Oracle® Database

2 Day + Performance Tuning Guide 11*g* Release 1 (11.1) **B28275-01**

July 2007



Oracle Database 2 Day + Performance Tuning Guide, 11g Release 1 (11.1)

B28275-01

Copyright © 2007, Oracle. All rights reserved.

Primary Authors: Immanuel Chan, Lance Ashdown

Contributing Author: Sushil Kumar

Contributors: Pete Belknap, Supiti Buranawatanachoke, Nancy Chen, Kakali Das, Karl Dias, Mike Feng, Yong Feng, Cecilia Grant, Connie Green, William Hodak, Andrew Holdsworth, Sue K. Lee, Herve Lejeune, Colin McGregor, Mughees Minhas, Valarie Moore, Deborah Owens, Mark Ramacher, Uri Shaft, Susan Shepard, Janet Stern, Hsiao-Te Su, Minde Sun, Mark Townsend, Stephen Wexler, Graham Wood, Khaled Yagoub, Michael Zampiceni

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

Contents

	eface	ix
	Audience	ix
	Documentation Accessibility	ix
	Related Documents	x
	Conventions	x
Wł	hat's New in Oracle Performance?	xi
Ра	art I Getting Started	
1	Introduction	
	About This Guide	1-1
	Common Oracle DBA Tasks	1-1
	Tools for Tuning the Database	1-2
2		
	Oracle Database Performance Method	
	Oracle Database Performance Method Gathering Database Statistics Using the Automatic Workload Repository	2-1
	Gathering Database Statistics Using the Automatic Workload Repository	2-2
	Gathering Database Statistics Using the Automatic Workload Repository Time Model Statistics	2-2 2-3
	Gathering Database Statistics Using the Automatic Workload Repository Time Model Statistics Wait Event Statistics	2-2 2-3 2-4
	Gathering Database Statistics Using the Automatic Workload Repository Time Model Statistics Wait Event Statistics Session and System Statistics	
	Gathering Database Statistics Using the Automatic Workload Repository Time Model Statistics Wait Event Statistics Session and System Statistics Active Session History Statistics	
	Gathering Database Statistics Using the Automatic Workload Repository Time Model Statistics Wait Event Statistics Session and System Statistics Active Session History Statistics High-Load SQL Statistics	
	Gathering Database Statistics Using the Automatic Workload Repository Time Model Statistics Wait Event Statistics Session and System Statistics Active Session History Statistics High-Load SQL Statistics Using the Oracle Performance Method	2-2 2-3 2-4 2-4 2-4 2-4 2-5 2-5
	Gathering Database Statistics Using the Automatic Workload Repository Time Model Statistics Wait Event Statistics Session and System Statistics Active Session History Statistics High-Load SQL Statistics Using the Oracle Performance Method Preparing the Database for Tuning	2-2 2-3 2-4 2-4 2-4 2-4 2-5 2-5 2-5 2-6
	Gathering Database Statistics Using the Automatic Workload Repository Time Model Statistics Wait Event Statistics Session and System Statistics Active Session History Statistics High-Load SQL Statistics Using the Oracle Performance Method Preparing the Database for Tuning Tuning the Database Proactively	2-2 2-3 2-4 2-4 2-4 2-4 2-4 2-5 2-5 2-5 2-5 2-6

Part II Proactive Database Tuning

3	Automatic Database	Performance	Monitoring
---	--------------------	-------------	------------

Overview of Automatic Database Diagnostic Monitor	3-1
ADDM Analysis	3-2

ADDM Recommendations	3-2
ADDM for Oracle Real Application Clusters	3-3
Configuring Automatic Database Diagnostic Monitor	3-3
Setting Initialization Parameters to Enable ADDM	3-3
Setting the DBIO_EXPECTED Parameter	3-4
Managing AWR Snapshots	3-4
Creating Snapshots	3-5
Modifying Snapshot Settings	3-5
Reviewing the Automatic Database Diagnostic Monitor Analysis	3-7
Interpretation of Automatic Database Diagnostic Monitor Findings	3-8
Implementing Automatic Database Diagnostic Monitor Recommendations	3-9
Viewing Snapshot Statistics	3-12

4 Monitoring Real-Time Database Performance

Monitoring User Activity	4-2
Monitoring Top SQL	4-4
Monitoring Top Sessions	4-5
Monitoring Top Services	4-6
Monitoring Top Modules	4-7
Monitoring Top Actions	4-7
Monitoring Top Clients	4-8
Monitoring Top PL/SQL	4-9
Monitoring Top Files	4-9
Monitoring Top Objects	4-10
Monitoring Instance Activity	4-10
Monitoring Throughput	4-11
Monitoring I/O	4-12
Monitoring I/O by Function	4-13
Monitoring I/O by Type	4-14
Monitoring I/O by Consumer Group	4-15
Monitoring Parallel Execution	4-16
Monitoring Services	4-16
Monitoring Host Activity	4-17
Monitoring CPU Utilization	4-19
Monitoring Memory Utilization	4-22
Monitoring Disk I/O Utilization	4-25
Customizing the Database Performance Page	4-27

5 Monitoring Performance Alerts

Setting Metric Thresholds for Performance Alerts	5-1
Responding to Alerts	5-2
Clearing Alerts	5-2

Part III Reactive Database Tuning

6 Manual Database Performance Monitoring

Manually Running ADDM to Analyze Current Database Performance	6-1
Manually Running ADDM to Analyze Historical Database Performance	6-3
Accessing Previous ADDM Results	6-4

7 Resolving Transient Performance Problems

7-1
7-2
7-3
7-3
7-4
7-4
7-4
7-5
7-5
7-6
7-6
7-7

8 Resolving Performance Degradation Over Time

Managing Baselines	8-1
Creating a Baseline	8-2
Creating a Single Baseline	8-2
Creating a Repeating Baseline	8-4
Deleting a Baseline	8-5
Computing Threshold Statistics for Baselines	
Setting Metric Thresholds for Baselines	8-7
Setting Metric Thresholds for the Default Moving Baseline	8-7
Setting Metric Thresholds for Selected Baselines	8-8
Running the AWR Compare Periods Reports	8-9
Comparing a Baseline to Another Baseline or Pair of Snapshots	8-10
Comparing Two Pairs of Snapshots	8-12
Using the AWR Compare Periods Reports	8-15
Summary of the AWR Compare Periods Report	8-16
Snapshot Sets	8-16
Host Configuration Comparison	8-17
System Configuration Comparison	8-17
Load Profile	8-17
Top Timed Events	8-17
Details of the AWR Compare Periods Report	8-18
Supplemental Information in the AWR Compare Periods Report	8-18

Part IV SQL Tuning

9 Identifying High-Load SQL Statements

Identification of High-Load SQL Statements Using ADDM Findings	9-1
Identifying High-Load SQL Statements Using Top SQL	9-2
Viewing SQL Statements by Wait Class	9-3
Viewing Details of SQL Statements	9-4
Viewing SQL Statistics	9-5
Viewing Session Activity	9-6
Viewing the SQL Execution Plan	9-7
Viewing the SQL Tuning Information	9-8

10 Tuning SQL Statements

Tuning SQL Statements Using SQL Tuning Advisor	10-2
Tuning SQL Manually Using SQL Tuning Advisor	10-2
Viewing Automatic SQL Tuning Results	10-5
Managing SQL Tuning Sets	10-8
Creating a SQL Tuning Set	10-8
Creating a SQL Tuning Set: Options	10-9
Creating a SQL Tuning Set: Load Method	10-10
Creating a SQL Tuning Set: Filter Options	10-13
Creating a SQL Tuning Set: Schedule	10-15
Dropping a SQL Tuning Set	10-16
Transporting SQL Tuning Sets	10-16
Exporting a SQL Tuning Set	10-16
Importing a SQL Tuning Set	10-18
Managing SQL Profiles	10-19
Managing SQL Execution Plans	10-20

11 Optimizing Data Access Paths

Running SQL Access Advisor	11-1
Running SQL Access Advisor: Initial Options	11-2
Running SQL Access Advisor: Workload Source	11-3
Using SQL Statements from the Cache	11-3
Using an Existing SQL Tuning Set	11-4
Using a Hypothetical Workload	11-4
Running SQL Access Advisor: Filter Options	11-5
Defining Filters for Resource Consumption	11-5
Defining Filters for Users	
Defining Filters for Tables	11-6
Defining Filters for SQL Text	11-6
Defining Filters for Modules	11-7
Defining Filters for Actions	11-7
Running SQL Access Advisor: Recommendation Options	11-7
Running SQL Access Advisor: Schedule	11-9
Reviewing the SQL Access Advisor Recommendations	11-13
Reviewing the SQL Access Advisor Recommendations: Summary	11-14
Reviewing the SQL Access Advisor Recommendations: Recommendations	11-15

Reviewing the SQL Access Advisor Recommendations: SQL Statements	11-18
Reviewing the SQL Access Advisor Recommendations: Details	11-19
Implementing the SQL Access Advisor Recommendations	11-20

12 Analyzing SQL Performance Impact

SQL Performance Analyzer Usage	12-1
SQL Performance Analyzer Methodology	12-2
Capturing and Transporting a SQL Workload	12-3
Setting Up the Database Environment on the Test System	12-4
Executing a SQL Workload	12-5
Running SQL Performance Analyzer	12-5
Performing an Optimizer Upgrade Simulation with SQL Performance Analyzer	12-7
Testing an Initialization Parameter Change with SQL Performance Analyzer	12-10
Following a Guided Workflow with SQL Performance Analyzer	12-12
Creating a SQL Performance Analyzer Task Based on a SQL Tuning Set	12-13
Establishing the Initial Environment	12-14
Collecting SQL Performance Data Before the Change	12-14
Making the System Change	12-16
Collecting SQL Performance Data After the Change	12-16
Comparing SQL Performance Before and After the Change	12-17
Reviewing the SQL Performance Analyzer Report	12-19
Reviewing the SQL Performance Analyzer Report: General Information	12-19
Reviewing the SQL Performance Analyzer Report: Global Statistics	12-20
Reviewing the SQL Performance Analyzer Report: Global Statistics Details	12-21

Index

Preface

This preface contains the following topics:

- Audience
- Documentation Accessibility
- Related Documents
- Conventions

Audience

This guide is intended for Oracle database administrators (DBAs) who want to tune and optimize the performance of Oracle Database. Before using this document, you should complete *Oracle Database 2 Day DBA*.

In particular, this guide is targeted toward the following groups of users:

- Oracle DBAs who want to acquire database performance tuning skills
- DBAs who are new to Oracle Database

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at

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

Accessibility of Code Examples in Documentation

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

Accessibility of Links to External Web Sites in Documentation

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

TTY Access to Oracle Support Services

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

Related Documents

For more information about the topics covered in this document, see the following documents:

- Oracle Database 2 Day DBA
- Oracle Database Administrator's Guide
- Oracle Database Concepts
- Oracle Database Performance Tuning Guide

Conventions

The following conventions are used in this document:

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

What's New in Oracle Performance?

This section describes new performance features of Oracle Database 11g Release 1 (11.1) and provides pointers to additional information. The features and enhancements described in this section comprise the overall effort to optimize database performance.

For a summary of all new features for Oracle Database 11g Release 1 (11.1), see Oracle Database New Features Guide.

The new and updated performance features in Oracle Database 11g Release 1 (11.1) include:

Automatic Database Diagnostic Monitor (ADDM) enhancements

ADDM has been enhanced to perform global analysis on all instances of an Oracle Real Application Clusters (Oracle RAC) cluster.

For more information, see "ADDM for Oracle Real Application Clusters" on page 3-3.

Wait activity details

You can view wait activity details for services, modules, and actions from the Top Activity page.

For more information, see "Monitoring User Activity" on page 4-2.

Improved Automatic Workload Repository (AWR) baselines

You can create a single, fixed-time AWR baseline or a repeating baseline. Additionally, improved manageability of AWR baselines is now available, including the ability to compute threshold statistics.

For more information, see "Managing Baselines" on page 8-1.

 Improved manageability of SQL Advisor, SQL Access Advisor, and SQL Tuning Sets

Significant improvements were made to managing SQL Tuning Advisor, SQL Access Advisor, and SQL Tuning Sets.

For more information, see Chapter 10, "Tuning SQL Statements" and Chapter 11, "Optimizing Data Access Paths".

SQL Performance Analyzer

SQL Performance Analyzer enables you to measure the impact of system changes on SQL performance by testing these changes using a SQL workload on a test system.

For more information, see Chapter 12, "Analyzing SQL Performance Impact".

Active Session History (ASH) enhancements

The ASH report has been enhanced to include row-level tracking of top SQL statements, top PL/SQL procedures, top Java workload, top phases of execution, and other session-related information.

For more information, see Chapter 7, "Resolving Transient Performance Problems".

Automated Maintenance Tasks

Key maintenance tasks such as optimizer statistics collection, Automatic SQL Tuning, and Segment Advisor are now managed through the Automatic Maintenance Task framework. This framework enables you to exercise finer control over the scheduling of maintenance tasks and on their CPU consumption.

For more information, see "Viewing Automatic SQL Tuning Results" on page 10-5.

Customizable Performance page

You can customize the Performance page to show only the graphs and charts that you want displayed. You can also choose whether to include AWR baselines in the performance charts.

For more information, see "Customizing the Database Performance Page" on page 4-27.

I/O Statistics

The Performance page now contains charts for Throughput, I/O, Parallel Execution, and Services. You can use the I/O charts to monitor I/O throughput and latency information based on various criteria such as function (for example, Recovery Manager), consumer groups, and file types.

For more information, see "Monitoring Instance Activity" on page 4-10.

Part I

Getting Started

Part I provides an introduction to this guide and explains the Oracle Database performance method. This part contains the following chapters:

- Chapter 1, "Introduction"
- Chapter 2, "Oracle Database Performance Method"

Introduction

As an Oracle database administrator (DBA), you are responsible for the performance of your Oracle database. Tuning a database to reach a desirable performance level may be a daunting task, especially for DBAs who are new to Oracle Database. *Oracle Database 2 Day + Performance Tuning Guide* is a quick start guide that teaches you how to perform day-to-day database performance tuning tasks using features provided by Oracle Diagnostics Pack, Oracle Tuning Pack, and Oracle Enterprise Manager (Enterprise Manager).

This chapter contains the following sections:

- About This Guide
- Common Oracle DBA Tasks
- Tools for Tuning the Database

About This Guide

Before using this guide, you need to do the following:

- Read Oracle Database 2 Day DBA in its entirety.
- Obtain the necessary products and tools described in "Tools for Tuning the Database" on page 1-2.

Oracle Database 2 Day + Performance Tuning Guide is task-oriented. The objective is to describe why and when tuning tasks need to be performed.

This guide is not an exhaustive discussion of all Oracle Database concepts. For this type of information, see *Oracle Database Concepts*.

This guide does not describe basic Oracle Database administrative tasks. For this type of information, see *Oracle Database 2 Day DBA*. For a complete discussion of administrative tasks, see *Oracle Database Administrator's Guide*.

The primary interface used in this guide is the Enterprise Manager Database Control console. This guide is not an exhaustive discussion of all Oracle Database performance tuning features and does not cover available application programming interfaces (APIs) that provide comparable tuning options to those presented in this guide. For this type of information, see *Oracle Database Performance Tuning Guide*.

Common Oracle DBA Tasks

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

Installing Oracle software

- Creating an Oracle database
- Upgrading the database and software to new releases
- Starting up and shutting down the database
- Managing the storage structures of the database
- Managing user accounts and security
- Managing schema objects, such as tables, indexes, and views
- Making database backups and performing database recovery, when necessary
- Proactively monitoring the condition of the database and taking preventive or corrective actions, as required
- Monitoring and tuning database performance

In a small-to-midsize database environment, you might be the sole person performing these tasks. In large, enterprise environments, the job is often divided among several DBAs—each with his or her own specialty—such as database security or database tuning. *Oracle Database 2 Day + Performance Tuning Guide* describes how to accomplish the last two tasks in this list.

Tools for Tuning the Database

The intent of this guide is to allow you to quickly and efficiently tune and optimize the performance of Oracle Database.

To achieve the goals of this guide, you will need to acquire the following products, tools, features, and utilities:

Oracle Database 11g Enterprise Edition

Oracle Database 11*g* Enterprise Edition offers enterprise-class performance, scalability and reliability on clustered and single-server configurations. It includes many performance features that are used in this guide.

Oracle Enterprise Manager

The primary tool to manage your database is Enterprise Manager, a Web-based interface. After you install the Oracle software, create or upgrade a database, and configure the network, you can use Oracle Enterprise Manager to manage your database. In addition, Enterprise Manager provides an interface for performance advisors and for database utilities, such as SQL*Loader and Recovery Manager (RMAN).

Oracle Diagnostics Pack

Oracle Diagnostics Pack offers a complete, cost-effective, and easy-to-use solution to manage the performance of Oracle Database environments by providing unique features, such as automatic identification of performance bottlenecks, guided problem resolution, and comprehensive system monitoring. Key features of Oracle Diagnostics Pack that are used in this guide include Automatic Database Diagnostic Monitor (ADDM) and Automatic Workload Repository (AWR).

Oracle Database Tuning Pack

Oracle Database Tuning Pack automates the database application tuning process, thereby significantly lowering database management costs while enhancing performance and reliability. Key features of Oracle Database Tuning Pack that are used in this guide include the following:

- SQL Tuning Advisor

This feature enables you to submit one or more SQL statements as input and receive output in the form of specific advice or recommendations for how to tune statements, along with a rationale for each recommendation and its expected benefit. A recommendation relates to collection of statistics on objects, creation of new indexes, restructuring of the SQL statements, or creation of SQL Profiles.

SQL Access Advisor

This feature enables you to optimize data access paths of SQL queries by recommending the proper set of materialized views and view logs, indexes, and partitions for a given SQL workload.

Oracle Real Application Testing

Oracle Real Application Testing consists of two key features: Database Replay and SQL Performance Analyzer. Database Replay enables you to capture the database workload on a production system, and replay it on a test system with the exact same timing and concurrency as the production system on the same or newer release of Oracle Database. SQL Performance Analyzer enables you to assess the effect of system changes on SQL performance by identifying SQL statements that have regressed, improved, or remained unchanged.

Note: Some of the products and tools in the preceding list, including Oracle Diagnostics Pack and Oracle Database Tuning Pack, require separate licenses. For more information, see *Oracle Database Licensing Information*.

Oracle Database Performance Method

Performance improvement is an iterative process. Removing the first bottleneck (a point where resource contention is at its highest) might not lead to performance improvement immediately because another bottleneck might be revealed that has an even greater performance impact on the system. For this reason, the Oracle performance method is iterative. Accurately diagnosing the performance problem is the first step toward ensuring that the changes you make to the system will result in improved performance.

Performance problems generally result from a lack of throughput, unacceptable user or job response time, or both. The problem might be localized to specific application modules, or it might span the entire system. Before looking at any database or operating system statistics, it is crucial to get feedback from the most important components of the system: the users of the system and the people ultimately paying for the application. Getting feedback from users makes determining the performance goal easier, and improved performance can be measured in terms of real business goals, rather than system statistics.

The Oracle performance method can be applied until performance goals are met or deemed impractical. Because this process is iterative, it is likely that some investigations will be made that have little impact on the performance of the system. It takes time and experience to accurately pinpoint critical bottlenecks in a timely manner. Automatic Database Diagnostic Monitor (ADDM) implements the Oracle performance method and analyzes statistics to provide automatic diagnosis of major performance problems. Using ADDM can significantly shorten the time required to improve the performance of a system, and it is the method used in this guide.

This chapter discusses the Oracle Database performance method and contains the following sections:

- Gathering Database Statistics Using the Automatic Workload Repository
- Using the Oracle Performance Method
- Common Performance Problems Found in Oracle Databases

Gathering Database Statistics Using the Automatic Workload Repository

Database statistics provide information about the type of load on the database and the internal and external resources used by the database. To accurately diagnose performance problems with the database using ADDM, statistics must be available.

Oracle Database generates many types of cumulative statistics for the system, sessions, and individual SQL statements. Oracle Database also tracks cumulative statistics about segments and services. Automatic Workload Repository (AWR) automates database

statistics gathering by collecting, processing, and maintaining performance statistics for database problem detection and self-tuning purposes.

By default, the statistics gathering process repeats every hour and results in an AWR snapshot, which is a set of data for a specific time that is used for performance comparisons. The delta values captured by the snapshot represent the changes for each statistic over the time period. Statistics gathered by AWR are queried from memory. The gathered data can be displayed in both reports and views.

Gathering database statistics using AWR is enabled by default and is controlled by the STATISTICS_LEVEL initialization parameter. The STATISTICS_LEVEL parameter should be set to TYPICAL or ALL to enable statistics gathering by AWR. The default setting is TYPICAL. Setting the STATISTICS_LEVEL parameter to BASIC disables many Oracle Database features, including AWR, and is not recommended. For information about the STATISTICS_LEVEL initialization parameter, see Oracle Database Reference.

The CONTROL_MANAGEMENT_PACK_ACCESS initialization parameter should be set to DIAGNOSTIC+TUNING (default) or DIAGNOSTIC to enable automatic database diagnostic monitoring. Setting CONTROL_MANAGEMENT_PACK_ACCESS to NONE disables many Oracle Database features, including ADDM, and is strongly discouraged. For information about the CONTROL_MANAGEMENT_PACK_ACCESS initialization parameter, see *Oracle Database Reference*.

The database statistics collected and processed by AWR include:

- Time Model Statistics
- Wait Event Statistics
- Session and System Statistics
- Active Session History Statistics
- High-Load SQL Statistics

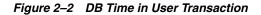
Time Model Statistics

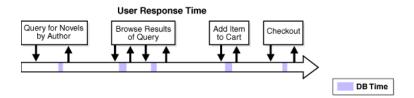
Time model statistics measure the time spent in the database by operation type. The most important time model statistic is database time, or DB time. Database time represents the total time spent in database calls, and is an indicator of the total instance workload. As shown in Figure 2–1, database time makes up a portion of an application's overall user response time.





A session is a specific connection of a user to an Oracle Database instance through a user process. Database time is calculated by aggregating the CPU time and wait time of all user sessions not waiting for idle wait events (user sessions that are not idle). For example, a user session may involve an online transaction made at a bookseller's Web site consisting of the actions shown in Figure 2–2.





1. Query for novels by author

The user performs a search for novels by a particular author. This action causes the application to perform a database query for novels by the author.

2. Browse results of query

The user browses the returned list of novels by the author and accesses additional details, such as user reviews and inventory status. This action causes the application to perform additional database queries.

3. Add item to cart

After browsing details about the novels, the user decides to add one of the novels to the shopping cart. This action causes the application to make a database call to update the shopping cart.

4. Checkout

The user completes the transaction by checking out, using the address and payment information previously saved at the bookseller's Web site from a previous purchase. This action causes the application to perform various database operations to retrieve the user's information, add a new order, update the inventory, and generate an E-mail confirmation.

For each of the preceding actions, the user makes a request to the database, as represented by the down arrow in Figure 2–2 on page 2-3. The CPU time spent by the database processing the request and the wait time spent waiting for the database are considered DB time, as represented by the shaded areas. After the request is completed, the results are returned to the user, as represented by the up arrow. The space between the up and down arrows represents the total user response time for processing the request, which contains other components besides DB time, as illustrated in Figure 2–1 on page 2-2.

The objective of database tuning is to reduce the time that users spend performing actions on the database, or reducing database time. In this way, you can improve the overall response time of user transactions on the application.

Wait Event Statistics

Wait events are incremented by a session to indicate that the session had to wait for an event to complete before being able to continue processing. When a session has to wait while processing a user request, the database records the wait by using one of a set of predefined wait events. The events are then grouped into wait classes, such as User I/O and Network. Wait event data reveals symptoms of problems that might be affecting performance, such as latch, buffer, and I/O contention.

See Also:

- Oracle Database Performance Tuning Guide
- Oracle Database Reference

Session and System Statistics

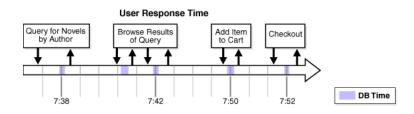
A large number of cumulative database statistics are available on a system and session level. Some of these statistics are collected by AWR.

Active Session History Statistics

The Active Session History (ASH) statistics are samples of session activity in the database. The database samples active sessions every second and stores them in a circular buffer in the System Global Area (SGA). Any session that is connected to the database and using CPU, or is waiting for an event that does not belong to the idle wait class, is considered an active session. By capturing only active sessions, a manageable set of data is represented. The size of the data is directly related to the work being performed, rather than the number of sessions allowed on the system.

Using the DB time example described in "Time Model Statistics" on page 2-2, samples of session activity are collected from the online transaction made at the bookseller's Web site, represented as vertical lines below the horizontal arrow in Figure 2–3.

Figure 2–3 Active Session History



The light vertical lines represent samples of inactive session activity that are not captured in the ASH statistics. The bold vertical lines represent samples of active sessions that are captured at:

- 7:38, while novels by the author are being queried
- 7:42, while the user is browsing the query results
- 7:50, when one of the novels is added to the shopping cart
- 7:52, during the checkout process

Table 2–1 lists the ASH statistics that are collected for the active sessions, along with examples of the session ID (SID), module, SQL ID, session state, and wait events that are sampled.

Time	SID	Module	SQL ID State E		Event		
7:38	213	Book by author	qa324jffritcf	Waiting	db file sequential read		
7:42	213	Get review ID	aferv5desfzs5	CPU			
7:50	213	Add item to cart	hk32pekfcbdfr	Waiting	buffer busy wait		
7:52	213	Checkout	abngldf95f4de	Waiting	log file sync		

Table 2–1 Active Session History

High-Load SQL Statistics

SQL statements that are consuming the most resources produce the highest load on the system, based on criteria such as elapsed time and CPU time.

Using the Oracle Performance Method

Performance tuning using the Oracle performance method is driven by identifying and eliminating bottlenecks in the database, and by developing efficient SQL statements. Database tuning is performed in two phases: proactively and reactively.

In the proactive tuning phase, you must perform tuning tasks as part of your daily database maintenance routine, such as reviewing ADDM analysis and findings, monitoring the real-time performance of the database, and responding to alerts.

In the reactive tuning phase, you must respond to issues reported by the users, such as performance problems that may occur for only a short duration of time, or performance degradation to the database over a period of time.

SQL tuning is an iterative process to identify, tune, and improve the efficiency of high-load SQL statements.

Applying the Oracle performance method involves the following:

- Performing pre-tuning preparations, as described in "Preparing the Database for Tuning" on page 2-5
- Tuning the database proactively on a regular basis, as described in "Tuning the Database Proactively" on page 2-6
- Tuning the database reactively when performance problems are reported by the users, as described in "Tuning the Database Reactively" on page 2-6
- Identifying, tuning, and optimizing high-load SQL statements, as described in "Tuning SQL Statements" on page 2-7

To improve the performance of your database, you will need to apply these principles iteratively.

Preparing the Database for Tuning

This section lists and describes the steps that must be performed before the database can be properly tuned.

To prepare the database for tuning:

1. Get feedback from users.

Determine the scope of the performance project and subsequent performance goals, and determine performance goals for the future. This process is key for future capacity planning.

2. Check the operating systems of all systems involved with user performance.

Check for hardware or operating system resources that are fully utilized. List any overused resources as possible concerns for later analysis. In addition, ensure that all hardware is functioning properly.

- **3.** Ensure that the STATISTICS_LEVEL initialization parameter is set to TYPICAL (default) or ALL to enable the automatic performance tuning features of Oracle Database, including AWR and ADDM.
- **4.** Ensure that the CONTROL_MANAGEMENT_PACK_ACCESS initialization parameter is set to DIAGNOSTIC+TUNING (default) or DIAGNOSTIC to enable ADDM.

See Also:

- "Gathering Database Statistics Using the Automatic Workload Repository" on page 2-1 for information about configuring AWR
- "Configuring Automatic Database Diagnostic Monitor" on page 3-3

Tuning the Database Proactively

This section lists and describes the steps required to keep the database properly tuned on a regular basis. These tuning procedures are considered proactive and should be performed as part of your daily maintenance of Oracle Database.

To tune the database proactively:

1. Review the ADDM findings, as described in Chapter 3, "Automatic Database Performance Monitoring".

ADDM automatically detects and reports on performance problems with the database, including most of the "Common Performance Problems Found in Oracle Databases" on page 2-8. The results are displayed as ADDM findings on the Database Home page in Oracle Enterprise Manager (Enterprise Manager). Reviewing these findings enables you to quickly identify the performance problems that require your attention.

2. Implement the ADDM recommendations, as described in Chapter 3, "Automatic Database Performance Monitoring".

ADDM automatically provides a list of recommendations for reducing the impact of the performance problem with each ADDM finding. Implementing a recommendation applies the suggested changes to improve the database performance.

3. Monitor performance problems with the database in real time, as described in Chapter 4, "Monitoring Real-Time Database Performance".

The Performance page in Enterprise Manager enables you to identify and respond to real-time performance problems. By drilling down to the appropriate pages, you can identify and resolve performance problems with the database in real time, without having to wait until the next ADDM analysis.

4. Respond to performance-related alerts, as described in Chapter 5, "Monitoring Performance Alerts".

The Database Home page in Enterprise Manager enables you to view performance-related alerts generated by the system. Typically, these alerts reveal performance problems whose resolutions will improve the performance of your database.

- **5.** Validate that the changes made have produced the desired effect, and verify that the users perceive performance improvements.
- **6.** Repeat these steps until your performance goals are met or become impossible to achieve because of other constraints.

Tuning the Database Reactively

This section lists and describes the steps required to tune the database based on user feedback. This tuning procedure is considered reactive. Perform this procedure periodically when performance problems are reported by the users.

To tune the database reactively:

1. Run ADDM manually to diagnose current and historical database performance when performance problems are reported by the users, as described in Chapter 6, "Manual Database Performance Monitoring".

This task is useful if you want to analyze current database performance before the next ADDM analysis, or to analyze historical database performance when you were not proactively monitoring the system.

2. Resolve transient performance problems, as described in Chapter 7, "Resolving Transient Performance Problems".

The Active Session History (ASH) reports enable you to analyze transient performance problems with the database that are short-lived and do not appear in the ADDM analysis.

3. Resolve performance degradation over time, as described in Chapter 8, "Resolving Performance Degradation Over Time".

The Automatic Workload Repository (AWR) Compare Periods report enables you to compare database performance between two periods of time, and resolve performance degradation that may happen from one time period to another.

- **4.** Validate that the changes made have produced the desired effect, and verify that the users perceive performance improvements.
- **5.** Repeat these steps until your performance goals are met or become impossible to achieve due to other constraints.

Tuning SQL Statements

This section lists and describes the steps required to identify, tune, and optimize high-load SQL statements.

To tune SQL statements:

1. Identify high-load SQL statements, as described in Chapter 9, "Identifying High-Load SQL Statements".

Use the ADDM findings and the Top SQL section to identify high-load SQL statements that are causing the greatest contention.

2. Tune high-load SQL statements, as described in Chapter 10, "Tuning SQL Statements".

You can improve the efficiency of high-load SQL statements by tuning them using SQL Tuning Advisor.

3. Optimize data access paths, as described in Chapter 11, "Optimizing Data Access Paths".

You can optimize the performance of data access paths by creating the proper set of materialized views, materialized view logs, and indexes for a given workload by using SQL Access Advisor.

4. Analyze the SQL performance impact of SQL tuning and other system changes, as described in Chapter 12, "Analyzing SQL Performance Impact".

You can analyze the performance impact of your SQL tuning activities or other system changes on a given SQL workload by using SQL Performance Analyzer.

5. Repeat these steps until all high-load SQL statements are tuned for greatest efficiency.

Common Performance Problems Found in Oracle Databases

This section lists and describes common performance problems found in Oracle databases. By following the Oracle performance method, you should be able to avoid these problems. If you have these problems, then repeat the steps in the Oracle performance method, as described in "Using the Oracle Performance Method" on page 2-5, or consult the appropriate section that addresses these problems:

CPU bottlenecks

Is the application performing poorly because the system is CPU-bound? Performance problems caused by CPU bottlenecks are diagnosed by ADDM, as described in Chapter 3, "Automatic Database Performance Monitoring". You can also identify CPU bottlenecks by using the Performance page in Enterprise Manager, as described in "Monitoring CPU Utilization" on page 4-19.

Undersized memory structures

Are the Oracle memory structures—such as the System Global Area (SGA), Program Global Area (PGA), and buffer cache—adequately sized? Performance problems caused by undersized memory structures are diagnosed by ADDM, as described in Chapter 3, "Automatic Database Performance Monitoring". You can also identify memory usage issues by using the Performance page in Enterprise Manager, as described in "Monitoring Memory Utilization" on page 4-22.

I/O capacity issues

Is the I/O subsystem performing as expected? Performance problems caused by I/O capacity issues are diagnosed by ADDM, as described in Chapter 3, "Automatic Database Performance Monitoring". You can also identify disk I/O issues by using the Performance page in Oracle Enterprise Manager, as described in "Monitoring Disk I/O Utilization" on page 4-25.

Suboptimal use of Oracle Database by the application

Is the application making suboptimal use of Oracle Database? Problems such as establishing new database connections repeatedly, excessive SQL parsing, and high levels of contention for a small amount of data (also known as application-level block contention) can degrade the application performance significantly. Performance problems caused by suboptimal use of Oracle Database by the application are diagnosed by ADDM, as described in Chapter 3, "Automatic Database Performance Monitoring". You can also monitor top activity in various dimensions—including SQL, session, services, modules, and actions—by using the Performance page in Enterprise Manager, as described in "Monitoring User Activity" on page 4-2.

Concurrency issues

Is the database performing suboptimally due to a high degree of concurrent activities in the database? A high degree of concurrent activities might result in contention for shared resources that can manifest in the forms of locks or waits for buffer cache. Performance problems caused by concurrency issues are diagnosed by ADDM, as described in Chapter 3, "Automatic Database Performance Monitoring". You can also identify concurrency issues by using Top Sessions in Enterprise Manager, as described in "Monitoring Top Sessions" on page 4-5.

Database configuration issues

Is the database configured optimally to provide desired performance levels? For example, is there evidence of incorrect sizing of log files, archiving issues, excessive number of checkpoints, or suboptimal parameter settings? Performance

problems caused by database configuration issues are diagnosed by ADDM, as described in Chapter 3, "Automatic Database Performance Monitoring".

Short-lived performance problems

Are users complaining about short-lived or intermittent performance problems? Depending on the interval between snapshots taken by AWR, performance problems that have a short duration may not be captured by ADDM. You can identify short-lived performance problems by using the Active Session History report, as described in Chapter 7, "Resolving Transient Performance Problems".

Degradation of database performance over time

Is there evidence that the database performance has degraded over time? For example, are you or your users noticing that the database is not performing as well as it was 6 months ago? You can generate an AWR Compare Periods report to compare the period when the performance was poor to a period when the performance is stable to identify configuration settings, workload profile, and statistics that are different between these two time periods. This technique will help you identify the cause of the performance degradation, as described in Chapter 8, "Resolving Performance Degradation Over Time".

Inefficient or high-load SQL statements

Are any SQL statements using excessive system resources that impact the system? Performance problems caused by high-load SQL statements are diagnosed by ADDM, as described in Chapter 3, "Automatic Database Performance Monitoring" and "Identification of High-Load SQL Statements Using ADDM Findings" on page 9-1. You can also identify high-load SQL statements by using Top SQL in Enterprise Manager, as described in "Identifying High-Load SQL Statements Using Top SQL" on page 9-2. After they have been identified, you can tune the high-load SQL statements using SQL Tuning Advisor, as described in Chapter 10, "Tuning SQL Statements".

Object contention

Are any database objects the source of bottlenecks because they are continuously accessed? Performance problems caused by object contention are diagnosed by ADDM, as described in Chapter 3, "Automatic Database Performance Monitoring". You can also optimize the data access path to these objects using SQL Access Advisor, as described in Chapter 11, "Optimizing Data Access Paths" on page 4-25.

Unexpected performance regression after tuning SQL statements

Is the performance of SQL statements degrading after they have been tuned? Tuning SQL statements may cause changes to execution plans of SQL statements, resulting in a significant impact on SQL performance. In some cases, the changes may result in the improvement of SQL performance. In other cases, the changes may cause SQL statements to regress, resulting in a degradation of SQL performance. Before making changes on a production system, you can analyze the performance impact from tuning SQL statements on a test system by using SQL Performance Analyzer, as described in Chapter 12, "Analyzing SQL Performance Impact".

Part II

Proactive Database Tuning

Part II describes how to tune Oracle Database proactively on a regular basis and contains the following chapters:

- Chapter 3, "Automatic Database Performance Monitoring"
- Chapter 4, "Monitoring Real-Time Database Performance"
- Chapter 5, "Monitoring Performance Alerts"

Automatic Database Performance Monitoring

Automatic Database Diagnostic Monitor (ADDM) automatically detects and reports performance problems with the database. The results are displayed as ADDM findings on the Database Home page in Oracle Enterprise Manager (Enterprise Manager). Reviewing the ADDM findings enables you to quickly identify the performance problems that require your attention.

Each ADDM finding provides a list of recommendations for reducing the impact of the performance problem. Reviewing ADDM findings and implementing the recommendations are tasks that you should perform daily as part of the regular database maintenance. Even when the database is operating at an optimal performance level, you should continue to use ADDM to monitor database performance on an ongoing basis.

This chapter contains the following sections:

- Overview of Automatic Database Diagnostic Monitor
- Configuring Automatic Database Diagnostic Monitor
- Reviewing the Automatic Database Diagnostic Monitor Analysis
- Interpretation of Automatic Database Diagnostic Monitor Findings
- Implementing Automatic Database Diagnostic Monitor Recommendations
- Viewing Snapshot Statistics

See Also:

• Oracle Database Performance Tuning Guide for information about using the DBMS_ADVISOR package to diagnose and tune the database with the Automatic Database Diagnostic Monitor

Overview of Automatic Database Diagnostic Monitor

ADDM is self-diagnostic software built into Oracle Database. ADDM examines and analyzes data captured in the Automatic Workload Repository (AWR) to determine possible performance problems in Oracle Database. ADDM then locates the root causes of the performance problems, provides recommendations for correcting them, and quantifies the expected benefits. ADDM also identifies areas where no action is necessary.

This section contains the following topics:

- ADDM Analysis
- ADDM Recommendations

ADDM for Oracle Real Application Clusters

ADDM Analysis

An ADDM analysis is performed after each AWR snapshot (every hour by default), and the results are saved in the database. You can then view the results by means of Oracle Enterprise Manager. Before using another performance tuning method described in this guide, first review the results of the ADDM analysis.

The ADDM analysis is performed from the top down, first identifying symptoms and then refining the analysis to reach the root causes of performance problems. ADDM uses the DB time statistic to identify performance problems. DB time is the cumulative time spent by the database in processing user requests, including both the wait time and CPU time of all user sessions that are not idle.

The goal of database performance tuning is to reduce the DB time of the system for a given workload. By reducing DB time, the database is able to support more user requests by using the same or a smaller amount of resources. ADDM reports system resources that are using a significant portion of DB time as problem areas and sorts them in descending order by the amount of related DB time spent. For more information about the DB time statistic, see "Time Model Statistics" on page 2-2.

ADDM Recommendations

In addition to diagnosing performance problems, ADDM recommends possible solutions. When appropriate, ADDM recommends multiple solutions from which you can choose. ADDM recommendations include the following:

Hardware changes

Adding CPUs or changing the I/O subsystem configuration

Database configuration

Changing initialization parameter settings

Schema changes

Hash partitioning a table or index, or using automatic segment space management (ASSM)

Application changes

Using the cache option for sequences or using bind variables

Using other advisors

Running SQL Tuning Advisor on high-load SQL statements or running the Segment Advisor on hot objects

ADDM benefits apply beyond production systems. Even on development and test systems, ADDM can provide an early warning of potential performance problems.

Performance tuning is an iterative process. Fixing one problem can cause a bottleneck to shift to another part of the system. Even with the benefit of the ADDM analysis, it can take multiple tuning cycles to reach a desirable level of performance.

See Also:

Oracle Database 2 Day DBA for information the Segment Advisor

ADDM for Oracle Real Application Clusters

In an Oracle Real Application Clusters (Oracle RAC) environment, you can use ADDM to analyze the throughput performance of a database cluster. ADDM for Oracle RAC considers DB time as the sum of database times for all database instances and reports findings that are significant at the cluster level. For example, the I/O levels of each cluster node may be insignificant when considered locally, but the aggregate I/O level may be a significant problem for the cluster as a whole.

See Also:

 Oracle Database 2 Day + Real Application Clusters Guide for information about using ADDM for Oracle RAC

Configuring Automatic Database Diagnostic Monitor

This section describes how to configure ADDM and contains the following topics:

- Setting Initialization Parameters to Enable ADDM
- Setting the DBIO_EXPECTED Parameter
- Managing AWR Snapshots

Setting Initialization Parameters to Enable ADDM

Automatic database diagnostic monitoring is enabled by default and is controlled by the CONTROL_MANAGEMENT_PACK_ACCESS and the STATISTICS_LEVEL initialization parameters.

The CONTROL_MANAGEMENT_PACK_ACCESS initialization parameter should be set to DIAGNOSTIC+TUNING (default) or DIAGNOSTIC to enable automatic database diagnostic monitoring. Setting CONTROL_MANAGEMENT_PACK_ACCESS to NONE disables many Oracle Database features, including ADDM, and is strongly discouraged.

The STATISTICS_LEVEL initialization parameter should be set to the TYPICAL (default) or ALL to enable automatic database diagnostic monitoring. Setting STATISTICS_LEVEL to BASIC disables many Oracle Database features, including ADDM, and is strongly discouraged.

To determine whether ADDM is enabled:

1. From the Database Home page, click **Server**.

The Server subpage appears.

2. In the Database Configuration section, click Initialization Parameters.

The Initialization Parameters page appears.

3. In the Name field, enter statistics_level and then click Go.

The table shows the setting of this initialization parameter.

- **4.** Do one of the following:
 - If the Value column shows ALL or TYPICAL, then do nothing.
 - If the Value column shows **BASIC**, then select **ALL** or **TYPICAL** and click **Apply**.
- 5. In the Name field, enter <code>control_management_pack_access</code> and then click Go.

The table shows the setting of this initialization parameter.

- **6.** Do one of the following:
 - If the Value column shows DIAGNOSTIC or DIAGNOSTIC+TUNING, then do nothing.
 - If the Value column shows NONE, then select DIAGNOSTIC or DIAGNOSTIC+TUNING and click Apply.

See Also:

- Oracle Database Reference for information about the STATISTICS_LEVEL initialization parameter
- Oracle Database Reference for information about the CONTROL_MANAGEMENT_PACK_ACCESS initialization parameter

Setting the DBIO_EXPECTED Parameter

ADDM analysis of I/O performance partially depends on a single argument, DBIO_EXPECTED, that describes the expected performance of the I/O subsystem. The value of DBIO_EXPECTED is the average time it takes to read a single database block, in microseconds. Oracle Database uses the default value of 10 milliseconds, which is an appropriate value for most hard drives. If your hardware is significantly different, then consider using a different value.

To determine the correct setting for the DBIO_EXPECTED initialization parameter:

1. Measure the average read time of a single database block for your hardware.

This measurement needs to be taken for random I/O, which includes seek time if you use standard hard drives. Typical values for hard drives are between 5000 and 20000 microseconds.

2. Set the value one time for all subsequent ADDM executions.

For example, if the measured value is 8000 microseconds, then execute the following PL/SQL code as the SYS user:

EXECUTE DBMS_ADVISOR.SET_DEFAULT_TASK_PARAMETER('ADDM', 'DBIO_EXPECTED', 8000);

Managing AWR Snapshots

By default, the Automatic Workload Repository (AWR) generates snapshots of performance data once every hour, and retains the statistics in the workload repository for 8 days. You can change the default values for both the snapshot interval and the retention period.

Oracle recommends that you adjust the AWR retention period to at least a month. You can also extend the period to one business cycle so you can compare data across time frames such as the close of the fiscal quarter. You can also create AWR baselines to retain snapshots indefinitely for important time periods.

The data in the snapshot interval is analyzed by ADDM. ADDM compares the difference between snapshots to determine which SQL statements to capture, based on the effect on the system load. The ADDM analysis shows the number of SQL statements that need to be captured over time.

This section contains the following topics:

Creating Snapshots

Modifying Snapshot Settings

Creating Snapshots

Manually creating snapshots is usually not necessary because AWR generates snapshots of the performance data once every hour by default. In some cases, however, it may be necessary to manually create snapshots to capture different durations of activity, such as when you want to compare performance data over a shorter period of time than the snapshot interval.

To create snapshots:

1. From the Database Home page, click **Performance**.

The Performance page appears.

2. Under Additional Monitoring Links, click Snapshots.

The Snapshots page appears with a list of the most recent snapshots.

3. Click Create.

The Confirmation page appears.

4. Click Yes.

The Processing: Create Snapshot page is displayed while the snapshot is being taken.

After the snapshot is taken, the Snapshots page reappears with a Confirmation message.

In this example, the ID of the snapshot that was created is 249.

				Create
Del	ete)	Actions Create SQL Tuning Set 💽 🗔	S Previo	ous 25 201-212 of 212 💌 Next 🛇
Select	ID	Capture Time \triangle	Collection Level	Within A Baseline
0	238	Jul 10, 2007 8:00:08 AM	TYPICAL	
0	239	Jul 10, 2007 9:00:23 AM	TYPICAL	
0	240	Jul 10, 2007 10:00:40 AM	TYPICAL	
0	241	Jul 10, 2007 11:00:58 AM	TYPICAL	
0	242	Jul 10, 2007 12:00:12 PM	TYPICAL	
- C	243	Jul 10, 2007 1:00:29 PM	TYPICAL	
0	244	Jul 10, 2007 2:00:44 PM	TYPICAL	
- C	245	Jul 10, 2007 3:00:00 PM	TYPICAL	
0	246	Jul 10, 2007 4:00:17 PM	TYPICAL	
0	247	Jul 10, 2007 5:00:34 PM	TYPICAL	
۲	248	Jul 10, 2007 6:00:49 PM	TYPICAL	
0	249	Jul 10, 2007 6:27:33 PM	TYPICAL	

Modifying Snapshot Settings

By default, AWR generates snapshots of performance data once every hour. Alternatively, you can modify the default values of both the interval between snapshots and their retention period.

To modify the snapshot settings:

1. From the Database Home page, click **Server**.

The Server subpage appears.

2. In the Statistics Management section, click Automatic Workload Repository.

The Automatic Workload Repository page appears.

General	
	Edit
Snapshot Retention (days)	
Snapshot Interval (minutes)	
Collection Level	TYPICAL
Next Snapshot Capture Time	Jan 30, 2007 11:30:20 AM

In this example, snapshot retention is set to 8 days and snapshot interval is set to 60 minutes.

3. Click Edit.

The Edit Settings page appears.

Snapshot Retention	Use Time-Based Retention Retention Period (Days) 8
Snapshot Collection	○ Retain Forever ● System Snapshot Interval Interval 1 Hour
	C Turn off Snapshot Collection
Collection Level TYP	ICAL

- **4.** For **Snapshot Retention**, do one of the following:
 - Select Use Time-Based Retention Period (Days), and in the associated field enter the number of days to retain the snapshots.
 - Select Retain Forever to retain snapshots indefinitely.

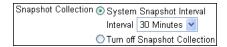
It is recommended that you increase the snapshot retention period whenever possible based on the available disk space.

In this example, the snapshot retention period is changed to 30 days.

Snapshot Retention 💿 Use Time-Based Retentio	n
Retention Period (Days)	
C Retain Forever	

- 5. For **Snapshot Collection**, do one of the following:
 - Select System Snapshot Interval and, in the Interval list, select the desired interval to change the interval between snapshots.
 - Select Turn off Snapshot Collection to disable snapshot collection.

In this example, the snapshot collection interval is changed to 30 minutes.



6. Click the link next to Collection Level.

The Initialization Parameter page appears.

To change the statistics level, select the desired value in the Value list for the statistics_level parameter. Click **Save to File** to set the value in the server parameter file.

In this example, the default value of Typical is used.

Name \triangle	Help	Revisions	¥alue	Comments	Туре	Basic	Modified	Dynamic	Category
statistics_level	Û		TYPICAL 💌		String				Diagnostics and Statistics
(Save to File)									

7. Click **OK** to apply the changes.

The Automatic Workload Repository page appears and displays the new settings.

General	
	(Edit)
Snapshot Retention (days)	30
Snapshot Interval (minutes)	30
Collection Level	
Next Snapshot Capture Time	Jan 30, 2007 11:00:20 AM

Reviewing the Automatic Database Diagnostic Monitor Analysis

By default, ADDM runs every hour to analyze snapshots taken by AWR during that period. If the database finds performance problems, then it displays the results of the analysis under Diagnostic Summary on the Database Home page.

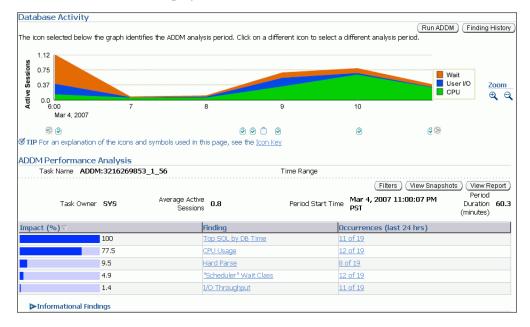
Diagnostic Su	nmary
ADDM Findings	5
Period Start	Mar 4, 2007 11:00:07 PM
Time	Mai 4, 2007 11.00.07 PM
Alert Log	No ORA- errors
SQL Response	Reference collection is empty.
Time	

The ADDM Findings link shows how many ADDM findings were found in the most recent ADDM analysis.

To view ADDM findings:

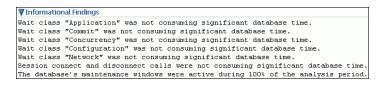
1. On the Database Home page, under Diagnostic Summary, click the link next to ADDM Findings.

The Automatic Database Diagnostic Monitor (ADDM) page appears. The results of the ADDM run are displayed.



On the Automatic Database Diagnostic Monitor (ADDM) page, the Database Activity chart shows the database activity during the ADDM analysis period. Database activity types are defined in the legend based on their corresponding colors in the chart. Each icon below the chart represents a different ADDM task, which in turn corresponds to a pair of individual Oracle Database snapshots saved in the Workload Repository. In this example, the largest block of activity from 8:00 onwards appears in green and corresponds to CPU usage, as described in the legend. This data suggests that CPU may be a performance bottleneck during the ADDM analysis period.

In the ADDM Performance Analysis section, the ADDM findings are listed in descending order, from highest impact to least impact. The Informational Findings section lists the areas that do not have a performance impact and are for informational purpose only.



- **2.** Optionally, click the Zoom icons to shorten or lengthen the analysis period displayed on the chart.
- 3. To view the ADDM findings in a report, click View Report.

The View Report page appears.

You can click Save to File to save the report for later access.

Interpretation of Automatic Database Diagnostic Monitor Findings

The ADDM analysis results are represented as a set of findings. Each ADDM finding belongs to one of three types:

Problem

Findings that describe the root cause of a database performance issue.

Symptom

Findings that contain information that often leads to one or more problem findings.

Information

Findings that are used to report areas of the system that do not have a performance impact.

Each problem finding is quantified with an estimate of the portion of DB time that resulted from the performance problem.

When a specific problem has multiple causes, ADDM may report multiple findings. In this case, the impacts of these multiple findings can contain the same portion of DB time. Because the performance problems can overlap, summing all the impacts of the reported findings can yield a number higher than 100 percent of DB time. For example, if a system performs many read I/O operations, ADDM may report a SQL statement responsible for 50 percent of DB time due to I/O activity as one finding, and an undersized buffer cache responsible for 75 percent of DB time as another finding.

A problem finding can be associated with a list of recommendations for reducing the impact of a performance problem. Each recommendation has a benefit that is an estimate of the portion of DB time that can be saved if the recommendation is implemented. When multiple recommendations are associated with an ADDM finding, the recommendations may contain alternatives for solving the same problem. In this case, the sum of the benefits may be higher than the impact of the finding. You do not need to apply all the recommendations to solve the same problem.

Recommendations are composed of actions and rationales. You must apply all the actions of a recommendation to gain the estimated benefit of that recommendation. The rationales explain why the set of actions was recommended, and provide additional information for implementing the suggested recommendation. An ADDM action may present multiple solutions to you. If this is the case, then choose the easiest solution to implement.

Implementing Automatic Database Diagnostic Monitor Recommendations

This section describes how to implement ADDM recommendations. ADDM findings are displayed in the Automatic Database Diagnostic Monitor (ADDM) page under ADDM Performance Analysis.

Impact (%) 🗸	Finding	Occurrences (last 24 hrs)
100	Top SQL by DB Time	<u>11 of 19</u>
77.5	CPU Usage	<u>13 of 19</u>
9.5	Hard Parse	8 of 19
4.9	"Scheduler" Wait Class	<u>11 of 19</u>
1.4	I/O Throughput	<u>11 of 19</u>

To implement ADDM recommendations:

1. On the Database Home page, under Diagnostic Summary, click the link next to ADDM Findings.

The Automatic Database Diagnostic Monitor (ADDM) page appears.

2. In the ADDM Performance Analysis section, click the ADDM finding that has the greatest impact.

In this example, the finding with the greatest impact is Top SQL by DB Time.

The Performance Finding Details page appears.

Performance Finding Details: Top SQL by DB Time		
Finding SQL statements consuming significant data	base time were found. Finding F	listory
Impact (Active Sessions) .84		
Impact (%) 100		
Period Start Time Mar 4, 2007 11:00:07 PM PST		
Period Duration (minutes) 60.3		
Filtered No (Filters)		
Recommendations		
Schedule SQL Tuning Advisor		
Select All Select None Show All Details Hide All Details		
Select Details Category		Benefit (%) ▽
□ ▼ <u>Hide</u> SQL Tuning		84.6
Action Tune the PL/SQL block with SQL_ID "68ccwtzvyn7qt". Refer to	he "Tuning PL/SQL Applications"	" chapter of Oracle's "PL/SQL
User's Guide and Reference".		
SQL Text DECLARE in number; BEGIN for i in 11000 loop select /*+ ORI SQL ID 68ccwtzvyn7qt	DERED USE NL(c) FUL	
		78.3
SqL Tuning		/8.3
Findings Path		
Expand All Collapse All		
Findings	Impact (%)	Additional Information
$oldsymbol{ abla}$ SQL statements consuming significant database time were found.	100	

3. Under Recommendations, review the recommendations and required actions for each recommendation.

The Category column displays the category of the recommendation. The Benefit (%) column displays the estimated benefit of implementing the recommendation.

Recommendations	
Schedule SQL Tuning Advisor	
Select All Select None Show All Details Hide All Details	
Select Details Category	Benefit (%) ∇
Vertical SQL Tuning	84.6
Tune the PL/SQL block with SQL_ID "68ccwtzvyn7qt". Refer to the "Tuning PL/SQL Applications" User's Guide and Reference". SQL Text DECLARE in number; BEGIN for i in 11000 loop select /*+ ORDERED USE NL(c) FUL SQL ID 68ccwtzvyn7qt	chapter of Oracle's "PL/SQL
✓ ▼Hide SQL Tuning	78.3
Action Run SQL Tuning Advisor on the SQL statement with SQL_ID "05b6pvb81dg8b". (Run Advisor Now	Filters

In this example, two recommendations are displayed for this finding. The first recommendation contains one action and is estimated to have a maximum benefit of up to 84.6% of DB time in the analysis period. The second recommendation contains one action and is estimated to have a maximum benefit of up to 78.3% of DB time in the analysis period.

4. If additional information about why the set of actions was recommended is available, then click **Additional Information**, or review the content displayed under Additional Information.

For example, the Undersized Buffer Cache finding contains additional information to indicate the value of the DB_CACHE_SIZE initialization parameter.

Recom	nendations		
Show All	Details Hide All Details		
Details	Category	Benefit (%) 🗸	
▼ <u>Hide</u>	DB Configuration		6.3
Action	Increase the buffer cache size by setting the value of parameter "db_c	ache_size" to 160 M. (Impleme	nt) (Filters)
Additio	nal Information		
The va	alue of parameter "db cache size" was "96 M" during	the analysis period.	

5. To view the history of a finding, click **Finding History**.

The Finding History page appears.

1 11-1-1	-Dar) There				
Fine	-		op SQL by DE					
View		Mar 4, 2007		Go) (Filters)				
Drag	; the	shaded box to c	hange the time pe	riod for the detail se	ection below.			
	2.0	0						
ŝio	1.	5				L		
essi								
Active Sessions	1.0	0						Other Load on System
cti	0.	5						Impact of Finding
Ā		-						
	0.0	0 4:52	6	7	8	9	10	11
		4.52 Mar 4, 2007	0	'	0	9	10	11
	(0						
E)eta	ail for Select	ed 3 Hour Inte	erval				
9	Shov	w All Details Hid	le All Details					
		ails Finding Det		In	npact (Active Se	ssions)	Start Dat	e
L.	⊳ sł	how ADDM:32162	269853 1 43	0.1	25		Mar 4, 200	7 8:22:56 PM PST
t	⊳ sł	how ADDM:32162	269853 1 44	1			Mar 4, 200	7 8:27:02 PM PST
t	⊳ sł	how ADDM:32162	269853 1 45	1			Mar 4, 200	7 8:30:02 PM PST
L.	▶ <u>s</u> ł	how ADDM:32162	269853 1 46	1.	5		Mar 4, 200	7 8:31:05 PM PST
t		how ADDM:32162	269853 1 47	1			Mar 4, 200	7 8:33:06 PM PST
t		how ADDM:32162	269853 1 48	0			Mar 4, 200	7 8:35:10 PM PST
L.		how ADDM:32162	269853 1 51	0.	06		Mar 4, 200	7 8:41:10 PM PST
t		how ADDM:32162	269853 1 52	0.	5		Mar 4, 200	7 8:57:16 PM PST
L.	► <u>sk</u>	how ADDM:32162	269853 1 53	1			Mar 4, 200	7 8:59:20 PM PST
E.		how ADDM:32162	269853 1 54	0.	78		Mar 4, 200	7 9:01:23 PM PST

The Finding History page shows how often a particular finding has occurred in a selected 3-hour interval. You can use this information to determine whether the finding was a transient or a persistent problem on the system. Based on this information, you can determine whether the actions associated with the finding should be implemented.

The Active Sessions chart shows the impact of the finding and of the other load on the system. You can change the display as follows:

- **a.** To move the 3-hour interval, click and drag the shaded box in the Active Sessions chart to the time period in which you are interested.
- **b.** To change dates, enter the desired date in the View field and click Go.
- **c.** To view details about a finding, under Detail for Selected 3 Hour Interval, click the link in the **Finding Details** column to display the Performance Finding Details page for the corresponding ADDM finding.
- **6.** Optionally, create a filter to suppress known findings that have been tuned or cannot be tuned further. To create filters for this ADDM finding:
 - a. Click Filters.

The Filters for Finding page appears.

Filter	s for Finding Top SQL by DB Time	
		Create
Select	Name	Definition
	No filters have been set	

b. Click Create.

The Create Filter for Finding page appears.

Create Filter for Finding	g: Top SQL by DB Time
Name	
Active Sessions <	Specify the filter criteria in terms of Active Sessions for this finding. The finding will be filtered for a future ADDM run if the number of Active Sessions for this finding is less than the filter criteria specified above.
% Active Sessions <	Specify the filter criteria in terms of percentage Active Sessions for this finding. The finding will be filtered for a future ADDM run if the number o Active Sessions for this finding is less than the above filter criteria.

- c. In the Name field, enter a name for the ADDM filter.
- **d.** In the **Active Sessions** field, specify the filter criteria, in terms of the number of active sessions, for this finding.

The ADDM finding will be filtered for future ADDM runs if the number of active sessions for this finding is less than the specified filter criteria.

e. In the % **Active Sessions** field, specify the filter criteria, in terms of percentage of active sessions, for this finding.

The ADDM finding will be filtered for future ADDM runs if the number of active sessions for this finding is less than the specified filter criteria.

- f. Click OK.
- 7. Perform the required action of a chosen recommendation.

Depending on the type of action you choose to perform, various buttons may be available, such as **Implement** or **Run Advisor Now**. These buttons enable you to implement the recommendation immediately with only a single mouse click.

In this example, the simplest solution is to click **Run Advisor Now** to immediately run a SQL Tuning Advisor task on the SQL statement.

See Also:

Chapter 10, "Tuning SQL Statements"

Viewing Snapshot Statistics

You can view the data contained in snapshots taken by AWR using Enterprise Manager. Typically, it is not necessary to review snapshot data because it consists primarily of raw statistics. Instead, you should rely on ADDM, which analyzes these statistics to identify performance problems. Snapshot statistics should be used primarily by advanced users, or by DBAs who are accustomed to using Statspack for performance analysis.

To view snapshot statistics:

1. From the Database Home page, click **Performance**.

The Performance page appears.

2. Under Additional Monitoring Links, click Snapshots.

The Snapshots page appears with a list of the most recent snapshots.

3. To view the statistics gathered in a snapshot, click the **ID** link of the snapshot you want to view.

The Snapshot Details appears, showing the Details subpage.

Beginning Snapshot ID 161 Beginning Snapshot Capture Jan 30, 2007 10:00:19 Time AM	Ending Si Ending Snapsh	Time AM	0, 2007 10:30:20
		🛇 Previou	s 1-27 of 27 Next 🛞
Name Δ	Value	Per Second	Per Transaction
DB cpu (seconds)	0.00	0.00	0.00
DB time (seconds)	7,820.36	4.34	11.62
db block changes	59,013.00	32.78	87.69
execute count	35,518.00	19.73	52.78
global cache cr block receive time (seconds)	0.00	0.00	0.00
global cache cr blocks received	0.00	0.00	0.00
global cache current block receive time (seconds)	0.00	0.00	0.00
global cache current blocks received	0.00	0.00	0.00
global cache get time (seconds)	0.00	0.00	0.00
global cache gets	0.00	0.00	0.00
opened cursors cumulative	33,061.00	18.37	49.12
parse count (total)	19,007.00	10.56	28.24
parse time cpu (seconds)	0.46	0.00	0.00
parse time elapsed (seconds)	0.54	0.00	0.00
physical reads	1,232.00	0.68	1.83
physical writes	1,037.00	0.58	1.54
redo size (KB)	9,274.00	5.15	13.78
session cursor cache hits	29,220.00	16.23	43.42
session logical reads	167,958.00	93.31	249.57
sql execute cpu time (seconds)	0.00	0.00	0.00
sql execute elapsed time (seconds)	0.00	0.00	0.00
user calls	7,786.00	4.33	11.57
user commits	545.00	0.30	0.81
user rollbacks	128.00	0.07	0.19
workarea executions - multipass	0.00	0.00	0.00
workarea executions - onepass	0.00	0.00	0.00
workarea executions - optimal	0.00	0.00	0.00

In this example, statistics gathered from the previous snapshot (snapshot 161) to the selected snapshot (snapshot 162) are displayed.

4. To view a Workload Repository report of the statistics, click Report.

The Workload Repository report appears.

5. Optionally, click **Save to File** to save the report for later access.

See Also:

Chapter 8, "Resolving Performance Degradation Over Time"

Monitoring Real-Time Database Performance

The Performance page in Oracle Enterprise Manager (Enterprise Manager) displays information in three sections that you can use to assess the overall performance of the database in real time.

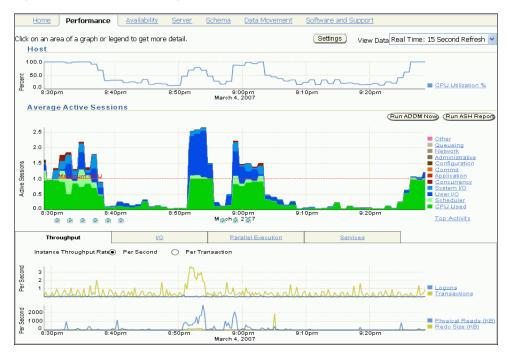


Figure 4–1 Performance Page

Typically, you should use the automatic diagnostic feature of Automatic Database Diagnostic Monitor (ADDM) to identify performance problems with the database, as described in Chapter 3, "Automatic Database Performance Monitoring". In some cases, however, you may want to monitor the database performance in real time to identify performance problems as they happen. For example, ADDM performs its analysis after each Automatic Workload Repository (AWR) snapshot, which by default is once every hour. However, if you notice a sudden spike in database activity on the Performance page, then you may want to investigate the incident before the next ADDM analysis.

By drilling down to appropriate pages from the Performance page, you can identify performance problems with the database in real time. If you find a performance problem, then you can choose to run ADDM manually to analyze it immediately, without having to wait until the next ADDM analysis. To learn how to run ADDM manually to analyze performance in real time, see "Manually Running ADDM to Analyze Current Database Performance" on page 6-1.

This chapter contains the following sections:

- Monitoring User Activity
- Monitoring Instance Activity
- Monitoring Host Activity
- Customizing the Database Performance Page

Monitoring User Activity

The Average Active Sessions chart of the Performance page shows potential problems inside the database, including how much CPU users are consuming. The wait classes show how much of the database activity is consumed by waiting for a resource such as disk I/O.

Figure 4–2 Monitoring User Activity



By following the performance method explained in Chapter 2, "Oracle Database Performance Method", you can drill down from the charts to identify the cause of instance-related performance issues and resolve them.

To monitor user activity:

1. From the Database Home page, click **Performance**.

The Performance page appears.

2. Locate the spikes in the Average Active Sessions chart.

When the CPU Used value reaches the Maximum CPU line (shown as a dotted line), the database instance is running at 100 percent of CPU time on the host system.

All other values in the chart represent users waiting and contention for resources, which are categorized by wait classes in the legend. Values that use a larger block of active sessions represent bottlenecks caused by a particular wait class, as indicated by the corresponding color in the legend.

In the chart shown in Figure 4–2 on page 4-2, the largest block of activity appears in green and corresponds to the CPU Used wait class as described in the legend.

3. To identify each wait class, move your cursor over the block in the Average Active Sessions chart corresponding to the class.

The corresponding wait class is highlighted in the chart legend.

4. Click the largest block of color on the chart or its corresponding wait class in the legend to drill down to the wait class with the most active sessions.

If you click **CPU Used**, then the Active Sessions Working page for the wait class appears. If you click a different wait class, such as **User I/O**, then the Active Sessions Waiting page appears.

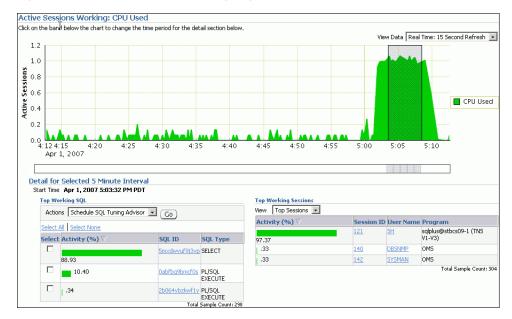


Figure 4–3 Active Sessions Working page

The Active Sessions Working page shows a 1-hour time line. Details for each wait class are shown in 5-minute intervals under Detail for Selected 5 Minute Interval.

You can view the details of wait classes in different dimensions by proceeding to one of the following sections:

- "Monitoring Top SQL" on page 4-4
- "Monitoring Top Sessions" on page 4-5
- "Monitoring Top Services" on page 4-6
- "Monitoring Top Modules" on page 4-7
- "Monitoring Top Actions" on page 4-7
- "Monitoring Top Clients" on page 4-8
- "Monitoring Top PL/SQL" on page 4-9
- "Monitoring Top Files" on page 4-9
- "Monitoring Top Objects" on page 4-10
- **5.** To change the time selected interval, move the slider below the chart to a different interval.

The information contained in the Detail for Selected 5 Minute Interval section is automatically updated to display the selected time period.

In the example shown in Figure 4–3, the 5 -minute interval from 5:03 to 5:08 is selected for the CPU Used wait class.

6. If you discover a performance problem, then you can attempt to resolve it in real time. On the Performance page, do one of the following:

• Click a snapshot below the chart that corresponds to the time when the performance problem occurred to run ADDM for that time period.

For information about ADDM analysis, see "Reviewing the Automatic Database Diagnostic Monitor Analysis" on page 3-7.

• Create a snapshot manually by clicking **Run ADDM Now**.

For information about creating snapshots manually, see "Creating Snapshots" on page 3-5. For information about running ADDM manually, see "Manually Running ADDM to Analyze Current Database Performance" on page 6-1.

 Click **Run ASH Report** to create an ASH report to analyze transient performance problems that last for only a short period of time.

For information about ASH reports, see "Active Session History Reports" on page 7-3.

Monitoring Top SQL

On the Active Sessions Working page, the Top Working SQL table shows the database activity for actively running SQL statements that are consuming CPU resources. The Activity (%) column shows the percentage of this activity consumed by each SQL statement. If one or several SQL statements are consuming a majority of the activity, then you should investigate them.

Figure 4–4 Monitoring Top SQL

Top Wo	Top Working SQL					
Actions	Actions Schedule SQL Tuning Advisor 🔽 🕝					
Select A	Select All Select None					
Select /	Activity (%) ∇	SQL ID	SQL Type			
	90.58	05b6pvb81dg8b	SELECT			
	1.81	bvf3fxv3hatw7	SELECT			
	1.09	<u>2b064ybzkwf1y</u>	PL/SQL EXECUTE			
	1.09	<u>gxxa073u093s4</u>	SELECT			
	1.09	fyddnrs5cctsu	SELECT			
	.72	gxxa073u093s4	SELECT			
	.72	40u9awsftkwn4	SELECT			
		Total Sa	ample Count: 276			

In the example shown in Figure 4–4, the SELECT statement is consuming over 90% of database activity and should be investigated.

To monitor the top working SQL statements:

1. On the Performance page, in the Average Active Sessions chart, click the CPU block on the chart or its corresponding wait class in the legend.

The Active Sessions Working page appears.

2. Under Detail for Selected 5 Minute Interval, click the **SQL ID** link of the most active SQL statement in the Top Working SQL table.

The SQL Details page appears.

For SQL statements that are using the majority of the wait time, use SQL Tuning Advisor or create a SQL Tuning Set to tune the problematic SQL statements.

See Also:

- "Viewing Details of SQL Statements" on page 9-4
- "Tuning SQL Statements Using SQL Tuning Advisor" on page 10-2

Monitoring Top Sessions

On the Active Sessions Working page, the Top Working Sessions table displays the top sessions waiting for the corresponding wait class during the selected time period. Sessions represent specific user connections to the database through a user process.

Figure 4–5 Monitoring Top Sessions

View 🛛 Top Sessions 💌			
Activity (%) 🗸	Session II	User Nam	e Program
	124	<u>SH</u>	sqlplus@stbcs09-1 (TNS V1-V3)
96.42			(
.98	<u>121</u>	DBSNMP	OMS
.33	135	SYSMAN	OMS

A session lasts from the time the user connects to the database until the time the user disconnects or exits the database application. For example, when a user starts SQL*Plus, the user must provide a valid database user name and password to establish a session. If a single session is using the majority of the wait time, then you should investigate it.

To monitor the top working sessions:

1. On the Performance page, in the Average Active Sessions chart, click the CPU Used block on the chart or its corresponding wait class in the legend.

The Active Sessions Working page appears.

2. Under Detail for Selected 5 Minute Interval, click the **Session ID** link of the session consuming the most database activity.

The Session Details page appears.

This page contains information such as session activity, session statistics, open cursors, blocking sessions, wait events, and parallel SQL for the selected session.

In the example shown in Figure 4–5, the SQL*Plus session for user sh is consuming over 96% of database activity and should be investigated.



Figure 4–6 Viewing Session Details

In this example, because the session is consuming 100 percent of database activity, consider ending the session by clicking **Kill Session**, and proceeding to tune the SQL statement that this session is running.

See Also:

Chapter 10, "Tuning SQL Statements"

Monitoring Top Services

The Top Services table displays the top services waiting for the corresponding wait event during the selected time period.

Services represent groups of applications with common attributes, service-level thresholds, and priorities. For example, the SYS\$USERS service is the default service name used when a user session is established without explicitly identifying its service name. The SYS\$BACKGROUND service consists of all Oracle Database background processes. If a single service is using the majority of the wait time, then you should investigate it.

To monitor a service:

1. On the Performance page, in the Average Active Sessions chart, click a block on the chart or its corresponding wait class in the legend.

The Active Sessions Working page appears.

2. Under Detail for Selected 5 Minute Interval, select **Top Services** from the View list.

The Top Services table appears.

Figure 4–7	Monitoring	Top Services
------------	------------	---------------------

Top S	ervices	
View	Top Services 💌	
Activ	rity (%) ∇	Service
	97.32	2 <u>SYS\$USERS</u>
2.01		orcl111
.67		SYS\$BACKGROUND
		Total Sample Count: 298

In the example shown in Figure 4–7, the SYS\$USERS service is consuming 97.32% of database activity. This service corresponds to the SQL*Plus session for user sh shown in Figure 4–5.

3. Click the Service link of the most active service.

The Service page appears.

This page contains information about the modules, activity, and statistics for the selected service.

Monitoring Top Modules

The Top Modules table displays the top modules waiting for the corresponding wait event during the selected time period.

Modules represent the applications that set the service name as part of the workload definition. For example, the DBMS_SCHEDULER module may assign jobs that run within the SYS\$BACKGROUND service. If a single module is using the majority of the wait time, then it should be investigated.

To monitor a module:

1. On the Performance page, in the Average Active Sessions chart, click a block on the chart or its corresponding wait class in the legend.

The Active Sessions Working page appears.

2. Under Detail for Selected 5 Minute Interval, select **Top Modules** from the View list.

The Top Modules table appears.

Figure 4–8 Monitoring Top Modules

Top Modules										
View Top Modules 💌										
Activity (%) ▽	Service	Module								
95.19	SYS\$USERS	<u>SQL*Plus</u>								
1.92	SYS\$USERS	Realtime Connection								
1.28	SYS\$USERS									
1.28	SYS\$USERS	emagent.exe								
.32	SYS\$USERS	DBMS SCHEDULER								
		Total Sample Count: 312								

3. Click the **Module** link of the module that is showing the highest percentage of activity.

The Module page appears.

This page contains information about the actions, activity, and statistics for the selected module.

In the example shown in Figure 4–8, the SQL*Plus module is consuming over 95% of database activity and should be investigated. As shown in Figure 4–5, the SQL*Plus session for user sh is consuming a huge percentage of database activity.

Monitoring Top Actions

The Top Actions table displays the top actions waiting for the corresponding wait event during the selected time period.

Actions represent the jobs that are performed by a module. For example, the DBMS_SCHEDULER module can run the GATHER_STATS_JOB action to gather statistics on all database objects. If a single action is using the majority of the wait time, then you should investigate it.

To monitor an action:

1. On the Performance page, in the Average Active Sessions chart, click a block on the chart or its corresponding wait class in the legend.

The Active Sessions Working page appears.

2. Under Detail for Selected 5 Minute Interval, select Top Actions from the View list.

The Top Actions table appears.

Figure 4–9 Monitoring Top Actions

Activity (%) 🗸 👘	Service	Module	Action		
	SYS\$USERS	SQL*Plus	SALES INFO		
96.04					
1.08	SYS\$USERS				
.72	SYS\$BACKGROUND				
.72	emdc	OEM.SystemPool			
.36	emdc	OEM.CacheModeWaitPool			
.36	SYS\$USERS	EM PING	AGENT STATUS MARKE		
.36	SYS\$USERS	emagent@stbcs09-1 (TNS V1-V3)			

3. Click the **Action** link of the most active action.

The Action page appears.

This page contains statistics for the selected action.

In the example shown in Figure 4–9, the action associated with the SQL*Plus module and SALES_INFO action is consuming 96% of the database activity. This information is consistent with Figure 4–5, which shows that the SQL*Plus session for user sh is consuming over 96% of database activity.

Monitoring Top Clients

The Top Clients table displays the top clients waiting for the corresponding wait event during the selected time period. A client can be a Web browser or any end-user process that initiates requests for an operation to be performed on the database. If a single client is using the majority of the wait time, then you should investigate it.

To monitor a client:

1. On the Performance page, in the Average Active Sessions chart, click a block on the chart or its corresponding wait class in the legend.

The Active Sessions Working page appears.

2. Under Detail for Selected 5 Minute Interval, select **Top Clients** from the View list.

The Top Clients table appears.

Figure 4–10	Monitoring	Top Clients
-------------	------------	-------------

Top Clients		
View Top Clients	*	
Activity (%) 🗸		Client ID
		SYS@130.35.182.5@Mozilla/5.0 (Windows; U; Windows NT 5.1;
		en-US Total Sample Count: 1

3. Click the Client ID link of the most active client.

The Clients page appears.

This page contains statistics for the selected user process.

Monitoring Top PL/SQL

The Top PL/SQL table displays the top PL/SQL subprograms waiting for the corresponding wait event during the selected time period. If a single PL/SQL subprogram is using the majority of the wait time, then you should investigate it.

To monitor a PL/SQL subprogram:

1. On the Performance page, in the Average Active Sessions chart, click a block on the chart or its corresponding wait class in the legend.

The Active Sessions Working page appears.

2. Under Detail for Selected 5 Minute Interval, select **Top PL/SQL** from the View list.

The Top PL/SQL table appears.

Figure 4–11 Monitoring Top PL/SQL

Top P	L/SQL		
View	Top PL/SQL	*	
Activ	ity (%) ⊽		PL/SQL Subprogram
		100.00	SYSMAN.MGMT_JOB_EXEC_UPDATE
			Total Sample Count: 1

3. Click the PL/SQL Subprogram link of the most active subprogram.

The PL/SQL Subprogram page appears.

This page contains statistics for the selected subprogram.

In Figure 4–11, the SYSMAN.MGMT_JOB_EXEC_UPDATE subprogram is consuming 100% of database activity.

Monitoring Top Files

The Top Files table displays the average wait time for specific files during the selected time period. This data is available from the Active Sessions Waiting: User I/O page.

To monitor a file:

1. On the Performance page, in the Average Active Sessions chart, click the User I/O block on the chart or its corresponding wait class in the legend.

The Active Sessions Waiting: User I/O page appears.

 Under Detail for Selected 5 Minute Interval, select Top Files from the View list. The Top Files table appears.

Figure 4–12 Monitoring Top Files

Top Files View Top Files 💌			
Activity (%)	Name	Tablespace	Average Wait Time (ms)
100.00	/ade/lasacle/dbs/t_db1.	f SYSTEM	322
			Total Sample Count: 3

3. Click the Tablespace link of the file with the highest average wait time.

The View Tablespace page appears.

In the example shown in Figure 4–12, the wait times are all associated with I/O for the file in the SYSTEM tablespace.

Monitoring Top Objects

The Top Objects table displays the top database objects waiting for the corresponding wait event during the selected time period. This data is available from the Active Sessions Waiting: User I/O page.

To monitor an object:

1. On the Performance page, in the Average Active Sessions chart, click the User I/O block on the chart or its corresponding wait class in the legend.

The Active Sessions Waiting: User I/O page appears.

2. Under Detail for Selected 5 Minute Interval, select Top Objects from the View list.

The Top Objects table appears.

Figure 4–13 Monitoring Top Objects

Top Objects								
View Top Objects 💌								
Activity (%) 🗸	Object Name	Object Type						
100.00	SYS.I SYSAUTH1	INDEX						
Total Sample Count: 1								

3. Click the Object Name link of the object with the highest average wait time.

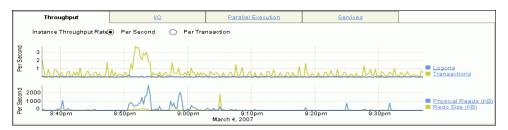
The View page for the object appears.

This example in Figure 4–13 shows that all the waits are for the SYS.I_SYSAUTH1 index. Given the information in Figure 4–4 and Figure 4–5, you can conclude that the performance problem is caused by the SELECT statement executed by user sh, which is waiting for access to the SYS.I_SYSAUTH1 index.

Monitoring Instance Activity

In the Average Active Sessions section of the Performance page, you can use the Throughput, I/O, Parallel Execution, and Services charts to monitor database instance activity. As explained in "Customizing the Database Performance Page" on page 4-27, you can also customize the Performance page so that the most useful charts are displayed by default.

Figure 4–14 M	onitoring In	istance Activit	y
---------------	--------------	-----------------	---



You can use the instance activity charts to perform the following tasks:

- Monitoring Throughput
- Monitoring I/O

- Monitoring Parallel Execution
- Monitoring Services

Monitoring Throughput

The Throughput charts show any contention that appears in the Average Active Sessions chart. The charts indicate how much work the database is performing for the user. The Throughput charts on the Performance page display:

- Number of physical reads, redo size, logons, and transactions per second
- Number of physical reads and redo size per transaction

Figure 4–15 Monitoring Throughput

Throughput	1/0	I/O Parallel Execution Services					
Instance Throughput Rate	e 🖲 Per Second 🔵 Per	Transactior					
달2000 중1500 생1000 월 500			Mar MMM	M-M- ■ Logons			
1000			Mulm-unn	Transactions			
§ 500	Dam 10:20am	10:30am 10:40a April 3, 2007	m 10:50am	Physical Reads (KB)			

Compare the peaks on the Throughput charts with the peaks on the Average Active Sessions chart. If the Average Active Sessions chart displays a large number of sessions waiting, indicating internal contention, but throughput is high, then the situation may be acceptable. The database is probably also performing efficiently if internal contention is low but throughput is high. However, if internal contention is high but throughput is low, then consider tuning the database.

To monitor throughput:

1. From the Database Home page, click **Performance**.

The Performance page appears.

2. In the instance activity chart, click Throughput.

The Throughput charts are shown with **Instance Throughput Rate** set to the default value of **Per Second**. You can select **Per Transaction** to show the throughput rate per transaction.

In the example in shown in Figure 4–15, the number of transactions and physical reads per second spiked at around 10:45 a.m. The number of transactions per second has remained between 1000 and 2000 for 25 minutes. The physical reads have remained between 500 and 1000 KB per second.

3. To view the top consumers for each type of activity, click the corresponding link in the legend.

The Top Consumers page appears. This page shows the top sessions for the selected activity.

			CPU	PGA												
Select				Memory (bytes)						user commits $ abla$	Status	Program	OS PID	Machine	05 User	SQL Trace
œ	<u>124</u>	SYS	545	5232788	1	70762	0	1803	0	13532	ACTIVE	sqlplus@stbcs09-1 (TNS V1-V3)	26374	stbcs09-1	lashdown	DISABLEI
0	<u>157</u>	CJQ0	0	2684008	0	8	0	0	0	0	ACTIVE	oracle@stbcs09-1 (CJQ0)	11056	stbcs09-1	lashdown	DISABLEI
0	<u>130</u>	W000	0	5143396	0	0	0	0	0	0	ACTIVE	oracle@stbcs09-1 (W000)	26761	stbcs09-1	lashdown	DISABLEI
0	<u>165</u>	PSPO	0	602028	0	0	0	0	0	0	ACTIVE	oracle@stbcs09-1 (PSP0)	11014	stbcs09-1	lashdown	DISABLEI
0	<u>160</u>	SMON	0	3010464	0	0	0	0	0	0	ACTIVE	oracle@stbcs09-1 (SMON)	11052	stbcs09-1	lashdown	DISABLEI
0	<u>148</u>	DBSNMP	7	2400192	0	3	0	5	0	0	ACTIVE	emagent@stbcs09-1 (TNS V1-V3)	26283	stbcs09-1	lashdown	DISABLED
0	<u>159</u>	СКРТ	0	1164332	4	0	0	0	0	0	ACTIVE	oracle@stbcs09-1 (CKPT)	11048	stbcs09-1	lashdown	DISABLE

In this example, a SQL*Plus session created by operating system user lashdown is responsible for the increase in database throughput.

4. Select any session and click View to obtain more information.

After you analyze the information, you can choose to end the session by clicking **Kill Session**, or return to the Performance page.

Monitoring I/O

The I/O charts show I/O statistics collected from all database clients. The I/O wait time for a database process represents the amount of time that the process could have been doing useful work if a pending I/O had completed. Oracle Database captures the I/O wait times for all important I/O components in a uniform fashion so that every I/O wait by any Oracle process can be deduced from the I/O statistics.

Figure 4–16 Monitoring I/O



The Latency for Synchronous Single Block Reads chart shows the total perceived I/O latency for a block read, which is the time difference between when an I/O is issued and when it is processed by the database. Most systems are performing satisfactorily if

latency is fewer than 10 milliseconds. This type of I/O request is the best indicator of I/O performance for the following reasons:

- Write operations may exhibit good performance because of write caches in storage.
- Because multiblock I/O requests have varying sizes, they can take different amounts of time.
- The latency of asynchronous I/O requests does not represent the full I/O wait time.

The other charts shown depend on your selection for **I/O Breakdown**, as described in the following sections:

- Monitoring I/O by Function
- Monitoring I/O by Type
- Monitoring I/O by Consumer Group

Monitoring I/O by Function

The I/O Function charts determine I/O usage level by application or job. The component-level statistics give a detailed view of the I/O bandwidth usage, which you can then use in scheduling jobs and I/O provisioning. The component-level statistics fall in the following categories:

Background type

This category includes ARCH, LGWR, and DBWR.

Activity

This category includes XML DB, Streams AQ, Data Pump, Recovery, and RMAN.

I/O type

The category includes Direct Write, which is a write issued by a foreground process that is not from the buffer cache; Direct Read, which is a physical I/O from a datafile that bypasses the buffer cache and reads the data block directly into process-private memory; and Buffer Cache Read.

Others

This category includes I/Os such as control file I/Os.

To monitor I/O by function:

1. From the Database Home page, click **Performance**.

The Performance page appears.

2. In the instance activity chart, click I/O.

The I/O Megabytes per Second and I/O Requests per Second charts appear.

3. For I/O Breakdown, select I/O Function.

The I/O Megabytes per Second by I/O Function and I/O Requests per Second by I/O Function charts appear.

The example in Figure 4–16 shows that a significant amount of I/O is being performed by the log writer. The log writer activity peaked at around 550 I/O requests per second.

4. Click the largest block of color on the chart or its corresponding function in the legend to drill down to the function with the highest I/O rate.

The I/O Details page appears.

You can view real-time or historical data for details on I/O megabytes or I/O requests.

See Also:

• Oracle Database Concepts to learn about database background processes such as ARCH, LGWR, and DBWR

Monitoring I/O by Type

The I/O Type charts enable you to monitor I/O by the types of read and write operations. Small I/Os are requests smaller than 128 KB and are typically single database block I/O operations. Large I/Os are requests greater than or equal to 128 KB. Large reads are generated by database operations such as table/index scans, direct data loads, backups, restores, and archiving.



If you are optimizing for low transaction times, then monitor the rate at which I/O requests are completed. Single-block performance is optimal when there is low I/O latency. High latencies typically indicate that the storage system is a bottleneck. Performance is negatively impacted by large I/O workloads.

If you are optimizing for large queries, such as in a data warehouse, then performance is dependent on the maximum throughput your storage system can achieve rather than the latency of the I/O requests. In this case, monitor the I/O megabytes per second rather than the synchronous single-block read latencies.

To monitor I/O by type:

1. From the Database Home page, click **Performance**.

The Performance page appears.

2. In the instance activity chart, click I/O.

The I/O Megabytes per Second and I/O Requests per Second charts appear

3. For I/O Breakdown, select I/O Type.

The I/O Megabytes per Second by I/O Type and I/O Requests per Second by I/O Type charts appear.

In this example, the number of small writes per second increased to 550. These writes correspond to the log writer I/O requests shown in Figure 4–16.

4. Click the largest block of color on the chart or its corresponding function in the legend to drill down to the function with the highest I/O rate.

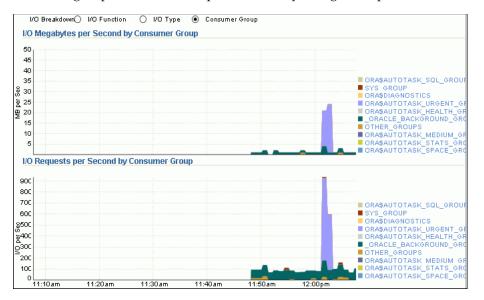
The I/O Details page appears.

You can view real-time or historical data for details on I/O megabytes or I/O requests.

Monitoring I/O by Consumer Group

When Oracle Database Resource Manager is enabled, the database collects I/O statistics for all consumer groups that are part of the currently enabled resource plan. The Consumer Group charts enable you to monitor I/O by consumer group.

A resource plan specifies how the resources are to be distributed among various users (resource consumer groups). Resource consumer groups let you group user sessions together by resource requirements. Note that the _ORACLE_BACKGROUND_GROUP_ consumer group contains I/O requests issued by background processes.



To monitor I/O requests by consumer group:

1. From the Database Home page, click **Performance**.

The Performance page appears.

2. In the instance activity chart, click I/O.

The I/O Megabytes per Second and I/O Requests per Second charts appear.

3. For I/O Breakdown, select Consumer Group.

The I/O Megabytes per Second by Consumer Group and I/O Requests per Second by Consumer Group charts appear.

Monitoring Parallel Execution

The Parallel Execution charts show system metrics related to parallel queries. A parallel query divides the work of executing a SQL statement across multiple processes. The charts show parallel queries that were waiting for a particular wait event that accounted for the highest percentages of sampled session activity.

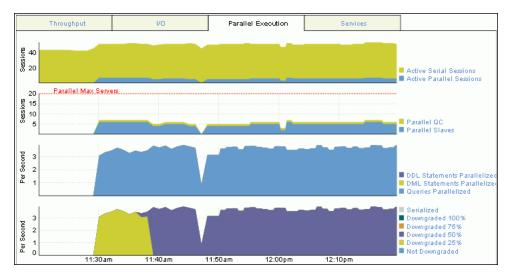


Figure 4–17 Monitoring Parallel Execution

To monitor parallel execution:

1. From the Database Home page, click **Performance**.

The Performance page appears.

2. In the instance activity chart, click **Parallel Execution**.

The Parallel Execution charts appear.

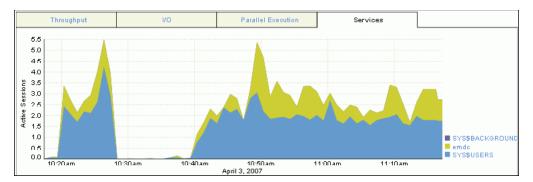
Two pairs of charts are displayed. The first pair of charts shows the number of sessions on the y-axis, whereas the second pair shows the per second rate on the y-axis.

In the example shown in Figure 4–17, query parallelization was active between 11:30 a.m. to 12:20 p.m.

Monitoring Services

The Services charts show services waiting for the corresponding wait event during the time period shown. Services represent groups of applications with common attributes, service-level thresholds, and priorities. For example, the SYS\$USERS service is the default service name used when a user session is established without explicitly identifying its service name. Only active services are shown.

Figure 4–18 Monitoring Services



To monitor services:

1. From the Database Home page, click **Performance**.

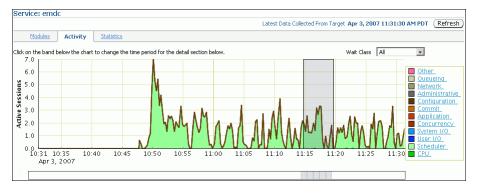
The Performance page appears.

2. In the instance activity chart, click Services.

The Services chart appears.

In Figure 4–18, the emdc and SYS\$USERS services have the greatest number of active sessions.

3. Click the largest block of color on the chart or its corresponding service in the legend to drill down to the service with the highest number of active sessions.



The Service page appears, showing the Activity subpage.

You can view real-time data showing the session load for all wait classes associated with the service.

Monitoring Host Activity

The Host chart on the Performance page displays utilization information about the system hosting the database.

Figure 4–19 Monitoring Host Activity



To determine if the host system has enough resources available to run the database, establish appropriate expectations for the amount of CPU, memory, and disk resources that your system should be using. You can then verify that the database is not consuming too many of these resources.

To view details about CPU, memory, and disk utilization:

1. From the Database Home page, click **Performance**.

The Performance page appears.

2. Click Load Average in the legend for the Host chart.

The Host page appears, showing the Performance subpage.

/iew Performance Summa	ry 💌			View Data Real	Time: Manual Refresh 🛛 👻
CPU Utilization		Memory Utilization	1	Disk I/O Utili	zation
100 50 9:36 9:45 10:00 Mar 4, 2007	10:15 10:30 10:45 CPU Utilization	100 50 9:36 9:45 10:00 Mar 4, 2007	10:15 10:30 10:45 Memory Utilization (%)	15 10 5 0 9:36 9:45 1 Mar 4, 2007	0:00 10:15 10:30 10:4
Curr	ent CPU Load 🗹 🧕	Swap U	Jtilization (%) 🗸 <u>19.3</u>	Longe	st Service Time (ms) 2.9
Additional Metric	s <u>CPU Usage</u>	Additional Metrics	aging Activity	Additional	Metrics Disk Activity
Processes Processes <u>69</u> Top 10 Processes					
View By CPU Utilizati	on (%) 💌				
Process ID	Command	CPU Utilization (%)	Resident Size	e (KB) 🛛 🕚	/irtual Size (KB) Owner
	iexplore	1	:	34,128	43,936 N/A
2320		0		88	1,476 N/A
2236		0		108	1,464 N/A
	SWSTRTR	0		128	428 N/A
	JRUNSVC	0		132	352 N/A
	SCARDSVR	0		132	880 N/A
	nmesrvc	0		168	292 N/A
	SMSS	0		172	172 N/A
	System	0		180	28 N/A
3396	MMTASK	0		196	636 N/A

Figure 4–20 Performance Summary

The Performance Summary view is shown by default. The Performance Summary view displays metric values for CPU utilization, memory utilization, disk I/O utilization, and the top 10 processes ordered by both CPU and memory utilization.

3. Determine whether sufficient resources are available and whether when your system is using too many resources.

Determine the amount of CPU, memory, and disk resources the database uses in the following scenarios:

- When your system is idle, or when little database and nondatabase activity exists
- At average workloads
- At peak workloads

Workload is an important factor when evaluating the level of resource utilization for your system. During peak workload hours, 90 percent utilization of a resource, such as a CPU with 10 percent idle and waiting time, can be acceptable. However, if your system shows high utilization at normal workload, then there is no room for additional workload. Use the procedures in the following sections to monitor the host activity for your database:

- Monitoring CPU Utilization
- Monitoring Memory Utilization
- Monitoring Disk I/O Utilization
- **4.** Set the appropriate threshold values for the performance metrics so the system can automatically generate alerts when these thresholds are exceeded.

For information about setting metric thresholds, see "Setting Metric Thresholds for Performance Alerts" on page 5-1.

Monitoring CPU Utilization

To address CPU problems, first establish appropriate expectations for the amount of CPU resources your system should be using. You can then determine whether sufficient CPU resources are available and recognize when your system is consuming too many resources. This section describes how to monitor CPU utilization.

To monitor CPU utilization:

1. From the Database Home page, click **Performance**.

The Performance page appears.

2. In the Host chart, click Load Average in the legend.

The Host page appears, showing the Performance subpage.

3. Select CPU Details from the View list.

The CPU Details view appears.

This view contains statistics about CPU utilization, I/O wait times, and load gathered over the last hour. The top 10 processes are also listed ordered by CPU utilization.

View CPU Details		View Data 🛛 Real Time: Manual Refresh 🛛 💌
CPU Utilization 100 50 11:49:12 AM 12:15 12:30 12:40 26 27 February 2007 CPU Utilization Current CPU Utilization Additional Metrics <u>All CPUs</u>	CPU I/O Wait	CPU Load
Top 10 Processes (ordered by CPU)		

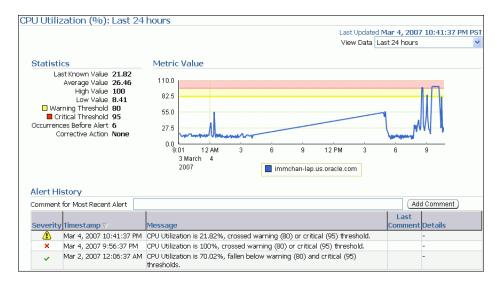
4. Verify the current CPU utilization using the CPU Utilization chart.

The CPU Utilization chart shows CPU utilization over the last hour. The current value is displayed below the chart. During standard workload hours, the value should not exceed the critical threshold.

5. Click CPU Utilization.

The CPU Utilization page appears.

This page contains CPU utilization statistics and related alerts generated over the last 24 hours.



In this example, the CPU utilization crossed the critical threshold value at 9:56 p.m., so an alert for CPU utilization is generated to indicate that a CPU performance problem may exist.

If you notice an unexpected spike in this value that is sustained through normal workload hours, then the CPU performance problem should be investigated.

6. Verify the current CPU I/O wait time using the CPU I/O Wait chart.

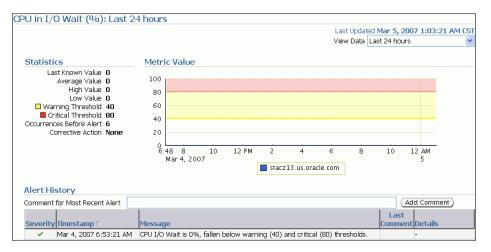
The CPU I/O Wait chart shows CPU I/O wait time over the last hour. The current value is displayed below the chart. During normal workload hours, the value of CPU I/O wait should not exceed the warning threshold.

CPU I/O wait represents the average number of jobs waiting for I/O during an interval.

7. Click CPU I/O Wait.

The CPU in I/O Wait page appears.

This page contains CPU I/O wait statistics and related alerts generated over the last 24 hours.



If you notice an unexpected spike in this value that is sustained through standard workload hours, then a CPU performance problem might exist.

8. Verify the current CPU load using the CPU Load chart.

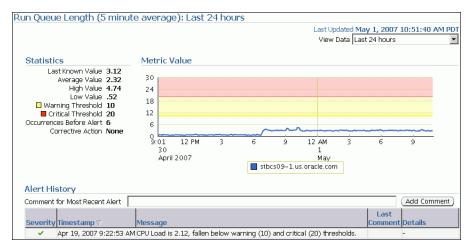
The CPU Load chart shows the CPU load over the last hour. The current value is displayed below the chart. During standard workload hours, the value of CPU load should not exceed the warning threshold.

CPU load represents the average number of processes waiting to be scheduled for CPU resources in the previous minute, or the level of CPU contention time over time.

9. Click CPU Load.

The Run Queue Length page appears.

This page contains CPU load statistics and related alerts generated over the last 24 hours.



In this example, the CPU load crossed the warning threshold, but it is still below the critical threshold, so an alert was not generated.

If you notice an unexpected spike in this value that is sustained through normal workload hours, then a CPU performance problem might exist.

10. Return to the CPU Details view of the Host Performance subpage and review the Top 10 Processes table.

If a process is consuming too much of the CPU utilization percentage, then this process should be investigated.

Command	CPU Utilization (%)	Resident Size (KB)	Virtual Size (KB)	Owner	Process ID
ORACLE	100	257,060	390,152	N/A	836
emdctl	0.97	22,340	20,836	N/A	5208
SWAGENT	0	84	1,872	N/A	240
cmd	0	88	1,476	N/A	2320
cmd	0	108	1,464	N/A	2236
SWSOC	0	116	2,148	N/A	348
SWSTRTR	0	128	428	N/A	248
SCARDSVR	0	132	880	N/A	1976
JRUNSVC	0	132	352	N/A	200
MMTASK	0	140	636	N/A	3396

In this example, the database is consuming 100 percent of CPU utilization. Therefore, the database is the likely source of a potential CPU performance problem and should be investigated.

11. If a CPU performance problem is identified, then you can try to resolve the issue by doing the following:

- Using Oracle Database Resource Manager to reduce the impact of peak-load-use patterns by prioritizing CPU resource allocation
- Avoiding running too many processes that use a lot of CPU
- Increasing hardware capacity, including changing the system architecture

See Also:

- Oracle Database Performance Tuning Guide for information about resolving CPU issues
- Oracle Database Administrator's Guide for information about Oracle Database Resource Manager

Monitoring Memory Utilization

Operating system performance issues commonly involve process management, memory management, and scheduling. This section describes how to monitor memory utilization and identify problems such as paging and swapping.

To monitor memory utilization:

1. From the Database Home page, click **Performance**.

The Performance page appears.

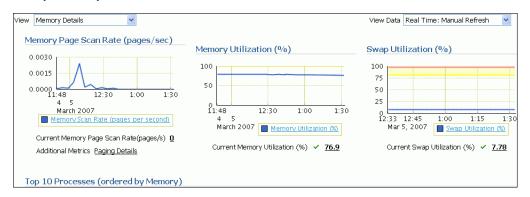
2. In the Host chart, click Load Average in the legend.

The Host page appears, showing the Performance subpage.

3. Select Memory Details from the View list.

The Memory Details view of the Performance subpage appears.

This view contains statistics about memory utilization, page scan rates, and swap utilization gathered over the last hour. The top 10 processes are also listed ordered by memory utilization.



4. Verify the current memory page scan rate using the Memory Page Scan Rate chart.

The current value of the memory page scan rate is displayed below the chart. On UNIX-based systems, this value represents the number of pages scanned per second. On Microsoft Windows, this value represents the rate at which pages are read from or written to disk to resolve hard page faults. This value is a primary indicator of the kinds of faults that may be causing systemwide delays.

5. Click Memory Scan Rate.

The Memory Page Scan Rate page appears.

This page contains memory page scan rate statistics and related alerts over the last 24 hours.

Memory Page Scan Rate (pe	* second): Last 24 hours
Statistics Last Known Value 0 Average Value 0 High Value 02 Low Value 0 Warning Threshold Not Defin Critical Threshold Not Defin Occurrences Before Alert 6 Corrective Action None	Last Updated Mar 5, 2007 2:33:21 AM CST View Data Last 24 hours
Alert History	
Comment for Most Recent Alert	Add Comment
Severity Timestamp \bigtriangledown	Message Comment Details
	Page scan rate is 0.0021 /sec, fallen below warning (/sec) and critical (/sec) thresho

In this example, an alert is not generated because a threshold is not defined.

If you notice an unexpected spike in this value that is sustained through standard workload hours, then a memory performance problem might exist.

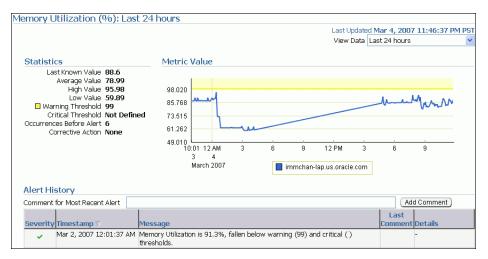
6. Verify the current memory utilization using the Memory Utilization chart.

The Memory Utilization chart shows how much memory is being used. The current value of memory utilization is displayed below the chart. During standard workload hours, the value should not exceed the warning threshold (shown in yellow).

7. Click Memory Utilization.

The Memory Utilization page appears.

This page contains memory utilization statistics and related alerts generated over the last 24 hours.



In this example, memory utilization is near, but does not exceed, the warning threshold value (99 percent), so an alert is not generated.

If you notice an unexpected spike in this value that is sustained through normal workload hours, then a memory performance problem might exist.

8. Verify current swap utilization using the Swap Utilization chart.

The Swap Utilization chart shows how much swap space is being used. The current value of swap utilization is displayed below the chart. During normal workload hours, the value should not exceed the warning threshold.

9. Click Swap Utilization.

The Swap Utilization page appears.

This page contains swap utilization statistics and related alerts generated over the last 24 hours.

Swap Utilization (%): Last 2	4 hours						
							7 11:51:37 PM PST
					View Data	Last 24 hours	~
Statistics	Metric Value						
Last Known Value 21.73 Average Value 13.96 High Value 21.73	110.0						
Low Value .98	82.5						
Warning Threshold 80 Critical Threshold 95	55.0						
Occurrences Before Alert 6 Corrective Action None	27.5						
	0.0 10:01 12 AM 3 4	3	6 9	12 PM	3	6 9	
	March 2007		📕 immchar	-lap.us.orac	le.com		
Alert History							
Comment for Most Recent Alert						A	dd Comment
Severity Timestamp \bigtriangledown	Message					Last Comment	t Details
Mar 2, 2007 12:01:37 AM	Swap Utilization is 10.15 thresholds.	%, fallen be	low warning (8	30) and criti	:al (95)		-

In this example, swap utilization is below the warning threshold, so an alert is not generated.

If you notice an unexpected spike in this value that is sustained through normal workload hours, then a memory performance problem might exist.

10. On the Memory Details view of the Host Performance subpage, verify the top processes in the Top 10 Processes table.

If a process is taking up too much memory, then this process should be investigated.

In this example, the database is consuming 82.86 percent of the CPU. The resident size is 259 MB, while the virtual size is 391 MB. Therefore, the database is the likely source of a potential memory problem and should be investigated.

op 10 Processes (ordered by Memory)									
Command	Resident Size (KB)	Virtual Size (KB)	CPU Utilization (%)	Owner	Process ID				
ORACLE	265,704	400,392	82.86	N/A	836				
java	169,544	192,704	0	N/A	2440				
iexplore	58,552	40,320	0	N/A	6060				
OUTLOOK	56,844	35,056	0	N/A	1896				
iexplore	34,688	44,020	0	N/A	3896				
sqlplus	29,964	26,440	0	N/A	5512				
FrameMaker	21,608	53,112	2.86	N/A	3340				
JRUN	21,472	40,788	9.52	N/A	292				
NOTEPAD	14,912	12,624	0	N/A	5568				
emagent	14,456	36,176	0	N/A	2500				

- **11.** If a memory performance problem is identified, you can attempt to resolve the issue by doing the following:
 - Using Automatic Memory Management to automatically manage and distribute memory between the System Global Area (SGA) and the aggregate program global area (PGA aggregate)

- Using the Memory Advisor to set SGA and PGA memory target values
- Using Automatic PGA Management to manage SQL memory execution
- Avoiding running too many processes that use a lot of memory
- Reducing paging or swapping
- Reducing the number of open cursors and hard parsing with cursor sharing

See Also:

- Oracle Database Administrator's Guide for information about using Automatic Memory Management
- Oracle Database 2 Day DBA for information about using the Memory Advisor
- Oracle Database Performance Tuning Guide for information about resolving memory issues

Monitoring Disk I/O Utilization

Because the database resides on a set of disks, the performance of the I/O subsystem is very important to the performance of the database. Important disk statistics include the disk I/Os per second and the length of the service times. These statistics show if the disk is performing optimally or if the disk is being overworked. This section describes how to monitor disk I/O utilization.

To monitor disk I/O utilization:

1. From the Database Home page, click **Performance**.

The Performance page appears.

2. In the Host chart, click Load Average in the legend.

The Host page appears, showing the Performance subpage.

3. Select Disk Details from the View list.

The Disk Details view appears.

This view contains disk I/O utilization and service time statistics gathered over the last hour, and the top disk devices ordered by busy percentage.

View	Disk Detai	s	*				View Data Re	al Time: Manual I	Refresh 🔽		
	Disk I/O Utilization Longest I/O service time (ms)										
1	20 10 0 9:36 10:00 10:30 11:00 11:30 12.4M Mar 4, 2007 5 11:11 11:20 11:30 11:40 11:50 12.4M Mar 4, 2007 5 10 Longest J/O service time (ms)										
Т	otal I/Os p	er second §	9.25			Longest Service	Time (ms) 3.9				
Т	Top Disk Devices (ordered by % Busy)										
6	status Dev	Busy ice (%)		Reads per second	Writes per second		Blocks Written (512 bytes per second)	Average Wait	Average Service Time (ms)		
	 OC: 	2.44	0.03	<u>3.33</u>	<u>5.36</u>	80.31	<u>104.94</u>	Q	<u>3.8</u>		
	🖌 1E:	0	0	0	<u>0</u>	Q	0	Q	0		

4. Verify the current disk I/O utilization using the Disk I/O Utilization chart.

The Disk I/O Utilization chart shows how many disk I/Os are being performed per second. The current value for total I/Os per second is displayed below the chart.

5. Click Total I/Os per Second.

The Total Disk I/O Per Second page appears.

This page contains disk utilization statistics and related alerts generated over the last 24 hours.



In this example, an alert is not generated because a threshold is not defined.

If you notice an unexpected spike in this value that is sustained through standard workload hours, then a disk I/O performance problem might exist and should be investigated.

6. Verify the current I/O service time using the Longest I/O Service Time chart.

The Longest I/O Service Time chart shows the longest service time for disk I/Os in milliseconds. The current value for longest I/O service time is displayed below the chart.

7. Click Longest I/O Service Time.

The Longest Service Time page appears.

This page contains I/O service time statistics and related alerts generated over the last 24 hours.

ongest Service Time (ms): Lasi	t 24 hours				ast Updated View Data			21:37 AM PST
Statistics	Metric Value							
Last Known Value 4 Average Value 28.88 High Value 510 Low Value 1.1 Warning Threshold Not Defined Critical Threshold Not Defined Occurrences Before Alert No data Corrective Action None	600 500 400 200 100 12:01 3 Mar 4, 2007	6	9 Immo	12 PM than-lap.us.c	3 racle.com	M6	9	12 AM 5
Alert History								
Severity Timestamp Me	ssage					La: Comn	st nent Deta	ils
(No alerts)								

In this example, an alert is not generated because a threshold is not defined.

If you notice an unexpected spike in this value that is sustained through normal workload hours, then a disk I/O performance problem might exist and should be investigated.

8. On the Disk Details page, verify the disk devices in the Top Disk Devices table.

If a particular disk is busy a high percentage of the time, then this disk should be investigated.

In this example, the drive that hosts Oracle Database (drive C) is only busy about 2.82 percent of the time, and there does not appear to be a disk performance problem.

Top Disk Devices (ordered by % Busy)											
Stat	us Device	Busy (%)	Average Outstanding Requests	Reads per			Blocks Written (512 bytes per second)	Average Wait	Average Service Time (ms)		
~	0 C:	2.82	0.06	<u>3.04</u>	5.14	120.85	<u>84.53</u>	<u>0</u>	7.3		
~	1 E:	0	<u>0</u>	0	Q	<u>0</u>	<u>0</u>	<u>0</u>	<u>0</u>		

- **9.** If a disk I/O performance problem is identified, you can attempt to resolve the problem by doing the following:
 - Using Automatic Storage Management (ASM) to manage database storage
 - Striping everything across every disk to distribute I/O
 - Moving files such as archived redo logs and online redo logs to separate disks
 - Storing required data in memory to reduce the number of physical I/Os

See Also:

Oracle Database Performance Tuning Guide for information about resolving disk I/O issues

Customizing the Database Performance Page

You can customize the Performance page so that it specifically addresses your requirements. As explained in "Monitoring Instance Activity" on page 4-10, you can specify which charts you want to appear by default in the Performance page, and how you want them to appear. You can also decide whether to include baseline values in the Throughput and Services charts.

Enterprise Manager stores persistent customization information for each user in the repository. Enterprise Manager retrieves the customization data when you access the Performance page and caches it for the remainder of the browser session until you change the settings.

To customize the Performance page:

1. From the Database Home page, click **Performance**.

The Performance page appears.

2. On the Performance page, click Settings.

The Performance Page Settings page appears.

Performance Page Set	Cancel) (OK)
You can change the settings for saved for this database and Ent	the performance page by setting chart defaults and baseline display options. The preferences will be
Detailed Chart Setting	ls
Default View	⊙ Throughput ○ I/O ○ Parallel Execution ○ Services
Throughput Chart Settings	• Per Second • • Per Transaction
I/O Chart Settings	I/O Function ○ I/O Type ○ Consumer Group
Baseline Display	
You may optionally show the	baseline values on performance page charts.
O not show the baseling	ne values
C Show the 99th percenti	le line using the system moving window baseline
C Show the 99th percenti Baseline Name AWR_E	le line using a static baseline with computed statistics BASELINE_040207 💌

- **3.** In the Detailed Chart Settings section, choose the defaults for display of the instance activity charts. Complete the following steps:
 - **a.** In **Default View**, select the instance activity chart to appear by default in the Average Active Session section.

See "Monitoring Instance Activity" on page 4-10 for a description of the Throughput, I/O, Parallel Execution, and Services charts.

b. In **Throughput Chart Settings**, select **Per Second** or **Per Transaction** as the default instance throughput rate to be displayed in the Throughput chart.

See "Monitoring Throughput" on page 4-11 to learn how to use the Throughput charts.

c. In **I/O Chart Settings**, select the default I/O breakdown to be displayed in the I/O chart.

See "Monitoring I/O" on page 4-12 to learn how to use the I/O charts.

- **4.** In the Baseline Display section, choose how Automatic Workload Repository (AWR) baselines will be displayed in the performance charts. Do one of the following:
 - Select **Do not show the baseline values** to prevent baselines from appearing.
 - Select Show the 99th percentile line using the system moving window baseline to specify a percentile to display for the Throughput and Services charts.

Note that the 99th percentile is a very high significance level.

• Select **Show the 99th percentile line using a static baseline with computed statistics** and then select a baseline name from the Baseline Name list.

Note that you can select only baselines that have undergone schedule statistics computation, as described in "Computing Threshold Statistics for Baselines" on page 8-6.

5. Click OK.

The Performance page reappears.

The charts are now displayed according to your customization settings.

Monitoring Performance Alerts

Oracle Database includes a built-in alerts infrastructure to notify you of impending problems with the database. By default, Oracle Database enables the following alerts:

- Tablespace Usage
- Snapshot Too Old
- Recovery Area Low on Free Space
- Resumable Session Suspended

For information about alerts and how to manage them, see Oracle Database 2 Day DBA.

In addition to these default alerts, you can use performance alerts to detect any unusual changes in database performance.

This chapter contains the following sections:

- Setting Metric Thresholds for Performance Alerts
- Responding to Alerts
- Clearing Alerts

Setting Metric Thresholds for Performance Alerts

A metric is defined as the rate of change in some cumulative statistic. This rate can be measured against a variety of units, including time, transactions, or database calls. For example, the number database calls per second is a metric.

Performance alerts are based on metrics that are performance-related. These alerts are either environment-dependent or application-dependent.

Environment-dependent performance alerts may not be relevant on all systems. For example, the AVERAGE_FILE_READ_TIME metric generates an alert when the average time to read a file exceeds the metric threshold. This alert may be useful on a system with only one disk. On a system with multiple disks, however, the alert may not be relevant because I/O processing is spread across the entire subsystem.

Application-dependent performance alerts are typically relevant on all systems. For example, the BLOCKED_USERS metric generates a performance alert when the number of users blocked by a particular session exceeds the metric threshold. This alert is relevant regardless of how the environment is configured.

To obtain the most relevant information from performance alerts, set the threshold values of performance metrics to values that represent desirable boundaries for your system. You can then fine-tune these values over time until an equilibrium is reached.

To set thresholds for performance metrics:

1. On the Database Home page, under Related Links, click **Metric and Policy Settings**.

The Metric and Policy Settings page appears, showing the Metric Thresholds subpage.

2. For each performance metric relevant for your system, click the Edit icon.

The Edit Advanced Settings page appears.

3. Follow the steps of the wizard to set the threshold value.

See Also:

- "Setting Metric Thresholds for Baselines" on page 8-7
- Oracle Database 2 Day DBA to learn how to set metric thresholds

Responding to Alerts

When an alert is generated by Oracle Database, it appears under Alerts on the Database Home page.

∀ Alerts									
Category All 🗸 🖸 Critical 🗙 1 Warning 🚯 1									
Severity \bigtriangledown	Category	Name	Impact	Message	Alert Triggered				
×		Generic Internal Error	PROCESS FAILURE	Internal error (3059024509) detected in /ade/kloute_sa0220/aracle/log/dlag/rdbms/sa0220/sa0220/alert/log.xml at time/line number: Mon Feb 26 04:44:39 2007/43205	Feb 26, 2007 4:45:25 AM				
1	Alert Log	Generic Alert Log Error		<u>ORA-error stack (00600/kcbzwb 41) logged</u> in /ade/kcloutie_sa0220/oracle/log/clbms/sa0220/sa0220/race/alert_sa0220.log.	Feb 26, 2007 4:50:34 AM				

Oracle Enterprise Manager enables you to configure alerts to be sent by means of e-mail, pager, or cellular phone text messaging.

To respond to an alert:

1. On the Database Home page, under Alerts, locate the alert that you want to investigate and click the **Message** link.

A page that contains further information about the alert appears.

- **2.** Do one of the following:
 - Follow the recommendations.
 - Run Automatic Database Diagnostic Monitor (ADDM) or another advisor to get more detailed diagnostics of the system or object behavior.

See Also:

 Oracle Database 2 Day DBA for information about how to configure the alert notification method

Clearing Alerts

Most alerts, such as the CPU Utilization alert, are cleared automatically when the cause of the problem disappears. However, other alerts, such as the Generic Alert Log Error alert, must be acknowledged.

After taking the necessary corrective measures, you can acknowledge an alert by clearing or purging it. Clearing an alert sends the alert to the Alert History, which can

be viewed from the Database Home page under Related Links. Purging an alert removes it from the Alert History.

To clear alerts:

1. On the Database Home page, under Diagnostic Summary, click the **Alert Log** link.

The Alert Log Errors page appears.

				Latest Data Collected From Target 2007-02-27 01:13:30 View Data All	l Days 🔽 🤇	Refresh
Alert	_og Entr	ries Cont	aining ORA-	Errors (Show Open Alerts) (Clear Every Open	Alert) (Purge Eve	n/ Alort
Clea	r)(Purge)			Act (Targe Loc	iy Acti
	All Select	<i>.</i>				
Select	Severity	Category	Time	Alert Log Error Stack	Alert Triggered	Lir Numbe
	⚠	Generic Alert Log Error	Feb 26, 2007 4:44:39 AM	ORA-00600: internal error code, arguments: [kcbzwb_4], [], [], [], [], [], [], [], [], [], [Feb 26, 2007 4:50:34 AM	2672
	~	Generic Alert Log Error	Feb 25, 2007 10:34:35 PM	ORA-12012: error on auto execute of job 63855 ORA-56708: Could not find any datafiles in managed pool with async i/o capability ORA-06512: at "SYS.DBMS_RMIN", line 416 ORA-06512: at "SYS.DBMS_RESOURCE_MANAGER", line 1099 ORA-06512: at line 5 Trace /ade/kloute_se0220/oracle/log/diag/rdbms/se0220/se0220/trace/se0220_j000_15587.trc File:		2669
	~	Generic Alert Log Error	Feb 25, 2007 10:26:04 PM	ORA-12012: error on auto execute of job 63854 ORA-29355: NULL or invalid MAX_LATENCY argument specified ORA-06512: at "SYS.DBMS_SYS_ERROR", line 86 ORA-06512: at "SYS.DBMS_RESOURCE_MANAGER", line 1096 ORA-06512: at line 5 Trace /ade/icloutie_se0220/oracle/log/diag/rdbms/se0220/se0220/trace/se0220_j000_14709.trc File:		2668
	*	Generic Alert Log Error	Feb 25, 2007 6:29:23 PM	ORA-56706: The specified Resource Manager plan is a subplan and cannot be set as a top-level plan Trace /ade/lcloutie_sa0220/oracle/log/diag/rdbms/sa0220/sa0220/trace/sa0220_dbrm_1725.trc File:		2666
	*	Generic Alert Log Error	Feb 25, 2007 5:55:41 PM	CREATE SMALLFILE TABLESPACE "LEO_TEST" DATAFILE '/ade//cloute_sa0220/oracle/dbs/leo_data_test1' SIZE 100M AUTOEXTEND ON NEXT 40M MAXSIZE UNLIMITED LOGGING EXTENT MANAGEMENT DICTIONARY DEFAULT NOCOMPRESS STORAGE (INITIAL 35M) ORA-1543 signalled during: CREATE SMALLFILE TABLESPACE "LEO_TEST" DATAFILE '/ade//cloutie_sa0220/oracle/dbs/leo_data_test1' SIZE 100M AUTOEXTEND ON NEXT 40M MAXSIZE UNLIMITED LOGGING EXTENT MANAGEMENT DICTIONARY DEFAULT NOCOMPRESS STORAGE (INITIAL 35M)		2660

- **2.** Do one of the following:
 - Select the alerts that you want to clear and click **Clear**.
 - To clear all open alerts, click Clear Every Open Alert.
- **3.** Do one of the following:
 - Select the alerts that you want to purge and click **Purge**.
 - To purge all alerts, click **Purge Every Alert**.

See Also:

Oracle Database 2 Day DBA to learn how to manage alerts

Part III

Reactive Database Tuning

Part III describes how to tune Oracle Database in response to a reported problem, such as when the user reports a performance problem with the database that needs to be tuned immediately.

This part contains the following chapters:

- Chapter 6, "Manual Database Performance Monitoring"
- Chapter 7, "Resolving Transient Performance Problems"
- Chapter 8, "Resolving Performance Degradation Over Time"

Manual Database Performance Monitoring

You can run the Automatic Database Diagnostic Monitor (ADDM) manually to monitor current and historical database performance. Typically, you use the automatic diagnostic feature of ADDM to identify performance problems with the database. As described in Chapter 3, "Automatic Database Performance Monitoring", ADDM runs once every hour by default. It is possible to configure ADDM to run more or less frequently. However, in some cases you may want to run ADDM manually.

One reason for running ADDM manually is to analyze a time period longer than one ADDM analysis period. For example, you may want to analyze database performance in a full workday by analyzing 8 consecutive hours of activity. One technique is to analyze each of the individual ADDM analyses within this 8-hour period. However, this approach may become complicated if performance problems exist for only part of the 8-hour period, because they may appear in only some of the individual ADDM analyses. Alternatively, you can run ADDM manually with a pair of Automatic Workload Repository (AWR) snapshots that encompass the 8-hour period. In this case, ADDM will identify the most critical performance problems in the entire time period.

This chapter contains the following sections:

- Manually Running ADDM to Analyze Current Database Performance
- Manually Running ADDM to Analyze Historical Database Performance
- Accessing Previous ADDM Results

Manually Running ADDM to Analyze Current Database Performance

By default, ADDM runs every hour to analyze snapshots taken by AWR during this period. In some cases you may notice performance degradation that did not exist in the previous ADDM analysis period, or a sudden spike in database activity on the Performance page, as described in Chapter 4, "Monitoring Real-Time Database Performance". If the next ADDM analysis is not scheduled to run for 30 minutes, then you may want to run ADDM manually to identify and resolve the performance problem.

When you run ADDM manually, a manual AWR snapshot is created automatically. This manual run may affect the ADDM run cycle. For example, if you scheduled ADDM to run hourly at the start of each hour and the last ADDM run was at 8:00 p.m., running ADDM manually at 8:30 p.m. will cause the next scheduled ADDM run to start at 9:30 p.m., not 9:00 p.m. All subsequent ADDM runs will continue on the new run cycle, occurring hourly at the half-hour instead of the start of each hour.

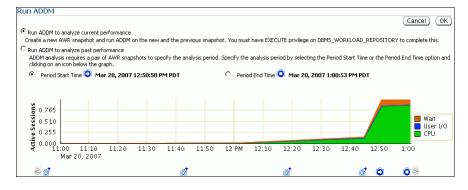
To analyze current database performance by manually running ADDM:

1. On the Database Home page, under Related Links, click Advisor Central.

The Advisor Central page appears.

2. Under Advisors, click **ADDM**.

The Run ADDM page appears.



In this example, CPU usage spiked in the last 10 minutes.

3. Select Run ADDM to analyze current instance performance and click OK.

The Confirmation page appears.

4. Click Yes.

The Processing: Run ADDM Now page appears while the database takes a new AWR snapshot.

An ADDM run occurs for the time period between the new and the previous snapshot. After ADDM completes the analysis, the Automatic Database Diagnostic Monitor (ADDM) page appears with the results.

Database Activity The icon selected below the graph identifies the ADDM analysis period. Click or	n a different icon to select a different analy	(Run ADDM) (Finding History
50 0.76 0.51 0.26 0.11:00 ▶ 11:10 11:20 11:30 11:40 11:50		Wait User I/O CPU Q
₩ 0.0 11:00 11:10 11:20 11:30 11:40 11:50 Mar 20, 2007	12 PM 12:10 12:20 1	12:30 12:40 12:50 1:00
⊗ Ø TIP For an explanation of the icons and symbols used in this page, see the	Con Key	9 9 19
ADDM Performance Analysis Task Name ADDM:1621828677_1_8	Time Range	Mar 20, 2007 1:08:01 PM to Mar 20, 2007 1:12:01 PM
Task Owner SYS Average Active Sessions 1	L Period:	Filters View Snapshots View Report Start Time Mar 20, 2007 1:00:54 PM PDT Period Duration 9J. (minutes)
Impact (%) \bigtriangledown	Finding	Occurrences (last 24 hrs)
100	CPU Usage	4 of 5
100	Top SQL by DB Time	<u>4 of 5</u>
>Informational Findings		

5. Click View Report.

The View Report page appears.

6. Optionally, click **Save to File** to save the results of the ADDM task in a report for later access.

See Also:

 "Reviewing the Automatic Database Diagnostic Monitor Analysis" on page 3-7

Manually Running ADDM to Analyze Historical Database Performance

You can run ADDM manually to analyze historical database performance by selecting a pair or range of AWR snapshots as the analysis period. This technique is useful when you have identified a time period in the past when database performance was poor.

In the Performance page, you can monitor historical performance by selecting **Historical** from the View Data list. In Historical view, you can monitor database performance in the past, up to the duration defined by the AWR retention period. If you notice performance degradation, then you can drill down to appropriate pages from the Performance page to identify historical performance problems with the database, as described in Chapter 4, "Monitoring Real-Time Database Performance". If you identify a problem, then you can run ADDM manually to analyze a particular time period.

To analyze historical database performance by manually running ADDM:

1. On the Database Home page, under Related Links, click Advisor Central.

The Advisor Central page appears.

2. Under Advisors, click ADDM.

The Run ADDM page appears.

- 3. Select Run ADDM to analyze past instance performance.
- **4.** Specify a time period for analysis by selecting a pair of AWR snapshots. Complete the following steps:
 - a. Select Period Start Time.
 - **b.** Below the chart for the starting snapshot, click the snapshot you want to use for the start time.

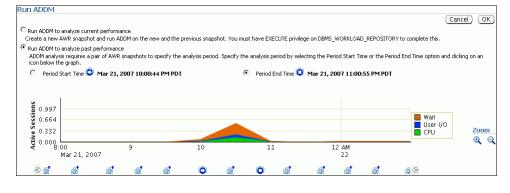
A play icon (displayed as an arrow) appears over the snapshot icon.

In this example, database activity peaked from 10 p.m. to 11 p.m., so the snapshot taken at 10:00 p.m. is selected for the start time.

- c. Select Period End Time.
- **d.** Below the chart for the ending snapshot, click the snapshot you want to use for the end time.

A stop icon (displayed as a square) appears over the snapshot icon.

In this example, the snapshot taken at 10:00 p.m. is selected.



5. Click OK.

After ADDM completes the analysis, the Automatic Database Diagnostic Monitor (ADDM) page appears with the results of the ADDM run.

1	below the graph	identifies the	ADDM analysis j	period. Click on	a different ic	on to select a	different analy	sis period.		(Run ADDM	L) (Find	ling Histo
1.0 0.66 0.33 0.0 9:00										Wa Use CPI	er I/O	Zoom E
	21, 2007	10		11		12 AM 22		1		2		
ഭീ	n	ß	3		r n	ñ	n	n	ീ	r 🔊		
DM Perforn	ame TASK_96	s i (End Time:M		00:55 PM)		×			Filters	o Mar 21, 2007 View Snapsho	ots) (Vi	ew Repo Period
DM Perforn Task N	nance Analysi	s i (End Time:M	ar 21, 2007 11:	00:55 PM) e Sessions 0.	3	•		art Time Mar	Filters	View Snapsho	nts) (Vie Du	ew Repo
DM Perforn	ame TASK_96	s · (End Time:M · YS	ar 21, 2007 11:	00:55 PM) e Sessions 0. Findir	3			art Time Mar	Filters	View Snapsho	nts) (Vie Du	ew Repo Period Iration 6
DM Perforn Task N	nance Analysi ame TASK_96 Task Owner S	s (End Time:M SYS	ar 21, 2007 11:	00:55 PM) e Sessions 0. Findir Under	3	Pool		art Time Mar	Filters	View Snapsho	nts) (Vie Du	ew Repo Period Iration 6
DM Perforn Task N	nance Analysi ame TASK_96 Task Owner 1	s (End Time:M. SYS 10	ar 21, 2007 11:	e Sessions 0. Findir Under Top SC	3 Ng Sized Shared I	<u>Pool</u>		art Time Mar	Filters) 21, 2007 10:0 ccurrences (la of 49	View Snapsho	nts) (Vie Du	ew Repo Period Iration 6
DM Perforn Task N	nance Analysi ame TASK_96 Task Owner 1 10 33	s (End Time:M. i¥5 0 .2 .7	ar 21, 2007 11:	e Sessions 0. Findir Under Top 50 "Sches	3 sized Shared I DL by DB Time	Pool 2 355		art Time Mar	Filters 21, 2007 10:0 currences (la of 49 of 49	View Snapsho	nts) (Vie Du	ew Repo Period Iration 6
DM Perforn Task N	nance Analysi ame TASK_96 Task Owner 1 10 37 13	s (End Time:M YS 0 2.2 1.7 .4	ar 21, 2007 11:	e Sessions 0. Findir Under Top 50 "Sches	3 sized Shared I DL by DB Time Juler" Wait Ck sized Buffer C	Pool 2 355		art Time Mar	(Filters) 21, 2007 10:0 ccurrences (la of 49 of 49 of 49	View Snapsho	nts) (Vie Du	ew Repo Period Iration 6
DM Perforn Task N	ame Analysi ame TASK_96 Task Owner 9 10 32 13 11	s (End Time:M iyy5 00 .2 .7 .4 3	ar 21, 2007 11:	00:55 PM) e Sessions 0. Findir Under Top 50 "Scher Under CPU U	3 sized Shared I DL by DB Time Juler" Wait Ck sized Buffer C	Pool 2 ass Cache		art Time Mar 2.1 10 2.1 2.1 10 2.1 10 2.1 10 10	Filters 21, 2007 10:0 courrences (la of 49 of 49 of 49 of 49	View Snapsho	nts) (Vie Du	ew Repo Period Iration 6

Figure 6–1 Analyzing Historical Database Performance

6. Click View Report.

The View Report page appears.

7. Optionally, click Save to File.

See Also:

 "Reviewing the Automatic Database Diagnostic Monitor Analysis" on page 3-7

Accessing Previous ADDM Results

If you ran ADDM manually to analyze current or historical database performance, the results are displayed on the Automatic Database Diagnostic Monitor (ADDM) page after the ADDM run has completed.

You can access the ADDM results at a later time, or access the ADDM results from previous executions.

To access the ADDM results:

1. On the Database Home page, under Related Links, click Advisor Central.

The Advisor Central page appears.

- **2.** Complete the following steps:
 - a. Under Advisor Tasks, select ADDM from the Advisory Type list.
 - **b.** Select the appropriate search criteria.

For example, you can select **All** in the Advisor Runs list to view all ADDM tasks.

c. Click Go.

visor T	asks					(Char	nge Default Pa	arameter
Search							-	
			isk name to filter the data that is			sults set.		
Advisor ADDM		Task Name	Advisor Runs		Status All	▼ (Go)		
1		eturos all unpercase matches her	inning with the string you entered. To	nin an	11		quote the search s	tring, You c
		l (%) in a double quoted string.	nining with the string you entered. To	Turr un	exact or case as	insidive matchy double	quote the search's	angi rou (
Results								
Viev	w Result)	Delete Actions Re-sched	ule 🔍 Go)			Previous 1-25	of 281 💌	Next 25
Select	Advisory Type	Name	Description	User	Status	Start Time ∇	Duration (seconds)	
۲	ADDM	ADDM:1620962274 1 283	ADDM auto run: snapshots [282, 283], instance 1, database id 1620962274	SYS	COMPLETED	Mar 19, 2007 7:50:52 PM	0	
0	ADDM	ADDM:1620962274_1_282	ADDM auto run: snapshots [281, 282], instance 1, database id 1620962274	SYS	COMPLETED	Mar 19, 2007 7:40:50 PM	1	
0	ADDM	ADDM:1620962274_1_281	ADDM auto run: snapshots [280, 281], instance 1, database id 1620962274	SYS	COMPLETED	Mar 19, 2007 7:30:47 PM	0	
0	ADDM	ADDM:1620962274_1_280	ADDM auto run: snapshots [279, 280], instance 1, database id 1620962274	SYS	COMPLETED	Mar 19, 2007 7:20:44 PM	0	
0	ADDM	ADDM:1620962274_1_279	ADDM auto run: snapshots [278, 279], instance 1, database id 1620962274	SYS	COMPLETED	Mar 19, 2007 7:10:41 PM	1	

The ADDM tasks are displayed under Results.

3. To view an ADDM result, select the desired ADDM task and click **View Result**.

The results from the selected ADDM task are shown in the Automatic Database Diagnostic Monitor (ADDM) page.

See Also:

 "Reviewing the Automatic Database Diagnostic Monitor Analysis" on page 3-7

Resolving Transient Performance Problems

Transient performance problems are short-lived and do not appear in the Automatic Database Diagnostic Monitor (ADDM) analysis. ADDM tries to report the most significant performance problems during an analysis period in terms of their effect on DB time. If a problem lasts for a brief time, then its severity might be averaged out or minimized by other performance problems in the entire analysis period. Therefore, the problem may not appear in the ADDM findings. Whether or not a performance problem is captured by ADDM depends on its duration compared to the interval between the Automatic Workload Repository (AWR) snapshots.

If a performance problem lasts for a significant portion of the time between snapshots, then it will be captured by ADDM. For example, if the snapshot interval is one hour, a performance problem that lasts 30 minutes should not be considered a transient performance problem because its duration represents a significant portion of the snapshot interval and will likely be captured by ADDM.

On the other hand, a performance problem that lasts 2 minutes could be a transient performance problem because its duration represents a small portion of the snapshot interval and will likely not show up in the ADDM findings. For example, if the system was slow between 10:00 p.m. and 10:10 p.m., but the ADDM analysis for the time period between 10:00 p.m. and 11:00 p.m. does not show a performance problem, a transient performance problem may have occurred for only a few minutes of the 10-minute interval reported by the user.

This chapter contains the following sections:

- Overview of Active Session History
- Running Active Session History Reports
- Active Session History Reports

Overview of Active Session History

To capture a detailed history of database activity, Oracle Database samples active sessions each second with the Active Session History (ASH) sampler. The Automatic Workload Repository (AWR) snapshot processing collects the sampled data into memory and writes it to persistent storage. ASH is an integral part of the Oracle Database self-management framework and is extremely useful for diagnosing performance problems.

Unlike instance-level statistics gathered by AWR, sampled data is gathered at the session level by ASH. By capturing statistics for only active sessions, a manageable set of data is represented. The size of this data is directly related to the work being performed, rather than to the entire database instance.

Sampled data captured by ASH can be aggregated based on the various dimensions that it captures, including the following:

- SQL identifier of a SQL statement
- Object number, file number, and block number
- Wait event identifier and parameters
- Session identifier and session serial number
- Module and action name
- Client identifier of the session
- Service hash identifier

You can run ASH reports to analyze transient performance problems with the database that only occur during specific times. This technique is especially useful when you are trying to do either of the following:

- Resolve transient performance problems that may last for only a short period of time, such as why a particular job or session is not responding when the rest of the instance is performing as usual
- Perform scoped or targeted performance analysis by various dimensions or their combinations, such as time, session, module, action, or SQL identifier

See Also:

"Active Session History Statistics" on page 2-4

Running Active Session History Reports

This section describes how to generate ASH reports using Oracle Enterprise Manager (Enterprise Manager).

To run ASH reports:

1. On the Performance page, under Average Active Sessions, click Run ASH Report.

The Run ASH Report page appears.

2. Enter the date and time for the start and end of the time period when the transient performance problem occurred.

In this example, database activity increased between 1:45 p.m. and 2:00 p.m., so an ASH report needs to be created for that time period.

Run ASH Report	
Specify the time period for the report.	(Generate Report)
Start Date 3/21/07 (Example: 12/15/03)	End Date 3/21/07
Start Time 1 🔽 45 💌 O AM 💿 PM	End Time 2 • 00 • C AM © PM
Filter SID 💽]

3. Click Generate Report.

The Processing: View Report page appears while the report is being generated.

After the report is generated, the ASH report appears under Report Results on the Run ASH Report page.

CPUs SGA Size	Buf					
		fer Cache		ed Pool	ASH Bu	ffer Size
1 355M (100%)		96M (27.0%)	19	6M (55.2%)		2.0M (0.6%)
		Sample Ti	me	[Jata Source	e
nalysis Begin Time:		21-Mar-07 1	13:45:19	V\$ACTIV	E_SESSION_	HISTORY
alysis End Time:		21-Mar-07 1	14:00:19	V\$ACTIV	E_SESSION	HISTORY
apsed Time:		15.	.0 (mins)			
ample Count:			817			
verage Active Sessions:			0.91			
vg. Active Session per CPU:			0.91			
oort Target:		None s	specified			

4. Optionally, click Save to File to save the report in HTML for future analysis.

Active Session History Reports

You can use an ASH report to identify the source of transient performance problems. The report is divided into titled sections. The following sections of the ASH report are a useful place to begin the investigation:

- Top Events
- Load Profile
- Top SQL
- Top Sessions
- Top DB Objects
- Top DB Files
- Activity Over Time

See Also:

Oracle Database Performance Tuning Guide for more detailed information about the ASH report

Top Events

The Top Events section of the report describes the top wait events of the sampled session activity categorized by user, background, and priority. Use this information to identify the wait events that may be the cause of the transient performance problem.

The Top Events section of the report contains the following subsections:

Top User Events

Top Background Events

Top User Events

The Top User Events subsection of the report lists the top wait events from user processes that accounted for the highest percentages of sampled session activity.

The example in Figure 7–1 shows that 84 percent of database activity is consumed by the CPU + Wait for CPU event. In this example, the Load Profile section should be examined next to determine the type of activity that is causing this wait event.

Figure 7–1 Top User Events



Top Background Events

The Top Background Events subsection of the report lists the top wait events from the background events that accounted for the highest percentages of sampled session activity.

The example in Figure 7–2 shows that 17.65 percent of sampled session activity is consumed by the CPU + Wait for CPU event.

Figure 7–2 Top Background Events

Top Background Events								
Event Event Class % Activity Avg Active S								
CPU + Wait for CPU	CPU	17.65	0.02					
control file parallel write	System I/O	17.65	0.02					
log file parallel write	System I/O	2.94	0.00					

Load Profile

The Load Profile section of the report describes the load analyzed in the sampled session activity. Use the information in this section to identify the service, client, or SQL command type that may be the cause of the transient performance problem. The Top Service/Module subsection lists the services and modules that accounted for the highest percentages of sampled session activity.

The example in Figure 7–3 shows that 81 percent of database activity is consumed by the SYS\$USERS service running the SQL*Plus module. In this example, it appears that the user is running a high-load SQL statement that is causing the performance problem indicated in Figure 7–1. The Top SQL section of the report should be analyzed next to determine whether a particular type of SQL statement makes up the load.

Top Service/Module										
Service	Module	% Activity	Action	% Action						
SYS\$USERS	SQL*Plus	81.52	UNNAMED	81.52						
SYS\$BACKGROUND	MMON_SLAVE	8.57	Auto-Purge Slave Action	6.36						
			Auto-Flush Slave Action	1.59						
	UNNAMED	4.90	UNNAMED	4.90						
SYS\$USERS	Realtime Connection	1.47	UNNAMED	1.47						
emdc	OEM.SystemPool	1.35	NotificationMgr	1.10						

See Also:

- "Monitoring Top Services" on page 4-6
- "Monitoring Top Modules" on page 4-7

Top SQL

The Top SQL section of the report describes the top SQL statements of the sampled session activity. Use this information to identify high-load SQL statements that may be the cause of the transient performance problem. One useful subsection is Top SQL with Top Events, which lists the SQL statements that accounted for the highest percentages of sampled session activity. The Sampled # of Executions column shows how many distinct executions of a particular SQL statement were sampled. To view the text of the SQL statements, click the **SQL ID** link.

The example in Figure 7–4 shows that 75 percent of database activity is consumed by a particular SELECT statement. This statement was executed in the SQL*Plus module shown in Figure 7–3. It appears that this high-load SQL statement is causing the performance problem. The Top Sessions section should be analyzed to identify the session running this SQL statement.

Figure 7–4 Top SQL with Top Events

SQL ID	Planhash	Sampled # of Executions	% Activity	Event	% Event	Top Row Source	% RwSrc	SQL Text
<u>5mxdwvuf9j3vp</u>	3219640484	583		CPU + VVait for CPU		SELECT STATEMENT		SELECT /*+ ORDERED USE_NL(c) F
0abfbq9bmcf0s		1		CPU + VVait for CPU		** Row Source Not Available **	5.63	DECLARE n number; BEGIN for i

See Also:

"Monitoring Top SQL" on page 4-4

Top Sessions

The Top Sessions section lists the sessions that were waiting for the wait event that accounted for the highest percentages of sampled session activity. Use this information to identify the sessions that accounted for the highest percentages of sampled session activity, which may be the cause of the performance problem.

The example in Figure 7–5 shows that 81 percent of database activity is used by the user SH with the session ID of 147. Thus, it appears that this user was running the high-load SQL statement identified in Figure 7–4. You should investigate this session to determine whether it is performing a legitimate operation and tune the SQL statement if possible. If tuning the SQL is not possible and the session is causing an unacceptable performance impact on the system, consider terminating the session.

Sid, Serial#	% Activity	Event	% Event	User	Program	#Samples Active	XIDs
147,9322	81.40	CPU + Wait for CPU	81.40		sqlplus@stbcs09-1 (TNS V1-V3)	665/900 [74%]	0
125,15415	6.36	CPU + Wait for CPU	3.55	SYS	oracle@stbcs09-1 (m000)	29/900 [3%]	2
		db file sequential read	1.96			16/900 [2%]	4
131,45		SQL*Net break/reset to client	0.98	SYSMAN	OMS	8/900 [1%]	0
161,1	1.10	log file parallel write	1.10	SYS	oracle@stbcs09-1 (LGWR)	9/900 [1%]	0
164, 3	1.10	CPU + Wait for CPU	1.10	SYS	oracle@stbcs09-1 (DIA0)	9/900 [1%]	0

Figure 7–5 Top Sessions

See Also:

- "Monitoring Top Sessions" on page 4-5
- Chapter 10, "Tuning SQL Statements"

Top DB Objects

The Top DB Objects subsection lists the database objects (such as tables and indexes) that accounted for the highest percentages of sampled session activity.

The example in Figure 7–6 shows that the objects accounting for the most session activity are in the SYSTEM and SYSAUX tablespaces. In each row of the table, the event is db file sequential read, which signifies that a user process is reading a buffer into the system global area (SGA) buffer cache and is waiting for a physical I/O call to return.

Figure 7–6 Top DB Objects

-	With respect to Application, Cluster, User I/O and buffer busy waits only.										
Object ID	% Activity	Event	% Event	Object Name (Type)	Tablespace						
512	2.94	db file sequential read	2.94	SYS.KOTTD\$ (TABLE)	SYSTEM						
523	2.94	db file sequential read	2.94	SYS.SYS_C00648 (INDEX)	SYSTEM						
60507	2.94	db file sequential read	2.94	SYSMAN.MGMT_TARGET_TYPES (TABLE)	SYSAUX						
60684	2.94	db file sequential read	2.94	SYSMAN.PK_MGMT_CREDENTIALS (INDEX)	SYSAUX						
60789	2.94	db file sequential read	2.94	SYSMAN.MGMT_BUG_ADVISORY (TABLE)	SYSAUX						

Top DB Files

The Top DB Files subsection lists the database files that accounted for the highest percentages of sampled session activity.

The example in Figure 7–7 shows that most of the session activity involves the datafile in the SYSTEM tablespace. This information is consistent with Figure 7–6, which shows that the objects accounting for the most session activity are located in the SYSTEM and SYSAUX tablespaces.

Figure 7–7 Top DB Files

Top Files View Top Files 💌			
Activity (%)	Name	Tablespace	Average Wait Time (ms)
100.00	/ade/lasacle/dbs/t_db1.f	<u>SYSTEM</u>	322
			Total Sample Count: 3

Activity Over Time

The Activity Over Time section of the ASH report is particularly useful for longer time periods because it provides in-depth details about activities and workload profiles during the analysis period. The Activity Over Time section is divided into multiple time slots.

Slot Time (Duration)	Slot Count	Event	Event Count	% Event
13:45:19 (41 secs)	2	CPU + Wait for CPU	1	0.12
		SQL*Net break/reset to client	1	0.12
13:46:00 (2.0 min)	19	CPU + Wait for CPU	17	2.08
		SQL*Net break/reset to client	2	0.24
13:48:00 (2.0 min)	130	CPU + Wait for CPU	122	14.93
		SQL*Net break/reset to client	3	0.37
		control file parallel write	2	0.24
13:50:00 (2.0 min)	133	CPU + Wait for CPU	129	15.79
		control file parallel write	2	0.24
		db file sequential read	1	0.12
13:52:00 (2.0 min)	153	CPU + Wait for CPU	141	17.26
		db file sequential read	5	0.61
		control file parallel write	2	0.24
13:54:00 (2.0 min)	175	CPU + Wait for CPU	143	17.50
		db file sequential read	15	1.84
		enq: WF - contention	5	0.61
13:56:00 (2.0 min)	130	CPU + Wait for CPU	124	15.18
		db file sequential read	4	0.49
		Data file init write	1	0.12
13:58:00 (2.0 min)	74	CPU + Wait for CPU	65	7.96
		log file parallel write	3	0.37
		log file sync	3	0.37
14:00:00 (19 secs)	1	CPU + Wait for CPU	1	0.12

Figure 7–8 Activity Over Time

Each of the time slots contains information regarding that particular time slot, as described in Table 7–1.

Table 7–1 Activity Over Time

Column	Description
Slot Time (Duration)	Duration of the slot
Slot Count	Number of sampled sessions in the slot
Event	Top three wait events in the slot
Event Count	Number of ASH samples waiting for the wait event
% Event	Percentage of ASH samples waiting for wait events in the entire analysis period

All inner slots are 2 minutes each and can be compared to each other. The first and last slots, which are also called the outer slots, are odd-sized because they are the only slots that do not have a fixed slot time.

When comparing the inner slots, perform a skew analysis by identifying spikes in the Event Count and Slot Count columns. A spike in the Event Count column indicates an increase in the number of sampled sessions waiting for a particular event. A spike in the Slot Count column indicates an increase in active sessions, because ASH data is sampled from active sessions only and a relative increase in database workload. Typically, when the number of active session samples and the number of sessions

associated with a wait event increase, the slot may be the cause of the transient performance problem.

The example in Figure 7–8 indicates that the number of sampled sessions rose sharply in the first inner slot and fell sharply in the last inner slot. The slot count and event count peaked in the 13:54 p.m. time slot.

Resolving Performance Degradation Over Time

Performance degradation of the database over time happens when your database was performing optimally in the past, such as 6 months ago, but has gradually degraded to a point where it becomes noticeable to the users. The Automatic Workload Repository (AWR) Compare Periods report enables you to compare database performance between two periods of time.

While an AWR report shows AWR data between two snapshots (or two points in time), the AWR Compare Periods report shows the difference between two periods (or two AWR reports, which equates to four snapshots). Using the AWR Compare Periods report helps you to identify detailed performance attributes and configuration settings that differ between two time periods. The two time periods selected for the AWR Compare Periods report can be of different durations. The report normalizes the statistics by the amount of time spent on the database for each time period and presents statistical data ordered by the largest difference between the periods.

For example, a batch workload that historically completed in the maintenance window between 10:00 p.m. and midnight is currently showing poor performance and completing at 2 a.m. You can generate an AWR Compare Periods report from 10:00 p.m. to midnight on a day when performance was good and from 10:00 a.m. to 2 a.m. on a day when performance was poor. The comparison of these reports should identify configuration settings, workload profile, and statistics that were different in these two time periods. Based on the differences identified, you can more easily diagnose the cause of the performance degradation.

This chapter contains the following sections:

- Managing Baselines
- Running the AWR Compare Periods Reports
- Using the AWR Compare Periods Reports

See Also:

 "Gathering Database Statistics Using the Automatic Workload Repository" on page 2-1

Managing Baselines

Baselines are an effective way to diagnose performance problems. AWR supports the capture of baseline data by enabling you to specify and preserve a pair or a range of snapshots as a baseline. The snapshots contained in a baseline are excluded from the automatic AWR purging process and are retained indefinitely.

A moving window baseline corresponds to all AWR data that exists within the AWR retention period. Oracle Database automatically maintains a system-defined moving window baseline. The default size of the window is the current AWR retention period, which by default is 8 days.

This section contains the following topics:

- Creating a Baseline
- Deleting a Baseline
- Computing Threshold Statistics for Baselines
- Setting Metric Thresholds for Baselines

Creating a Baseline

Before creating a baseline, carefully consider the time period you choose as a baseline because it should represent the database operating at an optimal level. In the future, you can compare these baselines with other baselines or snapshots captured during periods of poor performance to analyze performance degradation over time.

You can create the following types of baseline:

- Creating a Single Baseline
- Creating a Repeating Baseline

Creating a Single Baseline

A single baseline is captured at a single, fixed time interval. For example, a single baseline may be captured on March 5, 2007 from 5:00 p.m. to 8:00 p.m.

You can choose a start time and an end time that are in the future to create a baseline that captures future database activity. If both the start time and the end time are in the future, a baseline template with the same name as the baseline will also be created. A baseline template is a specification that enables Oracle Database to automatically generate a baseline for a future time period.

To create a single baseline:

1. From the Database Home page, click **Server**.

The Server subpage appears.

2. Under Statistics Management, click AWR Baselines.

The AWR Baselines page appears with a list of existing baselines displayed.

AWR	Baselines								
					Page Refreshed Mar	6, 2007 5:40:49	PM PST Refresh		
Search	ι	Go					Create		
Edit	Edit View Delete Actions Schedule Statistics Computation Go								
Select	Select Name								
۲	SYSTEM MOVING WINDOW	MOVING_WINDOW (8 Days)	No	Yes			Mar 6, 2007 5:00:03 PM		

3. Click Create.

The Create Baseline: Baseline Interval Type page appears.

4. Select Single.

Create Baseline: Baseline Interval Type	
	Cancel Continue
Choose one of the baseline interval types listed below.	
⊙ Single	
The single type of baseline has a single and fixed time interval. For example, from Jan 1, 2007 10:00 AM to Jan 1, 2007 12:1	00 PM.
○ Repeating	
The repeating type of baseline has a time interval that repeats over a time period. For example, every Monday from 10:00 A	VM to 12:00 PM for the
ear 2007.	

5. Click Continue.

The Create Baseline: Single Baseline page appears.

Create Baseline: Single Baseline		
	(Cancel) (Back) (Finish)
The single type of baseline has a single and fixed time interval. For example, from Jan 1, 2007 10:00 AM to Jan 1, 2007 12:00	\square	
	FINI.	
* Baseline Name		
Baseline Interval		
Snapshot Range		
Change Chart Time Period		
Select Time Period		
Choose the Period Start Time option, then click a snapshot icon in the chart to select the period start time. Repeat the p	rocess for the	period end time.
🔿 Period Start Time 🝳 Mar 6, 2007 7:00:22 PM PST 🛛 💿 Period End Time 🗘 Mar 6, 2007	7 8:00:31 PM	PST
1.347		
8 0.674		Wait
ů.		User I/O
5 0.674 9 0.000 12 AM 2 4 6 8 10 12 PM 2 4 5 March 6	6	E CPU
2007		
ඉබ් බ් බ් බ්බ් බ් බ් බ් බ් බ් බ්	ർർ മ	60
		-
○ Time Range		
Start Time Mar 7, 2007 🗒 8 🗸 20 🗸 PM 🗸 🛛 End Time Mar 7, 2007 🗒 8 🗸 30 🗸 PM 🗸		
(example: Mar 6, 2007) (example: Mar 6, 2007)		

- 6. In the **Baseline Name** field, enter a name for the baseline.
- **7.** Under Baseline Interval, select whether to use a snapshot range or a time range for the baseline. Do one of the following:
 - To use a snapshot range, select **Snapshot Range**. Complete the following steps:
 - Optionally, to view older snapshots that are not displayed below the Active Sessions chart, expand Change Chart Time Period. Enter the desired start date in the Chart Start Date field and the desired end date in the Chart End Date field, and click Go.
 - Under Select Time Period, select a start time for the baseline by selecting Period Start Time and the snapshot icon below the Active Sessions chart that corresponds to the desired start time.
 - Select an end time for the baseline by selecting **Period End Time** and the snapshot icon below the Active Sessions chart that corresponds to the desired end time.

In this example, a snapshot range on March 5, 2007 from 5:00 p.m. to 8:00 p.m. is selected.

Snapshot Range									
▼Change Chart Time Peri	od								
To search for a specific tim then click Go to see an activ		beginning and er	ding snapshots,	specify a wider range	of time frame	e, which includ	es the tim	ie period for wh	ich you are interested,
Chart Start Da	te 3/5/07 Go			Chart End D	ate 3/5/07				
Select Time Period									
Choose the Period Start Tir	ne option, ther	n click a snapsho	t icon in the char	rt to select the period	start time. Re	peat the proce	ss for the	period end time	э.
🔘 Period Start Time 🕻) Mar 5, 200	07 5:00:47 PM	PST	Period End	l Time 🛈 Ma	ar 5, 2007 8:0	0:11 PM	PST	
1.347									
SU 0.898 S 0.449								Wait User I/	- <u>200m</u>
.000 명 · · · · · · · · · · · · · · · · ·									⊕ .
12 AM 2 Mar 5, 2007	4	6 8	10 1	2 PM 2	4 6	8	10	12 AM 6	

- To use a time range, select **Time Range**. Complete the following steps:
 - In the **Start Time** fields, select a start time for the baseline.
 - In the **End Time** fields, select an end time for the baseline.

In this example, a time range from 5:00 p.m. to 8:00 p.m. on March 5, 2007 is selected.

💿 Time Range	
Start Time Mar 5, 2007 🗐 (example: Mar 6, 2007) 5 💌 00 💌 PM 💌	End Time Mar 5, 2007 (example: Mar 6, 2007)

8. Click Finish.

The AWR Baselines page reappears with the newly created baseline displayed.

Creating a Repeating Baseline

A repeating baseline is a baseline that repeats during a time interval over a specific period. For example, a repeating baseline may repeat every Monday from 5:00 p.m. to 8:00 p.m. for the year 2007.

To create a repeating baseline:

1. From the Database Home page, click **Server**.

The Server subpage appears.

2. Under Statistics Management, click AWR Baselines.

The AWR Baselines page appears with a list of existing baselines displayed.

3. Click Create.

The Create Baseline: Baseline Interval Type page appears.

```
Create Baseline: Baseline Interval Type

Choose one of the baseline interval types listed below.

Single
The single type of baseline has a single and fixed time interval. For example, from Jan 1, 2007 10:00 AM to Jan 1, 2007 12:00 PM.
Repeating
The repeating type of baseline has a time interval that repeats over a time period. For example, every Monday from 10:00 AM to 12:00 PM for the year 2007.
```

4. Select **Repeating** and then click **Continue**.

The Create Baseline: Repeating Baseline Template page appears.

5. In the **Baseline Name Prefix** field, enter a name prefix for the baseline.

- **6.** Under Baseline Time Period, specify the time of the day that you want the baseline to begin collecting AWR data and the duration of the baseline collection.
- 7. Under Frequency, do one of the following:
 - Select **Daily** if you want the baseline to repeat on a daily basis.
 - Select Weekly if you want the baseline to repeat on a weekly basis and select the day of the week on which the baseline will repeat.
- **8.** Under Interval of Baseline Creation, complete the following steps:
 - **a.** In the **Start Time** fields, select a date and time in the future when the data collection should begin.
 - **b.** In the **End Time** fields, select a date and time in the future when the data collection should end.
- **9.** Under Purge Policy, enter the number of days to retain baselines that have been captured.
- 10. Click Finish.

A baseline template with the same name as the baseline name prefix will also be created. A baseline template is a specification that enables Oracle Database to automatically generate a baseline for a future time period.

In this example, a repeating baseline that repeats weekly on Mondays from 5:00 p.m. to 8:00 p.m. for the year 2007 will be created.

Create Baseline: Repeating Baseline Template
Cancel Back Finish
The repeating type of baseline has a time interval that repeats over a time period. For example, every Monday from 10:00 AM to 12:00 PM for the year 2007.
* Baseline Name Prefix BASELINE_2007_MON_5PM
Baseline Time Period
Start Time 5 V PM V Duration (Hours) 3
Frequency
© Daily
Weekly
💿 Monday 🔿 Tuesday 🔿 Wednesday 🔿 Thursday 🔿 Friday 🔿 Saturday 🔿 Sunday
Interval of Baseline Creation
Mar 12, 200 S OO PM End Time Dec 31, 200 B OO PM End Time Dec 31, 200 B OO PM Image: Complex Start Start Start Start Start Start Start Sta
Purge Policy
Retention Time (Days) 30

Deleting a Baseline

To conserve storage space, you may want to periodically delete unused baselines stored in the database.

To delete a baseline:

1. From the Database Home page, click **Server**.

The Server subpage appears.

2. Under Statistics Management, click AWR Baselines.

The AWR Baselines page appears with a list of existing baselines displayed.

3. Select a baseline and click **Delete**.

The Confirmation page appears.

4. Select whether to purge the underlying data associated with the baseline.

The underlying data includes the individual snapshots preserved in the baseline and any statistics that are computed for the baseline. Do one of the following:

- To delete the underlying data, select **Purge the underlying data associated** with the baseline.
- To preserve the underlying data, select **Do not purge the underlying data associated with the baseline**.
- 5. Click Yes.

The AWR Baselines page reappears. A message informs you that the baseline was deleted successfully.

Computing Threshold Statistics for Baselines

Computing threshold statistics for baselines enables you to graphically display the computed statistics in the charts on the Performance page.

To compute threshold statistics for baselines:

1. From the Database Home page, click Server.

The Server subpage appears.

2. Under Statistics Management, click AWR Baselines.

The AWR Baselines page appears with a list of existing baselines displayed.

3. Select the baseline for which you want to compute statistics.

Select a baseline that does not already have computed statistics. These baselines are identified by No in the Statistics Computed column.

4. From the Actions list, select Schedule Statistics Computation, and then click Go.

The Compute Threshold Statistics page appears.

Compute Thresho	old Statistics: BASELINE_NORMAL_LOAD
	Cancel) (Submit
Task Informatio	n
* Name	BSLNSTATS_1173958899
Description	Compute metric statistics for baseline $\ensuremath{BASELINE_NORMAL_LOAD}$ (i
Schedule	
Schedule Type Star	ndard
Time Zone GM	T-07:00
Repeating	
Repeat Do	Not Repeat
Start	
🧭 TIP This op	peration may be resource-intensive and should be scheduled during off-peak hours.
💿 Immediat	æly
OLater	
	Date 3/15/07 (example: 3/15/07)
	Time 4 🔽 50 🔽 👁 AM 🔿 PM

5. In the Name field, enter a name for the task.

Alternatively, you can choose to use the system-generated name.

6. In the **Description** field, enter a description for the task.

Alternatively, you can choose to use the system-generated description.

- 7. Under Start, do one of the following:
 - Select **Immediately** to run the task immediately after it has been submitted.
 - Select Later to run the task at a later time as specified using the Date and Time fields.
- 8. Click Submit.

The AWR Baselines page appears. A message informs you that statistics computation has been scheduled for the selected baseline.

See Also:

- "Customizing the Database Performance Page" on page 4-27 for information about displaying computed statistics on the Performance page
- Oracle Database 2 Day DBA for information about thresholds and how to manage them

Setting Metric Thresholds for Baselines

As explained in "Setting Metric Thresholds for Performance Alerts" on page 5-1, a metric is the rate of change in a cumulative statistic. Alerts notify you when particular metric thresholds are crossed. When the metric thresholds are crossed, the system is in an undesirable state. You can edit the threshold settings for baseline metrics.

You can create the following types of baseline:

- Setting Metric Thresholds for the Default Moving Baseline
- Setting Metric Thresholds for Selected Baselines

Setting Metric Thresholds for the Default Moving Baseline

This section describes the easiest technique for setting the metric thresholds for the default moving baseline. You can choose a group of basic metric threshold settings based on common database workload profiles: OLTP, data warehousing, and OLTP with nighttime batch jobs. After choosing a workload profile, you can expand or change the threshold values as needed.

To set metric thresholds for the default moving baseline:

1. On the Database Home page, under Related Links, click **Baseline Metric Thresholds**.

The Threshold Configuration tab of the Baseline Metric Thresholds page appears.

2. Click Quick Configuration.

The Quick Configuration: Baseline Metric Thresholds page appears.

- **3.** In **Workload Profile**, select one of the following options, depending on how you are using the database:
 - Primarily OLTP (pure transaction processing 24 hours a day)
 - Primarily Data Warehousing (query and load intensive)
 - Alternating (OLTP during the daytime and batch during the nighttime)

In this example, select Primarily OLTP.

4. Click Continue.

The Quick Configuration: Review OLTP Threshold Settings page appears.

iick Configuratio	n: Review OLTI	P Thre	eshold Set	tings		
			C	ancel) (Back)	Finis
OLTP Threshold S	ettinas		_			
SETT THRESHOLD S	eccings					
Metric Name	AWR Baseline		Threshold Type	Warning Level	Critical Level	
Average Active Sessions	SYSTEM_MOVING_W		Significance Level	Very High (0.99)	Extreme (0.9999)	
Redo Generated (per second)	SYSTEM_MOVING_W		Percentage ol Maximum	100%	120%	
Response Time (per transaction)	SYSTEM_MOVING_W		Significance Level	Very High (0.99)	Extreme (0.9999)	
Session Logical Reads (per transaction)	SYSTEM_MOVING_W		Significance Level	Very High (0.99)	None	
Impact on Existin	g Thresholds					
Applying the OLTP	threshold settings will	also cle	ar the followir	ig settings.		
		Three			Tritical	
Metric Name	AWR Baseline	Type	L	evel I	.evel	
Cumulative Logons (pe second)	ər	Fixed	Values 10	00		
Current Open Cursors Count		Fixed	Values 1,	200		

5. Review the metric threshold settings and click Finish.

You are returned to the Baseline Metric Thresholds page, with the Threshold Configuration tab selected. The metric threshold settings are displayed.

Setting Metric Thresholds for Selected Baselines

This section explains how to select a baseline and edit its thresholds. You can configure the type of threshold, for example, whether it is based on significance levels, percentage of maximum values, or fixed values. You can also configure the threshold levels that determine when the system generates critical alerts and warnings.

You can edit thresholds for the default moving baseline or a baseline that you created in the AWR Baselines page. You can select a baseline in the Edit Thresholds page after you have scheduled statistics computation from the AWR Baselines page and the statistics have finished computing on the static baseline.

To set a metric threshold for the default moving baseline:

1. On the Database Home page, under Related Links, click **Baseline Metric Thresholds**.

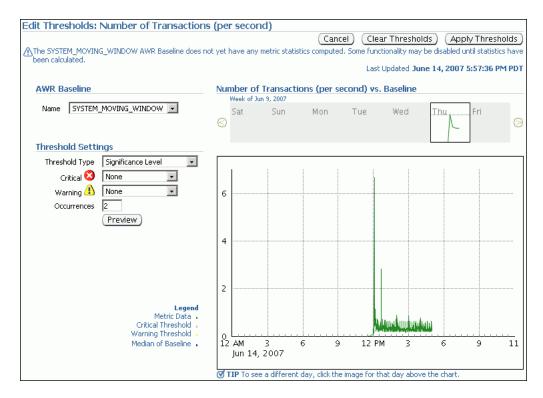
The Threshold Configuration tab of the Baseline Metric Thresholds page appears.

2. In the View list, select **Basic Metrics**.

The Baseline Metric Thresholds page appears.

3. In the **Category/Name** column, click the link for the metric whose threshold you want to set or change. For example, click **Number of Transactions (per second)**.

The Edit Thresholds: Number of Transactions (per second) appears.



The charts on this page provide thumbnail and detailed views of metric activity for a 24-hour period. In the top thumbnail chart, click a day to view the value of the metric plotted against a 24-hour period.

 Under AWR Baseline, in the Name list, select either the default SYSTEM_MOVING_WINDOW or the name of a baseline created in the AWR Baselines page.

A baseline appears in the AWR Baseline list after you have scheduled statistics computation from the AWR Baselines page and the statistics have finished computing on the static baseline.

In this example, select AWR_BASELINES_2007.

The page refreshes to show the charts for the baseline that you selected.

- **5.** In the Threshold Settings section, complete the following steps:
 - a. In Threshold Type, leave Significance Level selected.
 - b. In Critical, select Extreme.
 - c. In Warning, select Very High.
 - d. In Occurrences, leave the current value.
- 6. Click Apply Thresholds.

You are returned to the Baseline Metric Thresholds page. This page shows the altered metric threshold settings.

Running the AWR Compare Periods Reports

This section describes how to run the Automatic Workload Repository (AWR) Compare Periods reports using Oracle Enterprise Manager. You can use AWR Compare Periods reports to compare the database performance between two time periods by:

- Comparing a Baseline to Another Baseline or Pair of Snapshots
- Comparing Two Pairs of Snapshots

Comparing a Baseline to Another Baseline or Pair of Snapshots

When performance degradation happens to a database over time, you should run the AWR Compare Periods report to compare the degraded performance, captured as a new baseline or a pair of snapshots, to an existing baseline. You will need a baseline that represents the system operating at an optimal level. If an existing baseline is not available, then you can compare database performance between two periods of time by using two arbitrary pairs of snapshots, as described in "Comparing Two Pairs of Snapshots" on page 8-12.

To compare a baseline to another baseline:

1. From the Database Home page, click Server.

The Server subpage appears.

2. Under Statistics Management, click Automatic Workload Repository.

The Automatic Workload Repository page appears.



3. Under Manage Snapshots and Baselines, click the link next to Baselines.

The AWR Baselines page appears.

AWR	Baselines			Page Re	freshed Mar 22, 20	07 5:16:33 PM P	DT (Refresh)		
Search Go (Create)									
Edit View Delete Actions Schedule Statistics Computation 🔽 🕝									
Select	Name \triangle	Туре	Valid	Statistics Computed	Last Time Computed	Start Time	End Time		
œ	AWR BASELINE 2007	STATIC	Yes	No			Mar 22, 2007 5:00:00 PM		
0	SYSTEM MOVING WINDOW	MOVING_WINDOW (8 Days)	Yes	Pending	Mar 25, 2007 12:00:00 AM		Mar 22, 2007 5:00:00 PM		

- 4. Complete the following steps:
 - **a.** Select the baseline to use for the report.

At least one existing baseline must be available.

b. From the Actions list, select Compare Periods and click Go.

The Compare Periods: Second Period Start page appears. Under First Period, the selected baseline is displayed.

In this example, the baseline named AWR_BASELINE_2007 is selected.

First Period						
Baseline ID	1	Beginning Snapshot ID	101	Capture Time	Mar 22, 2007 4:30:52 PM	
Name	AWR_BASELINE_2007	Ending Snapshot ID	102	Capture Time	Mar 22, 2007 5:00:00 PM	

- **5.** Compare the baseline selected in the first period to another baseline or a pair of snapshots. Do one of the following:
 - To compare to another baseline, select **Select a Baseline** and the baseline you want to use in the second period, and then click **Next**.

In this example, the baseline named SYSTEM_MOVING_WINDOW is selected.

Select the second period by choosing a baseline or a beginning snapshot.						
Select a Baseline (You will skip the next step since you do not need an end to the period)						
O Sele	ct Beginning Sna	pshot	-		-	
Select	Baseline ID $ riangle$	Name		Beginning Snapshot Capture Time		Ending Snapshot Capture Time
۲	<u>0</u>	SYSTEM_MOVING_WINDOW	42	Mar 21, 2007 12:00:37 PM		Mar 22, 2007 5:30:09 PM

The Compare Periods: Review page appears. Go to Step 7.

• To compare to a pair of snapshots, select **Select Beginning Snapshot** and the beginning snapshot to use in the second period, and then click **Next**.

In this example, snapshot 102, taken on March 22, 2007 at 5:00 p.m., is selected.

Select th	Select the second period by choosing a baseline or a beginning snapshot.						
	C Select a Baseline (You will skip the next step since you do not need an end to the period)						
⊙ _{Sele}	ect Begi	nning Snapshot					
Go To Ti		22/07 (6:00 PM 💽 (Go ample: 12/15/03)	D				
			S Prev	rious 10 61-62 of 62 💌 Next 🛇			
Select ID Capture Time 🛆			Collection Level	Within A Baseline			
International Mar 22, 2007 5:00:00 PM		Mar 22, 2007 5:00:00 PM	TYPICAL	×			
O 103 Mar 22, 2007 5:30:09 PM		Mar 22, 2007 5:30:09 PM	TYPICAL				

The Compare Periods: Second Period End appears. Proceed to the next step.

6. Select the ending snapshot for the snapshot period that will be included in the report and click **Next**.

In this example, snapshot 103, taken on March 22, 2007 at 5:30 p.m., is selected.

Second Period							
Beginning Snapshot ID 102 Beginning Snapshot Capture Time Mar 22, 2007 5:00:00 PM							
Select ar	n ginding	snapshot for the second period.					
Go To Time 3/22/07 📄 6:00 PM 💌 Go							
Select		ample: 12/15/03) Capture Time $ abla$	Collection Level	Within A Baseline			
o	103	Mar 22, 2007 5:30:09 PM	TYPICAL				

The Compare Periods: Review page appears.

mpare Period	I I I COICH				Cancel Back	Step 5 of 5 Finis
Database datab First Period	ase					
Baseline	ID 1		Beginning Snapshot ID	101	Capture Tim	e Mar 22, 2007 4:30:52 PM
Na	ne AWR_BA	5ELINE_2007	Ending Snapshot ID	102	Capture Tim	e Mar 22, 2007 5:00:00 PM
Second Period						
Beginn	ng Snapshot ID	102		Beginning Sna	oshot Capture Mar 2: Time	2, 2007 5:00:00 PM
End	ng Snapshot ID	103		Ending Snapshot		2. 2007 5:30:09 PM

7. Review the periods to be included in the report and click Finish.

The Compare Periods: Results page appears.

Data from the selected periods appears under the General subpage. You can view data per second or per transaction by selecting an option from the View Data list.

Name 🛆	First Period Metric Ratio	Second Period Metric Ratio	First Period Value	Second Period Value	First Period Rate Per Second	Second Period Rate Per Second
DB cpu (seconds)			0.00	0.00	0.00	0.00
DB time (seconds)	1		5,887.68	5,963.94	3.37	3.30
db block changes			19,944.00	26,344.00	11.42	14.56
execute count			20,285.00	20,770.00	11.61	11.48
global cache cr block receive time (seconds)			0.00	0.00	0.00	0.00
global cache cr blocks received			0.00	0.00	0.00	0.00
global cache current block receive time (seconds)			0.00	0.00	0.00	0.00
global cache current blocks received			0.00	0.00	0.00	0.00
global cache get time (seconds)			0.00	0.00	0.00	0.00
global cache gets			0.00	0.00	0.00	0.00
opened cursors cumulative	I		18,051.00	18,514.00	10.33	10.23
parse count (total)			5,918.00	5,881.00	3.39	3.25
parse time cpu (seconds)			0.23	0.32	0.00	0.00
parse time elapsed (seconds)			0.26	1.13	0.00	0.00
physical reads	1		371.00	372.00	0.21	0.21
physical writes			1,202.00	923.00	0.69	0.51
redo size (KB)			3,680.53	4,473.99	2.11	2.47

In this example, parse time in the second period is much higher than the first.

8. Click **Report** to view the report.

The Processing: View Report page appears while the report is being generated. After it completes, the report will appear. To change periods, click **Change Periods**. To save the report as an HTML file, click **Save to File**.

See Also:

- "Creating a Baseline" on page 8-2
- "Using the AWR Compare Periods Reports" on page 8-15

Comparing Two Pairs of Snapshots

If an existing baseline is not available, then you can compare the database performance by using two arbitrary pairs of snapshots, one pair taken when the database is performing optimally, and another pair when the database is performing poorly. At least four existing snapshots must be available.

To compare performance using two pairs of snapshots:

1. From the Database Home page, click **Server**.

The Server subpage appears.

2. Under Statistics Management, click Automatic Workload Repository.

The Automatic Workload Repository page appears.

General
Edit
Snapshot Retention (days) 40
Snapshot Interval (minutes) 30
Collection Level TYPICAL
Next Snapshot Capture Time Mar 22, 2007 5:30:02 PM
·····
Manage Snapshots and Baselines
Run AWR Report
Snapshots 61
Baselines 2
Latest Snapshot Time Mar 22, 2007 5:00:02 PM
Earliest Snapshot Time Mar 21, 2007 12:00:37 PM

3. Under Manage Snapshots and Baselines, click the link next to **Snapshots**.

The Snapshots page appears.

	Select Beginning Snapshot Go To Time 3/22/07 📰 7:00 PM 🔽 Go							
(Example: 12/15/03) (Create) ☐ Delete Actions Create SQL Tuning Set								
Select	ID	Capture Time $ riangle$	Collection Level	Within A Baseline				
0	<u>92</u>	Mar 22, 2007 12:00:34 PM	TYPICAL					
0	<u>93</u>	Mar 22, 2007 12:30:42 PM	TYPICAL					
0	<u>94</u>	Mar 22, 2007 1:00:51 PM	TYPICAL					
0	95	Mar 22, 2007 1:30:01 PM	TYPICAL					
0	96	Mar 22, 2007 2:00:10 PM	TYPICAL					
0	97	Mar 22, 2007 2:30:19 PM	TYPICAL					
0	98	Mar 22, 2007 3:00:27 PM	TYPICAL					
0	99	Mar 22, 2007 3:30:36 PM	TYPICAL					
0	100	Mar 22, 2007 4:00:45 PM	TYPICAL					
0	<u>101</u>	Mar 22, 2007 4:30:52 PM	TYPICAL	¥				
0	<u>102</u>	Mar 22, 2007 5:00:00 PM	TYPICAL	¥				

4. From the Go To Time list, select the time for the starting snapshot and click Go.

This action filters the snapshots and displays only the snapshot taken at the start of the comparison period. The time in this example is 5:00 p.m. on March 21, 2007.

Select Beg	Select Beginning Snapshot					
Go To Time	3/21/07	🧱 5:00 PM	• G0			

5. Under Select Beginning Snapshot, select the starting point for the first snapshot period to be included in the report.

In this example, snapshot 53, taken on Mar 21, 2007 5:00 p.m., is selected.

\odot	<u>53</u>	Mar 21, 2007 5:00:09 PM	TYPICAL
0	<u>54</u>	Mar 21, 2007 5:30:18 PM	TYPICAL
0	<u>55</u>	Mar 21, 2007 6:00:26 PM	TYPICAL

6. From the Actions list, select **Compare Periods** and click **Go**.

The Compare Periods: First Period End page appears.

7. Select the ending point for the first snapshot period to be included in the report and click **Next**.

In this example, snapshot 55, taken on Mar 21, 2007 6:00 p.m., is selected.

\odot	<u>55</u>	Mar 21, 2007 6:00:26 PM	TYPICAL
0	<u>56</u>	Mar 21, 2007 6:30:34 PM	TYPICAL
0	<u>57</u>	Mar 21, 2007 7:00:42 PM	TYPICAL

The Compare Periods: Second Period Start page appears.

8. Select the starting point for the second snapshot period to be included in the report and click **Next**.

In this example, snapshot 104, taken on March 22, 2007 at 6:00 p.m., is selected.

 Image: 104
 Mar 22, 2007 6:00:18 PM
 TYPICAL

 Image: 105
 Mar 22, 2007 6:30:25 PM
 TYPICAL

 Image: 106
 Mar 22, 2007 7:00:33 PM
 TYPICAL

The Compare Periods: Second Period End page appears.

9. Select the end point for the second period that will be included in the report and click **Next**.

In this example, snapshot 106, taken on March 22, 2007 at 7:00 p.m., is selected.

\odot	106	Mar 22, 2007 7:00:33 PM	TYPICAL
0	107	Mar 22, 2007 7:30:41 PM	TYPICAL
0	108	Mar 22, 2007 8:00:50 PM	TYPICAL

The Compare Periods: Review page appears.

First Period			
Beginning Snapshot ID	53	Beginning Snapshot Capture Time	Mar 21, 2007 5:00:09 PM
Ending Snapshot ID	55	Ending Snapshot Capture Time	Mar 21, 2007 6:00:26 PM
Second Period			
Beginning Snapshot ID	104	Beginning Snapshot Capture Time	Mar 22, 2007 6:00:18 PM
Ending Snapshot ID	106	Ending Snapshot Capture Time	Mar 22, 2007 7:00:33 PM

10. Review the selected periods that will be included in the report and click Finish.

The Compare Periods: Results page appears.

Data from the selected periods appears under the General subpage. You can view data per second or per transaction by selecting an option from the View Data list.

					S Previous 1	-27 of 27 Next 🔅
Name 🛆	First Period Metric Ratio	Second Period Metric Ratio	First Period Value	Second Period Value	First Period Rate Per Second	Second Perio Rate Per Secon
DB cpu (seconds)			0.00	0.00	0.00	0.0
DB time (seconds)			12,646.31	12,034.85	3.50	3.3
db block changes			64,654.00	55,517.00	17.88	15.3
execute count			1,043,192.00	40,113.00	288.41	11.1
global cache cr block receive time (seconds)			0.00	0.00	0.00	0.0
global cache cr blocks received			0.00	0.00	0.00	0.0
global cache current block receive time (seconds)			0.00	0.00	0.00	0.0
global cache current blocks received			0.00	0.00	0.00	0.0
global cache get time (seconds)			0.00	0.00	0.00	0.0
global cache gets			0.00	0.00	0.00	0.0
opened cursors cumulative			1,038,542.00	35,535.00	287.13	9.8
parse count (total)			13,026.00	10,952.00	3.60	3.0
parse time cpu (seconds)			1.61	0.44	0.00	0.0
parse time elapsed (seconds)			4.52	0.53	0.00	0.0
physical reads			999.00	703.00	0.28	0.1
physical writes		I	1,930.00	2,063.00	0.53	0.5
redo size (KB)	1		10,350.50	9,106.48	2.86	2.5
session cursor cache hits			1,031,174.00	29,014.00	285.09	8.0
session logical reads			63,168,643.00	154,144.00	17,464.37	42.6

In this example, the first period shows significantly more activity, especially in session reads, than the second period.

11. To view the report, click the **Report** tab.

The Processing: View Report page appears while the report is being generated. After it completes, the report will appear. To change periods, click **Change Periods**. To save the report as an HTML file, click **Save to File**.

Using the AWR Compare Periods Reports

After an AWR Compare Periods report is generated for the time periods you want to compare, you can use it to perform an analysis of performance degradation with Oracle Database that may have happened over time. To learn how to generate AWR Compare Periods reports, see "Running the AWR Compare Periods Reports" on page 8-9.

Figure 8–1 shows an example of an AWR Compare Periods report.

								_		
				DSI	TC)R.	YC	OIVI	PARE	
PERI	UD	REPO	JK I							
Snapshot Set	DB Name	DB Id	Instance	Inst num	Rel	ease	Cluste	Host	Std Block Size	
First (1st) Second (2nd	EMDC) EMDC	1621828677 1621828677				.0.4.0 .0.4.0		stbcs09 stbcs09		
Snapshot Set	Begin Snap Id	Begin Snap	o Time	End Snap Id	E	nd Sna	ıp Time	Av Acti Use	ive Time	DB tim (mir
1st	50 21	I-Mar-07 13:5	9:33 (Wed)	52 3	21-Mai	r-07 16	:00:51 (V	Ved) 0.0	04 121.31	5.05
2nd	110 2	2-Mar-07 21:0	0:07 (Thu)	114	22-Ma	r-07 23	3:00:47 (50.4
%Diff								950	.00 -0.54	898.
Number of CF	Di le:			1st	1	21	nd 1	Diff	%Diff 0.00	
Physical Mer				1	773M		1773M	OM		
Load at Start				'	2.07		.52	-1.55		
Load at End S					.39		.87	.48	123.08	
%User Time:					3.97		7.72	3.75	94.46	
%System Tim	ne:				9.15		8.87	28	-3.06	
%ldle Time:				1	36.75		83.25	-3.5	-4.03	
%IO Wait Tim	e:				1.55		2.06	.51	32.90	
System C	Configur	ation Con	nparisor	1						
				1st		21	nd	Diff	%Diff	
SGA Target:					0		0	OM		
Buffer Cack					96M		96M	OM	0.00	
Shared Pool					196M		196M	OM		
Large Pool S					OM		OM	OM		
Java Pool Si Streams Pool					52M 0M		52M 0M	OM		
Streams Po Log Buffer:	ui size:				ом 120К		0M 6,120K	OM OK		
Log burier.				0,	120N		0,1200	UN	0.00	

Figure 8–1	AWR Compare	Periods Report
------------	-------------	----------------

The AWR Compare Periods report is divided into the following sections:

- Summary of the AWR Compare Periods Report
- Details of the AWR Compare Periods Report
- Supplemental Information in the AWR Compare Periods Report

Summary of the AWR Compare Periods Report

The report summary is at the beginning of the AWR Compare Periods report, and summarizes information about the snapshot sets and loads used in the report. The report summary contains the following sections:

- Snapshot Sets
- Host Configuration Comparison
- System Configuration Comparison
- Load Profile
- Top Timed Events

Snapshot Sets

The Snapshot Sets section displays information about the snapshot sets used for this report, such as instance, host, and snapshot information.

In the example shown in Figure 8–1 on page 8-16, the first snapshot period corresponds to the time when performance was stable on March 21, 2007 from 1:59 p.m. to 4:00 p.m. The second snapshot period corresponds to the time when performance degradation occurred on March 22, 2007 from 9:00 p.m. to 11:00 p.m.

Host Configuration Comparison

The Host Configuration Comparison section compares the host configurations used in the two snapshot sets. For example, the report compares physical memory and number of CPUs. Any differences in the configurations are quantified as percentages in the %Diff column.

System Configuration Comparison

The System Configuration Comparison section compares the database configurations used in the two snapshot sets. For example, the report compares the SGA and log buffer size. Any differences in the configurations are quantified as percentages in the %Diff column.

Load Profile

The Load Profile section compares the loads used in the two snapshot sets. Any differences in the loads are quantified as percentages in the %Diff column.

	1st per sec	2nd per sec	%Diff	1st per txn	2nd per txn	%Diff
DB time:	0.01	0.25	2,400.00	0.04	0.70	1,650.00
CPU time:	0.01	0.04	300.00	0.02	0.12	500.00
Redo size:	2,676.70	8,280.03	209.34	7,945.55	23,502.95	195.80
Logical reads:	43.25	754.87	1,645.36	128.39	2,142.71	1,568.91
Block changes:	16.59	202.59	1,121.16	49.24	575.05	1,067.85
Physical reads:	0.10	8.19	8,090.00	0.29	23.23	7,910.34
Physical writes:	0.61	1.76	188.52	1.81	5.01	176.80
User calls:	3.37	3.17	-5.93	9.99	9.00	-9.91
Parses:	2.97	6.99	135.35	8.83	19.84	124.69
Hard parses:	0.01	0.38	3,700.00	0.02	1.07	5,250.00
Sorts:	2.18	3.88	77.98	6.48	11.03	70.22
Logons:	0.03	0.03	0.00	0.09	0.09	0.00
Executes:	11.04	23.38	111.78	32.78	66.38	102.50
Transactions:	0.34	0.35	2.94			
				1st	2nd	Diff
% Blocks change	d per Read:			38.35	26.84	-11.51
Recursive Call %:			88.68 97.50		8.82	
Rollback per trans	action %:		20.14 0.63		-19.51	
Rows per Sort:			3.18 101.17		97.98	
Avg DB time per (Call (sec):			0.00	0.08	0.07

Top Timed Events

The Top 5 Timed Events section displays the five timed events or operations that consumed the highest percentage of total DB time in each of the snapshot sets.

1st					2nd						
Event	Wait Class	Waits	Time(s)	Avg Time(ms)	%DB time	Event	Wait Class	Waits	Time(s)	Avg Time(ms)	%DB time
CPU time			111.72		36.84	CPU time			912.19		30.14
db file sequential read	User I/O	7,132	99.95	14.01	32.96	resmgr:cpu quantum	Scheduler	1,622	333.99	205.91	11.04
control file parallel write	System I/O	2,635	36.66	13.91	12.09	library cache lock	Concurrency	3	265.85	88,617.53	8.78
log file parallel write	System I/O	2,652	30.51	11.50	10.06	db file sequential read	User I/O	17,442	247.98	14.22	8.19
log file sync	Commit	622	19.33	31.07	6.37	db file scattered read	User I/O	3,229	84.87	26.28	2.80
-db file scattered read	User I/O	932	17.09	18.33	5.63	-log file parallel write	System I/O	3,373	63.08	18.70	2.08
-						-control file parallel write	System I/O	2,774	27.83	10.03	0.92
-						-log file sync	Commit	607	25.54	42.08	0.84

In this example, CPU time is over eight times higher in the second period than in the first. The number of waits for the db file sequential read event in the second period is over double the number in the first.

Details of the AWR Compare Periods Report

The details section follows the summary of the AWR Compare Periods report, and provides statistics about the snapshot sets and loads used in the report. For example, the section includes statistics for database time, wait events, SQL execution time, and instance activity.

Supplemental Information in the AWR Compare Periods Report

The supplemental information is at the end of the AWR Compare Periods report, and provides additional information about initialization parameters and SQL statements. The init.ora Parameters section lists all the initialization parameter values for the first snapshot set. The Complete List of SQL Text section lists each statement by SQL ID and shows the text of the SQL statement.

Part IV SQL Tuning

Part IV describes how to effectively tune SQL statements and contains the following chapters:

- Chapter 9, "Identifying High-Load SQL Statements"
- Chapter 10, "Tuning SQL Statements"
- Chapter 11, "Optimizing Data Access Paths"
- Chapter 12, "Analyzing SQL Performance Impact"

Identifying High-Load SQL Statements

High-load SQL statements may consume a disproportionate amount of system resources. These SQL statements often cause a large impact on database performance and must be tuned to optimize their performance and resource consumption. Even when a database itself is properly tuned, inefficient SQL statements can significantly degrade database performance.

Identifying high-load SQL statements is an important SQL tuning activity that you must perform regularly. Automatic Database Diagnostic Monitor (ADDM) automates this task by proactively identifying potential high-load SQL statements. Additionally, you can use Oracle Enterprise Manager (Enterprise Manager) to identify high-load SQL statements that require further investigation. After you have identified the high-load SQL statements, you can tune them with SQL Tuning Advisor and SQL Access Advisor.

This chapter describes how to identify high-load SQL statements and contains the following sections:

- Identification of High-Load SQL Statements Using ADDM Findings
- Identifying High-Load SQL Statements Using Top SQL

Identification of High-Load SQL Statements Using ADDM Findings

By default, ADDM runs proactively once every hour. It analyzes key statistics gathered by the Automatic Workload Repository (AWR) over the last hour to identify any performance problems, including high-load SQL statements. When the system finds performance problems, it displays them as ADDM findings in the Automatic Database Diagnostic Monitor (ADDM) page.

ADDM provides recommendations with each ADDM finding. When a high-load SQL statement is identified, ADDM gives appropriate recommendations, such as running SQL Tuning Advisor on the SQL statement. You can begin tuning as described in Chapter 10, "Tuning SQL Statements".

See Also:

- "Overview of Automatic Database Diagnostic Monitor" on page 3-1
- "Interpretation of Automatic Database Diagnostic Monitor Findings" on page 3-8
- "Implementing Automatic Database Diagnostic Monitor Recommendations" on page 3-9

Identifying High-Load SQL Statements Using Top SQL

ADDM automatically identifies high-load SQL statements that may be causing systemwide performance degradation. Under normal circumstances, manual identification of high-load SQL statements is not necessary. In some cases, however, you may want to monitor SQL statements at a more granular level. The Top SQL section of the Top Activity page in Enterprise Manager enables you to identify high-load SQL statements for any 5-minute interval.

Figure 9–1 shows an example of the Top Activity page.

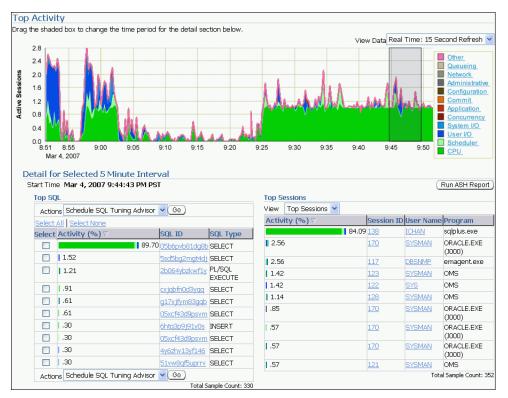


Figure 9–1 Top Activity Page

To access the Top Activity page:

1. From the Database Home page, click **Performance**.

The Performance page appears.

2. Under Additional Monitoring Links, click **Top Activity**.

The Top Activity page appears.

This page shows a 1-hour time line of the top activity running on the database. SQL statements that are using the highest percentage of database activity are listed under the Top SQL section, and are displayed in 5-minute intervals.

3. To move the 5-minute interval, drag and drop the shaded box to the time of interest.

The information contained in the Top SQL section will be automatically updated to reflect the selected time period. Use this page to identify high-load SQL statements that may be causing performance problems.

4. To monitor SQL statements for a longer duration than one hour, select **Historical** from the View Data list.

In Historical view, you can view the top SQL statements for the duration defined by the AWR retention period.

This section contains the following topics:

- Viewing SQL Statements by Wait Class
- Viewing Details of SQL Statements

Viewing SQL Statements by Wait Class

The SQL statements that appear in the Top SQL section of the Top Activity page are categorized into various wait classes, based on their corresponding color as described in the legend on the Top Activity chart.

To view the SQL statements for a particular wait class, click the block of color on the chart for the wait class, or its corresponding wait class in the legend. The Active Sessions Working page for the selected wait class appears, and the Top SQL section will be automatically updated to show only the SQL statements for that wait class.

The example in Figure 9–2 shows the Active Sessions Working page for the CPU Used wait class. Only SQL statements that are consuming the most CPU time are displayed in the Top Working SQL section.

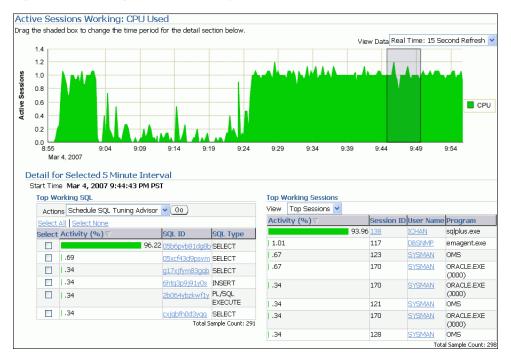


Figure 9–2 Viewing SQL Statement by Wait Class

See Also:

 "Monitoring User Activity" on page 4-2 for information about using the Active Sessions Working page

Viewing Details of SQL Statements

The Top SQL section of the Top Activity page displays the SQL statements executed within the selected 5-minute interval in descending order based on their resource consumption. The SQL statement at the top of this table represents the most resource-intensive SQL statement during that time period, followed by the second most resource-intensive SQL statement, and so on.

In the example shown in Figure 9–1 on page 9-2, the SQL statement with SQL_ID 05b6pvb81dg8b is consuming 89.7 percent of database activity and should be investigated.

To view details of SQL statements:

1. From the Database Home page, click **Performance**.

The Performance page appears.

2. Under Additional Monitoring Links, click Top Activity.

The Top Activity page appears.

3. In the Top SQL section, click the SQL ID link of the SQL statement.

The SQL Details page for the selected SQL statement appears.

4. To view SQL details for a longer time period, select **Historical** from the View Data list.

You can now view SQL details in the past, up to the duration defined by the AWR retention period.

5. Review the SQL text for the SQL statement.

The Text section contains the SQL text for the selected SQL statement.

```
Text

SELECT /*+ ORDERED USE_NL(c) FULL(c) FULL(s)*/ COUNT(*)

FROM SH.SALES S, SH.CUSTOMERS C

WHERE C.CUST ID = S.CUST ID AND CUST FIRST NAME='Dina' ORDER BY TIME ID
```

If only part of the SQL statement is displayed, then a plus sign (+) icon will appear next to the Text heading. To view the SQL text for the entire SQL statement, click the plus sign (+) icon.

6. If the SQL statement has multiple plans, then select **All** from the Plan Hash Value list to show SQL details for all plans.

Alternatively, you can select a particular plan to display SQL details for that plan only.

```
Details
Select the plan hash value to see the details below. Plan Hash Value 2043253752 💙
```

- **7.** Access the subpages from the SQL Details page to gather more information about the SQL statement, as described in the following sections:
 - Viewing SQL Statistics
 - Viewing Session Activity
 - Viewing the SQL Execution Plan
 - Viewing the SQL Tuning Information

8. If the SQL statement is a high-load SQL statement, then tune it as described in Chapter 10, "Tuning SQL Statements".

Viewing SQL Statistics

The Statistics subpage of the SQL Details page displays statistical information about the SQL statement.

To view statistics for the SQL statement:

1. On the SQL Details page, under Details, click **Statistics**.

The Statistics subpage appears.



- 2. View the statistics for the SQL statement, as described in the following sections:
 - SQL Statistics Summary
 - General SQL Statistics
 - Activity by Wait Statistics and Activity by Time Statistics
 - Elapsed Time Breakdown Statistics
 - Shared Cursors Statistics and Execution Statistics
 - Other SQL Statistics

SQL Statistics Summary The Summary section displays SQL statistics and activity on a chart.

In the Real Time view, the Active Sessions chart shows the average number of active sessions executing the SQL statement in the last hour. If the SQL statement has multiple plans and **All** is selected in the Plan Hash Value list, then the chart will display each plan in different colors, enabling you to easily spot if the plan changed and whether this may be the cause of the performance degradation. Alternatively, you can select a particular plan to display that plan only.

In the Historical view, the chart shows execution statistics in different dimensions. To view execution statistics, select the desired dimension from the View list:

- Elapsed time per execution
- Executions per hour
- Disk reads per execution
- Buffer gets per execution

This enables you to track the response time of the SQL statement using different dimensions and determine if the performance of the SQL statement has degraded based on the dimension selected.

To view statistics of the SQL statement for a particular time interval, click the snapshot icon below the chart. You can also use the arrows to scroll the chart to locate a desired snapshot.

General SQL Statistics The General section enables you to identify the origin of the SQL statement by listing the following information:

- Module, if specified using the DBMS_APPLICATION_INFO package
- Action, if specified using the DBMS_APPLICATION_INFO package
- Parsing schema, or the database users account that is used to execute the SQL statement
- PL/SQL source, or the code line if the SQL statement is part of a PL/SQL program unit

Activity by Wait Statistics and Activity by Time Statistics The Activity by Wait and Activity by Time sections enable you to identify where the SQL statement spent most of its time. The Activity by Wait section contains a graphical representation of how much elapsed time is consumed by CPU and by remaining waits. The Activity by Time section breaks out the total elapsed time into CPU time and wait time by seconds.

Elapsed Time Breakdown Statistics The Elapsed Time Breakdown section enables you to identify if the SQL statement itself is consuming a lot of time, or if the total elapsed time is inflated due to the amount of time the originating program or application is spending with the PL/SQL or Java engine. If the PL/SQL time or Java time makes up a significant portion of the elapsed time, then there may be minimal benefit gained by tuning the SQL statement. Instead, you should examine the application to determine how the PL/SQL time or Java time can be reduced.

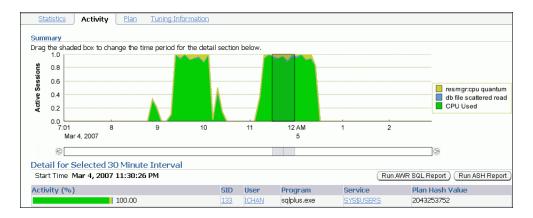
Shared Cursors Statistics and Execution Statistics The Shared Cursors Statistics and Execution Statistics sections provide information about the efficiency of various stages of the SQL execution process.

Other SQL Statistics The Other Statistics section provides additional information about the SQL statement, such as average persistent and run-time memory.

Viewing Session Activity

To view session activity for the SQL statement, in the Details section, click Activity.

The Activity subpage contains a graphical representation of the session activity.



The Activity subpage displays details of various sessions executing the SQL statement. The Active Sessions chart profiles the average number of active sessions over time. You can drag the shaded box to select a 5-minute interval. The Detail for Selected 5 Minute Interval section lists the sessions that executed the SQL statement during the selected 5-minute interval. The multicolored bar in the Activity % column depicts how the database time is divided for each session while executing the SQL statement. To view more details for a particular session, click the link in the **SID** column of the session you want to view to display the Session Details page.

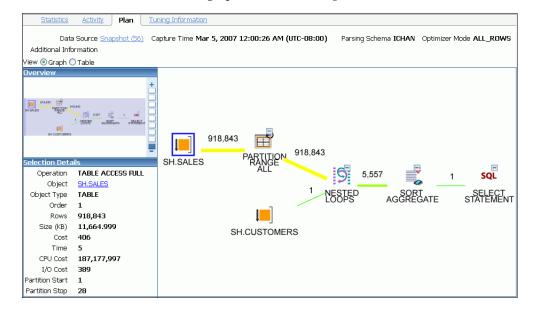
See Also:

 "Monitoring Top Sessions" on page 4-5 for information about monitoring session activity and details

Viewing the SQL Execution Plan

To view the execution plan for the SQL statement, in the Details section, click **Plan**. The execution plan for a SQL statement is the sequence of operations Oracle performs to run the statement.

The Plan subpage displays the execution plan for the SQL statement in a graph view and a table view.



To view the SQL execution in a graph view, click **Graph**.

In the graph view, you can display details about the operations shown in the execution plan by selecting the operation in the graph. Details about the selected operations are displayed under Selection Details. If the selected operation is on a particular database object (such as a table), then you can view further details about the database object by clicking the **Object** link.

To view the SQL execution in a table view, click **Table**.

<u>Statistics</u> <u>Activity</u> Plan	Tuning Information									
Data Source <u>Snapshot (56)</u> Additional Information	Capture Time Mar 5, 2007	7 12:00	:26 AM	(UTC-08:	00) Pars	ing S	chema IC I	HAN Opt	imizer Moc	e ALL_ROWS
/iew 🔘 Graph 💽 Table										
Expand All Collapse All										
						СРИ				Query Block Name/Object
Operation	Object	Order	Rows	Bytes	Cost	(%)	Time	Start	Stop	Alias
SELECT STATEMENT		6	0	0	300,620,031	100	0:0:0			
▼SORT AGGREGATE		5	1	25	0		0:0:0			SEL\$1
VINESTED LOOPS		4	5,557	135.669K	300,620,031	1	1002:4:1			
V PARTITION RANGE ALL		2	918,843	11.392M	406	4	0:0:5	1	28	
TABLE ACCESS FULL	SH.SALES	1	918,843	11.392M	406	4	0:0:5	1		SEL\$1 / S@SEL\$1
TABLE ACCESS FULL	SH.CUSTOMERS	3	1	12	327	1	0:0:4			SEL\$1 / C@SEL\$1

Oracle Database compares the cost for the query, with and without query rewrite, and selects the least costly alternative. If a rewrite is necessary, then the query rewrite and its cost benefit are displayed in the Explain Rewrite section.

See Also:

• Chapter 10, "Tuning SQL Statements" for information about execution plan and the query optimizer

Viewing the SQL Tuning Information

To view the tuning information for the SQL statement, in the Details section, click **Tuning Information**.

The Tuning Information subpage contains information about the SQL tuning tasks and the SQL profiles recommended by SQL Tuning Advisor for the SQL statement.

The SQL Profiles and Outlines section displays SQL profiles and outlines associated with the SQL statement. A SQL profile contains additional statistics of the SQL statement. An outline contains hints for the SQL statement for the query optimizer. Both are used by the query optimizer to generate a better execution plan for the SQL statement.

The SQL Tuning History section displays a history of SQL Tuning Advisor or SQL Access Advisor tasks.

The ADDM Findings for this SQL During Historic Period section displays the number of occurrences of ADDM findings that are associated with the SQL statement.

Statistics Activity	<u>Plan</u>	Tuning Inform	nation						
SQL Profiles and Outli	nes								
A SQL Profile contains a An outline contains hint						enerate a better execution plan. execution plan.			
Change Category	Delete) Di:	sable/Enable)							
Select Name			Туре	Category	Status	Created			
SYS_SQLPROF_	_0144488f7d	65c000	SQL Profile	DEFAULT	ENABLED	Mar 5, 2007 5:15:57 AM			
SQL Tuning History									
The following SQL tunir	ng tasks prov	ide the recomm	endations to	tune this SQL s	tatement.				
Advisor Task Name		Ad	lvisor Task ()wner	Task C	Completion			
SQL TUNING 1173098	908778	SY	5		Mar 5,	2007 5:06:22 AM			
ADDM Findings for th	is SQL durin	g historic perio	bd						
Finding Name Occurrences (last 24 hrs) ▽									
Top SQL by DB Time	12 of 24								
CPU Usage	10 of 24								

See Also:

- Chapter 10, "Tuning SQL Statements" for information about SQL Tuning Advisor and SQL profiles
- "Managing SQL Profiles" on page 10-19
- Chapter 11, "Optimizing Data Access Paths" for information about SQL Access Advisor

Tuning SQL Statements

A SQL statement expresses the data you want Oracle Database to retrieve. For example, you can use a SQL statement to retrieve the names of all employees in a department. When Oracle Database executes the SQL statement, the query optimizer (also called simply the optimizer) first determines the best and most efficient way to retrieve the results.

The optimizer determines whether it is more efficient to read all data in the table, called a full table scan, or use an index. It compares the cost of all possible approaches and chooses the approach with the least cost. The access method for physically executing a SQL statement is called an execution plan, which the optimizer is responsible for generating. The determination of an execution plan is an important step in the processing of any SQL statement, and can greatly affect execution time.

The query optimizer can also help you tune SQL statements. By using SQL Tuning Advisor and SQL Access Advisor, you can invoke the query optimizer in advisory mode to examine a SQL statement or set of statements and determine how to improve their efficiency. SQL Tuning Advisor and SQL Access Advisor can make various recommendations, such as creating SQL profiles, restructuring SQL statements, creating additional indexes or materialized views, and refreshing optimizer statistics. Additionally, Oracle Enterprise Manager (Enterprise Manager) enables you to accept and implement many of these recommendations with just a few mouse clicks.

SQL Access Advisor is primarily responsible for making schema modification recommendations, such as adding or dropping indexes and materialized views. SQL Tuning Advisor makes other types of recommendations, such as creating SQL profiles and restructuring SQL statements. If significant performance improvements can be gained by creating a new index, then SQL Tuning Advisor may recommend it. However, such recommendations should be verified by running SQL Access Advisor using a SQL workload that contains a set of representative SQL statements.

This chapter describes how to tune SQL statements using the SQL Tuning Advisor and contains the following sections:

- Tuning SQL Statements Using SQL Tuning Advisor
- Managing SQL Tuning Sets
- Managing SQL Profiles
- Managing SQL Execution Plans

See Also:

- Chapter 9, "Identifying High-Load SQL Statements"
- Chapter 11, "Optimizing Data Access Paths" for information about SQL Access Advisor

Tuning SQL Statements Using SQL Tuning Advisor

You can use SQL Tuning Advisor to tune one or more SQL statements. When tuning multiple statements, keep in the mind that SQL Tuning Advisor does not recognize interdependencies between the SQL statements. Instead, it is intended as a convenient way to run SQL Tuning Advisor for a large number of SQL statements.

Besides enabling you to tune SQL statements manually, Oracle Database generates SQL tuning reports automatically. 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.

Tuning SQL Manually Using SQL Tuning Advisor

As described in Chapter 9, "Identifying High-Load SQL Statements", ADDM automatically identifies high-load SQL statements. In such cases, simply click **Schedule/Run SQL Tuning Advisor** on the Recommendation Detail page to invoke SQL Tuning Advisor.

To tune SQL statements manually using SQL Tuning Advisor:

1. On the Database Home page, under Related Links, click Advisor Central.

The Advisor Central page appears.

2. Under Advisors, click SQL Advisors.

The SQL Advisors page appears.

3. Under SQL Tuning Advisor, click SQL Tuning Advisor.

The Schedule SQL Tuning Advisor page appears.

Schedule SQL Tuning Advisor	
Specify the following parameters to schedule a job to run the SQL Tuning Advisor.	Cancel
* Name SOL TUNING 1173098703162	SQL Tuning Advisor Data Source Links
Description	The SQL Tuning Advisor analyzes individual SQL statements and makes recommendations for improving
* SQL Tuning Set	their performance. You can click on one of the following sources, which will lead you to a data source where you
SQL Tuning Set Description	can tune SQL statements using the SQL Tuning Advisor.
SQL Statements Counts 0	Top Activity Historical SQL (AWR) SQL Tuning Sets
SQL Statements	
Scope	
CLimited	
Ocomprehensive	
Time Limit per Statement (minutes) 5	
Total Time Limit (minutes) 30	
Schedule	
Time Zone Pacific/Pago_Pago 🔽	
 Immediately 	
OLater	
Date Mar 5, 2007	
Time 4 ¥ 45 ¥ 00 ¥ • AM O PM	

4. In the Name field, enter a name for the SQL tuning task.

If unspecified, a system-generated name is used.

- **5.** Do one of the following:
 - To run a SQL tuning task for one or more high-load SQL statements, under SQL Tuning Advisor Data Source Links, click Top Activity.

The Top Activity page appears.

Under Top SQL, select the SQL statement you want to tune and click **Schedule SQL Tuning Advisor**. For information about identifying high-load SQL statements using the Top Activity page, see "Identifying High-Load SQL Statements Using Top SQL" on page 9-2.

 To run a SQL tuning task for historical SQL statements from the Automatic Workload Repository (AWR), under SQL Tuning Advisor Data Source Links, click Historical SQL (AWR).

The Historical SQL (AWR) page appears.

Under Historical SQL (AWR), click the band below the chart, and select the 24-hour interval for which you want to view SQL statements that ran on the database. Under Detail for Selected 24 Hour Interval, select the SQL statement you want to tune, and click **Schedule SQL Tuning Advisor**.

• To run a SQL tuning task for a SQL Tuning Set, click **SQL Tuning Sets**.

The SQL Tuning Sets page appears.

Select the SQL Tuning Set that contains the SQL statements you want to tune and click **Schedule SQL Tuning Advisor**. For information about creating SQL Tuning Sets, see "Creating a SQL Tuning Set" on page 10-8.

The Schedule SQL Tuning Advisor page reappears.

6. To display the SQL text of the selected statement, expand SQL Statements.

▼SQL Statements	
	Parsing Schema
SELECT /*+ ORDERED USE_NL(c) FULL(c) FULL(s)*/ COUNT(*) FROM SH.SALES S, SH.CUSTOMERS C WHERE C.CUST_ID = S.CUST_ID AND CUST_FIRST_NAME='Dina' ORDER BY TIME_ID	ICHAN

- 7. Under Scope, select the scope of tuning to perform. Do one of the following:
 - Select Limited.

A limited scope takes approximately 1 second to tune each SQL statement but does not recommend a SQL profile.

• Select **Comprehensive**, and then set a time limit (in minutes) for each SQL statement in the **Time Limit per Statement** field, and a total time limit (in minutes) in the **Total Time Limit** field.

A comprehensive scope performs a complete analysis and recommends a SQL profile, when appropriate, but may take much longer. Note that setting the time limit too small may affect the quality of the recommendations.

Running a SQL Tuning Advisor task in comprehensive mode may take several minutes to tune a single SQL statement. This mode is both time and resource intensive because each time a query must be hard-parsed. Thus, you should only use comprehensive scope for high-load SQL statements that have a significant impact on the entire system.

Scope
 Comprehensive
Time Limit per Statement (minutes) 5
Total Time Limit (minutes) 30

For information about SQL profiles, see "Managing SQL Profiles" on page 10-19.

- **8.** Under Schedule, do one of the following:
 - Select **Immediately** to run the SQL tuning task immediately, and then proceed to Step 10 after the SQL Tuning Results page appears.
 - Select Later to schedule a specific time in the future, and click OK.

Schedule		
Time Zone	Pacif	ific/Pago_Pago 🛛 👻
💿 Immedia	ately	
OLater		
1	Date	Mar 5, 2007
		(example: Mar 5, 2007)
T	Time	4 🔽 48 🔽 00 👻 💿 AM 🔾 PN

- 9. Optionally, on the Advisor Central page, do one of the following:
 - To view results for the SQL tuning task after it completes, select the SQL Tuning Advisor task and click **View Result**.

The SQL Tuning Results page appears. Select the recommendation you want to implement and click **View**.

Proceed to the next step.

- To delete a SQL tuning task, select the SQL Tuning Advisor task and click Delete.
- To reschedule a SQL tuning task, select the SQL Tuning Advisor task. From the Actions list, select **Re-schedule** and click **Go**.
- To interrupt a SQL tuning task that is running, select the SQL Tuning Advisor task. From the Actions list, select **Interrupt** and click **Go**.
- To cancel a scheduled SQL tuning task, select the SQL Tuning Advisor task. From the Actions list, select **Cancel** and click **Go**.
- To change the expiration of a SQL tuning task, select the SQL Tuning Advisor task. From the Actions list, select **Change Expiration** and click **Go**.

Results of each advisor run are stored in the database so that they can be referenced later. This data is stored until it expires, at which point it will be deleted by the AWR purging process.

• To edit a scheduled SQL tuning task, select the SQL Tuning Advisor task. From the Actions list, select **Edit** and click **Go**.

Result	8										
Viev	View Result) Delete) Actions Re-schedule Go										
Select	Advisory Type			Description	User	Status	Start Time 🗸	Duration (seconds)	Expires In (days)		
۲	SQL Tuning Advisor	SOL TUN	Cancel Change Expiration Edit		SYS		Mar 5, 2007 5:30:00 AM		30		
0	SQL Tuning Advisor	<u>SYS AUT</u>	O SQL TUNING TASK	Automatic SQL Tuning Task	SYS		Mar 4, 2007 4:42:33 PM	965	UNLIMITED		

10. On the SQL Tuning Results page, click **View**.

The Recommendations for SQL ID page appears.

If you used a SQL Tuning Set, then multiple recommendations may be displayed. To help you decide whether or not to implement a recommendation, an estimated benefit of implementing the recommendation is displayed in the Benefit (%) column. The Rationale column displays an explanation of why the recommendation is made.

- **11.** Optionally, if multiple recommendations are displayed, then do one of the following:
 - To view the original execution plan for the SQL statement, click **Original Explain Plan**.
 - To compare the new and original execution plans, click the icon in the **Compare Explain Plans** column.
 - To view the new execution plan for the SQL statement, click the icon in the **New Explain Plan** column.

For information about viewing execution plans, see "Viewing the SQL Execution Plan" on page 9-7.

12. To implement the recommendation, click **Implement**.

Re	ecomr	nendat	ions for SQL ID:05b6pvb81dg	8b							
							Return				
Onl	Only one recommendation should be implemented.										
	SQL 1	fext									
	SELEC1	Г /*+ ORE	DERED USE NL(c) FULL(c) FULL(s)*/ COUN	IT (*) FROM SH.SALES S, SH.CUSTOME	ERS C WHER	E C.CUST ID = S.CUST	ID AND				
	CUST	FIRST NA	ME='Dina' ORDER BY TIME ID								
	~ 1		1.0								
	Selec	t Recon	nmendation								
	Origin	nal Explain	Plan (Annotated)								
	Imp	lement)									
						Benefit New Explain	Compare Explain				
	Select	Туре	Findings	Recommendations	Rationale	(%) Plan	Plans				
	۲		A potentially better execution plan was found for this statement.	Consider accepting the recommended SQL profile.		99.63 👧	øa				

The SQL Tuning Results page appears with a confirmation that the recommended action was completed.

SQL_TUNING Results:SQL_TUNING_11730989	81180							
Confirmation The recommended SQL Profile has been created success	sfully.							
Status COMPLETED SQLID 05b6pvb81dg8b Time Limit (seconds) 1800 Recommendations			Pa Running	- Com	Started pleted	ar 5, 2007 5:0 Mar 5, 2007 Mar 5, 2007 61	5:05:21 AM	resh
View								
Select SQL Text	Parsing Schema	SQL ID	Statistics	SQL Profile	Index	Restructure SQL	Miscellaneous	Error
 SELECT /*+ ORDERED USE_NL(c) FULL(c) FULL(s) */ COUNT(*) FROM SH.SALES S, SH.CUSTOMERS C WHERE C.CUST 		05b6pvb81dg8b		~				

Viewing Automatic SQL Tuning Results

By analyzing information stored in the Automatic Workload Repository (AWR), the database can identify routine maintenance tasks that need to be run. The automated maintenance tasks infrastructure (known as AutoTask) schedules these tasks to run in Oracle Scheduler time periods known as maintenance windows. By default, one window is scheduled for each day of the week. You can customize attributes of these maintenance windows, including start and end time, frequency, and days of the week.

AutoTask automatically schedules SQL Tuning Advisor to run during the maintenance windows. You can view the results of automated execution of SQL Tuning Advisor on observed high-load SQL statements.

To view automatic SQL tuning results:

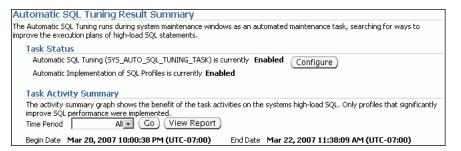
- 1. On the Database Home page, under Related Links, click Advisor Central.
 - The Advisor Central page appears.
- 2. Under Advisors, click SQL Advisors.

The SQL Advisors page appears.

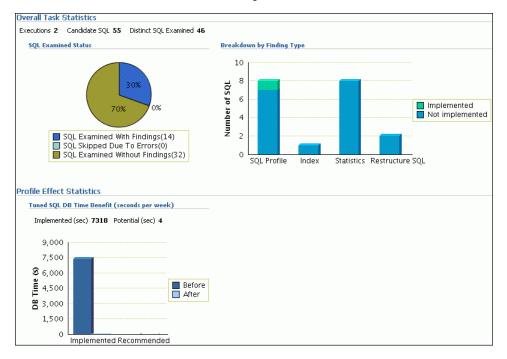
3. Under SQL Tuning Advisor, click Automatic SQL Tuning Results.

The Automatic SQL Tuning Result Summary page appears.

The top half of the page includes sections for the status and activity summary of the SQL Tuning task.



The bottom half of the Automatic SQL Tuning Result Summary page shows statistics for the overall task and the profile effect.



The Tuned SQL DB Time Benefit chart estimates the weekly DB time saved by SQL profiles that were automatically implemented. The chart also shows time that could be saved if other recommended SQL profiles were implemented. For example, the **Before** bar in the **Implemented** set aggregates the DB times during the week before tuning for all SQL statements with profiles implemented. The **After** bar projects the new weekly cumulative DB time, calculated by lowering the time of each SQL statement according to the benefit found by test execution. Thus, the **Implemented** set shows DB time benefit that has already been realized, whereas the **Recommended** set shows the potential benefit of profiles that were not automatically accepted by the SQL Tuning Advisor.

4. Optionally, in the Task Status section, click **Configure** to change the attributes of the Automatic SQL Tuning task.

The Automated Maintenance Tasks Configuration page appears.

In this page, you can enable or disable the Automatic SQL Tuning task and specify which days it should run. Click **Apply** or **Revert** to return to the previous page.

5. In the Task Activity Summary section, leave **All** selected for the **Time Period** and then click **View Report**.

The Automatic SQL Tuning Result Details page appears.

The page lists SQL statements that have been automatically selected by the database as candidates for SQL tuning.

n Date	Mar 20, 2007 10:00:38 PM (UTC-07:00)			End Date	Mar 22, 2007	11:38:09	AM (UTC-07:	00)	
Recom	mendations								
	ofiles that significantly improve SQL performan	nce were in	nplemented.				~		-
Viev	w Recommendations							1-25 of 55	 Next 25
Select	SQL Text	Parsing Schema		Statistics	SQL Profile $ abla$	Index	Restructure SQL	Miscellaneous	Error Dat
	SELECT /*+ ORDERED USE_NL(c) FULL(c) FUL	SH	5mxdwvuf9j3vp		(99.6%) 🗸	(62%) 🗸			3/20
	SELECT COUNT(*) FROM ALL_PROCEDURES WHER	SYSMAN	<u>Oms7aqju3b8s9</u>		(98.4%) 🗸				3/20
	SELECT SUM(USED), SUM(TOTAL) FROM (SELEC	SYSMAN	<u>gjm43un5cy843</u>	~	(84%) 🧹				3/20
0	SELECT TASK_TGT.TARGET_GUID TARGET_GUID,	SYSMAN	<u>4jrfrtx4u6zcx</u>	~	(71.7%) 🗸				3/20
0	SELECT /*+ ordered */ 'TABLE' OBJ_TYPE, 	SYS	<u>3xua7cqdv0qhr</u>		(50%) 🗸		×		3/20
	SELECT /*+ NO_MERGE(t) USE_NL(t) USE_NL(DBSNMP	<u>5qmzpdt3hb×ur</u>	~	(44.7%) 🗸			~	3/20
0	SELECT /*+ ordered full(t) use_hash(t) *	SYS	f72wuq53yw45k	-	(42.1%) 🧹				3/20
0	select file# from file\$ where ts#=:1	SYS	bsa0wjtftq3uw		(33.3%) 🌙				3/20

6. Under Recommendations, select a SQL statement and then click **View Recommendations**.

The Recommendations for SQL ID page appears.

Rec	omm	enda	tions for S	SQL ID:5mxdwvuf9j3vp				
Only	one rec	ommer	ndation should	be implemented.				Return
· ·	SOL Te		laadon shoala					
3	SELECT.	/*+ OF	RDERED USE IN IAME='Dina' OF	IL(c) FULL(c) FULL(s)*/ COUNT(*) FROM SA RDER BY TIME_ID	LES S, CUSTOMERS C WHERE C	.CUST II) = 5.CU9	5T ID AND
5			mmendatio	······································				
(Origi	nal Ex	plain Plan (A	nnotated)				
	Imp	lemer	nt)					
	Select	Туре	Findings	Recommendations	Rationale	Benefit	Explain	Compare Explain Plans
		Profile		Consider accepting the recommended SQL profile.		99.56	.00 .	øa
	C		execution plan of this statement	Consider running the Access Advisor to improve the physical schema design or creating the recommended index. SH.CUSTOMERS("CUST_FIRST_NAME") SH.SALES("CUST_ID")	Creating the recommended indices significantly improves the execution plan of this statement. However, it might be preferable to run "Access Advisor" using a representative SQL workload as opposed to a single statement. This will allow to get comprehensive index recommendations which takes into account index maintenance overhead and additional space consumption.	62.03	.00.	ØQ

This page can include recommendations for SQL profiles and indexes. See "Tuning SQL Manually Using SQL Tuning Advisor" on page 10-2 to learn how to implement recommendations made by SQL Tuning Advisor.

Managing SQL Tuning Sets

A SQL Tuning Set is a database object that includes one or more SQL statements and their execution statistics and execution context. You can use the set as an input source for various advisors, such as SQL Tuning Advisor, SQL Access Advisor, and SQL Performance Analyzer. You can load SQL statements into a SQL Tuning Set from different SQL sources, such as AWR, the cursor cache, or high-load SQL statements that you identified.

A SQL Tuning Set includes the following:

- A set of SQL statements
- Associated execution context, such as user schema, application module name and action, list of bind values, and the cursor compilation environment
- Associated basic execution statistics, such as elapsed time, CPU time, buffer gets, disk reads, rows processed, cursor fetches, the number of executions, the number of complete executions, optimizer cost, and the command type
- Associated execution plans and row source statistics for each SQL statement (optional)

SQL statements can be filtered using the application module name and action, or any of the execution statistics. In addition, SQL statements can be ranked based on any combination of execution statistics.

SQL Tuning Sets are transportable across databases and can be exported from one system to another, allowing SQL workloads to be transferred between databases for remote performance diagnostics and tuning. When high-load SQL statements are identified on a production system, it may not be desirable to perform investigation and tuning activities on the production system directly. This feature enables you to transport the high-load SQL statements to a test system, where they can be safely analyzed and tuned. For information about transporting SQL Tuning Sets, see *Oracle Database Performance Tuning Guide*.

Using Oracle Enterprise Manager, you can manage SQL Tuning Sets by doing the following:

- Creating a SQL Tuning Set
- Dropping a SQL Tuning Set
- Transporting SQL Tuning Sets

Creating a SQL Tuning Set

This section describes how to create a SQL Tuning Set by using Oracle Enterprise Manager.

To create a SQL Tuning Set:

- 1. Specify the initial options for the SQL Tuning Set, as described in "Creating a SQL Tuning Set: Options" on page 10-9.
- 2. Select the load method to use for collecting and loading SQL statements into the SQL Tuning Set, as described in "Creating a SQL Tuning Set: Load Method" on page 10-10.
- **3.** Specify the filter options for the SQL Tuning Set, as described in "Creating a SQL Tuning Set: Filter Options" on page 10-13.

4. Schedule and submit a job to collect the SQL statements and load them into the SQL Tuning Set, as described in "Creating a SQL Tuning Set: Schedule" on page 10-15.

Creating a SQL Tuning Set: Options

The first step in creating a SQL Tuning Set is to specify initial options for the set such as name, owner, and description.

To specify options for creating a SQL Tuning Set:

1. On the Database Performance page, under Additional Monitoring Links, click SQL Tuning Sets.

The SQL Tuning Sets page appears. Existing SQL Tuning Sets are displayed on this page.

SQL Tuning Sets										
Page Refreshed Mar 6, 2007 2:37:51 AM PST (Refresh)										
A SQL Tuning Set is a collection of SQL Statements that can be used for tuning purposes.										
Search Go										
Filter on a name or partial name					Create Import					
Details Drop Export Sch	nedule SQL A	Advisor								
Select Name	Schema	Description	SQL Count	Created	Last Modified \bigtriangledown					
O CURSOR CACHE 070306	SYS	Cursor cache	762	3/6/07 2:33 AM	3/6/07 2:34 AM					

2. Click Create.

The Create SQL Tuning Set: Options page appears.

- 3. In the SQL Tuning Set Name field, enter a name for the SQL Tuning Set.
- 4. In the **Owner** field, enter the owner of the SQL Tuning Set.
- 5. In the **Description** field, enter a description of the SQL Tuning Set.

Create SQL Tuning	g Set: Options	
Database orcl111		Cancel Step 1 of 5 Next
* SQL Tuning Set Name	AWR_SNAPSHOTS_070306_CUSTOM	
Owner	eve	-
Owner	515	
Description	Contains custom snapshots from 3/4/07 to 3/	n,
_	· · · · · · · · · · · · · · · · · · ·	-
Create an empty SQL	tuning set	

- **6.** Optionally, if you want to create an empty SQL Tuning Set and add SQL statements to it at a later time, then complete the following steps:
 - a. Enable Create an empty SQL tuning set.
 - **b.** Click Next.

The Create SQL Tuning Set: Review page appears.

c. Review the SQL Tuning Set options that you have selected and click Submit.

The empty SQL Tuning Set is created. You can add SQL statements to it at a later time.

7. Click Next.

The Create SQL Tuning Set: Load Methods page appears.

Create SQL Tuning Set: Load Meth	iods
Database orcl111	Finish Cancel Back Step 2 of 5 Next
Pick one of the load methods to collect and load	SQL statements into the SQL tuning set.
 Incrementally capture active SQL statement 	x over a period of time from the cursor cache
Specify the duration within which the SQL statements from the cursor cache will be o	statements will be collected, and specify frequency over which the active SQL collected repeatedly.
Duration 24 Hours 💌	
Frequency 5 Minutes 💙	
\bigcirc Load SQL statements one time only	
Data Source Cursor Cache 💌	

8. Proceed to the next step, as described in "Creating a SQL Tuning Set: Load Method" on page 10-10.

Creating a SQL Tuning Set: Load Method

After options are specified for the SQL Tuning Set, select the load method to use for collecting and loading SQL statements into the SQL Tuning Set, as described in the following sections:

- Loading Active SQL Statements Incrementally from the Cursor Cache
- Loading SQL Statements from the Cursor Cache
- Loading SQL Statements from AWR Snapshots
- Loading SQL Statements from AWR Baselines
- Loading SQL Statements from a User-Defined Workload

Tip: Before selecting the load method for the SQL Tuning Set, create a SQL Tuning Set and specify the initial options, as described in "Creating a SQL Tuning Set: Options" on page 10-9

Loading Active SQL Statements Incrementally from the Cursor Cache You can load active SQL statements from the cursor cache into the SQL Tuning Set incrementally over a specified period of time. This enables you to not only collect current and recent SQL statements stored in the SQL cache, but also SQL statements that will run during the specified time period in the future.

To load active SQL statements incrementally from the cursor cache:

- 1. On the Create SQL Tuning Set: Load Methods page, select **Incrementally capture active SQL statements over a period of time from the cursor cache**.
- **2.** In the **Duration** field, specify how long active SQL statements will be captured.
- **3.** In the **Frequency** field, specify how often active SQL statements will be captured during the specified duration.
- 4. Click Next.

The Create SQL Tuning Set: Filter Options page appears.

5. Proceed to the next step, as described in "Creating a SQL Tuning Set: Filter Options" on page 10-13.

Loading SQL Statements from the Cursor Cache You can load SQL statements from the cursor cache into the SQL Tuning Set. However, because only current and recent SQL statements are stored in the SQL cache, collecting these SQL statements only once may

result in a SQL Tuning Set this is not representative of the entire workload on your database.

To load SQL statements from the cursor cache:

- 1. On the Create SQL Tuning Set: Load Methods page, select Load SQL statements one time only.
- 2. In the Data Source field, select Cursor Cache.

Output to the second				
Data Source	Cursor Cache	*		

3. Click Next.

The Create SQL Tuning Set: Filter Options page is shown.

4. Proceed to the next step, as described in "Creating a SQL Tuning Set: Filter Options" on page 10-13.

Loading SQL Statements from AWR Snapshots You can load SQL statements captured in AWR snapshots. This is useful when you want to collect SQL statements for specific snapshot periods of interest that can be used for later comparison or tuning purposes.

To load SQL statements from AWR snapshots:

- 1. On the Create SQL Tuning Set: Load Methods page, select Load statements one time only.
- 2. In the Data Source field, select AWR Snapshots.
- **3.** In the **AWR Snapshots** field, select the snapshots to include. Do one of the following:
 - Select Last 24 hours and then go to Step 5.

Only snapshots that are captured and stored in AWR in the last 24 hours will be included.

• Select Last 7 days and then go to Step 5.

Only snapshots that are captured and stored in AWR in the last 7 days will be included.

Select Last 31 days and then go to Step 5.

Only snapshots that are captured and stored in AWR in the last 31 days will be included.

• Select **ALL** and then go to Step 5.

All snapshots that are captured and stored in AWR will be included.

• Select **Customize** and then proceed to Step 4.

Only snapshots that are captured and stored in AWR during a customized time period that you specify will be included.

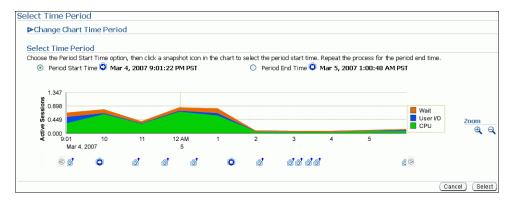
- **4.** To select a customized time period of snapshots to include, complete the following steps:
 - a. Select Customize and click Go.

0	●Load SQL statements one time only				
	Data Source	AWR Snapshots			1
	AWR Snapshots Customize				Go

The Select Time Period window opens.

- **b.** For the starting snapshot, select **Period Start Time** and click the snapshot icon below the Active Session graph that corresponds to the desired start time.
- **c.** For the ending snapshot, select **Period End Time** and click the snapshot icon below the Active Session graph that corresponds to the desired end time.
- d. Click Select.

In this example, the snapshot taken on March 4, 2007 at 9:01 p.m. is selected as the start time, and the snapshot taken on March 5, 2007 at 1:00 a.m. is selected as the end time.



5. Click Next.

The Create SQL Tuning Set: Filter Options page is shown.

6. Proceed to the next step, as described in "Creating a SQL Tuning Set: Filter Options" on page 10-13.

Loading SQL Statements from AWR Baselines You can load SQL statements captured in AWR baselines. This is useful when you want to collect SQL statements that are representative of a time period during known performance levels that can be used for later comparison or tuning purposes.

To load SQL statements from AWR baselines:

- 1. On the Create SQL Tuning Set: Load Methods page, select Load SQL statements one time only.
- 2. In the Data Source field, select AWR Baseline.
- 3. In the AWR Baseline field, select the baseline to include.

⊙ Load SQL state	ements one time only	
Data Source	AWR Baseline 💙	
AWR Base	eline BASELINE_PEAK_LOAD	, s

4. Click Next.

The Create SQL Tuning Set: Filter Options page is shown.

5. Proceed to the next step, as described in "Creating a SQL Tuning Set: Filter Options" on page 10-13.

Loading SQL Statements from a User-Defined Workload You can load SQL statements by importing from a table or view. This is useful if the workload you want to analyze is not currently running on the database or captured in an existing AWR snapshot or AWR baseline.

There are no restrictions on which schema the workload resides in, the name of the table, or the number of tables that you can define. The only requirement is that the format of the table must match format of the USER_WORKLOAD table.

To load SQL statements from a user-defined workload:

- 1. On the Create SQL Tuning Set: Load Methods page, select Load statements one time only.
- 2. In the Data Source field, select User-Defined Workload.
- 3. In the User-Defined Workload field, select the table or view to include.

O Load SQL statements one time only			
Data Source User-Defined	Workload 💌		
User-Defined Workload	I IMMCHAN.BAD_SQL		

4. Click Next.

The Create SQL Tuning Set: Filter Options page is shown.

5. Proceed to the next step, as described in "Creating a SQL Tuning Set: Filter Options" on page 10-13.

Creating a SQL Tuning Set: Filter Options

After the load method is selected, you can apply filters to reduce the scope of the SQL statements found in the SQL Tuning Set. While using filters is optional, it can be very beneficial due to the following:

- Using filters directs the various advisors that use the SQL Tuning Set as a workload source—such as SQL Tuning Advisor, SQL Access Advisor, and SQL Performance Analyzer—to make recommendations based on a specific subset of SQL statements, which may lead to better recommendations.
- Using filters removes extraneous SQL statements from the SQL Tuning Set, which
 may greatly reduce processing time when it is used as a workload source for the
 various advisors.

Tip: Before you can specify the filter options for the SQL Tuning Set, do the following:

- Create a SQL Tuning Set and specify the initial options, as described in "Creating a SQL Tuning Set: Options" on page 10-9
- Select the load method, as described in "Creating a SQL Tuning Set: Load Method" on page 10-10

To specify filter options for a SQL Tuning Set:

1. On the Create SQL Tuning Set: Filter Options page, specify the values of filter conditions that you want use in the search in the **Value** column, and an operator or a condition in the **Operator** column.

Only the SQL statements that meet all of the specified filter conditions will be added to the SQL Tuning Set. Unspecified filter values will not be included as filter conditions in the search.

By default, the following filter conditions are displayed:

- Parsing Schema Name
- SQL Text
- SQL ID
- Elapsed Time (sec)

Create SQL Tuning Se	t: Filter Options						
Database orcl111			Finish Cancel	Back Step 3 of 5 Next			
Total Number of SQL Statements							
Top N <all></all>	Sorted By EL/	APSED_TIME					
Filter Conditions							
		r conditions will be included as sea	rch results. Rows with an em	pty value in the 'Value'			
column will not be included		arch.					
Plan Hash Value 🛛 🗸 🗛	dd a Filter or Column)						
Filter Attribute	Operator	Value	Show Column	Remove			
Parsing Schema Name	= *						
SQL Text	LIKE 🛩						
SQL ID	= 💙						
Elapsed Time (sec)	>= 🗸			2			

2. To add filter conditions, under Filter Conditions, select the filter condition you want to add and click **Add a Filter or Column**.

The available filter conditions include the following:

- Plan hash value
- Module
- Action
- Buffer gets
- Disk reads
- Disk writes
- Rows processed
- Fetches
- Executions
- End of fetch count
- Command type

After the desired filter conditions have been added, specify their values in the **Value** column, and an operator or a condition in the **Operator** column.

- **3.** To remove any unused filter conditions, click the icon in the **Remove** column for the corresponding filter condition you want to remove.
- 4. Click Next.

The Create SQL Tuning Set: Schedule page appears.

5. Proceed to the next step, as described in "Creating a SQL Tuning Set: Schedule" on page 10-15.

Creating a SQL Tuning Set: Schedule

After the filter options are specified for the SQL Tuning Set, you can schedule and submit a job to collect the SQL statements and load them into the SQL Tuning Set.

Tip: Before you can schedule a job to create the SQL Tuning Set, do the following:

- Create a SQL Tuning Set and specify the initial options, as described in "Creating a SQL Tuning Set: Options" on page 10-9.
- Select the load method, as described in "Creating a SQL Tuning Set: Load Method" on page 10-10.
- Specify the filter options, as described in "Creating a SQL Tuning Set: Filter Options" on page 10-13.

To schedule and submit a job to create a SQL Tuning Set:

- 1. On the Create SQL Tuning Set: Schedule page, under Job Parameters, enter a name in the **Job Name** field if you do not want to use the system-generated job name.
- 2. In the **Description** field, enter a description of the job.
- **3.** Under Schedule, do one of the following:
 - Immediately to run the job immediately after it has been submitted
 - Later to run the job at a later time as specified using the Time Zone, Date, and Time fields

Create SQL Tuning Set: Schedule	
Database orcl111	Finish Cancel Back Step 4 of 5 Next
A job will be created and scheduled to collect SQL statements an	d load them into the new SQL tuning set.
Job Parameters	
Job Name CREATE_STS_1173267160578	
Description	
Schedule	
 Immediately 	
O Later	
Time Zone 🛛 Pacific/Pago_Pago 🛛 🔽	
Date Mar 7, 2007	
Time 3 💙 32 💙 00 🌱 💿 AM 🔿 PM	

4. Click Next.

The Create SQL Tuning Set: Review page appears.

Create SQL Tuning Set: Review	
Database orcl111	Cancel Back Step 5 of 5 Submit
Review the SQL Tuning Set options you have :	selected.
SQL Tuning Set Name	STS_BASELINE_PEAK_LOAD
Owner	IMMCHAN
Description	Contains SQL from BASELINE_PEAK_LOAD
Create an empty SQL tuning set	No
Load Methods	Load SQL statements one time only
Data Source	AWR Baseline
Baseline Name	BASELINE_PEAK_LOAD
Top N	<all></all>
Job Name	CREATE_STS_1173267160578
Scheduled Start Time	Run Immediately
▶Show SQL	

5. Review the SQL Tuning Set options that you have selected.

To view the SQL statements used by the job, expand Show SQL.

6. Click Submit.

The SQL Tuning Sets page appears.

If the job was scheduled to run immediately, then a message is displayed to inform you that the job and the SQL Tuning Set were created successfully. If the job was scheduled to run at a later time, a message is displayed to inform you that the job was created successfully.

7. To view details about the job, such as operation status, click View Job Details.

The View Job page appears to display details about the job.

Dropping a SQL Tuning Set

This section describes how to drop a SQL Tuning Set. To conserve storage space, you may want to periodically drop unused SQL Tuning Sets stored in the database.

To drop a SQL Tuning Set:

1. On the Database Performance page, under Additional Monitoring Links, click SQL Tuning Sets.

The SQL Tuning Sets page appears.

Existing SQL Tuning Sets are displayed on this page.

2. Select the SQL Tuning Set you want to drop and click Drop.

The Confirmation page appears to verify if you want to delete the selected SQL Tuning Set.

3. Click Yes.

The SQL Tuning Sets page appears.

A confirmation message is displayed to indicate that the SQL Tuning Set was successfully deleted.

Transporting SQL Tuning Sets

You can transport SQL Tuning Sets from one system to another by first exporting a SQL Tuning Set from one system, then importing it into another system.

This section contains the following topics:

- Exporting a SQL Tuning Set
- Importing a SQL Tuning Set

Exporting a SQL Tuning Set

This section describes how to export a SQL Tuning Set, thereby enabling it to be transported to another system.

To export a SQL Tuning Set:

1. On the Database Performance page, under Additional Monitoring Links, click SQL Tuning Sets.

The SQL Tuning Sets page appears.

Existing SQL Tuning Sets are displayed on this page.

2. Select the SQL Tuning Set you want to export and click Export.

The Export SQL Tuning Set page appears.

Expo	Export SQL Tuning Set					
				Cancel OK		
SQL		ing Set Name STS_BASELINE_PEAK_LOAD Owner IMMCHAN				
		irectory Object DATA_PUMP_DIR Directory Name C:\oracle\admin\orcl111\dpdump\				
	* Export File	EXPDAT_STS_BA	SELINE_PEAK_LO/			
	Log File	EXPDAT_STS_BA	SELINE_PEAK_LO			
Select	a tablespace which v	will be used tempo	rarily to store the data for the export operation. By default, SYSAUX will be used.			
Select	Tablespace Name		Available Space (MB)			
۲	SYSAUX		30.625			
0	EXAMPLE		24.5625			
0	SYSTEM		2.5625			
0	USERS		15.3125			
Joł	Parameters					
Jo	b Name EXPDAT_S	STS_BASELINE_PE	4K_LOAD			
De	scription					
Scl	nedule					
۲	Immediately					
C	OLater					
	Time Zone Pacific/Pago_Pago 💌					
	Date Mar 12, 2007					
	Time 2 • 21 • 00 • • AM OPM					

3. In the **Directory Object** field, select a directory where the export file will be created.

For example, to use the Oracle Data Pump directory, select DATA_PUMP_DIR. The Directory Name field refreshes automatically to indicate the selected directory.

4. In the **Export File** field, enter a name for the dump file that will be exported.

Alternatively, you can accept the name generated by the system.

5. In the **Log File** field, enter a name for the log file for the export operation.

Alternatively, you can accept the name generated by the system.

- **6.** Select a tablespace to temporarily store the data for the export operation. By default, SYSAUX is used.
- 7. Under Job Parameters, in the **Job Name** field, enter a name for the job.

Alternatively, you can accept the name generated by the system.

- **8.** Under Schedule, do one of the following:
 - Select Immediately to run the job immediately after it has been submitted.
 - Select Later to run the job at a later time as specified by selecting or entering values in the Time Zone, Date, and Time fields.
- 9. Click OK.

The SQL Tuning Sets page appears.

A confirmation message is displayed to indicate that the job was successfully created.

10. Optionally, transport the export file to another system using the mechanism of choice (such as Oracle Data Pump or a database link).

Importing a SQL Tuning Set

Before a SQL Tuning Set can be imported, you must first export a SQL Tuning Set from another system and transport it to your current system. For more information, see "Exporting a SQL Tuning Set" on page 10-16.

To import a SQL Tuning Set:

1. On the Database Performance page, under Additional Monitoring Links, click SQL Tuning Sets.

The SQL Tuning Sets page appears.

2. Click Import.

The Import SQL Tuning Set page appears.

Import SQL Tuni	ng Set		
			Cancel OK
* Directory Object	DATA_PUMP_DIR	×	
Directory Name	C:\oracle\admin\c	rcl111\dpdump\	
* Import File	EXPDAT_STS_BASEL	INE_PEAK_LO/	
Log File	EXPDAT_STS_BASEL	INE_PEAK_LO;	
Replace the existing	g SQL tuning set if one	exists.	
Select a tablespace wh	ich will be used tempo	prarily to store the data for the import operation. By default, SYSAUX will be used.	
Select Tablespace Na	ame	Available Space (MB)	
SYSAUX		30.625	
EXAMPLE		24.5625	
SYSTEM		2.5625	
USERS		15.3125	
Job Parameters	5		
Job Name IMPOR	T_STS_BASELINE_PE	AK_LOAD	
Description			
Schedule			
 Immediately 			
OLater			
Time Zone P	acific/Pago_Pago	v	
Date L	1ar 12, 2007 xample: Mar 12, 2007)		
Time 2	✓ 51 ✓ 00 ✓) AM O PM	

3. In **Directory Object**, select a directory where the import file is stored.

The directory should contain the export file that was transported to your current system. For example, if the file resides in the Data Pump directory, then select DATA_PUMP_DIR. The Directory Name field refreshes automatically to indicate the selected directory.

- 4. In the **Import File** field, enter the name of the dump file that will be imported.
- 5. In the Log File field, enter a name for the log file for the import operation.
- 6. To replace an existing SQL Tuning Set with the one that you are importing, select **Replace the existing SQL tuning set if one exists**.
- **7.** Select a tablespace to temporarily store the data for the import operation.

By default, SYSAUX is used.

8. Under Job Parameters, in the Job Name field, enter a name for the job.

Alternatively, you can accept the name generated by the system.

- 9. Under Schedule, do one of the following:
 - Select **Immediately** to run the job immediately after it has been submitted.

- Select Later to run the job at a later time as specified by selecting or entering values in the Time Zone, Date, and Time fields.
- 10. Click OK.

The SQL Tuning Sets page appears.

A confirmation message is displayed to indicate that the job was successfully created. If the job is scheduled to run immediately, then the imported SQL Tuning Set will be displayed on this page. You may need to refresh the page.

Managing SQL Profiles

When running a SQL Tuning Advisor task with a limited scope, the query optimizer makes estimates about cardinality, selectivity, and cost. These estimates can sometimes be off by a significant amount, resulting in poor execution plans.

To address this problem, consider running a SQL Tuning Advisor task with a comprehensive scope to collect additional information using sampling and partial execution techniques to verify and, if necessary, adjust these estimates. These auxiliary statistics about the SQL statement are collected into a SQL profile.

During SQL profiling, the query optimizer uses the execution history information about the SQL statement to create appropriate settings for optimizer parameters. After the SQL profiling completes, the query optimizer uses the information stored in the SQL profile, in conjunction with regular database statistics, to generate execution plans. The availability of the additional information makes it possible to produce well-tuned plans for corresponding SQL statements.

After running a SQL Tuning Advisor task with a comprehensive scope, a SQL profile may be recommended. If you accept the recommendation, then the SQL profile will be created and enabled for the SQL statement.

In some cases, you may want to disable a SQL profile. For example, you may want to test the performance of a SQL statement without using a SQL profile to determine if the SQL profile is actually beneficial. If the SQL statement is performing poorly after the SQL profile is disabled, then you should enable it again to avoid performance degradation. If the SQL statement is performing optimally after you have disabled the SQL profile, you may want to remove the SQL profile from your database.

To enable, disable, or delete a SQL profile:

1. On the Performance page, click **Top Activity**.

The Top Activity page appears.

2. Under Top SQL, click the **SQL ID** link of the SQL statement that is using a SQL profile.

The SQL Details page appears.

3. Click the **Tuning Information** tab.

A list of SQL profiles is displayed under SQL Profiles and Outlines.

SQL Pr	ofiles and Outlines						
	A SQL Profile contains additional statistics of this SQL statement for the query optimizer to generate a better execution plan.						
An outli	An outline contains hints for this SQL statement for the query optimizer to generate a better execution plan.						
Cha	Change Category Delete Disable/Enable						
Select	Select Name Type Category Status Created						
۲	SYS_SQLPROF_0144488f7d65c000	SQL Profile	DEFAULT	ENABLED	Mar 5, 2007 5:15:57 AM		

4. Select the SQL profile you want to manage. Do one of the following:

- To enable a SQL profile that is disabled, click **Disable/Enable**.
- To disable a SQL profile that is enabled, **Disable/Enable**.
- To remove a SQL profile, click **Delete**.

A confirmation page appears.

5. Click Yes to continue, or No to cancel the action.

Managing SQL Execution Plans

SQL plan management records and evaluates the execution plans of SQL statements over time. This mechanism builds SQL plan baselines composed of a set of existing plans known to be efficient. If the same SQL statement is run repeatedly, and if the optimizer generates a new plan that differs from the baseline, then the database compares the new plan with the baseline and chooses the best one.

Some events can cause changes to SQL execution plans, such as new optimizer statistics, changes to initialization parameter values, a database upgrade that causes a change to the optimizer, and so on. These changes can cause regressions in SQL performance, which can be difficult and time-consuming to fix manually. As a preventative measure, you can fix SQL plan baselines to preserve performance of corresponding SQL statements, regardless of changes occurring in the system.

To load SQL execution plans:

1. From the Database Home page, click **Server**.

The Server subpage appears.

2. Under Query Optimizer, click SQL Plan Control.

The SQL Profile subpage of the SQL Plan Control page appears.

3. Click SQL Plan Baseline.

The SQL Plan Baseline subpage appears.

QL Plan Control					
SQL Profile SQL Patch SQL Plan Baseline					
A SQL Plan Baseline is an execution plan deemed to have accept	(Refresh				
	Jobs for SQL Plan Baselines				
Settings	Pending Completed				
Capture SQL Plan Baselines FALSE	Evolve				
Use SQL Plan Baselines TRUE	Jobs				
Plan Retention(Weeks) 53 Configure	Load Jobs <u>SPM_LOAD_20070626_154521</u>				
Search					
SQL Text					
 By default, the search returns all uppercase matches beginning with the search string. You can use the wildcard symbol (%) in a double guote 	he string you entered. To run an exact or case-sensitive match, double quote the d string.				

4. Under Settings, click the link next to Capture SQL Plan Baselines.

The Initialization Parameters page appears.

5. In the Value column of the table, select TRUE and then click OK.

You are returned to the SQL Plan Baseline subpage, which now shows **Capture SQL Baselines** set to **TRUE**.

Because you configured baselines to be captured, the database automatically keeps a history of execution plans for all SQL statements executed more than once.

6. Click Load.

The SQL Plan Control page appears.

SQL Plan Control		
	Cancel	ОК
Load SQL Plan Baselines		
Plans can be bulk loaded from either an existing SQL Tuning Set or directly from the cursor cache	۱.	
Load plans from SQL Tuning Set(STS)		
SQL Tuning Set		
C Load plans from cursor cache		
SQL ID		
*		
Job Parameters		
Job Name SPM_LOAD_20070627_105654		
Description		
Schedule		
© Immediately		
C Later		
Time Zone (UTC+00:00) Universal Time		
Date Jun 27, 2007		
Time 10 🗸 56 🗸 00 🗸 🖲 AM C PM		

- 7. Select the SQL plan baselines to be loaded. Complete the following steps:
 - a. Under Load SQL Plan Baselines, select Load plans from SQL Tuning Set (STS).

In this example, load plans from the SQL Tuning Set that you created in "Creating a SQL Tuning Set" on page 10-8.

- **b.** In **Job Name**, enter a name for the job. In this example, enter SPM_LOAD_TEST.
- c. Under Schedule, select Immediately.
- d. Click OK.

The SQL Profile subpage of the SQL Plan Control page appears. The table displays a list of SQL plans that are stored as SQL plan baselines.

Ena	ble)(Disable)(Drop) Evolve)	(Pack) Fixed - Yes				S Previous	25 26-50 c	Load Unpack	
Select All Select None										
Select	Name	9	5QL Text	Enabled	Accepted	Fixed	Auto Purge	Created	Last Modified $ abla$	
	SYS SQL PLAN 45e3b1c6	<u>5c5243ac L</u>	JPDATE employees SET salary = (SELECT salary FR	YES	YES	NO		Jun 28, 2007 9:17:08 AM	Jun 28, 2007 9:17:08 AM	
	SYS SQL PLAN 91fe5df50	<u>)150911c</u> U	JPDATE employees SET salary = 4000 WHERE last n	YES	YES	NO		Jun 28, 2007 9:17:08 AM	Jun 28, 2007 9:17:08 AM	
	<u>SYS SQL PLAN 92da6e62</u>	<u>d8aea094</u> 9	SELECT salary FROM employees AS OF TIMESTAMP (S	YES	YES	NO		Jun 28, 2007 9:17:08 AM	Jun 28, 2007 9:17:08 AM	
	SYS SQL PLAN cd8e8229	43cbde50 9	SELECT DECODE(GROUPING(department_name), 1, 'All D	YES	YES	NO		Jun 28, 2007 9:17:08 AM	Jun 28, 2007 9:17:08 AM	
	<u>SYS SQL PLAN df31a9ee</u>	<u>c03e7818</u> 9	SELECT * FROM sales PARTITION (sales q2 2000) s	YES	YES	NO		Jun 28, 2007 9:17:08 AM	Jun 28, 2007 9:17:08 AM	
	<u>SYS SQL PLAN f5879c99f</u>	5e4b1833 9	SELECT department id, MIN(salary), MAX (salary)	YES	YES	NO		Jun 28, 2007 9:17:08 AM	Jun 28, 2007 9:17:08 AM	
	SYS SQL PLAN 3428d6ab	<u>cb8ddfac</u> <u>\</u>	WITH dept_costs AS (SELECT department_na	YES	YES	NO		Jun 28, 2007 9:17:07 AM	Jun 28, 2007 9:17:07 AM	

- **8.** Optionally, fix the execution plan of a baseline so that the database will not use any alternative SQL plan baseline. Complete the following steps:
 - **a.** Select a SQL plan baseline that is not fixed.
 - **b.** Select **Fixed Yes** from the list preceding the baseline table.
 - c. Click Go.

The table is refreshed to show the SQL execution plan with the value ${\tt YES}$ in the Fixed column of the table.

See Also:

Oracle Database Performance Tuning Guide to learn how to use SQL plan management

Optimizing Data Access Paths

To achieve optimum performance for data-intensive queries, materialized views and indexes are essential when tuning SQL statements. Implementing these objects, however, does not come without a cost. Creation and maintenance of these objects can be time-consuming, and space requirements can be significant. SQL Access Advisor enables you to optimize data access paths of SQL queries by recommending the proper set of materialized views and view logs, indexes, SQL profiles, and partitions for a given workload.

A materialized view provides access to table data by storing query results in a separate schema object. Unlike an ordinary view, which does not take up storage space or contain data, a materialized view contains the rows resulting from a query against one or more base tables or views. A materialized view log is a schema object that records changes to a master table's data, so that a materialized view defined on the master table can be refreshed incrementally. SQL Access Advisor recommends how to optimize materialized views so that they can be rapidly refreshed and take advantage of the general query rewrite feature. For more information about materialized views and view logs, see *Oracle Database Concepts*.

SQL Access Advisor also recommends bitmap, function-based, and B-tree indexes. A bitmap index provides a reduced response time for many types of ad hoc queries and reduced storage requirements compared to other indexing techniques. A function-based index derives the indexed value from the table data. For example, to find character data in mixed cases, a function-based index can be used to look for the values as if they were all in uppercase characters. B-tree indexes are most commonly used to index unique or near-unique keys.

Using SQL Access Advisor involves the following tasks:

- Running SQL Access Advisor
- Reviewing the SQL Access Advisor Recommendations
- Implementing the SQL Access Advisor Recommendations

See Also:

- Chapter 9, "Identifying High-Load SQL Statements"
- Chapter 10, "Tuning SQL Statements" for information about SQL Tuning Advisor

Running SQL Access Advisor

This section describes how to run SQL Access Advisor to make recommendations on a SQL workload.

To run SQL Access Advisor:

- 1. Select the initial options, as described in "Running SQL Access Advisor: Initial Options" on page 11-2.
- 2. Select the workload source you want to use for the analysis, as described in "Running SQL Access Advisor: Workload Source" on page 11-3.
- **3.** Define the filters options, as described in "Running SQL Access Advisor: Filter Options" on page 11-5.
- **4.** Choose the types of recommendations, as described in "Running SQL Access Advisor: Recommendation Options" on page 11-7.
- **5.** Schedule the SQL Access Advisor task, as described in "Running SQL Access Advisor: Schedule" on page 11-9.

Running SQL Access Advisor: Initial Options

The first step in running SQL Access Advisor is to select the initial options on the SQL Access Advisor: Initial Options page.

To select initial options:

1. On the Database Home page, under Related Links, click Advisor Central.

The Advisor Central page appears.

2. Under Advisors, click SQL Advisors.

The SQL Advisors page appears.

3. Click SQL Access Advisor.

The SQL Access Advisor: Initial Options page appears.

- 4. Select the initial options. Do one of the following:
 - Select Verify use of access structures (indexes, materialized views, partitioning, and so on) only to verify existing structures.
 - Select **Recommend new access structures** to use the recommended options defined in the Oracle Enterprise Manager default template.

If you select this option, then you can optionally complete the following additional steps:

- Select Inherit Options from a previously saved Task or Template to use the options defined in an existing SQL Access Advisor task or another template.
- In Tasks and Templates, select the task or template that you want to use.

In this example, **Recommend new access structures** is selected.

× .	ess Advisor: Initial Opt	lions				
elect a set	of initial options.			(Cancel) (Contin		
O Verify	y use of access structures (inde)	kes, materialized views, partitioning, etc) only	Overview			
Recommend new access structures The SQL Access Advisor evaluates SQL statements in						
V	Inherit Options from a previousl	Source, and can suggest indexes, partitionin materialized view logs that will improve perfo as a whole.	ng, materialized views an			
111 100	rare selecting the starting point	for the wizard. All options can be changed from	wich in the wizard.			
	and Templates Templates Only	or the wizard, willoptions can be changed noin:	warmin are wezard.			
Tasks View	and Templates	or die wizard, wir opdons dan de changed nom	wumin une wezeru.			
Tasks ^{View}	and Templates	Description	Last Modified	Туре		
Tasks ^{View} Vie Select	and Templates Templates Only w Options			Type Default Template		
Tasks View Vie Select	and Templates Templates Only W Options Name	Description	Last Modified			
Tasks View Vie Select	and Templates Templates Only W Options Name SQLACCESS_EMTASK	Description Default Enterprise Manager task template	Last Modified Mar 26, 2007 1:28:21 PM PDT	Default Template		

5. Click Continue.

The SQL Access Advisor: Workload Source page appears.

6. Proceed to the next step, as described in "Running SQL Access Advisor: Workload Source" on page 11-3.

Running SQL Access Advisor: Workload Source

After initial options are specified for SQL Access Advisor, select the workload source that you want to use for the analysis, as described in the following sections:

- Using SQL Statements from the Cache
- Using an Existing SQL Tuning Set
- Using a Hypothetical Workload

Tip: Before you can select the workload source for SQL Access Advisor, select the initial options, as described in "Running SQL Access Advisor: Initial Options" on page 11-2.

Using SQL Statements from the Cache

You can use SQL statements from the cache as the workload source. However, because only current and recent SQL statements are stored in the SQL cache, this workload source may not be representative of the entire workload on your database.

To use SQL statements from the cache as the workload source:

 On the SQL Access Advisor: Workload Source page, select Current and Recent SQL Activity.

In this example, Use Default Options is selected.

SQL Acces	s Advisor: Workload Source	
Database Logged In As	database DBA1	Cancel) Step 1 of 4 Ne <u>x</u> t
	ce of the workload that you want to use for the analysis. The best wa at access the underlying tables.	orkload is one that fully represents all the SQL
Current SQL wi	and Recent SQL Activity I be selected from the cache.	

2. Proceed to the next step, as described in "Running SQL Access Advisor: Filter Options" on page 11-5.

Using an Existing SQL Tuning Set

You can use an existing SQL Tuning Set as the workload source. This option is useful because SQL Tuning Sets can be used repeatedly as the workload source for not only SQL Access Advisor, but also SQL Tuning Advisor.

To use a SQL Tuning Set as the workload source:

- On the SQL Access Advisor: Workload Source page, select Use an existing SQL Tuning Set.
- 2. Click the SQL Tuning Set search icon to use an existing SQL Tuning Set.

The Search and Select: SQL Tuning Set dialog box appears.

3. In the **Schema** field, enter the name of the schema containing the SQL Tuning Set you want to use and click **Go**.

A list of SQL Tuning Sets contained in the selected schema appears.

4. Select the SQL Tuning Set to be used for the workload source and click Select.

The Search and Select: SQL Tuning Set dialog box closes and the selected SQL Tuning Set now appears in the **SQL Tuning Set** field.

5. Proceed to the next step, as described in "Running SQL Access Advisor: Filter Options" on page 11-5.

See Also:

"Managing SQL Tuning Sets" on page 10-8

Using a Hypothetical Workload

A dimension table stores all or part of the values for a logical dimension in a star or snowflake schema. You can create a hypothetical workload from dimension tables containing primary or foreign key constraints. This option is useful if the workload to be analyzed does not exist. In this case, SQL Access Advisor examines the current logical schema design, and provides recommendations based on the defined relationships between tables.

To use a hypothetical workload as the workload source:

- 1. On the SQL Access Advisor: Workload Source page, select **Create a Hypothetical Workload from the Following Schemas and Tables**.
- 2. Leave Schemas and Tables empty and click Add to search for tables.

The Workload Source: Search and Select Schemas and Tables page appears.

3. In the Tables section, enter a schema name in the Schema field and click Search.

A list of tables in the selected schema is displayed.

4. Select the tables to be used in creating the hypothetical workload and click **Add Tables**.

The selected tables now appear in the **Schemas and Tables** field.

5. Click OK.

The SQL Access Advisor: Workload Source page appears with the selected tables now added.

6. Proceed to the next step, as described in "Running SQL Access Advisor: Filter Options" on page 11-5.

Running SQL Access Advisor: Filter Options

After the workload source is selected, you can apply filters to reduce the scope of the SQL statements found in the workload. While using filters is optional, it can be very beneficial due to the following:

- Using filters directs SQL Access Advisor to make recommendations based on a specific subset of SQL statements from the workload, which may lead to better recommendations.
- Using filters removes extraneous SQL statements from the workload, which may
 greatly reduce processing time.

Tip: Before you can select the filter options for the workload, do the following:

- Select initial options, as described in "Running SQL Access Advisor: Initial Options" on page 11-2.
- Select the workload source, as described in "Running SQL Access Advisor: Workload Source" on page 11-3.

To apply filters to the workload source:

- On the SQL Access Advisor: Workload Source page, click Filter Options. The Filter Options section expands.
- 2. Select Filter Workload Based on these Options.

The Filter Options section is enabled.

- **3.** Define the filters you want to apply, as described in the following sections:
 - Defining Filters for Resource Consumption
 - Defining Filters for Users
 - Defining Filters for Tables
 - Defining Filters for SQL Text
 - Defining Filters for Modules
 - Defining Filters for Actions
- 4. Click Next.

The Recommendation Options page appears.

5. Proceed to the next step, as described in "Running SQL Access Advisor: Recommendation Options" on page 11-7.

Defining Filters for Resource Consumption

The resource consumption filter restricts the workload to include only the number of high-load SQL statements that you specify.

To define a filter for resource consumption:

- On the SQL Access Advisor: Workload Source page, under User Resource Consumption, enter the number of high-load SQL statements in the Number of Statements field.
- **2.** From the Order by list, select one of the methods by which the SQL statements are to be ordered.

Defining Filters for Users

The users filter restricts the workload to include or exclude SQL statements executed by users that you specify.

To define a filter for users:

- 1. On the SQL Access Advisor: Workload Source page, under Users, select Include only SQL statements executed by these users or Exclude all SQL statements executed by these users.
- **2.** To search for available users, click the Users search icon.

The Search and Select: Users dialog box appears.

3. Select the users for which you want to include or exclude SQL statements and click **Select**.

The Search and Select: Users dialog box closes and the selected tables now appear in the **Users** field.

In this example, a filter is defined to include only SQL statements executed by the user SH.

users		
Include only SQL statements executed by these users C Exclude all SQL statements executed by these users	SH	_
executed by these users	Comma-separated list	-

Defining Filters for Tables

The tables filter restricts the workload to include or exclude SQL statements that access a list of tables that you specify. Table filters are not permitted if you selected the **Create a Hypothetical Workload from the Following Schemas and Tables** option, as described in "Using a Hypothetical Workload" on page 11-4.

To define a filter for tables:

- 1. To include only SQL statements that access a specific list of tables, enter the table names in the **Include only SQL statements that access any of these tables** field.
- **2.** To exclude all SQL statements that access a specific list of tables, enter the table names in the **Exclude all SQL statements that access any of these tables** field.
- **3.** To search for available tables, click the Tables search icon.

The Search and Select: Schema and Table dialog box appears.

4. Select the tables for which you want to include or exclude SQL statements and click **Select**.

The Search and Select: Schema and Table dialog box closes and the selected tables now appear in the corresponding Tables field.

Defining Filters for SQL Text

The SQL text filter restricts the workload to include or exclude SQL statements that contains SQL text substrings that you specify.

To define a filter for SQL text:

 To include only SQL statements that contains specific SQL text, enter the SQL text to be included in the Include only SQL statements containing these SQL text substrings field. To exclude all SQL statements that contain specific SQL text, enter the SQL text to be excluded in the Exclude all SQL statements containing these SQL text substrings field.

Defining Filters for Modules

The module filter restricts the workload to include or exclude SQL statements that are associated with modules that you specify.

To define a filter for module ID:

- 1. To include only SQL statements associated with a specific module ID in the workload, select **Include only SQL statements associated with these modules**.
- **2.** To exclude all SQL statements associated to a specific module ID from the workload, select **Exclude all SQL statements associated with these modules**.
- **3.** In the Modules field, enter the names of the modules for which associated SQL statements will be included or excluded.

Defining Filters for Actions

The actions filter restricts the workload to include or exclude SQL statements that are associated with actions that you specify.

To define a filter for actions:

- 1. To include only SQL statements associated with a specific action in the workload, select **Include only SQL statements associated with these actions**.
- To exclude all SQL statements associated with a specific action from the workload, select Exclude all SQL statements associated with these actions.
- **3.** In the **Actions** field, enter the actions for which associated SQL statements will be included or excluded.

Running SQL Access Advisor: Recommendation Options

To improve the underlying data access methods chosen by the optimizer for the workload, SQL Access Advisor provides recommendation for indexes, materialized views, and partitioning. Using these access structures can significantly improve the performance of the workload by reducing the time required to read data from the database. However, you must balance the benefits of using these access structures against the cost to maintain them.

Tip: Before you can select the recommendation options for SQL Access Advisor, do the following:

- Select initial options, as described in "Running SQL Access Advisor: Initial Options" on page 11-2.
- Select the workload source, as described in "Running SQL Access Advisor: Workload Source" on page 11-3.
- Define the filter options, as described in "Running SQL Access Advisor: Filter Options" on page 11-5.

To specify recommendation options:

1. On the SQL Access Advisor: Recommendation Options page, under Access Structures to Recommend, select the type of access structures to be recommended by SQL Access Advisor:

- Indexes
- Materialized Views
- Partitioning

In this example, all of the preceding access types are selected.

Access Structures to Recommend

Indexes
Materialized Views
Partitioning

- **2.** Under Scope, select the mode in which SQL Access Advisor will run. Do one of the following:
 - Select Limited Mode.

In limited mode, SQL Access Advisor focuses on SQL statements with the highest cost in the workload. The analysis is quicker, but the recommendations may be limited.

• Select Comprehensive Mode.

In comprehensive mode, SQL Access Advisor analyzes all SQL statements in the workload. The analysis can take much longer, but the recommendations will be exhaustive.

In this example, **Limited Mode** is selected.

```
Advisor Mode
The advisor can run in one of two modes, Limited or Comprehensive. Limited Mode is meant to return quickly after processing the statements with the highest cost, potentially ignoring statements with a cost below a certain threshold. Comprehensive Mode will perform an exhaustive analysis.
Comprehensive Mode
Analysis will be exhaustive
```

3. Optionally, click Advanced Options.

The Advanced Options section expands. This section contains the following subsections:

Workload Categorization

In this section, you can specify the type of workload for which you want a recommendation. The following categories are available:

- Workload Volatility

Select **Consider only queries** if the workload contains primarily read-only operations, as in data warehouses. Volatility data is useful for online transaction processing (OLTP) systems, where the performance of INSERT, UPDATE, and DELETE operations is critical.

Workload Scope

Select **Recommend dropping unused access structures** if the workload represents all access structure use cases.

Space Restrictions

Indexes and materialized views increase performance at the cost of space. Do one of the following:

- Select No, show me all recommendations (unlimited space) to specify no space limits. When SQL Access Advisor is invoked with no space limits, it makes the best possible performance recommendations.
- Select Yes, limit additional space to and then enter the space limit in megabytes, gigabytes, or terabytes. When SQL Access Advisor is invoked with a space limit, it produces only recommendations with space requirements that do not exceed the specified limit.
- Tuning Prioritization

This section enables you to specify how SQL statements will be tuned. Complete the following steps:

- From the Prioritize tuning of SQL statements by list, select a method by which SQL statements are to be tuned and then click **Add**.
- Optionally, select Allow Advisor to consider creation costs when forming recommendations to weigh the cost of creating access structures against the frequency and potential improvement of SQL statement execution time. Otherwise, creation cost will be ignored. You should select this option if you want specific recommendations generated for SQL statements that are executed frequently.
- Default Storage Locations

Use this section to override the defaults defined for schema and tablespace locations. By default, indexes are in the schema and tablespace of the table they reference. Materialized views are in the schema and tablespace of the first table referenced in the query. Materialized view logs are in the default tablespace of the schema of the table that they reference.

4. Click Next.

The SQL Access Advisor: Schedule page appears.

 Proceed to the next step, as described in "Running SQL Access Advisor: Schedule" on page 11-9.

Running SQL Access Advisor: Schedule

Use the SQL Access Advisor Schedule page to set or modify the schedule parameters for the SQL Access Advisor task.

QL Access Advisor: Schedule	
Database database Logged In As DBA1	Cancel Back Step 3 of 4 Next
Advisor Task Information	
* Task Name SQLACCESS9084523	
Task Description SQL Access Advisor	
Journaling Level Basic The level of journaling controls the amount of information that is log appears on the Details tab when viewing task results.	ged to the advisor journal during execution of the task. This information
* Task Expiration (days) 30 Number of days this task will be retained in the database	efore being purged
* Total Time Limit (minutes)	
Scheduling Options	
Schedule Type Standard	
Time Zone PST8PDT	
Repeating	
Repeat Do Not Repeat 💌	
Start	
Immediately	
C Later	
Date Mar 26, 2007	
(example: Mar 26, 2007) Time 6 ▼ 50 ▼ 00 ▼ C AM ☉ PN	

Figure 11–1 Scheduling a SQL Access Advisor Task

Tip: Before you can schedule a SQL Access Advisor task, do the following:

- Select initial options, as described in "Running SQL Access Advisor: Initial Options" on page 11-2.
- Select the workload source, as described in "Running SQL Access Advisor: Workload Source" on page 11-3.
- Define the filter options, as described in "Running SQL Access Advisor: Filter Options" on page 11-5.
- Specify the recommendation options, as described in "Running SQL Access Advisor: Recommendation Options" on page 11-7.

To schedule a SQL Access Advisor task:

1. On the SQL Access Advisor: Schedule page, under Advisor Task Information, enter a name in the **Task Name** field if you do not want to use the system-generated task name.

In the example shown in Figure 11–1, SQLACCESS9084523 is entered.

2. In the Task Description field, enter a description of the task.

In the example shown in Figure 11–1, SQL Access Advisor is entered.

3. From the Journaling Level list, select the level of journaling for the task.

Journaling level controls the amount of information that is logged to the SQL Access Advisor journal during task execution. This information appears on the Details subpage when viewing task results. In the example shown in Figure 11–1 on page 11-10, **Basic** is selected.

4. In the **Task Expiration (Days)** field, enter the number of days the task will be retained in the database before it is purged.

In the example shown in Figure 11–1 on page 11-10, 30 is entered.

5. In the **Total Time Limit (minutes)** field, enter the maximum number of minutes that the job is permitted to run.

You must enter a time in this field rather than use the default of UNLIMITED. In the example shown in Figure 11–1 on page 11-10, 10 is entered.

- **6.** Under Scheduling Options, in the Schedule Type list, select a schedule type for the task and a maintenance window in which the task should run. Do one of the following:
 - Click Standard.

This schedule type enables you to select a repeating interval and start time for the task. Complete the following steps:

- Enter your time zone code in the **Time Zone** field or click the search icon to locate the code for your area.
- In the Repeat list, select **Do Not Repeat** to perform the task only once, or select a unit of time and enter the number of units in the **Interval** field.
- Under Start, select Immediately to start the task now, or Later to schedule the task to start at a time specified using the Date and Time fields.
- Click Use predefined schedule.

This schedule type enables you to select an existing schedule. Do one of the following:

- In the Schedule field, enter the name of the schedule to be used for the task.
- To search for a schedule, click the search icon.

The Search and Select: Schedule dialog box appears.

Select the desired schedule and click **Select**. The selected schedule now appears in the **Schedule** field.

Click Standard using PL/SQL for repeated interval.

This schedule types enables you to select a repeating interval and an execution time period (window) for the task. Complete the following steps:

- Enter your time zone code in the **Time Zone** field or click the search icon to locate the code for your area.
- Under Available to Start, select Immediately to start the task now, or Later to schedule the task to start at a time specified using the Date and Time fields.
- In the Repeat list, select **Do Not Repeat** to perform the task only once, or select a unit of time and enter the number of units in the **Interval** field.
- In the Repeated Interval field, enter a PL/SQL schedule expression, such as SYSDATE+1.
- Under Not Available After, select No End Date to indicate that there is no end date for the execution window, or Specified End Date to specify an end date using the Date and Time fields.

• Click Use predefined window.

This schedule type enables you to select an existing window. Select **Stop on Window Close** to stop the job when the window closes. Do one of the following:

- In the Window field, enter the name of the window to be used for the task.
- To search for a window, click the search icon.

The Search and Select: Window and Window Groups dialog box appears.

Select the desired window and click **Select**. The selected window now appears in the **Schedule** field.

• Click Event.

Complete the following steps:

- Enter your time zone code in the **Time Zone** field or click the search icon to locate the code for your area.
- Under Event Parameters, enter values in the **Queue Name** and **Condition** fields.
- Under Start, select **Immediately** to start the task now, or **Later** to schedule the task to start at a time specified using the **Date** and **Time** fields.
- Under Not Available After, select No End Date to indicate that there is no end date for the execution window, or Specified End Date to specify an end date using the Date and Time fields.
- Click Calendar.

Complete the following steps:

- Enter your time zone code in the **Time Zone** field or click the search icon to locate the code for your area.
- Under Calendar Expression, enter a calendar expression.
- Under Start, select Immediately to start the task now, or Later to schedule the task to start at a time specified using the Date and Time fields.
- Under Not Available After, select No End Date to indicate that there is no end date for the execution window, or Specified End Date to specify an end date using the Date and Time fields.

In the example shown in Figure 11–1 on page 11-10, **Standard** is selected for schedule type. The task will not repeat and is scheduled to start immediately.

7. Click Next.

The SQL Access Advisor: Review page appears.

SQL Acce	ss Advisor: Revie	w							
Database Logged In As	DBA1		Cancel Show SQL Back Step 4 of 4 Submit						
Please review	Please review the SQL Access Advisor options and values you have selected.								
Ta	ask Name SQLACCESS	9817748							
Task De	escription SQL Access	Advisor							
Scheduled S	tart Time Run Imme	diately							
Option	8								
-	All Options)								
Modifie	d Option	Value	Description						
× .	Analysis Scope	All Tuning Artifacts	The type of recommendations that are allowed						
×	Total Time Limit (minutes)	30	Specifies the total time limit in minutes for the current SQL Access Advisor task						
1	Workload SQL Limit	25	Specifies the number of SQL statements to be analyzed						
			Cancel Show SQL Back Step 4 of 4 Submit						

Under Options, a list of modified options for the SQL Access Advisor task is shown. To display both modified and unmodified options, click **Show All Options**. To view the SQL text for the task, click **Show SQL**.

8. Click Submit.

The Advisor Central page appears. A message informs you that the task was created successfully.

Reviewing the SQL Access Advisor Recommendations

SQL Access Advisor graphically displays the recommendations and provides hyperlinks so that you can quickly see which SQL statements benefit from a recommendation. Each recommendation produced by the SQL Access Advisor is linked to the SQL statement it benefits.

Tip: Before reviewing the SQL Access Advisor recommendations, run SQL Access Advisor to make the recommendations, as described in "Running SQL Access Advisor" on page 11-1.

To review the SQL Access Advisor recommendations:

1. On the Advisor Central page, select the SQL Access Advisor task for review and click **View Result**.

In this example, **Limited Mode** is selected.

Results								
Viev	View Result Delete Actions Re-schedule							
Select	Advisory Type	Name	Description	User	Status	Start Time $ abla$	Duration (seconds)	Expires In (days)
	SQL Access Advisor	SQLACCESS2710801	SQL Access Advisor	SYS	CREATED			30
	SQL Access Advisor	SQLACCESS9817748	SQL Access Advisor	SYS	CREATED			30

If the task is not displayed, then you may need to refresh the screen. The Results for Task page appears.

- 2. Review the Summary subpage, which provides an overview of the SQL Access Advisor analysis, as described in "Reviewing the SQL Access Advisor Recommendations: Summary" on page 11-14.
- **3.** Review the Recommendations subpage, which enables you to view the recommendations ranked by cost improvement, as described in "Reviewing the SQL Access Advisor Recommendations: Recommendations" on page 11-15.

- 4. Review the SQL statements analyzed in the workload, as described in "Reviewing the SQL Access Advisor Recommendations: SQL Statements" on page 11-18.
- **5.** Review the details of the workload, task options, and the SQL Access Advisor task, as described in "Reviewing the SQL Access Advisor Recommendations: Details" on page 11-19.

Reviewing the SQL Access Advisor Recommendations: Summary

The Summary subpage displays an overview of the SQL Access Advisor analysis.

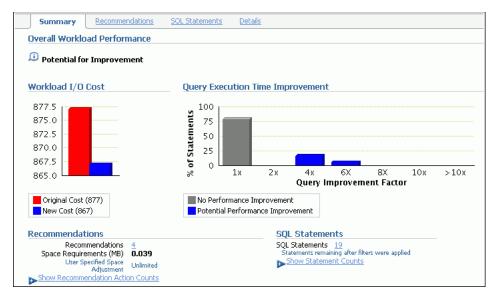
In this example, **Limited Mode** is selected.

To review the recommendations summary:

1. On the Results for Tasks page, click **Summary**.

The Summary subpage appears.

In this example, **Limited Mode** is selected.



- **2.** Under Overall Workload Performance, assess the potential for improvement in implementing the recommendations.
- **3.** Use the Workload I/O Cost chart to compare the original workload I/O cost (in red) with the new cost (in blue).

In this example, the workload I/O cost will decrease from 877 to 867 by implementing the recommendations.

4. Use the Query Execution Time Improvement chart to compare the improvement in query execution time.

This chart shows the percentage of SQL statements in the workload whose execution time will improve by accepting the recommendations. The SQL statements are grouped by the projected improvement factor along the horizontal axis on the chart (1x to >10x). The percentage of SQL statements that will improve by the projected improvement factor are along the vertical axis (0% to 100%).

In this example, approximately 75 percent of SQL statements in the workload will gain no performance improvement in execution time, but about 25 percent will have the potential for improvement of over 4x or more.

5. Under Recommendations, click Show Recommendation Action Counts.

In this example, creating 1 index, 4 materialized views, and 6 materialized view logs is recommended.

In this example, Limited Mode is selected.

Recommendations							
Recommendation Space Requirements (M	B)	<u>4</u> 0.039					
User Specified Spa Adjustme	int	Unlimited					
Hide Recommendation Ac	tio	n Counts					
Indexes	:	Create	1	Drop	0	Retain	0
Materialized Views	:	Create	4	Drop	0	Retain	0
Materialized View Logs	:	Create	6	Retain	0	Alter	0
Partitions	:	Tables	0	Indexes	0	Materialized Views	0

6. Under SQL Statements, click **Show Statement Counts** to display the type of SQL statement.

In this example, 19 SELECT statements are analyzed.

In this example, Limited Mode is selected.

SQL Statements	
SQL Statements <u>19</u> Statements remaining after filters were applie <u>Hide Statement Counts</u>	d
Insert	0
Select	19
Update	0
Delete	0
Merge	0
Skipped (Parsing or Privilege Errors)	0

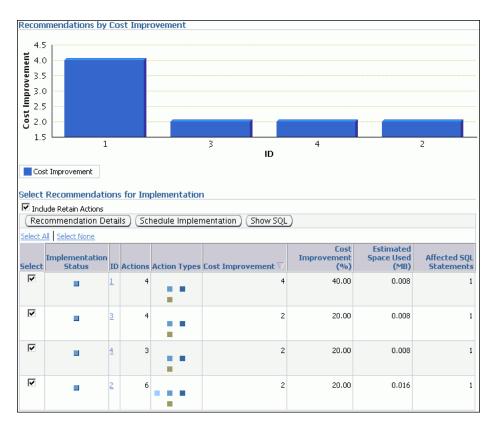
Reviewing the SQL Access Advisor Recommendations: Recommendations

The Recommendations subpage ranks the SQL Access Advisor recommendations by cost improvement. You can also view details about each recommendation.

To review recommendation details:

1. On the Results for Tasks page, click **Recommendations**.

The Recommendations subpage appears.



2. Use the Recommendations by Cost Improvement chart to view recommendations ordered by the cost improvement.

Under Select Recommendations for Implementation, each recommendation is listed with its implementation status, recommendation ID, cost improvement, space consumption, and the number of affected SQL statements for each recommendation. Implementing the top recommendation will have the biggest benefit to the total performance of the workload.

3. To view details for a particular recommendation, select the recommendation and click **Recommendation Details**.

The Recommendation Details page appears.

Actions							
Set Tablespac	e for All Actions	4	🖋 Go)			
Implement Status	ation Recommendation	Action		Ob	ject Name	Object Attributes	Base Table
-	3	CREATE MATERIALIZE	D VIEW LO	<u>)G</u>			SH.CUSTOMERS
-	3	CREATE MATERIALIZE	D VIEW LO	<u>)G</u>			SH.SALES
	3	CREATE MATERIALIZE	<u>D_VIEW</u>	MV	/\$\$_00650002	General Match	
	3	GATHER TABLE STATE	STICS	MV:	\$\$_00650002		
Statement ID	Statement	Recommendation ID			Cost Improvement $ abla$	Cost Improvement (%)	Execution Count
2283	SELECT c.cust_id, SUM(amount_sold) AS dollar_sales FROM sales s, customers c WHERE s.cust_id= c.cust_id GROUF BY c.cust_id	3	8	6	2	25.00	2

The Recommendation Details page displays all actions for the specified recommendation.

Under Actions, you can choose to modify the schema name, tablespace name, and storage clause for each action. To view the SQL text of an action, click the link in the Action column for the specified action.

Under SQL Affected by Recommendation, the SQL text of the SQL statement and cost improvement information are displayed.

4. Click OK.

The Recommendations subpage appears.

5. To view the SQL text of a recommendation, select the recommendation and click **Show SQL**.

The Show SQL page for the selected recommendation appears.

```
Show SQL
```

```
Done
Rem SQL Access Advisor: Version 11.1.0.4.0 - Production
Rem
Rem Username: SYS
Rem Task: SQLACCESS6607720
Rem Execution date:
Rem
CREATE MATERIALIZED VIEW LOG ON
"SH". "CUSTOMERS"
WITH ROWID, SEQUENCE ("CUST ID", "CUST LAST NAME", "CUST STATE PROVINCE")
INCLUDING NEW VALUES;
CREATE MATERIALIZED VIEW LOG ON
"SH", "SALES"
WITH ROWID, SEQUENCE ("PROD_ID", "CUST_ID", "QUANTITY_SOLD", "AMOUNT_SOLD")
INCLUDING NEW VALUES;
CREATE MATERIALIZED VIEW "SYS". "MV$$_00650002"
REFRESH FAST WITH ROWID
ENABLE QUERY REWRITE
AS SELECT SH.SALES.CUST_ID C1, SUM("SH"."SALES"."AMOUNT_SOLD") M1,
COUNT ("SH". "SALES". "AMOUNT_SOLD")
M2, COUNT(*) M3 FROM SH.CUSTOMERS, SH.SALES WHERE SH.SALES.CUST_ID = SH.CUSTOMERS.CUST_ID
GROUP BY SH. SALES. CUST ID;
begin
dbms_stats.gather_table_stats('"SYS"','"MV$$_00650002"',NULL,dbms_stats.auto_sample_size);
end;
```

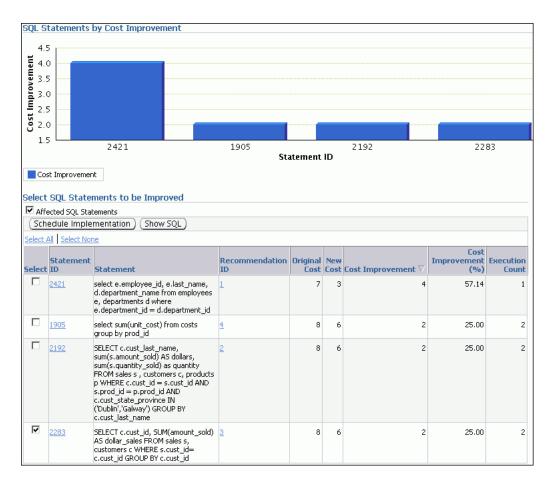
Reviewing the SQL Access Advisor Recommendations: SQL Statements

The SQL Statements subpage ranks SQL statements in the workload by cost improvement. You can use this page to view details about the SQL statements analyzed in the workload.

To review SQL statements:

1. On the Results for Tasks page, click **SQL Statements**.

The SQL Statements subpage appears.



2. Use the SQL Statements by Cost Improvement chart to view SQL statements in the workload ordered by the cost improvement.

Under Select SQL Statements to be Improved, each SQL statement is listed with its statement ID, SQL text, associated recommendation, cost improvement, and execution count.

Implementing the recommendation associated with the top SQL statement will have the biggest benefit to the total performance of the workload. In this example, implementing the recommendation with ID 1 will produce the biggest benefit, a cost improvement of 57.14 percent, for the SQL statement with ID 2421.

3. To view the SQL text of a recommendation, select the recommendation and click **Show SQL**.

The Show SQL page for the selected recommendation appears.

Reviewing the SQL Access Advisor Recommendations: Details

The Details subpage displays a list of all the workload and task options used in the analysis. You can also use this subpage to view a list of journal entries for the task, based on the journaling level used when the task was created.

To review workload and task details:

On the Results for Tasks page, click Details.

The Details subpage appears.

Summar	y <u>Reco</u>	mmendations S	QL Statements	Details				
Workloa	d and Tas	k Options						
		that were selected (when the advisor	task was crea	ted.			
					Previous 1-10 of 29 🔹 Next	10		
Option		Value	Description					
Advisor Mo	ode				L Access Advisor will operate during an analysis, either limite sive (higher quality recommendations)	be		
Analysis So	cope	All Tuning Artifacts	The type of recor	mmendations I	hat are allowed			
Creation C	lost	Consider creation cost			s Advisor will weigh the cost of creation of access structure Jeries and potential improvement in query execution time	s		
Default Inc	dex Schema	None	Specifies the defa	ault owner for	new index recommendations			
Default Ind Tablespace		None	Specifies the defa	ault tablespac	e for new index recommendations			
Default MV Tablespace		None	Specifies the defa	ault tablespac	e for new materialized view log recommendations			
Default MV Schema	/iew	None	Specifies the defa	ault owner for	new materialized view recommendations			
Default MV Tablespace		None	Specifies the default tablespace for new materialized view recommendations					
	efault Partitioning None ablespaces		Specifies the default tablespaces for new Partitioning recommendations					
Excluded A	Actions	None	Contains a list of application actions that are NOT eligible for tuning					
			S Previous 1-10 of 29 🔽 Next 10					
					· <u> </u>			
Journal B	Entries							
		es that were logged naling Level" option			task was executing. The amount of information logged is			
	,				S Previous 1-10 of 18 💌 Next	: 8		
Severity	Entry				Ord	ler		
<u> </u>	Preparing w	orkload for analysis						
<u> </u>	Filter Summ	ary: Valid username	ne: Unused					
<u>í</u>	Filter Summ	ary: Invalid usernar	ne: Unused					
	Filter Summary: Valid module: Unused							
<u>í</u>	Filter Summary: Invalid module: Unused							
<u>í</u>	Filter Summ	ary: Valid action: Ur	iused					
(i)	Filter Summary: Invalid action: Unused							

Under Workload and Task Options, a list of options that were selected when the advisor task was created is displayed.

Under Journal Entries, a list of messages that were logged to the SQL Access Advisor journal while the task was executing is displayed.

Implementing the SQL Access Advisor Recommendations

A SQL Access Advisor recommendation can range from a simple suggestion to a complex solution that requires partitioning a set of existing base tables and implementing a set of database objects such as indexes, materialized views, and materialized view logs. You can select the recommendations for implementation and schedule when the job should be executed.

Tip: Before implementing the SQL Access Advisor recommendations, review them for cost benefits to determine which ones, if any, should be implemented. For more information, see "Reviewing the SQL Access Advisor Recommendations" on page 11-13.

To implement the SQL Access Advisor recommendations:

1. On the Results for Tasks page, click **Recommendations**.

The Recommendations subpage appears.

2. Under Select Recommendations for Implementation, select the recommendation you want to implement and click **Schedule Implementation**.

✓ Incl (Rec	Recommendation ude Retain Actions ommendation De MI Select None					n nentation) (Show SQL)		
	Implementation		Actions	Actio	n Types	Cost Improvement $ abla$	Cost Improvement (%)	Space Used	Affected SQL Statements
V	•	1	4	1	•	4	40.00	0.008	1
		3	4			2	20.00	0.008	1
		4	3			2	20.00	0.008	1
		2	6		-	2	20.00	0.016	1

In this example, the recommendation with ID value 1 is selected.

The Schedule Implementation page appears.

- **3.** In the **Job Name** field, enter a name for the job if you do not want to use the system-generated job name.
- **4.** Determine whether or not the implementation job should stop if an error is encountered. Do one of the following:
 - To stop processing if an error occurs, select **Stop on Error**.
 - To continue processing even if an error occurs, deselect **Stop on Error**.
- **5.** Under Scheduling Options, in the Schedule Type list, select a schedule type for the task and a maintenance window in which the task should run. Do one of the following:
 - Click Standard.

This schedule type enables you to select a repeating interval and start time for the task. Complete the following steps:

- Enter your time zone code in the **Time Zone** field or click the search icon to locate the code for your area.
- In the Repeat list, select **Do Not Repeat** to perform the task only once, or select a unit of time and enter the number of units in the **Interval** field.
- Under Start, select **Immediately** to start the task now, or **Later** to schedule the task to start at a time specified using the **Date** and **Time** fields.
- Click Use predefined schedule.

This schedule type enables you to select an existing schedule. Do one of the following:

- In the Schedule field, enter the name of the schedule to be used for the task.
- To search for a schedule, click the search icon.
 - The Search and Select: Schedule dialog box appears.
 - Select the desired schedule and click **Select**. The selected schedule now appears in the **Schedule** field.
- Click Standard using PL/SQL for repeated interval.

This schedule type enables you to select a repeating interval and an execution window for the task. Complete the following steps:

- Enter your time zone code in the **Time Zone** field or click the search icon to locate the code for your area.
- Under Available to Start, select Immediately to start the task now, or Later to schedule the task to start at a time specified using the Date and Time fields.
- In the Repeat list, select **Do Not Repeat** to perform the task only once, or select a unit of time and enter the number of units in the **Interval** field.
- In the **Repeated Interval** field, enter a PL/SQL schedule expression, such as SYSDATE+1.
- Under Not Available After, select No End Date to indicate that there is no end date for the execution window, or Specified End Date to specify an end date using the Date and Time fields.
- Click Use predefined window.

This schedule type enables you to select an existing window. Select **Stop on Window Close** to stop the job when the window closes. Do one of the following:

- In the **Window** field, enter the name of the window to be used for the task.
- To search for a window, click the search icon.

The Search and Select: Window and Window Groups dialog box appears.

Select the desired window and click **Select**. The selected window now appears in the **Schedule** field.

Click Event.

Complete the following steps:

- Enter your time zone code in the Time Zone field or click the search icon to locate the code for your area.
- Under Event Parameters, enter values in the **Queue Name** and **Condition** fields.
- Under Start, select **Immediately** to start the task now, or **Later** to schedule the task to start at a time specified using the **Date** and **Time** fields.
- Under Not Available After, select No End Date to indicate that there is no end date for the execution window, or Specified End Date to specify an end date using the Date and Time fields.
- Click Calendar.

Complete the following steps:

- Enter your time zone code in the Time Zone field or click the search icon to locate the code for your area.
- Under Calendar Expression, enter a calendar expression.
- Under Start, select Immediately to start the task now, or Later to schedule the task to start at a time specified using the Date and Time fields.
- Under Not Available After, select No End Date to indicate that there is no end date for the execution window, or Specified End Date to specify an end date using the Date and Time fields.

In this example, **Standard** is selected for schedule type. The job will not repeat and is scheduled to start immediately.

* Job Name SQLACCESSIMPL6626256
Stop on Error If checked, this implementation job will stop processing if an error occurs. If not checked, this job will ignore errors and will continue processing all actions of selected recommendations.
Scheduling Options
Schedule Type Standard
Time Zone PST8PDT
Repeating
Repeat Do Not Repeat 💌
Start
Immediately
O Later
Date Mar 29, 2007 (example: Mar 29, 2007)
Time 12 V 05 V 00 V C AM © PM

- 6. Optionally, click **Show SQL** to view the SQL text for the job.
- 7. To submit the job, click **Submit**.

If the job is scheduled to start immediately, then the Results for Tasks page for the SQL Access Advisor task appears with a confirmation that the job was successfully created.

Confirmation SQL Access Advisor implem	entation job SYS.SQLACCESSI	MPL6626256 created successfu	illy.
Task Name	SQLACCE559384670	Started	Mar 29, 2007 11:50:17 AM PDT
Status	COMPLETED	Ended	Mar 29, 2007 11:51:15 AM PDT
Advisor Mode Scheduler Job	LIMITED ADV_SQLACCES59384670	Running Time (seconds) Time Limit (seconds)	58

- **8.** Do one of the following, depending on whether the job is scheduled to start immediately or later:
 - If you submitted the job immediately, and if the Results for Task page is shown, then click the link in the **Scheduler Job** field to display the View Job page. Go to Step 10.
 - If the job is scheduled to start at a later time, then proceed to Step 9.
- **9.** Complete the following steps:
 - **a.** On the Server page, under Oracle Scheduler, click **Jobs**.

The Scheduler Jobs page appears.

b. Select the implementation job and click **View Job Definition**.

The View Job page for the selected job appears.

10. On the View Job page, under Operation Detail, check the status of the operation.

Operation Detail			
View			
Select Log ID	Log Date 🗸	Operation	Status
• <u>1153</u>	Mar 28, 2007 6:48:43 PM -07:00	RUN	SUCCEEDED

11. Optionally, select the operation and click **View**.

The Operation Detail page appears.

This page contains information (such as start date and time, run duration, CPU time used, and session ID) that you can use to troubleshoot the failure.

12. On the Schema subpage, verify that the access structure recommended by SQL Access Advisor is created.

Depending on the type of access structure that is created, you can display the access structure using the Indexes page, Materialized Views page, or the Materialized View Logs page.

In this example, a materialized view named MV\$\$_00690000 is created in the SH schema.

Search	1								
Enter a	schema na	ame and an object name to fil	ter the data	that is displa	yed in you	r results	set.		
s	chema SH	1	_ 💉						
Object	Name								
		20							
		h returns all uppercase matches be u can use the wildcard symbol (%)			ntered. To n	in an exa	ct or case-sensi	tive mat	ch, double quote
		can use the wildcard symbol (70)	, in a double (• • • • • • • • •		• • • • • • • • • • •	• • • • • •	
Selecti	on Mode	Single 💌							Create
Edi	t) (View	Delete Actions Create	e Like	•	G0				
Select	Schema	Materialized View(Snapshot)	Master Owner	Master Table	Master Link	Туре	Updatable	Can Use Log	Last Refresh
œ	SH	FWEEK PSCAT SALES MV	SH	PRODUCTS		FORCE	NO	YES	Mar 26, 2007 12:10:15 PM PDT
0	SH	<u>CAL MONTH SALES MV</u>	SH	TIMES		FORCE	NO	YES	Mar 26, 2007 12:10:15 PM PDT
0	зн	<u>MV\$\$_00690000</u>	SH	COSTS		FAST	NO	YES	Mar 28, 2007 7:14:05 PM PDT

Analyzing SQL Performance Impact

System changes, such as upgrading a database or adding an index, may cause changes to execution plans of SQL statements, resulting in a significant impact on SQL performance. In some cases, the system changes may cause SQL statements to regress, resulting in performance degradation. In other cases, the system changes may improve SQL performance. Being able to accurately forecast the potential impact of system changes on SQL performance enables you to tune the system beforehand in cases where the SQL statements regress, or to validate and measure the performance gain in cases where the performance of the SQL statements improves.

SQL Performance Analyzer enables you to forecast the impact of system changes on a SQL workload by:

- Measuring the performance before and after the change
- Generating a report that describes the change in performance
- Identifying the SQL statements that regressed or improved
- Providing tuning recommendations for each SQL statement that regressed
- Enabling you to implement the tuning recommendations when appropriate

This chapter contains the following sections:

- SQL Performance Analyzer Usage
- SQL Performance Analyzer Methodology
- Running SQL Performance Analyzer
- Reviewing the SQL Performance Analyzer Report

See Also:

 Oracle Database Performance Tuning Guide to learn how to run SQL Performance Analyzer with the DBMS_SQLPA package

SQL Performance Analyzer Usage

You can use SQL Performance Analyzer to analyze the SQL performance impact of any type of system changes. Examples of system changes include the following:

Database upgrade

When performing a database upgrade, you may not be able to predict how the system will perform after the upgrade, or if an existing functionality may be adversely affected. For example, a database upgrade installs a new version of the optimizer, which has an effect on SQL performance. SQL Performance Analyzer enables you to compare the SQL performance between two versions of Oracle

Database. In this way, you can identify and tune SQL statements that may potentially regress after the database upgrade without affecting your production system.

Application upgrade

An application upgrade may involve changes to the database, such as redesigning tables, or adding and removing indexes. As is the case with database upgrades, it is difficult to predict how the system will perform after the changes, or if an existing functionality may be adversely affected. SQL Performance Analyzer enables you to compare the SQL performance between two versions of an application. Thus, you can tune SQL statements that may potentially regress after the upgrade.

Changes to the operating system, hardware, and database configuration

Changes to the operating systems (such as installing a new operating system), hardware (such as adding more CPU or memory), or database configuration (such as moving from a single instance database environment to Oracle Real Application Clusters) may have a significant effect on SQL performance. SQL Performance Analyzer enables you to determine the improvement or deterioration to SQL performance when making these changes.

Schema change

Changing a schema, such as altering indexes or creating new ones, almost inevitably affects SQL performance. SQL Performance Analyzer enables you to determine the effect on SQL performance when making a schema change.

Database initialization parameter change

Changing the value of a database parameter may produce unexpected results. For example, you may enable a specific initialization parameter to improve performance, but this change may produce unexpected results because the system constraints may have changed. SQL Performance Analyzer enables you to determine the effect on SQL performance when changing a database initialization parameter.

SQL tuning

Accepting recommendations from an advisor (such as ADDM, SQL Tuning Advisor, or SQL Access Advisor) may require you to tune problematic SQL statements. For example, SQL Tuning Advisor may recommend that you accept a SQL profile for a particular SQL statement. SQL Performance Analyzer enables you to measure the performance improvement that may be gained by tuning SQL statements as recommended by the advisors, and determine whether to accept these recommendations.

SQL Performance Analyzer Methodology

You can run SQL Performance Analyzer on a test system that closely resembles the production system, or on the production system itself. Performing a SQL Performance Analyzer analysis is resource-intensive, so performing the analysis on the production system may cause significant performance degradation.

Any global changes made on the system to test the performance effect may also affect other users of the system. For smaller changes, such as adding or dropping an index, the effect on other users may be acceptable. However, for systemwide changes, such as a database upgrade, using a production system is not recommended and should be considered only if a test system is unavailable. If a separate test system is available, then running SQL Performance Analyzer on the test system enables you to test the effects of the changes without affecting the production system.

To ensure that SQL Performance Analyzer can accurately analyze the SQL performance impact, the test system should be as similar to the production system as possible. For these reasons, running SQL Performance Analyzer on a test system is recommended and is the methodology described here. If you choose to run SQL Performance Analyzer on the production system, then substitute the production system for the test system where applicable.

Analyzing the SQL performance effect of system changes using SQL Performance Analyzer is an iterative process that involves the following steps:

- 1. Capture the SQL workload that you want to analyze on the production system, as described in "Capturing and Transporting a SQL Workload" on page 12-3.
- **2.** Transport the SQL workload from the production system to the test system, as described in "Capturing and Transporting a SQL Workload" on page 12-3.
- **3.** Create a SQL Performance Analyzer task on the test system using the SQL workload as its input source, as described in "Following a Guided Workflow with SQL Performance Analyzer" on page 12-12.
- **4.** Set up the environment on the test system to match the production system as closely as possible, as described in "Establishing the Initial Environment" on page 12-14.
- **5.** Build the pre-change performance data by executing the SQL workload on the system before the change, as described in "Collecting SQL Performance Data Before the Change" on page 12-14.
- **6.** Perform the system change on the test system, as described in "Making the System Change" on page 12-16.
- Build the post-change performance data by executing the SQL workload on the system after the change, as described in "Collecting SQL Performance Data After the Change" on page 12-16.
- Compare and analyze the pre-change and post-change versions of performance data, as described in "Comparing SQL Performance Before and After the Change" on page 12-17.
- **9.** Generate and review a report to identify the SQL statements in the SQL workload that have improved, remain unchanged, or regressed after the system change.
- **10.** Tune any regressed SQL statements that are identified, as described in Chapter 10, "Tuning SQL Statements".
- **11.** Ensure that the performance of the tuned SQL statements is acceptable by repeating Step 5 through Step 10 until your performance goals are met.

This section contains the following topics:

- Capturing and Transporting a SQL Workload
- Setting Up the Database Environment on the Test System
- Executing a SQL Workload

Capturing and Transporting a SQL Workload

Before running SQL Performance Analyzer, capture a set of SQL statements on the production system that represents the SQL workload that you intend to analyze and

transport to the test system. The captured SQL statements should include the following information:

- SQL text
- Execution environment
 - SQL binds, which are bind values needed to execute a SQL statement and generate accurate execution statistics
 - Parsing schema under which a SQL statement can be compiled
 - Compilation environment, including initialization parameters under which a SQL statement is executed
- Number of times a SQL statement was executed

You can store captured SQL statements in SQL Tuning Sets and use them as an input source for SQL Performance Analyzer. Capturing a SQL workload using a SQL Tuning Set enables you to do the following:

- Store the SQL text and any necessary auxiliary information in a single, persistent database object
- Populate, update, delete, and select captured SQL statements in the SQL Tuning Set
- Load and merge content from various data sources, such as the Automatic Workload Repository (AWR) or the cursor cache
- Export the SQL Tuning Set from the system where the SQL workload is captured and import it into another system
- Reuse the SQL workload as an input source for other advisors, such as SQL Tuning Advisor and SQL Access Advisor

After you have captured the SQL workload into a SQL Tuning Set on the production system, you must transport it to the test system.

See Also:

- "Creating a SQL Tuning Set" on page 10-8
- Oracle Database Performance Tuning Guide for information about transporting SQL Tuning Sets

Setting Up the Database Environment on the Test System

Depending on the system change that you intend to analyze, you must set up the database environment on the test system before and after performing the system change. Before the system change, set up the database environment on the test system to match the database environment in the production system as closely as possible. In this way, SQL Performance Analyzer can more accurately forecast the effect of the system change on SQL performance.

For example, to test how changing a database initialization parameter will affect SQL performance, complete the following steps:

- **1.** Set the database initialization parameter on the test system to the same value as the production system.
- **2.** Build the pre-change SQL performance data.
- 3. Set the database initialization parameter to the new value you want to test.
- 4. Build the post-change SQL performance data.

5. Compare the two sets of performance data.

Similarly, to test the performance effect of a database upgrade from release 10g to release 11g, perform the following tasks:

- **1.** Install Oracle Database 11*g* on the test system.
- **2.** Revert the OPTIMIZER_FEATURES_ENABLE initialization parameter to the database version on the production system.
- **3.** Build the pre-change SQL performance data.
- **4.** Set the OPTIMIZER_FEATURES_ENABLE initialization parameter to the database version to which you are upgrading.
- 5. Build the post-change SQL performance data.
- **6.** Compare the two sets of performance data.

See Also:

 Oracle Database Reference to learn about the OPTIMIZER_FEATURES_ENABLE initialization parameter

Executing a SQL Workload

After the SQL workload is captured and transported to the test system, and the initial database environment is properly configured, execute the SQL workload to build the pre-change performance data before making the system change. Executing a SQL workload runs each of the SQL statements contained in the workload to completion. During execution, SQL Performance Analyzer generates execution plans and computes execution statistics for each SQL statement in the workload. After the pre-change performance data is built, you can perform the system change.

After performing the system change, execute the SQL workload again to build the post-change performance data. SQL Performance Analyzer generates execution plans and computes execution statistics for each SQL statement in the workload a second time, resulting in a new set of performance data that can be used to compare to the pre-change version of the performance data.

Depending on its size, executing a SQL workload can be resource-intensive and cause a significant performance impact. When executing a SQL workload, you can choose to generate execution plans only, without collecting execution statistics. This technique shortens the time to run the execution and lessens the effect on system resources, but the results of the comparison analysis may not be as accurate. If you are running SQL Performance Analyzer on the production system, then consider executing the SQL workload using a private session to avoid affecting the rest of the system.

Running SQL Performance Analyzer

SQL Performance Analyzer enables you to analyze the effects of environmental changes on execution of SQL statements in SQL Tuning Sets. As explained in "Managing SQL Tuning Sets" on page 10-8, a SQL Tuning Set is a database object that includes one or more SQL statements along with their execution statistics and execution context. In addition to the performance analysis, SQL Performance Analyzer can invoke SQL Advisor and provide tuning recommendations.

SQL Performance Analyzer guides you through the SQL workload comparison by means of the following workflows:

Optimizer Upgrade Simulation

Use this workflow to simulate a database upgrade and measure its effect on a workload.

Parameter Change

Use this workflow to determine how a database initialization parameter change will affect SQL performance.

Guided Workflow

Use this workflow to compare the performance of SQL execution on different databases.

In each of the preceding workflows, you must create a SQL Performance Analyzer task. A task is a container for the results of SQL replay trials. A replay trial captures the execution performance of a SQL Tuning Set under specific environmental conditions.

To run SQL Performance Analyzer:

1. On the Database Home page, click Advisor Central.

The Advisor Central page appears.

2. Click SQL Performance Analyzer.

SQL Performance Analyzer page appears. A list of existing SQL Performance Analyzer tasks are displayed.

3QL Perf	formance Analyzer						
					Page Refr	eshed Mar 31, 2007 1:27	04 PM PDT Refresh
5QL Perform Funing Set.	iance Analyzer allows you t	o analyze t	he effect	s of environme	ental changes on th	e execution performance of	SQL contained in a SQL
SQL Pe	erformance Analyzer \	Workflow	s				
Create a	and execute SQL Performar	nce Analyze	r Task ex	(periments of (different types usin	g the following links.	
Optimize	er Upgrade Simulation	Test the e	effects of	optimizer ver:	sion changes on SQ	L Tuning Set performance.	
Parame	ter Change	Test and	compare	an initializatior	n parameter change	on SQL Tuning Set perform	ance.
Guided '	Workflow	Create a : trials.	5QL Perf	ormance Analy	zer Task and execu	ite custom experiments usin	g manually created replay
SQL Pe	erformance Analyzer 1 ete)	Tasks					
	SQL Performance Analy	yzer Task	Owner	Description	Last Run Status	Created	Last Modified $ abla$
Select	aver chormance minar						Last Mounteu V
0	<u>COMP_WK</u>		DBA1		~	Mar 30, 2007 11:05:35 AM	
•			DBA1 DBA1		•	Mar 30, 2007 11:05:35 AM Mar 30, 2007 11:04:15 AM	Mar 30, 2007 11:07:09 AM

- **3.** Do one of the following:
 - Proceed to "Performing an Optimizer Upgrade Simulation with SQL Performance Analyzer" on page 12-7.
 - Proceed to "Testing an Initialization Parameter Change with SQL Performance Analyzer" on page 12-10.
 - Proceed to "Following a Guided Workflow with SQL Performance Analyzer" on page 12-12.

See Also:

Oracle Database Performance Tuning Guide for information about running SQL Performance Analyzer using APIs

Performing an Optimizer Upgrade Simulation with SQL Performance Analyzer

SQL Performance Analyzer can automatically simulate the effect of a database upgrade on SQL performance. The simulation is based on changing the OPTIMIZER_FEATURES_ENABLE initialization parameter. All SQL statements use the optimizer, which is a part of Oracle Database that determines the most efficient means of accessing the specified data. You could use SQL Performance Analyzer to compare the performance of SQL execution when using the 10.2.0.2 and 11.1.0.1 versions of the optimizer.

After you select a SQL Tuning Set and a comparison metric, SQL Performance Analyzer creates two replay trials. The first trial captures SQL performance by simulating the optimizer from the user-selected previous release, whereas the second trial uses the optimizer from the current release. The system-generated Replay Trial Comparison report evaluates SQL regression. If performance was degraded, then you can then use SQL Tuning Advisor to develop SQL profiles for regressed SQL.

To simulate an optimizer upgrade:

1. On the SQL Performance Analyzer page, click **Optimizer Upgrade Simulation**.

The Optimizer Upgrade Simulation page appears.

Optimizer Upgrade Simulation	
Test the effects of optimizer version changes on SQL Tuning Set performance.	Cancel) (Submit
Task Information * Task Name * SQL Tuning Set Description Per-SQL Time Limit UNLIMITED Image: Comparison of SQL EXPLAIN ONLY generates plans without test execution of SQL EXPLAIN ONLY generates plans Optimizer Versions Version 1 Version 2 Interview	 Simulating and pre-tuning 11g upgrades SQL Performance Analyzer can automatically simulate the effects of upgrading from 10g to 11g releases of Oracle. This is accomplished as follows: SQL Tuning Set of representative workload from the 10g database has been migrated to this 11g database. (This step is not automated, refer to 10g documentation.) Two Replay Trials are created: the first captures STS performance with the optimizer simulating the 10g optimizer, the second uses the native 11g optimizer. A Replay Trial Comparison report is run for the two trials. The specified comparison metric is used as a basis for regression evaluation. SQL Tuning Advisor can be used to develop SQL profiles for regressed SQL, allowing you to pre-tune any SQL that will be negatively impacted by an upgrade. NOTE: This feature can be used to test both planned major upgrades to 11g and 11g Critical Patch Upgrades.
Evaluation Comparison Metric Execute Elapsed Time Schedule Time Zone America/Los_Angeles Grandlately C Later Date Mar 31, 2007 (example: Mar 31, 2007) Time 1 S 5 0 00 C AM PM	

2. In the Task Name field, enter the name of the task.

For example, enter 111_UPG_OPT_COST.

- **3.** Select the SQL Tuning Set to use for the comparison. Do one of the following:
 - In SQL Tuning Set, enter the name the SQL Tuning Set that contains the SQL workload to be analyzed.
 - Click the search icon to search for a SQL Tuning Set, and then select the set.
 The tuning set now appears in the SQL Tuning Set field.
- **4.** In the **Description** field, optionally enter a description of the task.

For example enter the following text: This task simulates an upgrade from 10.2.0.2 to 11.1.0.1.

5. In the Per-SQL Time Limit list, select the time limit for SQL execution during the replay. Do one of the following:

• Select UNLIMITED.

The execution will run each of the SQL statements in the SQL Tuning Set to completion and gather performance data. Collecting execution statistics provides greater accuracy in the performance analysis but takes a longer time.

• Select EXPLAIN ONLY.

The task will generate execution plans only. This option shortens the execution time, but the performance analysis may not be as accurate.

- Select **Customize** and then enter the specified number of hours, minutes, or seconds.
- 6. In the Optimizer Versions section, select the optimizer versions for comparison in the Version 1 and Version 2 lists.

In this example, the optimizer versions 10.2.0.2 and 11.1.0.1 are selected for comparison.

- **7.** In the Comparison Metric list, select the comparison metric to use for the analysis. Do one of the following:
 - If you selected **EXPLAIN ONLY** in Step 5, then select **Optimizer Cost**.
 - If you selected UNLIMITED or Customize in Step 5, then select any of the following options:
 - Elapsed Time
 - CPU Time
 - Buffer Gets
 - Disk Reads
 - Direct Writes
 - Optimizer Cost

To perform the comparison analysis by using more than one comparison metric, perform separate comparison analyses by repeating this procedure with different metrics.

- **8.** In the Schedule section, do one of the following:
 - **a.** Enter your time zone code in the **Time Zone** field or click the search icon to locate the code for your area.
 - **b.** Select **Immediately** to start the task now, or **Later** to schedule the task to start at a time specified with the **Date** and **Time** fields.
- 9. Click Submit.

A confirmation message appears.

In the SQL Performance Analyzer Tasks table, the status icon of this task changes to an arrow while the execution is in progress. To refresh the status icon, click **Refresh**. After the task completes, the Status icon changes to a check mark.

SQL P	erformance Analyzer T	asks							
Del	Delete								
Select	SQL Performance Analyzer Task	Owner	Description	Last Run Status	Created	Last Modified $ abla$			
œ	111 UPG OPT COST	DBA1	This task simulates an upgrade from 10.2.0.2 to 11.1.0.1	1	Mar 31, 2007 2:22:30 PM	Mar 31, 2007 2:22:31 PM			
0	<u>COMP_WK</u>	DBA1		× .	Mar 30, 2007 11:05:35 AM	Mar 30, 2007 11:07:09 AM			
0	WORK COMP	DBA1		×	Mar 30, 2007 11:04:15 AM	Mar 30, 2007 11:04:40 AM			
0	COMP_STS	DBA1		× .	Mar 30, 2007 10:51:40 AM	Mar 30, 2007 10:58:32 AM			

10. In the SQL Performance Analyzer Tasks table, select the optimizer task and click the link in the **SQL Performance Analyzer Task** column.

The SQL Performance Analyzer Task page appears.

SQL Performance	e Analyz	er Task: DBA	1.111_UPG	_OP1	r_cc	DST		
			Page Refres	ned Ma	ır 31,	2007 2:4	3:07 PM P	DT Refresh
The SQL Performance Ar changed environmental								
SQL Tuning Set								
		CURSOR_CACH DBA1	E_MAR2907	Numi	oer of	Descriptio Statement		
Replay Trials								
A Replay Trial captu	res the exe	cution performanc	e of the SQL Tu	ning Sel	: unde	r specific e	nvironment	al conditions.
							Create	e Replay Trial)
Replay Trial Name	Descript	ion			Crea	ted	SQL Executed	Status
initial_sql_trial	parameter '10.2.0.2'	r optimizer_featur	es_enable set to)	3/31/ PM	07 2:22	Yes	COMPLETED
second_sql_trial	parameter '11.1.0.1.	r optimizer_featur 1'	es_enable set to)	3/31/ PM	07 2:22	Yes	COMPLETED
Replay Trial Con Compare Replay Tria			f environmental	differer	ices oi			cution costs. Il Comparison)
		C		_			. ,	
Trial 1 Name Tria	l 2 Name	Compare Metric	Created	Statu		Compari: Report	son	SQL Tune Report
initial_sql_trial seco	nd_sql_trial	Optimizer Cost	3/31/07 2:22 PM	COMPL	ETED	øœ		

This page contains the following sections:

SQL Tuning Set

This section summarizes information about the tuning set, including the name, owner, description, and number of statements in the set.

Replay Trials

This section includes a table that lists the replay trials used in the SQL Performance Analyzer task.

Replay Trial Comparisons

This section contains a table lists the results of the workload comparisons.

11. Click the icon in the **Comparison Report** column.

The SQL Performance Analyzer Task Result page appears.

12. Review the results of the performance analysis, as described in "Reviewing the SQL Performance Analyzer Report" on page 12-19.

See Also:

 Oracle Database Reference for more information about the OPTIMIZER_FEATURES_ENABLE initialization parameter

Testing an Initialization Parameter Change with SQL Performance Analyzer

The Parameter Change workflow enables you to test the performance effect on a SQL Tuning Set when you vary a single environment initialization parameter between two values. For example, you can compare SQL performance when the sort area size is increased from 1 MB to 2 MB.

After you select a SQL Tuning Set and a comparison metric, SQL Performance Analyzer creates a task and performs a trial with the initialization parameter set to the original value. The Analyzer then performs a second trial with the parameter set to the new value. The system-generated Replay Trial Comparison report evaluates the regression.

To test an initialization parameter change:

1. On the SQL Performance Analyzer page, click **Parameter Change**.

The Parameter Change page appears.

Parameter Change	
	Cancel Submit
	Measuring parameter change effects
Task Information	Parameter Change allows you to test the performance impact on a SQL Tuning Set by varying a single environment initialization parameter between two values.
* Task Name	A SQL Performance Analyzer Task is created and initial Trial run is performed with
* SQL Tuning Set 📃 🚀	 A SQL Performance Analyzer Lask is created and initial maintains performed with parameter set to the Base Value.
Description	 A second Trial run is performed with parameter set to the Changed Value.
Per-SQL Time Limit UNLIMITED	 A Replay Trial Comparison report is run for the two trials. The specified comparison
Ø TIP Time limit is on elapsed time of test execution	metric is used as a basis for regression evaluation.
of SQL. EXPLAIN ONLY generates plans without test execution.	
without test execution.	
Parameter Change	
* Parameter Name 🛛 🚽	
* Base Value	
* Changed Value	
- Changed Value	
Evaluation	
Comparison Metric Execute Elapsed Time 💌	
Companson Medic Execute Liapsed nine	
Schedule	
Time Zone Pacific/Pago_Pago 🔹	
• Immediately	
O Later	
Date Mar 31, 2007	
Time 5 🔹 39 🔹 00 🔹 C AM 👁 PM	

2. In the Task Name field, enter the name of the task.

For example, enter SORT_TIME.

- **3.** Select the SQL Tuning Set. Do one of the following:
 - In SQL Tuning Set, enter the name the SQL Tuning Set that contains the SQL workload to be analyzed.
 - Click the search icon to search for a SQL Tuning Set, and then select the set.

The tuning set now appears in the SQL Tuning Set field.

4. In the **Description** field, optionally enter a description of the task.

For example enter the following text: Double the value of sort_area_size.

- **5.** In the Per-SQL Time Limit list, determine the time limit for SQL execution during the replay. Do one of the following:
 - Select UNLIMITED.

The execution will run each of the SQL statements in the SQL Tuning Set to completion and gather performance data. Collecting execution statistics provides greater accuracy in the performance analysis but takes a longer time.

Select EXPLAIN ONLY.

The task will generate execution plans only. This option shortens the execution time, but the performance analysis may not be as accurate.

- Select Customize and then enter the specified number of hours, minutes, or seconds.
- 6. In the Parameter Change section, complete the following steps:
 - **a.** In the **Parameter Name** field, enter the name of the initialization parameter whose value you want to modify, or click the Search icon to review the current parameter settings.

For example, enter sort_area_size.

b. In the **Base Value** field, enter the current value of the initialization parameter.

For example, enter 1048576.

c. In the **Changed Value** field, enter the new value of the initialization parameter.

For example, enter 2097152.

- **7.** In the Comparison Metric list, select the comparison metric to use for the analysis. Do one of the following:
 - If you selected EXPLAIN ONLY in Step 5, then select Optimizer Cost.
 - If you selected **UNLIMITED** or **Customize** in Step 5, then select any of the following options:
 - Elapsed Time
 - CPU Time
 - Buffer Gets
 - Disk Reads
 - Direct Writes
 - Optimizer Cost

To perform the comparison analysis by using more than one comparison metric, perform separate comparison analyses by repeating this procedure with different metrics.

- 8. In the Schedule section, do one of the following:
 - **a.** Enter your time zone code in the **Time Zone** field or click the search icon to locate the code for your area.
 - **b.** Select **Immediately** to start the task now, or **Later** to schedule the task to start at a time specified with the **Date** and **Time** fields.
- 9. Click Submit.

A confirmation message appears.

In the SQL Performance Analyzer Tasks table, the status icon of this task changes to an arrow while the execution is in progress. To refresh the status icon, click **Refresh**. After the task completes, the Status icon changes to a check mark.

SQL Performance Analyzer Tasks									
Delete									
	SQL Performance Analyzer Task	Owner	Description	Last Run Status	Created	Last Modified $ abla$			
œ	SORT TIME		Double the value of sort_area_size		Mar 31, 2007 5:56:34 PM	Mar 31, 2007 5:56:34 PM			

10. In the SQL Performance Analyzer Tasks table, select the optimizer task and click the link in the **SQL Performance Analyzer Task** column.

The SQL Performance Analyzer Task page appears.

e SQL Performanc	e Analyzer Tas	k is	Task: DBA1.SC	Page Ref rimental results of	executing a	specific :			Refresh changed
SQL Tuning		Sing	, the impact of enviro	ninentai chariyes	UII 515 EXEC	uon pei	ronnance,		
oge running i	Nar Own		CURSOR_CACHE_ DBA1	MAR2907	Number of	Descrip Stateme			
Replay Trials									
		ecuti	ion performance of t	he SQL Tuning Set	t under specif	ic enviro	nmental conditi	ons.	
							Crea	ite Ri	eplay Trial
Replay Trial N	ame Des	rip	tion		Created		Crea		eplay Trial Status
Replay Trial M initial_sql_trial			tion er sort_area_size sel	: to 1048576	Created 3/31/07 5:	56 PM		d 9	
	para	mete					SQL Execute	d 9	Status
initial_sql_trial second_sql_tria Replay Trial I	para I para Comparison:	meta meta	er sort_area_size sel	: to 2097152	3/31/07 5: 3/31/07 5:	56 PM 'uning_Se	SQL Execute Yes Yes	d s	Status COMPLETED COMPLETED
initial_sql_trial second_sql_tria Replay Trial (Compare Replay	para al para Comparison: y Trials to asses	meto meto s ch	er sort_area_size sel er sort_area_size sel nange impact of envir	: to 2097152 ronmental differer	3/31/07 5: 3/31/07 5: nces on SQL T	56 PM uning Se	SQL Execute Yes Yes	d g o ts. ial Co	Status COMPLETED COMPLETED

This page contains the following sections:

SQL Tuning Set

This section summarizes information about the tuning set, including the name, owner, description, and number of statements in the set.

Replay Trials

This section includes a table that lists the replay trials used in the SQL Performance Analyzer task.

Replay Trial Comparisons

This section contains a table lists the results of the workload comparisons.

11. Click the icon in the **Comparison Report** column.

The SQL Performance Analyzer Task Result page appears.

12. Review the results of the performance analysis, as described in "Reviewing the SQL Performance Analyzer Report" on page 12-19.

Following a Guided Workflow with SQL Performance Analyzer

You can use the guided workflow to compare the performance of SQL statements before and after a variety of system changes that can impact the performance of the SQL workload.

Tip: Before you can create a SQL Performance Analyzer task, capture the SQL workload to be used in the performance analysis into a SQL Tuning Set on the production system. Afterward, transport the workload to the test system where the performance analysis will be performed, as described in "Capturing and Transporting a SQL Workload" on page 12-3.

To initiate a guided workflow:

1. On the SQL Performance Analyzer page, click **Guided Workflow**.

The Guided Workflow page appears.

This page lists the required steps in the SQL Performance Analyzer task in sequential order. Each step must be completed in the order displayed before you can begin the next step.

Guide	ed Workflow			
	Page Refreshed Apr 27, 2007 2:20:30 PM PDT (Refresh) View I	Data Real Time	: 15 Second	l Refresh 👤
The foll Analyze	owing guided workflow contains the sequence of steps necessary to execute a su er test.	ccessful two-trial	SQL Perfor	mance
Note: B	e sure that the Trial environment matches the tests you want to conduct.			
Step	Description	Excecuted	Status	Execute
1	Create SQL Performance Analyzer Task based on SQL Tuning Set			
2	Replay SQL Tuning Set in Initial Environment			
3	Replay SQL Tuning Set in Changed Environment			
4	Compare Step 2 and Step 3			
5	View Trial Comparison Report			

2. Proceed to the next step, as described in "Creating a SQL Performance Analyzer Task Based on a SQL Tuning Set" on page 12-13.

Creating a SQL Performance Analyzer Task Based on a SQL Tuning Set

To run SQL Performance Analyzer, you must create a SQL Performance Analyzer task. The task requires you to select the SQL Tuning Set containing the workload to be used in the performance analysis.

The SQL Tuning Set remains constant in the SQL Performance Analyzer task and is executed in isolation during each replay trial. Thus, performance differences between trials are caused by environmental differences.

Tip: Before you can create a SQL Performance Analyzer task based on a SQL Tuning Set, capture the SQL workload for the performance analysis in a SQL Tuning Set on the production system. Transport the set to the test system, as described in "Capturing and Transporting a SQL Workload" on page 12-3.

To create a task based on a SQL Tuning Set:

1. In the Guided Workflow page, click the **Execute** icon for the Create SQL Performance Analyzer Task based on SQL Tuning Set step.

The Create SQL Performance Analyzer Task page appears.

Create SQL Performance Analyzer Task
The SQL Performance Analyzer Task is a container for the execution of replay experiments designed to test the effects of changes in execution environment on the SQL performance of an STS.
*Name
Owner DBA1
Description of TIP Use the description to characterize the intended SQL Performance Analyzer investigations.
SQL Tuning Set
The SQL Tuning Set is the basis for SQL Performance Analyzer Task experiments. The ST5 should represent a coherent set of SQL for the changes being investigated (e.g. full workload for an upgrade test).
* Name
♂ TIP You can create a new STS here: Link to STS Creation Wizard

- 2. In the Name field, enter the name of the task.
- **3.** In the **Description** field, optionally enter a description of the task.
- 4. In the SQL Tuning Set section, do one of the following:
 - In Name, enter the name the SQL Tuning Set that contains the SQL workload to be analyzed.
 - Click the search icon to search for a SQL Tuning Set, and then select the set.

The tuning set now appears in the Name field.

5. Click Create.

The Guided Workflow page appears.

The Status icon of this step has changed to a check mark and the **Execute** icon for the next step is now enabled.

6. Proceed to the next step, as described in "Establishing the Initial Environment" on page 12-14.

Establishing the Initial Environment

After selecting a SQL Tuning Set as the input source, establish the initial environment on the test system. This step is not included in the Guided Workflow page because you must perform it manually. For more information about setting up the database environment, see "Setting Up the Database Environment on the Test System" on page 12-4.

Tip: Before you establish the initial environment, select a SQL Tuning Set, as described in "Creating a SQL Performance Analyzer Task Based on a SQL Tuning Set" on page 12-13.

To establish the initial environment:

1. On the test system, manually make any necessary environmental changes affecting SQL optimization and performance.

These changes could include changing initialization parameters, gathering or setting optimizer statistics, and creating indexes.

2. Proceed to the next step, as described in "Collecting SQL Performance Data Before the Change".

Collecting SQL Performance Data Before the Change

After you have properly configured the initial environment on the test system, build the pre-change version of performance data by executing the SQL workload before performing the system change. For more information about executing a workload, see "Executing a SQL Workload" on page 12-5.

Tip: Before computing the pre-change version of performance data, establish the initial environment, as described in "Establishing the Initial Environment" on page 12-14.

To collect SQL performance data before the change:

1. On the Guided Workflow page, click the **Execute** icon for the Replay SQL Tuning Set in Initial Environment step.

The Create Replay Trial page appears. A summary of the selected SQL Tuning Set containing the SQL workload is displayed.

Create Replay Trial	
	Cancel Submit
Replay Trials capture execution performance of the STS under a given optimizer environment.	Trial environment determines results
SQL Performance Analyzer Task DBA1.PERF_ANALYZER_TASK	The SQL Tuning Set remains constants under the SQL Performance
SQL Tuning Set	Analyzer Task and its SQL is executed in isolation to create each Replay Trial. Performance differences between trials are thus
*Replay Trial Name SQL_REPLAY_1175211780874	attributed to environmental differences between trials.
Replay Trial Description	Environmental changes affecting SQL optimization and performance may need to be made manually prior to execution of the Trial. These could include changing initialization parameters, gathering or setting optimizer statistics and creating indexes.
Per-SQL Time Limit Customize 🔹 5 minute(s) 💌	
TIP Time limit is on elapsed time of test execution of SQL. EXPLAIN	NOTE: Be sure trial environment has been established prior to submitting.
ONLY generates plans without test execution.	Trial environment established 🔽
Schedule	
Time Zone Pacific/Pago_Pago 🔹	
Immediately	
O Later	
Date Mar 29, 2007 (example: Mar 29, 2007)	
Time 3 • 10 • 00 • C AM @ PM	

- 2. In the **Replay Trial Name** field, enter the name of the replay trial.
- 3. In the **Replay Trial Description** field, enter a description of the replay trial.
- **4.** In the Per-SQL Time Limit list, determine the time limit for SQL execution during the replay. Do one of the following:
 - Select UNLIMITED.

The execution will run each of the SQL statements in the SQL Tuning Set to completion and gather performance data. Collecting execution statistics provides greater accuracy in the performance analysis but takes a longer time.

Select EXPLAIN ONLY.

The task will generate execution plans only. This option shortens the execution time, but the performance analysis may not be as accurate.

- Select Customize and then enter the specified number of minutes.
- **5.** Ensure that the database environment on the test system matches the production environment as closely as possible, and select **Trial environment established**.
- **6**. In the Schedule section, do one of the following:
 - **a.** Enter your time zone code in the **Time Zone** field or click the search icon to locate the code for your area.
 - **b.** Select **Immediately** to start the task now, or **Later** to schedule the task to start at a time specified with the **Date** and **Time** fields.
- 7. Click OK.

The Guided Workflow page appears when the execution begins.

The status icon of this step changes to a clock while the execution is in progress. To refresh the status icon, click **Refresh**. Depending on the options selected and the size of the SQL workload, the execution may take a long time to complete. After the execution is completed, the Status icon will change to a check mark and the Execute icon for the next step is enabled.

8. Proceed to the next step, as described in "Making the System Change" on page 12-16.

Making the System Change

After computing the pre-change SQL performance data, perform the system change on the test system. This step is not included in the Guided Workflow page because you must perform it manually. Depending on the type of change, it may be necessary to reconfigure the environment on the test system to match the new environment for which you want to perform SQL performance analysis, as described in "Setting Up the Database Environment on the Test System" on page 12-4.

SQL Performance Analyzer can analyze the SQL performance impact of any type of system change. For example, you may want to test an application upgrade that involves changes such as database table redesign, adding or removing indexes, and so on. For examples of different types of system changes that can be analyzed by SQL Performance Analyzer, see "SQL Performance Analyzer Usage" on page 12-1.

Tip: Before making the system change, build the pre-change version of performance data, as described in "Collecting SQL Performance Data Before the Change" on page 12-14.

To make the system change:

- 1. Make the necessary changes to the test system.
- **2.** Proceed to the next step, as described in "Collecting SQL Performance Data After the Change" on page 12-16.

Collecting SQL Performance Data After the Change

After you have made the system change, build the post-change version of performance data by executing the SQL workload again. For more information about executing a workload, see "Executing a SQL Workload" on page 12-5.

Tip: Before you can build the post-change version of performance data, make the system change, as described in "Making the System Change" on page 12-16.

To collect SQL performance after the change:

1. On the Guided Workflow page, click the **Execute** icon for the Replay SQL Tuning Set in Changed Environment step.

The Create Replay Trial page appears.

- 2. In the **Replay Trial Name** field, enter the name of the execution.
- 3. In the **Replay Trial Description** field, enter a description of the execution.
- **4.** In the Per-SQL Time Limit list, determine the time limit for SQL execution during the replay. Do one of the following:
 - Select UNLIMITED.

The execution will run each of the SQL statements in the selected SQL Tuning Set to completion and gather performance data. Collecting execution statistics provides greater accuracy in the performance analysis but take a longer time to run.

Select EXPLAIN ONLY.

The task will generate execution plans only. This option shortens the execution time but the performance analysis may not be as accurate.

- Select Customize and then enter the specified number of minutes.
- **5.** Ensure that the database environment on the test system is set up to match the production environment as closely as possible, and select **Trial environment established**.
- **6.** In the Schedule section, complete the following steps:
 - **a.** Enter your time zone code in the **Time Zone** field or click the search icon to locate the code for your area.
 - **b.** Select **Immediately** to start the task now, or **Later** to schedule the task to start at a time specified using the **Date** and **Time** fields.
- 7. Click OK.

The Guided Workflow page appears when the execution begins.

The status icon of this step changes to an arrow icon while the execution is in progress. To refresh the status icon, click **Refresh**. Depending on the options selected and the size of the SQL workload, the execution may take a long time to complete. After the execution is completed, the Status icon will change to a check mark and the Execute icon for the next step is enabled.

8. Proceed to the next step, as described in "Comparing SQL Performance Before and After the Change" on page 12-17.

Comparing SQL Performance Before and After the Change

After the post-change SQL performance data is built, compare the pre-change version of performance data to the post-change version by running a comparison analysis.

Tip: Before you can compare the pre-change version of performance data with the post-change version, build the post-change version of performance data, as described in "Collecting SQL Performance Data After the Change" on page 12-16.

To analyze SQL performance before and after the change:

On the Guided Workflow page, click the Execute icon for Compare Step 2 and Step 3.

The Run Replay Trial Comparison page appears.

Run Replay Trial Comparison	
	(Cancel) (Submit)
Task Name SYS.PERF_ANALYZER_TASK	Compare trials to assess change impact
SQL Tuning Set DBA1.CURSOR_CACHE_MAR2907	SQL Performance Analyzer trial comparison allows you to assess the impact on SQL Tuning Set performance of changes made between two trials.
Trial 1 Name SQL_REPLAY_1175211780874	
Description	It is important to know the difference between Trial 1 and Trial 2 execution environments in order to properly assign impacts to the
SQL Executed Yes	changes between trials. Tracking environmental changes between trials is currently a user responsibility.
Trial 2 Name SQL_REPLAY_1175217780829 💌	The selected comparison metric is used as the basis for comparison,
Description	and defaults to execute elapsed time when both trials contain test
SQL Executed Yes	execution statistics. When execution statistics are not available, a less accurate comparison can be made using optimizer cost.
Comparison Metric Elapsed Time	
Schedule	
Time Zone America/Los_Angeles	
©	
Immediately	
C Later	
Date Mar 29, 2007 (example: Mar 29, 2007)	
Time 5 • 40 • 00 • C AM	
• PM	

In this example, the SQL_REPLAY_1175211780874 and SQL_REPLAY_1175217780829 trials are selected for comparison.

2. To compare trials other than those listed by default, select the desired trials in the **Trial 1 Name** and **Trial 2 Name** lists.

Note that you cannot compare a statistical trial with a trial that tests the explain plan only.

3. In the Comparison Metric list, select the comparison metric to use for the comparison analysis.

The types of comparison metrics you can use include:

- Elapsed Time
- CPU Time
- Buffer Gets
- Disk Reads
- Direct Writes
- Optimizer Cost

To perform the comparison analysis by using more than one comparison metric, perform separate comparison analyses by repeating this procedure with different metrics.

- 4. In the Schedule section, complete the following steps:
 - **a.** Enter your time zone code in the **Time Zone** field or click the search icon to locate the code for your area.
 - **b.** Select **Immediately** to start the task now, or **Later** to schedule the task to start at a time specified using the **Date** and **Time** fields.
- 5. Click Submit.

The Guided Workflow page appears when the comparison analysis begins.

The status icon of this step changes to an arrow icon while the comparison analysis is in progress. To refresh the status icon, click **Refresh**. Depending on the amount of performance data collected from the pre-change and post-change executions, the comparison analysis may take a long time to complete. After the comparison analysis is completed, the Status icon changes to a check mark.

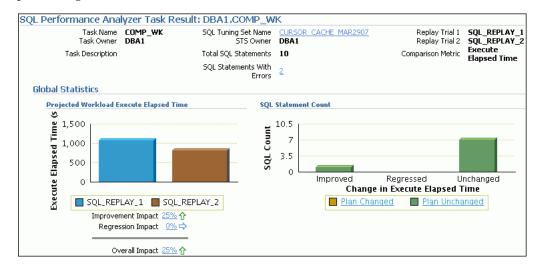
6. Click the Execute icon for View Trial Comparison Result.

The SQL Performance Analyzer Task Result page appears.

7. Review the results of the analysis, as described in "Reviewing the SQL Performance Analyzer Report" on page 12-19.

Reviewing the SQL Performance Analyzer Report

When a SQL Performance Analyzer task is completed, the resulting data is generated into a report. This section shows a sample of a SQL Performance Analyzer report. This sample report uses the elapsed time comparison metric to compare the pre-change and post-change executions of a SQL workload.



To review the SQL Performance Analyzer report:

- Review the general information about the performance analysis, as described in "Reviewing the SQL Performance Analyzer Report: General Information" on page 12-19.
- **2.** Review general statistics, as described in "Reviewing the SQL Performance Analyzer Report: Global Statistics" on page 12-20.
- **3.** Optionally, review the detailed statistics, as described in "Reviewing the SQL Performance Analyzer Report: Global Statistics Details" on page 12-21.

Reviewing the SQL Performance Analyzer Report: General Information

The General Information section contains basic information and metadata about the workload comparison performed by SQL Performance Analyzer.

Task Name	COMP_WK	SQL Tuning Set Name	CURSOR CACHE I	MAR2907	Replay Trial 1	SQL_REPLAY_1
Task Owner	DBA1	STS Owner	DBA1		Replay Trial 2	SQL_REPLAY_2
Task Description		Total SQL Statements	10		Comparison Metric	Execute Elapsed Time
		SQL Statements With Errors	2			

To review general information:

1. On the SQL Performance Analyzer Task Result page, review the information at the top of the page.

This summary at the top of the page includes the following information:

- The name, owner, and description of the SQL Performance Analyzer task
- The name and owner of the SQL Tuning Set
- The total number of SQL statements in the tuning set and the number of failing statements
- The names of the replay trials and the comparison metric used
- 2. Optionally, click the link next to SQL Tuning Set Name.

The SQL Tuning Set page appears.

This page contains information the SQL ID, SQL text, and related information about every SQL statement in the set.

3. Click the link next to SQL Statements With Errors if errors were found.

The SQL Performance Analyzer Task Result page appears.

The Errors table reports all errors that occurred while executing a given SQL workload. An error may be reported at the SQL Tuning Set level if it is common to all statements executions in the SQL Tuning Set, or at the execution level if it is specific to a SQL statement or execution plan.

4. Review general statistics, as described in "Reviewing the SQL Performance Analyzer Report: Global Statistics" on page 12-20.

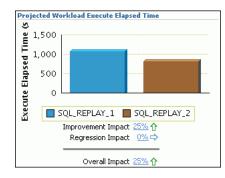
Reviewing the SQL Performance Analyzer Report: Global Statistics

The Global Statistics section reports statistics that describe the overall performance of the entire SQL workload. This section is a very important part of the SQL Performance Analyzer analysis because it reports on the impact of the system change on the overall performance of the SQL workload. Use the information in this section to understand the tendency of the workload performance, and determine how the workload performance will be affected by the system change.

To review the global statistics:

1. Review the chart in the Projected Workload Execute Elapsed Time subsection.

The chart shows the two replay executions on the x-axis and the execute elapsed time (in seconds) on the y-axis.

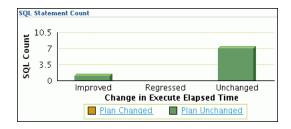


The most important statistic is the overall impact, which is given as a percentage. The overall impact is the difference between the improvement impact and the regression impact. You can click the link for any impact statistic to obtain more details, as described in "Reviewing the SQL Performance Analyzer Report: Global Statistics Details" on page 12-21.

In this example, the improvement impact is 25%, while the regression impact is 0%, so the overall impact of the system change is an improvement of 25%.

2. Review the chart in the SQL Statement Count subsection.

The x-axis of the chart shows the number of SQL statements that are improved, regressed, and unchanged after the system change. The y-axis shows the number of SQL statements. The chart also indicates whether the explain plan was changed or unchanged for the SQL statements.



This chart enables you to quickly weigh the relative performance of the SQL statements. You can click any bar in the chart to obtain more details, as described in "Reviewing the SQL Performance Analyzer Report: Global Statistics Details" on page 12-21.

In this example, all SQL statements were either improved or unchanged after the system change. Most statements were unchanged.

Reviewing the SQL Performance Analyzer Report: Global Statistics Details

You can use the SQL Performance Analyzer Report to obtain detailed statistics for the SQL workload comparison. The details chart enables you to drill down into the performance of SQL statements that appears in the Result Summary section of the report. Use the information in this section to investigate why the performance of a particular SQL statement regressed.

Im	Improved SQL Statements										
		Net Impact	Execute Elapsed Time		Net Impact	an from 1.1 - 1					
	SQL ID	on Workload (%)		SQL_REPLAY_2	on SQL (%)		SQL_REPLAY_2	Plan Changed			
ᡎ	<u>31h2wmu3q47u6</u>	25.370	0.319	0.238	25.390	99.900	99.730	N			

To review the global statistics details:

1. Click the bar in any chart on the SQL Performance Analyzer Task Result page, or click the impact percentages in the Projected Workload Execute Elapsed Time subsection.

A table including the detailed statistics appears. Depending on the table, the following columns are included:

SQL ID

This column indicates the ID of the SQL statement.

Executions

This column indicates the number of times this SQL statement was executed.

Net Impact on Workload (%)

This column indicates the impact of the system change relative to the performance of the SQL workload.

Execute Elapsed Time

This column indicates the total time (in seconds) of the SQL statement execution.

• Net Impact on SQL (%)

This column indicates the local impact of the change on the performance of a particular SQL statement.

% of Workload

This column indicates the percentage of the total workload consumed by this SQL statement.

Plan Changed

This column indicates whether the SQL execution plan changed.

2. Click SQL ID for any SQL statement in the table.

The SQL Details page appears.

You can use this page to access the text of the SQL statement and obtain low-level details such as CPU time, buffer gets, and optimizer cost.

Index

A

actions about, 4-7 Active Session History about, 7-1 report about, 7-2 activity over time, 7-7 load profile, 7-4 running, 7-2 top events, 7-3 Top SQL, 7-5 using, 7-3 sampled data, 7-1 statistics, 2-4 alerts clearing, 5-3 default, 5-1 performance, 5-1 purging, 5-3 responding to, 5-2 Automatic Database Diagnostic Monitor about, 3-1 accessing results, 6-4 analysis, 3-2 configuring, 3-3 DB time, 3-2 enabling, 2-5 findings about, 3-8 viewing, 3-7 for Oracle RAC, 3-3 identifying high-load SQL, 9-1 recommendations actions, 3-9 implementing, 3-9 interpreting, 3-8 rationales, 3-9 types, 3-2 report, 3-8 reviewing results, 3-7 running manually analyzing current database performance, 6-1 analyzing historical database performance, 6-3 Automatic SQL Tuning

modifying task attributes, 10-7 viewing recommendations, 10-7 viewing results, 10-5 Automatic Workload Repository about, 2-1 baselines, 8-1 compare periods report about, 8-1 details, 8-18 saving, 8-12, 8-15 summary, 8-16 supplemental information, 8-18 using, 8-15 using another baseline, 8-10 using snapshot pairs, 8-13 configuring, 2-2 enabling, 2-2, 2-5 snapshots, 2-2 statistics collected, 2-2 using, 3-4

В

baselines about, 8-1 baseline template about, 8-2, 8-5 comparing, 8-10 computing threshold statistics for, 8-6 creating single, 8-2 deleting, 8-5

С

clients about, 4-8 CONTROL_MANAGEMENT_PACK_ACCESS parameter and ADDM, 3-3 CPU I/O wait, 4-20 load, 4-21 performance problems, 4-21 utilization about, 4-18 monitoring, 4-19 customizing the Performance page, 4-27

D

data access paths, optimizing, 11-1 database statistics, 2-1 time, 2-2, 3-2, 3-8 database performance alerts, 5-1 automatic monitoring, 3-1 comparing, 8-1 current analysis, 6-1 degradation over time, 8-1 historical analysis, 6-3 manual monitoring, 6-1 overview, 2-1 Database Resource Manager using, 4-22 database tuning performance degradation over time, 8-1 preparing the database, 2-5 proactive tuning, 2-6 reactive tuning, 2-7 real-time performance problems, 4-1 SQL tuning, 2-7 tools, 1-2 transient performance problems, 7-1 using the Performance page, 4-1 database upgrade simulating with SQL Performance Analyzer, 12-7 DB time about, 2-2 and ADDM, 3-2 and ADDM finding, 3-8 DBIO_EXPECTED parameter about, 3-4 setting, 3-3, 3-4 DBMS ADVISOR package configuring ADDM, 3-4 setting DBIO_EXPECTED, 3-4 disk performance problems, 4-27 utilization about, 4-18 monitoring, 4-25

Ε

execution plan about, 10-1 viewing for a SQL statement, 9-7

Η

high-load SQL about, 9-1 identifying using ADDM, 9-1 identifying using Top SQL, 9-2 identifying using Top SQL by wait class, 9-3 statistics, 2-4 tuning, 10-2, 10-5 viewing details, 9-4 viewing execution plans, 9-7 viewing session activity, 9-6 viewing SQL profiles, 9-8 viewing SQL text, 9-4 viewing SQL Tuning Advisor tasks, 9-8 viewing statistics, 9-5 viewing tuning information, 9-8 host activity, monitoring, 4-17

I

index about, 11-1 bitmap, 11-1 B-tree, 11-1 creating, 11-2 functional, 11-1 indexes creating, 2-7 instance activity monitoring J/O wait times, 4-12 monitoring parallel execution, 4-16 monitoring services, 4-16 monitoring throughput, 4-11 I/O wait times, monitoring, 4-12

Μ

materialized view logs about, 11-1 creating, 2-7, 11-2 materialized views creating, 2-7, 11-2 memory performance problems, 4-24 swap utilization, 4-24 utilization about, 4-18 monitoring, 4-22 metrics, 5-1, 8-7 modules, 4-7

Ν

new features, list of, xi

0

optimizer, use of SQL profiles in, 10-19 Oracle performance method about, 2-1 pretuning tasks, 2-5 proactive database tuning tasks, 2-6 reactive database tuning tasks, 2-7 SQL tuning tasks, 2-7 using, 2-5 Ρ

parallel execution, monitoring, 4-16 parameters DBIO_EXPECTED, 3-4 initialization, 8-18 STATISTICS_LEVEL, 2-2, 2-5 Performance page customization, 4-27 performance problems common, 2-8 CPU, 4-21 diagnosing, 3-1 disk, 4-27 memory, 4-24 real-time, 4-1 transient, 7-1

S

services about, 4-6, 4-16 monitoring, 4-6, 4-16 sessions, monitoring, 4-5 snapshots about, 2-2 comparing, 8-13 creating, 3-5 default interval, 3-4 filtering, 8-13 modifying settings, 3-5 viewing statistics, 3-12 SOL Access Advisor about, 10-1, 11-1 filters, 11-5 initial options, 11-2 recommendations about, 11-1 details, 11-15 implementing, 11-20 options, 11-7 reviewing, 11-13 SQL, 11-18 summary, 11-14 running, 11-2 scheduling, 11-10 task options, 11-19 workload options, 11-19 workload source, 11-3 SQL Performance Analyzer about, 12-1 database environment setting up, 12-4 following guided workflow, 12-12 initial environment establishing, 12-14 methodology, 12-2 performance data collecting post-change version, 12-16 collecting pre-change version, 12-14 comparing, 12-17 running, 12-5

simulating database upgrade, 12-7 SQL Performance Analyzer report general information, 12-19 global statistics, 12-20 global statistics details, 12-21 reviewing, 12-19 SQL Tuning Set selecting, 12-13 SQL workload capturing, 12-3 executing, 12-5 transporting, 12-4 system change making, 12-16 testing parameter change, 12-10 usage, 12-1 SQL profiles about, 10-19 deleting, 10-19 disabling, 10-19 enabling, 10-19 viewing information, 9-8 SQL Tuning Advisor about, 10-1 automated maintenance tasks, 10-5 comprehensive scope, 10-3 implementing recommendations, 10-5 limited scope, 10-3 using, 10-2, 10-5 viewing information, 9-8 viewing results, 10-4 SQL Tuning Sets about, 10-8, 12-5 creating, 10-8 load method, 10-10 transporting, 10-8 statistics Active Session History, 2-4 baselines, 8-1 databases, 2-1 DB time, 2-2, 3-2, 3-8 default retention, 3-4 gathered by the Automatic Workload Repository, 2-2 gathering, 2-1 high-load SQL, 2-4, 9-5, 9-6 sampled data, 7-1 session, 2-4 system, 2-4 time model, 2-2 wait events, 2-3 STATISTICS_LEVEL parameter and ADDM, 3-3 and the Automatic Workload Repository, 2-2 setting, 2-5

Т

throughput, monitoring, 4-11 time model statistics

about, 2-2 Top Actions user activity, 4-7 top activity top SQL, 9-2, 9-4 Top Clients user activity, 4-8 Top Files user activity, 4-9 Top Modules user activity, 4-7 Top Objects user activity, 4-10 Top PL/SQL user activity, 4-9 **Top Services** user activity, 4-6 Top SQL Active Session History report, 7-5 by wait class, 9-3 identifying high-load SQL, 9-2 user activity, 4-4 Top Working Sessions user activity, 4-5

U

user activity Top Actions, 4-7 Top Clients, 4-8 Top Files, 4-9 Top Modules, 4-7 Top Objects, 4-10 Top PL/SQL, 4-9 Top Services, 4-6 top services, 4-6 Top SQL, 4-4 Top Working Sessions, 4-5

W

wait class about, 4-2 legend, 4-2 viewing high-load SQL by, 9-3 wait events statistics, 2-3