If no priority is assigned to a job, its priority defaults to 3.

Windows

Scheduler windows:

- Can start jobs or change resource allocation among jobs for various time periods
- One active at a time
- Created with the `CREATE_WINDOW` procedure



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The priority of jobs can change 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.

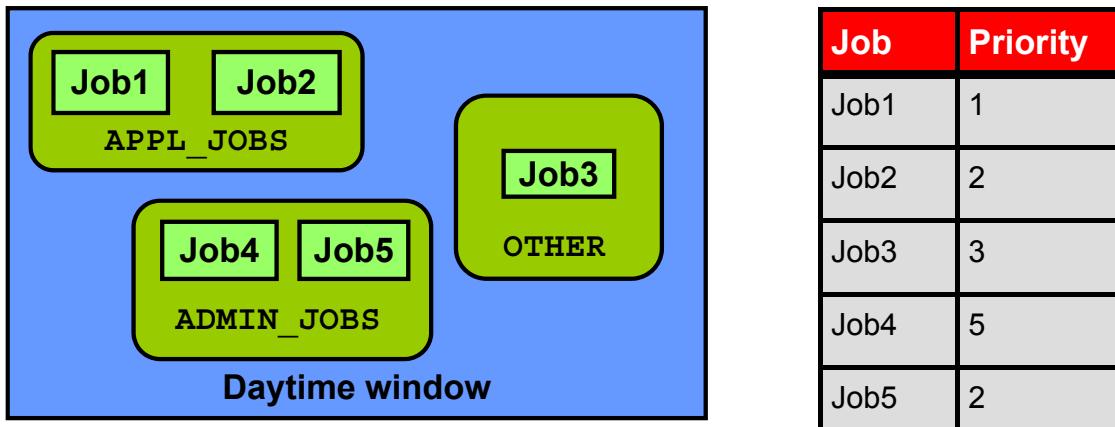
- Scheduler windows can automatically start jobs or change the resource allocation among jobs during various time periods of the day, week, and so on. A window is represented by an interval of time with a well-defined beginning and end, such as “from 12:00 AM to 6:00 AM.”
- Only one window can be in effect at any given time.
- You can create a window with the `CREATE_WINDOW` procedure.

Scheduler windows work with job classes to control resource allocation. Each window specifies the resource plan to activate when the window opens (becomes active), and each job class specifies a resource consumer group or specifies a database service, which can map to a consumer group. A job that runs within a window, therefore, has resources allocated to it according to the consumer group of its job class and the resource plan of the window (as shown in the graphic in this slide).

Prioritizing Jobs Within a Window

Prioritizing jobs:

- At the class level (via resource plans)
- At the job level (with the job priority attribute)
- Not guaranteed for jobs in different job classes



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

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.

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, with 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 resources available, 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.

Creating a Job Array

1. Declare variables of types `sys.job` and `sys.job_array`:

```
DECLARE
    newjob sys.job;
    newjobarr sys.job_array;
```

2. Initialize the job array:

```
BEGIN
    newjobarr := SYS.JOB_ARRAY();
```

3. Size the job array to hold the number of jobs needed:

```
newjobarr.EXTEND(100);
```

(... *continued*)

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

A more efficient way to create a set of jobs is the use of a job array. This also applies to lightweight jobs. In the example in the slide, 100 job specifications are created in a job array and submitted to the job queue in a single transaction. Notice that for a lightweight job, there is a very limited amount of information needed. In the example, the `START_TIME` parameter defaults to `NULL`, so the job is scheduled to start immediately.

1. Declare the variable to hold a job definition and a job array variable.
2. Initialize the job array using the `SYS.JOB_ARRAY` constructor. This creates a place for one job in the array.
3. Set the size of the array to the number of expected jobs.
4. Create each job and place it in the array. In the example in the slide, the only difference is the name of the job. The `START_TIME` variable of the job is omitted and it defaults to `NULL`, indicating that the job will run immediately.
5. Use the `CREATE_JOBS` procedure to submit all the jobs in the array as one transaction.

Note: If the array is very small, the performance will not be significantly better than submitting a single job.

Creating a Job Array

4. Place jobs in the job array:

```
FOR i IN 1..100 LOOP
    newjob := SYS.JOB(job_name => 'LWTJK'||to_char(i),
                      job_style => 'LIGHTWEIGHT',
                      job_template => 'MY_PROG',
                      enabled => TRUE );
    newjobarr(i) := newjob;
END LOOP;
```

5. Submit the job array as one transaction:

```
DBMS_SCHEDULER.CREATE_JOBS(newjobarr,
                            'TRANSACTIONAL');
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The full code of this example is:

```
DECLARE
    newjob sys.job;
    newjobarr sys.job_array;
BEGIN
    -- Create an array of JOB object types
    newjobarr := sys.job_array();
    -- Allocate sufficient space in the array
    newjobarr.extend(100);
    -- Add definitions for jobs
    FOR i IN 1..100 LOOP
        -- Create a JOB object type
        newjob := sys.job(job_name => 'LWTJK' || to_char(i),
                          job_style => 'LIGHTWEIGHT',
                          job_template => 'PROG_1',
                          enabled => TRUE );
        -- Add job to the array
        newjobarr(i) := newjob;
    END LOOP;
    -- Call CREATE_JOBS to create jobs in one transaction
    DBMS_SCHEDULER.CREATE_JOBS(newjobarr, 'TRANSACTIONAL');
END;
/
```

Quiz

Select the statements that are true about the advanced Scheduler features and functionality:

- a. Lightweight jobs can be created with a job array.
- b. Prioritizing jobs at the class level (via resource plans) and at the job level (with the job priority attribute) are mutually exclusive.
- c. Scheduler windows work with job classes to control resource allocation.
- d. Job chains are used to implement “dependency scheduling.”



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: a, c, d

Creating a File Watcher and an Event-Based Job

Perform the following tasks:

1. Create a Scheduler credential object and grant EXECUTE.
2. Create a file watcher and grant EXECUTE.
3. Create a Scheduler program object with a metadata argument that references the event message.
4. Create an event-based job that references the file watcher.
(Optionally, enable the job to run for each instance of the file arrival event.)
5. Enable the file watcher, the program, and the job.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Perform the following tasks to create a file watcher and create the event-based job that starts when the designated file arrives:

1. Create a Scheduler credential object (a credential) with which to authenticate with the host operating system for access to the file and grant EXECUTE on the credential to the schema that owns the event-based job that the file watcher will start.
2. Create a file watcher and grant EXECUTE on the file watcher to any schema that owns an event-based job that references the file watcher. Refer to the *Oracle Database Administrator's Guide* for a detailed example of creating a file watcher.
3. Create a Scheduler program object with a metadata argument that references the event message.
 - Define the metadata argument using the EVENT_MESSAGE attribute.
 - Create the stored procedure with an argument of the SYS.SCHEDULER_FILEWATCHER_RESULT type that the program invokes. The stored procedure must have an argument of the SYS.SCHEDULER_FILEWATCHER_RESULT type, which is the data type of the event message. The position of that argument must match the position of the defined metadata argument. The procedure can access attributes of this abstract data type to learn about the arrived file.

4. Create an event-based job that references the file watcher. You can use the DBMS_SCHEDULER.SET_ATTRIBUTE procedure to enable the job to run for each instance of the file arrival event, even if the job is already processing a previous event. Set the PARALLEL_INSTANCES attribute to TRUE.

```
BEGIN  
  DBMS_SCHEDULER.SET_ATTRIBUTE(' ', 'PARALLEL_INSTANCES', TRUE);  
END;
```

This enables the job to run as a lightweight job so that multiple instances of the job can be started quickly. If PARALLEL_INSTANCES is set to the default value of FALSE, file watcher events that occur while the event-based job is already processing another will be discarded.

5. Enable the file watcher, the program, and the job.

Enabling File Arrival Events from Remote Systems

Perform the following tasks to enable the raising of file arrival events at remote systems:

1. Set up the database to run remote external jobs.
2. Install, configure, register, and start the Scheduler agent on the first remote system.
3. Repeat step 2 for each additional remote system.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To receive file arrival events from a remote system, you must install the Scheduler agent on that system, and you must register the agent with the database. The remote system does not need an Oracle Database instance to generate file arrival events.

Refer to the *Oracle Database Administrator's Guide* for detailed information.

Scheduling Remote Database Jobs

- Create a job that runs stored procedures and anonymous PL/SQL blocks on another database instance on the same host or a remote host.
- The target database can be any release of Oracle Database.
- DBMS_SCHEDULER.CREATE_DATABASE_DESTINATION and DBMS_SCHEDULER.CREATE_CREDENTIAL can be used for remote database jobs.
- Jobs with job types of PLSQL_BLOCK and STORED PROCEDURE can be the subject of SET_ATTRIBUTE calls for the DESTINATION and CREDENTIAL attributes.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can now create a job that runs stored procedures and anonymous PL/SQL blocks on another database instance on the same host or a remote host. The target database can be any release of Oracle Database.

There are no new procedures to support remote database jobs, rather there have been changes to existing DBMS_SCHEDULER procedures to support this functionality. Additional information is provided over the next few pages.

Creating Remote Database Jobs

Perform the following tasks to create a remote job:

1. Set up the originating database for remote jobs.
2. Create the job by using DBMS_SCHEDULER.CREATE_JOB.
3. Create a credential by using DBMS_SCHEDULER.CREATE_CREDENTIAL.
4. Set the job CREDENTIAL_NAME attribute by using DBMS_SCHEDULER.SET_ATTRIBUTE.
5. Set the job DESTINATION attribute by using DBMS_SCHEDULER.SET_ATTRIBUTE.
6. Enable the job by using DBMS_SCHEDULER.ENABLE.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can perform the tasks listed in the slide to create a remote database job.

To set up the originating database for remote jobs, perform the following steps:

1. Verify that XML DB is installed.
2. Enable HTTP connections to the database.

```
BEGIN  
  DBMS_XDB.SETHTTPPORT(port) ;  
END ;
```

3. Execute the `prvtrscl.plb` script.
4. Set a registration password for the Scheduler agents.

```
BEGIN  
  DBMS_SCHEDULER.SET_AGENT_REGISTRATION_PASS('password') ;  
END ;
```

Refer to the *Oracle Database Administrator's Guide* for a detailed example.

Scheduling Multiple Destination Jobs

- This enables you to specify several targets on which your jobs should execute.
- It provides the ability to monitor and control the jobs from the database on which they were created.
- While running, a multiple-destination job is viewed as a collection of jobs, which are near-identical copies of each other.
- All jobs will execute based on the time zone that is specified in the start date of the job or will use the time zone of the source database.

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The multiple destination jobs feature enables you to specify multiple targets on which the jobs should execute. You have the ability to monitor and control the jobs from the database at which you created them. The following capabilities are included:

- Specifying several databases or machines on which a job must execute
- Modifying a job scheduled on multiple targets as a single entity
- Stopping or dropping jobs running on one or more remote targets
- Finding out the status of job instances on all of a job's targets

Note that in the initial release of this feature, all destinations will run based on the time zone that is specified in the start date of the job or will default to the time zone of the source database.

Viewing Scheduler Meta Data

Scheduler management views, displaying:

- * _SCHEDULER_JOBS: All jobs, enabled and disabled
- * _SCHEDULER_SCHEDULES: All schedules
- * _SCHEDULER_PROGRAMS: All programs
- * _SCHEDULER_RUNNING_JOBS: Active job states
- * _SCHEDULER_JOB_LOG: All job state changes
- * _SCHEDULER_JOB_RUN_DETAILS: All completed job runs

```
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

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The job table is a container for all the jobs, with one table per database. The job table stores information for all jobs, such as the owner name or the level of logging. You can find this information in the * _SCHEDULER_JOBS views.

Jobs are database objects, and can, therefore, accumulate and take up too much space. To avoid this, job objects are automatically dropped by default after completion. This behavior is controlled by the AUTO_DROP job attribute.

There are many views available to the DBA and privileged users that provide essential operating information regarding the scheduler, jobs, schedules, windows, and so on. These views include:

- * _SCHEDULER_PROGRAM_ARGS: Shows all arguments defined for all programs as well as the default values if they exist
- * _SCHEDULER_JOBS: Shows all jobs, enabled as well as disabled
- * _SCHEDULER_JOB_RUN_DETAILS show all completed (failed or successful) job runs. It has a row for each job instance. Each row contains information about the job execution for that instance. The ERROR# column shows the number of the first error encountered
- * _SCHEDULER_GLOBAL_ATTRIBUTE: Shows the current values of Scheduler attributes
- * _SCHEDULER_JOB_ARGS: Shows all set argument values for all jobs
- * _SCHEDULER_JOB_CLASSES: Shows all job classes

For job chains:

- *SCHEDULER_RUNNING_CHAINS: Shows all active chains
- *SCHEDULER_CHAIN_STEPS: Shows all steps for all chains
- *SCHEDULER_CHAINS: Shows all chains
- *SCHEDULER_CHAIN_RULES: Shows all rules for all chains

For windows and other advanced objects:

- *SCHEDULER_WINDOWS show all windows.
- *SCHEDULER_WINDOW_GROUPS show all window groups.
- *SCHEDULER_WINGROUP_MEMBERS show the members of all window groups, one row for each group member.
- *SCHEDULER_JOB_LOG shows all state changes made to jobs.
- *SCHEDULER_JOB_ROLES show all jobs by database role.

Lightweight jobs are visible through the same views as regular jobs:

- *SCHEDULER_JOBS: Shows all jobs, including the JOB_STYLE='LIGHTWEIGHT'
- *SCHEDULER_JOB_ARGS: Shows all set argument values also for lightweight jobs

Because lightweight jobs are not database objects, they are not visible through the *_OBJECTS views.

Starting with the Oracle Database 11g Release 2:

- *SCHEDULER_NOTIFICATIONS shows which email notifications have been set.
- *SCHEDULER_FILE_WATCHERS shows file watcher configuration information.

The following views display information about multiple destination jobs:

- *SCHEDULER_DESTS: Shows all the destinations on which remote jobs can be scheduled. The views contain both the external destinations (for remote external jobs) as well as the database destinations for remote database jobs.
- *SCHEDULER_EXTERNAL_DESTS: Shows all the agents that have registered with the database and can be used as a destination for remote external jobs.
- *SCHEDULER_DB_DESTS: Shows all the databases on which you can schedule remote database jobs.
- *SCHEDULER_GROUPS: Shows the groups in your schema or all groups in the database.
- *SCHEDULER_GROUP_MEMBERS: Shows the group members in your schema or all group members in the database.
- *SCHEDULER_JOB_DESTS: Shows the state of a job at a remote database.

Note: In the views listed above, the asterisk at the beginning of a view name can be replaced with DBA, ALL, or USER.

Quiz

Select the statements that are true about the Oracle Scheduler.

- a. Creating remote database jobs is a manual task requiring the use of OS-specific commands.
- b. A Scheduler credential is an object with which to authenticate with the host operating system for file access.
- c. You can specify several targets on which your jobs should execute and monitor them from the database on which they were created.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b, c

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
- Describe the use of windows, window groups, job classes, and consumer groups
- Use email notification
- Use job chains to perform a series of related tasks
- Describe Scheduler jobs on remote systems
- Use advanced Scheduler features to prioritize jobs



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice: Overview

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
- Creating a lightweight job
- Monitoring job runs



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

This practice uses both Enterprise Manager Cloud Control and SQL*Plus.

Working with Oracle Support

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

- Work with My Oracle Support
- Search My Oracle Support
- Log service requests (SRs)
- Manage patches
 - Apply a patch
 - Stage a patch
- Use the Enterprise Manager Cloud Control Support Workbench



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Working with Oracle Support

- Oracle Support Services (OSS) provides 24x7 solution support.
- Support is delivered in the following ways:
 - My Oracle Support website
 - Telephone
 - Oracle Direct Connect (ODC) remote diagnostic tool
- The Customer Support Identifier (CSI) number is used to track the software and support that are licensed to each customer.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle Support Services (OSS) provides 24x7 solution support to all Oracle customers throughout the world. OSS has support centers around the globe to provide this coverage whenever it is required, 365 days a year.

Support is delivered to Oracle customers through the My Oracle Support Web site, on the telephone, and by using the Oracle Direct Connect (ODC) remote diagnostic tool.

After purchasing Oracle software, customers are provided with a Customer Support Identifier (CSI) number. This number is used to track the software and support licensed to each customer. The CSI number provides access to all the available patches, documentation, and troubleshooting information on My Oracle Support. The CSI number enables customers to log a service request (SR) with OSS.

Note: Service requests were formerly called technical assistance requests (TARs).

Using My Oracle Support

The screenshot shows the Oracle My Oracle Support web interface. At the top, there's a navigation bar with links for Dashboard, Knowledge, Service Requests, Patches & Updates, Community, Certifications, On Demand, More..., Search Knowledge Base, Advanced, Welcome, Contact Us, Help, and Sign Out. Below the navigation is a 'Dashboard' section with news items like 'New Patch Nomenclature for Oracle Database, Fusion Middleware, Enterprise Manager, Grid Control and Collaboration Suite Products' and 'My Oracle Support社区已推出中文社区 - Introducing New Chinese Category in My Oracle Support Community'. There's also a 'Getting Started' section with links to User Resource Center, Webcasts and Recordings, Discover the Get Proactive Portfolio, Quick Video Training (with links to My Oracle Support Overview (01:46) and Searching and Browsing (04:38)), and social media links for Follow Us and Follow Oracle. The main content area has sections for 'Knowledge Base', 'Service Requests', and 'Knowledge Articles'. The 'Knowledge Articles' section shows a list of alerts from November 29, 2012, such as 'EI: 75A: 2011/2012 Australia Year-End Processing [1315052.1]' and 'EI: 75A: Australia Medicare Levy Surcharge [1470813.1]'. At the bottom, there's a footer with copyright information: 'Copyright (c) 2012, Oracle. All rights reserved.' and links to Legal Notices and Terms of Use, Privacy Statement, and Oracle support.

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To register for My Oracle Support, go to <http://myoraclesupport.oracle.com/> and click the “New User? Register here” link. At the prompt, enter your CSI number and answer a few basic questions. After registering, you are ready to use My Oracle Support. Note that each CSI number has an administrator designated by the customer who controls new-user access to My Oracle Support. Customers must designate this individual, and then new users must work with this individual to create new accounts and grant appropriate My Oracle Support access.

My Oracle Support has a variety of tools and methods available for researching problems. Searching for answers on My Oracle Support through the standard and advanced search engines is relatively straightforward. A common problem is that too many results are returned. The following are some simple steps that can improve the quality and relevance of search results:

- Use full and exact error text when performing your search. For example, ORA-1400 : mandatory (NOT NULL) column returns more relevant answers than ORA-1400.
- When researching errors in Oracle E-Business Suite, enter the name of the code as part of the search criteria. For example, APXINWKB ORA-1400: mandatory (NOT NULL) column returns fewer and better results than if you supply only the error message.

You can use the Knowledge tab to access the Knowledge Browser if you prefer a drilldown method of searching for information rather than searching by keyword. The Knowledge Browser provides easy-to-use access to OSS's most frequently used technical content.

The Knowledge Browser is organized to provide up-to-date information at your fingertips:

- Recent announcements and information in the *Featured News and Articles* section
- Information by product category
- Case studies
- Tools and training
- Online documentation
- Electronic technical reference manuals (eTRMs)
- Oracle Integration Repository
- Customer Knowledge Exchange

My Oracle Support Forums (Forums) enables you to interact with other Oracle customers to share ideas and discuss Oracle products. You can use My Oracle Support Forums to find out how other customers perform complex tasks or meet various business requirements with Oracle products. You should not use Forums as a substitute for logging an SR.

Customers can use the patch engine to search for patches by using a variety of methods. The following are the most common patch searches:

- **Patch Number:** If you know the patch number, you can enter it.
- **Latest Consolidated Patch:** You can use this when upgrading to determine the latest patches for the products you are using.
- **Includes File:** When a problem is encountered in a specific piece of code, a patch is often available to fix the issue. For this reason, support representatives often recommend that customers apply a patch to update code to the most current version available for the release. You can find and apply the latest versions of Oracle software by identifying the name and version of the code and then using the patch search utility to find out whether a more current version of the code is available.

Note: For detailed information about performing these searches, refer to My Oracle Support Technical Note 166650.1 ("Working Effectively with Global Customer Support").

You can use the BUGs link to search the BUG database when researching issues. A variety of methods are available for searching the BUG database.

Researching an Issue

To research an issue on My Oracle Support, perform the following steps:

1. Perform a keyword search.
2. Review the documentation.
3. Use the self-service toolkits.
4. Use the automated diagnostic tests and business flows.
5. Search for applicable patches.
6. Log a service request (SR).



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

My Oracle Support provides several resources that can be used to research an issue. The following steps outline basic troubleshooting techniques that use My Oracle Support resources:

1. **Keyword search:** Most issues can be resolved quickly and easily by using the keyword search utility on My Oracle Support. Effective searches can provide much information about a specific problem and its solutions.
2. **Documentation:** If keyword searching fails to yield a solution, you should review the documentation to ensure that setup problems are not the root cause. Setup issues account for more than one-third of all service requests; it is always good to review setups early in the troubleshooting process. Documentation consists of user guides and implementation manuals published in PDF format as well as product README files and installation notes published in HTML. Both of these document types are available on My Oracle Support and can be accessed through the self-service toolkits for each product.

3. **Self-service toolkits:** Self-service toolkits (SSTKs) provide a wealth of information about each product. In most cases, they contain FAQs, patch listings, and other helpful information that can assist you in researching and troubleshooting the issues that you are facing. Because SSTKs contain the most frequently used content about each product, you should reference them periodically to identify known issues before they cause problems within your environment.
4. **Diagnostics and flows:** Many recent innovations in Oracle Support Services have been in the area of automated diagnostic tests and business flows. Tests and flows have been created for you to check the setup of your system or gather information about a problem. In the case of diagnostic tests, this can be done by running a Java or SQL script. The output of these tests can help you in resolving issues and can also help Oracle Support Services identify the cause of your problem if it becomes necessary to log a service request.
5. **Patches and BUGs:** There are times when BUGs are found in Oracle products, and patches are required to correct the problem. When troubleshooting a problem, you should review your system to see whether patches are available to provide you with a more recent release of the product. With the patch search tool, you can search for patches that contain specific files. Searching for the latest code and patching your environment to the most recent version improves the troubleshooting process by eliminating existing BUGs that could be possible candidates for the problem. You should also leverage the BUG search engine to see whether a BUG has been logged for your issue but not yet fixed.
6. **Logging a service request (SR):** When all self-service options fail, it may become necessary to engage a support representative to assist in resolving your issue.

Logging Service Requests

- Log an SR by clicking the Service Request tab on the My Oracle Support home page.
- My Oracle Support performs searches based on the CSI number and SR profile.
- Provide the following information when logging an SR:
 - Explanation of the issue, including error messages
 - Steps taken to troubleshoot the issue
 - Software version
 - Steps required to reproduce the problem
 - Business impact of the issue



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You may research an issue on My Oracle Support, but may be unable to locate a solution. In this case, you should log a service request (SR) through My Oracle Support. You can log an SR by clicking the Service Request tab on the My Oracle Support home page.

The first step in creating an SR is the selection of a CSI number and SR profile. After the required profile information has been submitted, My Oracle Support gathers some specifics about the problem, including the problem type, error message, brief summary of the issue, and language preference. My Oracle Support performs a search by using this information and attempts to find a solution.

The search conducted during this phase may provide different results than the searches you have performed earlier. Both searches retrieve notes and BUGs from the same database; however, the search engines and weighting are slightly different. Because the search results can differ, it is important that the search results are reviewed during the SR creation process, even if previous searches have been conducted by using the My Oracle Support search engine.

If the search results fail to resolve the issue, the SR creation process continues with a series of questions and requests for information. After the questions are answered, the SR is submitted electronically and routed to a support representative who analyzes the issue further. Any files, screenshots, or other additional information must be uploaded immediately after the SR is logged by using the upload utility provided in the SR section of My Oracle Support.

You must ensure that the following items are clearly documented in the SR. By providing the following information, you can equip the support representative effectively to prioritize and work on the issue:

- Clear explanation of the problem, including exact error messages
- Explanation of the steps taken to troubleshoot the problem and the findings
- Exact versions of the software
- Steps required to reproduce the problem
- Business impact of this issue, including milestones, dates, and costs

Each SR is assigned a unique identifier called an *SR number*. When you log an SR, My Oracle Support provides you with the SR number (or your support representative advises you about the SR number if you log the SR by telephone). The support representative subsequently receives the SR in his or her queue through an automated allocation process that Oracle Support Services uses to distribute all phone- and web-sourced service requests. This automated process ensures that all SRs are assigned to the support representative who is best able to work on the specific issue that is being reported.

Note: For more information, refer to My Oracle Support Technical Note 166650.1 (“Working Effectively with Global Customer Support”).

Accessing My Oracle Support Community

The screenshot shows the Oracle My Oracle Support Community interface. At the top, there's a navigation bar with links for Main Home, Discussions, Documents, Private Messages (0), Contacts, Tags, Profile, Subscriptions Off, Welcome, User496924 - Oracle | My Profile | Sign out | Help, Search..., This Community, and Go.

The main content area has several sections:

- My Communities:** A sidebar with a "Quick find" search bar and a list of communities including Oracle Weblogic Server, Patch Reviews - Middleware, SQL*Plus, WebCenter Interaction, Oracle Commerce, ATG Oracle Live Help, ATG Web Commerce, Endeca, Other ATG Applications, Oracle Database, DB Diagnostic Repository (ADR), Database - RAC/Scalability, Database Administration, Database Backup and Recover, Database DataWarehousing, and a "Subscribe to this Community" button.
- Top Participants:** A list of active users in the last 90 days:
 - MarkDPowell (Guru, 3003 points / 7343 total)
 - BobB (Guru, 2679 points / 9341 total)
 - VenkataB (Guru, 1713 points / 5935 total)
 - BPeasland (Guru, 1565 points / 7426 total)
- Database Administration News & Announcements:** A message about PSU and CPU OCT 2012 being ready for download.
- Database Administration Webcasts:**
 - Upcoming Webcasts:** No live webcasts currently scheduled.
 - Archived Webcasts:** A list of recorded webcasts including "An Introduction to Concurrency Control", "Introduction to DMU (Database Migration Assistant for Unicode)", "Database Transaction Recovery", "Getting a Testcase for Support - The Way Without Pain", "Reorganizing data in tables and tablespaces - Why? When? How?", "Getting a Testcase for Support - The Way Without Pain", "Client side NLS: NLS_LANG and other client NLS parameters explained", "Oracle JVM 11g New Features", "What is an ORA-600/ORA-7445 Error and What Should I do About it?", "This one-hour session is recommended for technical users who want to have a introduction overview on what steps are needed to change the character set of an Oracle RDBMS. Intended audience: Oracle DBA's.", "This one-hour session will present the features and functionality of Oracle Connection Manager.", and "Look here for Advisor Webcast Archive across all products".
 - Database Administration Featured Discussions:** Links to "Database Administration All Community Discussions", "Database Administration All Community Documents", "Database Administration Tips and Tricks", "Database Administration Best Practices", "Database Administration Product Specific", and "Database Administration White Papers".
- Popular Discussions:** A list of popular discussions with their titles and view counts:
 - Welcome to YOUR Database Admin (1911 views)
 - Tips to effectively use YOUR DB (1128 views)
 - Alter table add column taking a long time (803 views)
 - Tracing ORA-07445 (696 views)
 - CLONE PROCESS (668 views)
 - I can't find my posting in the Database (640 views)
 - how to make DB auto startup at boot (566 views)
 - RDA zip file issue, download from (537 views)
 - ORA-809 : oipdr aborting process (509 views)
 - Init Parameters and SPFILE (491 views)
- Popular Documents:** A list of popular documents with their titles and view counts:
 - Script to identify Long Running (7916 views)
 - Tablespace space forecasting (4169 views)
 - A Case Study on Oracle Statistics (2690 views)
 - A couple of scripts for finding a (834 views)
 - UNDERSTANDING ORACLE LOGGING (535 views)
 - A CHECKLIST TO MOVE THE DATA (487 views)

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

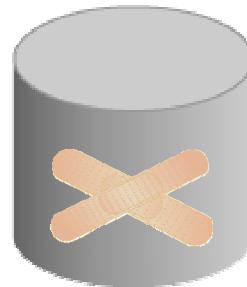
My Oracle Support Community is a multi-channel interactive community for sharing information, posting questions and answers, and providing suggestions about Oracle products, services, and related technologies.

If you have access to My Oracle Support you are automatically a member. There is no additional registration required.

Managing Patches

Kinds of patches

- Interim patches
 - For specific issues
 - No regression testing
- Security Patch Updates (SPUs)
 - Critical security issues
 - Regression testing
 - Does not advance version number
- Patch Set Updates (PSUs)
- Patch releases



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can apply different kinds of patches at different times for different reasons.

- Interim patches (also known as *one-off* or *one-of patches*) are created to solve a specific problem. They do not go through a full regression test. Interim patches are typically installed with the `opatch` utility. The Enterprise Manager Patching Wizard can help automate the patching process by downloading, applying, and staging the patches. This wizard uses the `opatch` utility in the background.
- Security Patch Updates (SPU) patches—formally, Critical Patch Update (CPU) patches—are cumulative, which means fixes from previous Oracle security alerts and critical patch updates are included.
- Patch Set Update (PSU) patches include Security Patch Updates (SPU) and dependent non-security patches. PSU patches are cumulative. It is not required to have previous security patches applied before applying PSU patches. However, you must be on the stated patch set level. PSU patches are for a specific patch release level (such as 10.2.0.3). PSU patches are installed with the `opatch` utility or through EM Patching Wizard. PSU patches are issued quarterly. PSU patches and interim patches can also be removed from your system with `opatch rollback -id <patch id>`. Oracle does extensive testing of Patch Set Updates with its own applications, as well as running regression tests for the PSUs themselves. To verify that a patch has been applied, query the inventory with `opatch -lsinventory` and see if the patch is listed.

Applying a Patch Release

- Patch releases are fully tested product fixes that:
 - Affect only the software residing in your Oracle home on installation
 - Contain individual bug fixes
 - Carry version numbers
- To apply a patch:
 1. Determine your Oracle software environment.
 2. Set your My Oracle Support login credentials.
 3. Stage the patch release.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Software management involves keeping your Oracle software up-to-date with the latest product fixes. Periodically, Oracle issues patch releases (product fixes) for its software. Patch releases are fully tested product fixes. Application of a patch release affects only the software residing in your Oracle home, with no upgrade or change to the database.

Patches are individual bug fixes. Patch sets are a collection of bug fixes up to the time of the patch set release. All patch and patch set releases carry version numbers. For example, 11.1.0.3 is an available patch set for Oracle Database 11g Release 11.1.0.2. Every patch or patch set also has a patch number to identify it. Every patch release has an associated README file that describes its bug fixes. The README also has instructions for manually applying the patch.

Enterprise Manager enables you to find the latest patch release on the My Oracle Support website and download it to your Oracle home.

Enterprise Manager Cloud Control: My Oracle Support Integration

- Enterprise Manager Cloud Control automatically alerts users to new critical patches.
- The Enterprise Manager Cloud Control patch wizard can be used to select an interim patch.
- You can review the patch's README file from within Enterprise Manager Cloud Control.
- You can download the selected patches from My Oracle Support into the Enterprise Manager Cloud Control patch cache.

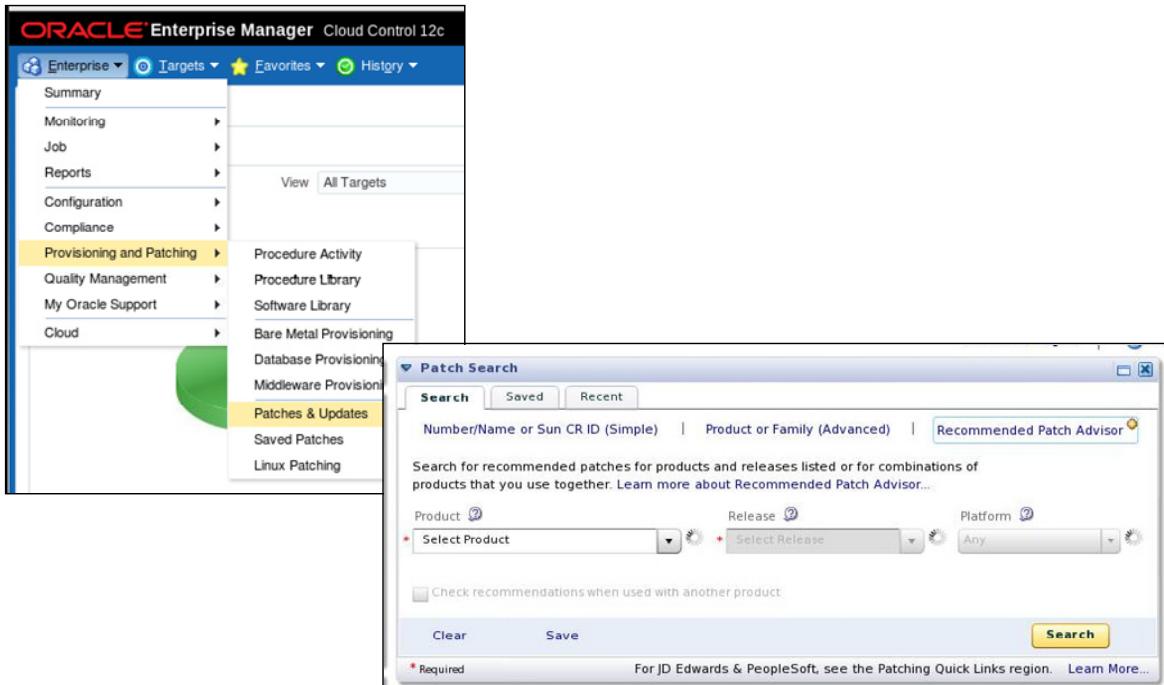


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle Enterprise Manager Cloud Control significantly facilitates software patching with its built-in My Oracle Support integration. Enterprise Manager Cloud Control automatically alerts users to new critical patches and flags all systems that require a specific patch. You can invoke the Enterprise Manager Cloud Control patch wizard to determine what interim patches are available for installation. Alternatively, you can use the patch wizard to select an interim patch and determine whether any of your systems require that patch. You can review the patch details and README patch notes directly from within Enterprise Manager Cloud Control.

You can use the patch wizard to download interim patches from My Oracle Support into the Enterprise Manager Cloud Control patch cache, eliminating the need for repeated downloads. You can stage appropriate patches on the destination system or systems for manual application at a later time. To further automate the patching process, you can also provide a customizable patch application script that is executed on the destination system at a user-defined time by the resident Enterprise Manager Cloud Control agents. As patches are applied to a system, the corresponding Oracle Universal Installer (OUI) inventory is automatically updated to keep track of the systems' correct patch level.

Using the Patch Advisor



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In Enterprise Manager Cloud Control, you can access the Patches & Updates page by expanding Enterprise and selecting Patches & Updates in the Provisioning and Patching menu. Note that this menu is at the Enterprise, not database target level.

In the Patch Search section, you can search for patches by patch number and name, you can search for all patches for a specific product, and you can invoke the Patch Advisor. The Patch Advisor enables you to search for recommended patches for specified products and combinations of products.

Refer to the Enterprise Manager Cloud Control documentation and online help for additional information. You may also want to attend the *Using Oracle Enterprise Manager Cloud Control 12c* course for detailed information about using Enterprise Manager Cloud Control.

Online Patching: Overview

For a bug fix or diagnostic patch on a running Oracle instance, online patching provides the ability to do the following:

- Install
- Enable
- Disable



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Online patching provides the ability to install, enable, and disable a bug fix or diagnostic patch on a live, running Oracle instance. Using online patching is the recommended solution for avoiding down time when applying online patches. Oracle provides the capability to perform online patching with any Oracle database using the opatch command-line utility. Online patches can be provided when the changed code is small in scope and complexity (for example, with diagnostic patches or small bug fixes).

Installing an Online Patch

- Applying an online patch does not require instance shutdown, relinking of the Oracle binary, or instance restart.
- OPatch can be used to install or uninstall an online patch.
- OPatch detects conflicts between two online patches, as well as between an online patch and a conventional patch.
- To determine if a patch is an online patch:

```
opatch query -is_online_patch <patch location>
OR
opatch query <patch location> -all
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Unlike traditional patching mechanisms, applying an online patch does not require instance shutdown or restart.

Similar to traditional patching, you can use OPatch to install an online patch.

You can determine whether a patch is an online patch by using the following commands:

```
opatch query -is_online_patch <patch location> or
opatch query <patch location> -all
```

Note: The patched code is shipped as a dynamic/shared library, which is then mapped to memory by each Oracle process.

Benefits of Online Patching

- No down time and no interruption of business
- Extremely fast installation and uninstallation times
- Integration with OPatch:
 - Conflict detection
 - Listed in patch inventory
 - Works in RAC environment
- Persistence across instance shutdown and startup



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You do not have to shut down your database instance while you apply the online patch. Unlike conventional patching, online patching enables fast installation and uninstallation. Because online patching uses OPatch, you get all the benefits that you already have with conventional patching that uses OPatch. It does not matter how long or how many times you shut down your database—an online patch always persists across instance shutdown and startup.

Conventional Patching and Online Patching

Conventional Patches	Online Patches
Require down time to apply or remove	Do not require down time to apply or remove
Are installed and uninstalled via OPatch	Are installed and uninstalled via OPatch
Persist across instance startup and shutdown	Persist across instance startup and shutdown
Take several minutes to install or uninstall	Take only a few seconds to install or uninstall



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Conventional patching basically requires a shutdown of your database instance.

Online patching does not require any down time. Applications can keep running while you install an online patch. Similarly, online patches that have been installed can be uninstalled with no down time.

Online Patching Considerations

- Some extra memory is consumed.
 - Exact amount depends on the size of the patch and number of concurrently running Oracle processes
 - Minimum amount of memory: Approximately one OS page per running Oracle process
- There may be a small delay (a few seconds) before every Oracle process installs or uninstalls an online patch.
- Not all bug fixes and diagnostic patches are available as an online patch.
- Use online patches in situations when down time is not feasible.
- When down time is possible, you should install all relevant bug fixes as conventional patches.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

One operating system (OS) page is typically 4 KB on Linux x86 and 8 KB on Solaris SPARC64. With an average of approximately one thousand Oracle processes running at the same time, this represents around 4 MB of extra memory for a small online patch.

A vast majority of diagnostic patches are available as online patches. For bug fixes, it really depends on their nature. Not every bug fix or diagnostic patch is available as an online patch. But the long-term goal of the online-patching facility is to provide online-patching capabilities for Critical Patch Updates.

Note: You must uninstall the online patch before applying the conventional patch.

Quiz

Which of the following statements are true about online patches?

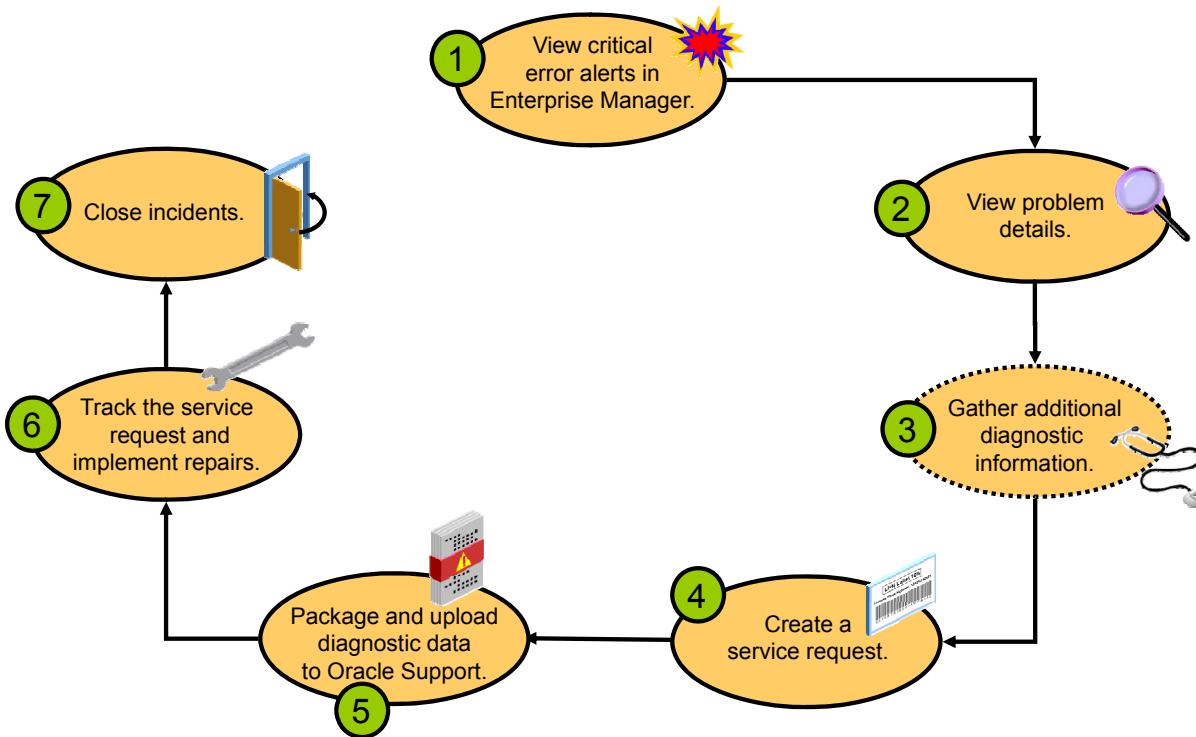
- a. They can be installed by using OPatch.
- b. They require down time to apply.
- c. They persist across instance startup and shutdown.
- d. They do not require down time to remove.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: a, c, d

Using the Support Workbench



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Using the Enterprise Manager Cloud Control Support Workbench, you can investigate, report, and (in some cases) resolve a problem by performing the following general steps:

1. On the Database Home page in Enterprise Manager, review critical error alerts. View the details by selecting an alert.
2. Examine the problem details and view a list of all incidents that were recorded for the problem. Display findings from any health checks that were automatically run.
3. (Optional) Run additional health checks and invoke the SQL Test Case Builder, which gathers all required data related to a SQL problem and packages the information in a way that enables the problem to be reproduced at Oracle Support.
4. Create a service request with My Oracle Support and (optionally) record the service request number with the problem information.
5. Invoke the Incident Packaging Service, which packages all gathered diagnostic data for a problem and (optionally) uploads the data to Oracle Support. You can edit the data to remove sensitive information before uploading.
6. You can maintain an activity log for the service request in the Support Workbench. Run Oracle advisors to help repair SQL failures or corrupted data.
7. Set the status for one, some, or all incidents for the problem to be closed.

Accessing the Support Workbench

The screenshot shows the Oracle Support Workbench interface. At the top, it displays 'Database Instance: orcl > Support Workbench'. Below this, there are tabs for 'Problems (2)', 'Checker Findings (0)', and 'Packages (0)'. The 'Problems' tab is selected. A summary table shows 'New Problems in Last 24 Hours' (0), 'All Active Problems' (1), 'All Problems' (2), 'New Incidents in Last 24 Hours' (2), 'All Active Incidents' (2), and 'All Incidents' (12). A search bar at the top right includes 'Filter by problem key', 'Go', and 'Advanced Search' buttons.

Below the summary, there are buttons for 'View' (selected) and 'Last 24 Hours'. Underneath are buttons for 'View' and 'Package'. A toolbar below these buttons includes 'Select All', 'Select None', 'Show All Details', and 'Hide All Details' buttons. A table follows, with columns for 'Select', 'Details', 'ID', 'Description', 'Number Of Incidents', 'Last Incident', 'Last Comment', 'Active', 'Packaged', and 'SR#'. One row is visible: ID 2, Description ORA-4036, Number Of Incidents 11, Last Incident November 28, 2012 1:10:54 PM UTC, Last Comment Yes, Active No, Packaged No, SR# blank.

A section titled 'Incidents (11)' lists several entries, each with an ID and description. A yellow callout box points to the first entry: '19538 ORA-4036'. The callout box contains the text 'Click to access the Problem Details page.' Below this section is a note: 'There are more incidents...'. To the right of the incident list is a column of dates: November 28, 2012 1:10:54 PM UTC, November 28, 2012 1:10:48 PM UTC, November 26, 2012 10:12:40 AM UTC, November 26, 2012 10:12:40 AM UTC, and November 21, 2012 12:52:35 PM UTC.

At the bottom of the page, there are 'Related Links' including 'Advisor Central' and 'Incident Packaging Configuration', 'Alert Log Contents', and 'Create User-Reported Problem'.

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can access the Support Workbench page directly by expanding the Oracle Database menu, and then selecting Support Workbench in the Diagnostics submenu.

Note: All the tasks described in this section are performed in Enterprise Manager Cloud Control. You can also accomplish all these tasks with the ADRCI command-line utility. See the *Oracle Database Utilities* guide for more information about the ADRCI utility.

Viewing Problem Details

The screenshot shows the 'Problem Details' page for problem ID ORA 4036. At the top, it displays the database instance (orcl), support workbench, and problem details. The page is refreshed on November 29, 2012, at 1:30:18 PM UTC. The main section is titled 'Summary' and includes fields for SR#, Bug#, Problem Id (2), Number of Incidents (11), First Incident (November 8, 2012, 10:11:54 AM UTC), Active (No), and Packaged (No). Below this is the 'Last Dumped Incident' section, which shows a timestamp of November 28, 2012, 1:10:54 PM UTC, an incident source of System Generated, and user impact information. The 'Incidents' subpage is active, showing a table with columns: Select, Details, ID, Description, Data Dumped, Active, Status, and Timestamp. A single row is selected, showing ID 19538 and Description ORA-4036. A yellow callout box points to the 'Details' link in the table, with the text 'Click to access the Incident Details page.' To the right, the 'Investigate and Resolve' section is visible, containing links for Quick Package, Assess Damage, Collect and Send Diagnostic Data, Run Checkers, Package the Problem, Database Instance Health, Track and Close, Diagnose, Alert Log, Related Problems Across Topology, Diagnostics for Last Dumped Incident, and Search Knowledge Base.

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

On the Problems subpage on the Support Workbench page, click the ID of the problem that you want to investigate. This takes you to the corresponding Problem Details page.

On this page, you can see all incidents that are related to your problem. You can associate your problem with a My Oracle Support service request and bug number. In the "Investigate and Resolve" section of the page, you see a Self Service subpage that has direct links to the operations that you can perform for this problem. In the same section, the Oracle Support subpage has direct links to My Oracle Support.

The Activity Log subpage shows you the system-generated operations that have occurred on your problem so far. This subpage enables you to add your own comments while investigating the problem.

On the Incidents subpage, you can click a related incident ID to access the corresponding Incident Details page.

Viewing Incident Details

Incident Details: 19538

Page Refreshed November 29, 2012 1:41:06 PM UTC [Refresh](#)

Summary

Problem Key	ORA 4036	Data Dumped	Yes
Problem Id	2	ECID	Unknown
Status	Ready	Error	ORA 4036
Active	No	Correlation Keys	SID = 282.255, ProcId = 41.32
Timestamp	November 28, 2012 1:10:54 PM UTC	ECID	004nsqrDbD6DWbI_4tYBUI0002lb000K8R.662, PQ = (0, 1354108248)
User Impact		Client ProcId	= oracle@EDRSR11P1 (TNS V1-V3).17543_139781106417216
Source	System Generated	Purge Date	December 28, 2012 1:10:54 PM UTC (Purging Enabled)
Disable Purging			

Dump Files [Checker Findings](#) [Additional Diagnostics](#)

File Name	Size (MB)	Timestamp	Path	View Contents
orcl ora_17543_i19538.trc	6.25	November 28, 2012 1:10:59 PM UTC	/u01/app/oracle/diag/rdbms/orcl/orcl/incident/incdir_19538	
orcl ora_17543.trc	0.03	November 28, 2012 1:11:01 PM UTC	/u01/app/oracle/diag/rdbms/orcl/orcl/trace	

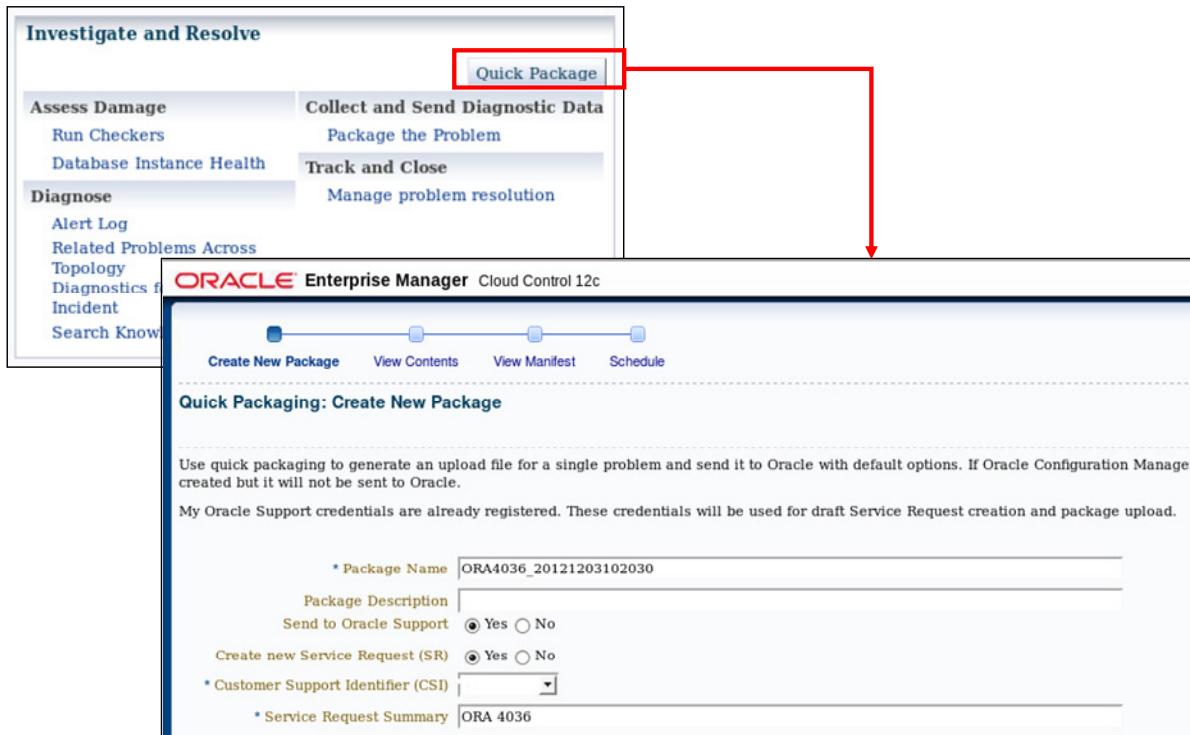


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

When you access the Incident Details page, the Dump Files subpage lists all corresponding dump files. You can then click the eyeglass icon for a particular dump file to visualize the file content with its various sections.

On the Incident Details page, click Checker Findings to view the Checker Findings subpage. This page displays findings from any health checks that were automatically run when the critical error was detected. You will usually have the opportunity to select one or more findings and invoke an advisor to fix the issue.

Packaging and Uploading Diagnostic Data to Oracle Support



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Support Workbench provides two methods for creating and uploading an incident package: the Quick Packaging method and the Advanced Packaging method. The example in the slide shows how to use Quick Packaging.

Quick Packaging is a more automated method with minimum steps. You select a single problem, provide an incident package name and description, and then schedule the incident package upload, either immediately or at a specified date and time. The Support Workbench automatically places diagnostic data related to the problem into the incident package, finalizes the incident package, creates the ZIP file, and then uploads the file. With this method, you do not have the opportunity to add, edit, or remove incident package files or add other diagnostic data such as SQL test cases.

If you have not registered with My Oracle Support, a link will appear so that you can register. You must register with My Oracle Support to send the package to Oracle Support.

Tracking the Service Request and Implementing Repairs

The screenshot shows the Oracle Support interface. At the top, a navigation bar includes 'Investigate and Resolve' (selected), 'Quick Package', 'Assess Damage', 'Collect and Send Diagnostic Data', 'Run Checkers', 'Package the Problem', 'Database Instance Health', 'Track and Close', 'Diagnose', 'Manage problem resolution' (highlighted with a red box and arrow), 'Alert Log', 'Related Problems Across Topology', and 'Diagnostics for Last Dumped'. Below this is the 'Incident Manager' page for 'Problem: ORA 4036'. It displays 'Problem Details' (ID: 2942, Problem Key: ORA 4036, Target: orcl (Database Instance)), 'Tracking' (Escalated: No, Priority: None, Status: New, Owner: -, Acknowledged: No), and 'Guided Resolution' (Diagnostics: Support Workbench: Problem Details, Actions: Support Workbench: Package Diagnostic). The page is refreshed on Dec 3, 2012 at 10:28:23 AM UTC.

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

After uploading diagnostic information to Oracle Support, you can perform various activities to track the service request and implement repairs. Among these activities are the following:

- Add an Oracle bug number to the problem information. On the Problem Details page, click the Edit button that is adjacent to the Bug# label. This is for your reference only.
- Add comments to the problem activity log:
 1. Access the Problem Details page for the problem.
 2. Click Activity Log to display the Activity Log subpage.
 3. In the Comment field, enter a comment, and then click Add Comment. Your comment is recorded in the activity log.
- Respond to a request by Oracle Support to provide additional diagnostics. Your Oracle Support representative can provide instructions for gathering and uploading additional diagnostics.

Summary

In this lesson, you should have learned how to:

- Use the Support Workbench
- Work with Oracle Support
- Search My Oracle Support
- Log service requests
- Manage patches
 - Apply a patch release
 - Stage a patch release



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

