Performance Analysis Overview

Performance Analysis offers a comprehensive suite for diagnosing and tuning Oracle instance and application workloads. Through advanced diagnostic features and specially-designed interfaces, Performance Analysis transforms Oracle performance management into a faster, easier and more efficient experience.

Performance Analysis empowers the Oracle DBA to do his job more efficiently and increases his scalability by:

- Providing at-a-glance <u>root-cause analysis</u> through a performance management digital dashboard.
- Applies automated expertise through "touchless" baseline behavior processing, performance advisories, and industry best practices, all providing uncomplicated yet intelligent alerts and tuning recommendations, to enhance and accelerate the diagnosis and tuning process.
- Exposing bottlenecks, anomalies, and application trends rapidly through sophisticated historical analysis. The analysis, supported by a unique graphical interface, reduces data sprawl and adds intelligence to facilitate quick and confident responses.
- Identifying top resource consumers through a powerful drill down to the specific resource intensive dimensions (SQL statements, users, programs, machines, and so forth) and files.
- Consolidating all mission-critical performance data in highlycustomizable reports.

The Performance Analysis application is built on StealthCollect <u>agent</u> technology. This resource-light, high sampling rate technology binds anonymous database activity to tangible users, programs and Oracle resource consumers (<u>dimensions</u>) in an intuitive, user-friendly interface.

For sites using RAC systems, Performance Analysis simplifies cluster performance management by consolidating the activity originating in all individual cluster nodes into a single logical cluster view. In this encapsulation of the physical structure, Oracle's grid concept is fully maximized and articulated. Performance Analysis gives you the ability to:

- monitor and optimize current Oracle database performance at a glance
- gather and diagnose business views
- identify and anticipate performance issues

Click <u>here</u> for a product demonstration.

Related Topics Measuring System Health Performance Advisories Best Practices Analysis

Opening Performance Analysis

Performance Analysis consists of client and <u>agent</u> components, supporting a variety of agent platforms and versions. The Performance Analysis Windows-based client installation file is available both as a <u>download</u> and on the installation CD.

To open the Performance Analysis client (GUI)

• Click the active instance node in Quest Database Management Solutions or

• Through the Performance Analysis Profiles dialog, access can be gained from either:

- Clicking the **Performance Analysis** () icon on the desktop, or
 - Navigating to start | Programs | Quest Software | Performance Analysis | Performance Analysis

NOTE: Installation of both the client and the agent components must be completed before Performance Analysis can be used. As a prerequisite to successfully using Performance Analysis ensure the monitored Oracle host is:

- Visible to the client (GUI) on the network
- Running and accessible through SQL*Net from the client

Managing Connections and Logging On

Performance Analysis maintains monitored Oracle instance specifics as logon profiles. The **All Connections** dialog can be used to create new connections (using the New Connections icon) and to manage existing profiles (by right-clicking an existing profile).

Supply the following parameters to create a new logon profile: For a new connection, supply the following parameters:

Parameter	Description

Database	TNS entry alias of the monitored instance
SID	The monitored instance identifier also referred to as the instance's name. If the TNS entry was not specified with the SID qualifier, you will have to manually specify the instance's SID.
Agent Port	The communication port number used to communicate with the Performance Analysis agent on the Oracle server host (on which the monitored instance resides).
Use default agent authentication	Select this option (default) when default agent authentication is used. If the checkbox is cleared, you must provide valid alternative Performance Analysis agent authentication details. Leaving this option selected causes Performance Analysis to attempt to connect via the default agent user <i>quest</i> created during agent installation (does not require agent logon information). This option will function as long as user <i>quest</i> 's password has not been modified. Note: Quest recommends that this password be changed upon agent installation.
User Name	Enter the username employed for connecting to the agent.
Password	Enter the password employed for connecting to the agent. Note : View and modify the agent user's logon details through the <u>Agent User's</u>

	Manager dialog.
Save password details (for	(Available only when non-default StealthCollect agent authentication details are used)
this connection)	If this checkbox is cleared, user connection details must be provided upon each connection to the agent.

Click here for more information on managing connections.

Options

Click **Tools** | **Options** to display the connection configuration options.

- The Always prompt for Username and Password checkbox requires users to enter the appropriate username and password each time the client connects to the specified Oracle instance. To allow users to bypass this step, uncheck this box.
- The **Save password default** checkbox sets the StealthCollect agent's logon password to the default value and allows users to monitor the specified Oracle instance without specifying a username and password.
- The **Connection Profile Directory** field allows users to specify an alternate to the default directory for the **Profile.xml** files, which define the collection configuration parameters; the default profile directory is:

C:\Documents and Settings\All Users\Application Data\Quest Software\Shared\Profiles\<Instance>

RAC Configuration (if present)

The Performance Analysis GUI consolidates the activity originating in all individual cluster nodes into a single logical cluster view. No modifications to the Performance Analysis agent (StealthCollect) installation are required for individual RAC nodes.

To configure Performance Analysis for RAC

 Install the Performance Analysis agent (StealthCollect) on each node participating in the RAC. Ensure that identical configuration settings (including those for the performance repository if created) are specified during each installation.
 Note: The Performance Analysis agent should not be installed on

the shared disk.

- 2. Launch Performance Analysis from a RAC TNS Entry in the displayed **Database** field in the **All Connections | New Properties** dialog.
- 3. Upon initial connection, specify the appropriate instance name [SID] for each node participating in the RAC in the displayed **TNSNames Additional Information** dialog.
- 4. Click **OK** to complete the configuration and to connect to the Performance Analysis RAC view.

Caution: Performance Analysis will only recognize nodes explicitly listed in the RAC TNSNames entry.

Related Topics

Support for Real Application Clusters

The Console

The Performance Analysis window features four modes:



- <u>Home Mode</u>—Offers at-a-glance monitoring of primary <u>system</u> <u>health measurement</u>s, and provides the starting point for drilldown analysis in one easy-to-understand digital dashboard. This view offers user-configurable, alarm-enabled, normative baseline, advisory and applied knowledgebase toolsets to optimize user monitoring and tuning efforts; Performance Analysis's Home mode intuitively focuses users' attention on those instance and system aspects most in need of attention. The Home view is divided into four distinct tabs:
- <u>Overview</u> tab: Displays overall instance and system activity in a set of intuitive gauges; the top set delineates instance activity for baseline analysis, and displays individual resource performance as compared to established thresholds, while the bottom set depicts both instance-level resource breakdowns compared to established thresholds, and system component performance indicators.
- <u>Performance Advisories</u> tab: Identifies and describes significant performance issues that affect instance and/or database operation. One click accesses both Quest's context-sensitive knowledgebase and suggests intelligent solutions to the problem(s).
- <u>Best Practices</u> tab: Analyzes and identifies database configuration and operating conditions that are non-optimal or that violate industry "best practices" for specific Oracle versions.
- <u>E-Business</u> tab: Provides analytical reports on key Oracle E-Business operational areas, including application performance and security characteristics.
- <u>Real-Time Mode</u>— Offer both real-time and recent (up to the last 60 minutes) monitoring of both instance and session level activity. Current activity can be tracked down to the statement currently executing by double-clicking any active session displayed in the <u>Current Sessions</u> node.

- <u>History Mode</u>— Facilitates problem detection, diagnosis, and resolution by means of <u>baseline</u> and <u>dimensional</u> analysis. Historical performance data can be analyzed down to increasingly refined levels through the ilnstance <u>overview</u>, and users can create and maintain custom <u>shortcuts</u> (cube views) and <u>user defined collections</u>.
- <u>Reports Mode</u>—Assists the DBA/Manager in automating key tasks and in understanding system behavior with a variety of reports. Reports can be created to span all dimensions and filters in a variety of output formats for further inquiry and analysis.
- A second row of icons, consists of two sets:
- individual icons for:

Agent Refresh (2), Client Configuration (1) agent configuration (2) Help (?);

- context-sensitive toolbar icons for the various modes:
- Home: 🖶 🖾 <u>3/19/2006 11:55 AM 3/19/2006 12:55 PM</u>

and Real-time and History: $\bigcirc \bigcirc \boxminus \checkmark \textcircled{} \bullet \boxtimes \bullet \bullet \bullet$

Related Topics

Opening Performance Analysis Common Elements in the Drilldown Console Common Elements in the Home Page View

Performance Baseline

Database administrators face the ongoing challenge of effectively planning for the future of their database systems. Performance Analysis's adaptive IntelliProfile engine dynamically analyzes time-sensitive operating metrics to produce a baseline (expected) operating range for the instance. A form of quantitative analysis, this process looks at historical performance data to benchmark current performance, and, moreover, to estimate future performance. Good quantitative analysis requires as much data as is available, and IntelliProfile incorporates data on hardware and software configurations, concurrent users, workload size, instance availability, as well as other day-to-day operating specifics.

The baseline algorithm parses sampled data into pre-defined time units (e.g. day of the week) and distinct episodes (e.g. systematically recurring spikes in CPU consumption). These episodes form the basis for projecting normative instance behavior, allowing for both exception monitoring and performance diagnostics. These dynamically-created statistical ranges help administrators configure high and low metric threshold values. The IntelliProfile engine learns the instance's "signature" behavior characteristics via sophisticated algorithms that identify both cyclical and periodic behavior to minimize and/or neutralize random noise distortion (for example, an employee absent due to illness). The engine is adaptive, i.e. it grows "smarter" as the sampling period increases so acceptable/normal activity range projections are more accurate for each point in time.

This time-sensitive baseline analysis:

- is resource-light and runs in the middle tier
- features configurable color alerts
- supports full <u>compare</u> capabilities
- features intelligent calibration of charts and bars display

The baseline perspectives are always relative to the latest norms, so if the instance changes, the baseline is re-applied. Thus, behavior which in its original context was reported as within a normative range can subsequently, and even retrospectively, be deemed exceptional. **Note**: A performance baseline is not a definition of overall system performance; rather a realistic workload baseline gives valuable insight into expected real performance. This insight then allows database and administrative staff to effectively anticipate and plan for future needs.

Use case scenarios — identifying abnormal behavior

A key business problem solved by IntelliProfile is that of identifying, diagnosing and characterizing abnormal behavior. Almost all performance indicators are "relative", but administrators must specify behavioral ranges specific to each of their instances. Are peaks presently encountered in fact exceptional? How exceptional are they? Do they reflect desired or undesired system responses? Are they one-time or cyclical? The baseline-generated exceptions trigger advisory alerts which help identify and correct abnormal instance behavior, and explain why exceptions occurred (what were the underlying causes).

See <u>Tuning Scenarios</u> for possibilities that lend themselves to baseline analysis.

Display and usage within Performance Analysis

The baseline graphic can be toggled on three different timeline views: on both the Home and History modes' Overview tabs in the instance workload view, and within the History mode's overall Resource <u>time-line</u> instance view.

Graphical interface:

The baseline range is displayed as a light blue band in the background of the <u>Home</u> mode > Overview tab's **Instance Workload** graph and in the <u>History</u> mode | Average Active Sessions timelines, the baseline range is displayed as a background light blue band.

The baseline ranges are depicted as a range-marking vertical element

on the Overview Workload, Resource Breakdown and System Utilization bar graphs..

The Overview, Workload, Resource Breakdown and System Utilization bar graphs feature <u>configurable color alerts</u> for behavior exceeding specified thresholds, which also offer a hyperlink starting point for further analysis.

Related Topics

<u>Agent Options - IntelliProfile (Baseline)</u> <u>Display Options - Baseline</u>

New Features

Performance Analysis features a new highly-configurable digital dashboard consolidating key performance indicators and providing at-aglance understanding of instance health and root-cause problem detection; the dashboard is comprised of:

Baselines	Performance Analysis's adaptive IntelliProfile engine dynamically analyzes time- sensitive operating metrics to produce an expected baseline operating range for the instance.
<u>Home Page</u>	PA's default view, the home page offers fully- configurable, prioritized performance alerts, current and baseline-normalized system resource utilization indicators, key Oracle performance advisories, and industry best practice information presented side- by-side in a clean, intuitive format.
<u>Performance</u> <u>Advisories</u>	Groupings of key Oracle tuning checks, organized and presented as both individual alerts and a coherent, step-by-step action plan to maximize database performance. These advisories can be seamlessly integrated with Knowledge Expert.

<u>Best</u> <u>Practices</u>	These advisories identify, by category, non-optimal processes or configuration elements that violate industry "best practices" for specific Oracle versions.
<u>Oracle E-</u> Business	This analysis checks key Oracle E-Business areas for performance-degrading issues.
Executive Summary Report	The Executive Summary is a one-click report- generating feature designed to deliver to key decision- makers critical instance and system performance parameters in a useful configurable format to key decision-makers.
Historical Execution plan	Compares statement execution plans created by Oracle's query optimizer.
Dimension filters	Supports specifying complex filter expressions to intelligently exclude or include dimensions of interest. For example, filter in all the SQL statements starting with "INSERT" and excluding tables named "DUAL".
Dynamic link from metrics to help file	Hyperlinks now offer one- click connectivity between displayed metrics and their explanations.

Kill Session	Terminate a specific session directly from Performance Analysis' real-time interface (via in-context launching of TOAD).
Improved flow of agent user management	Performance Analysis now provides a utility to create and manage StealthCollect agent users from one option in the Support Tools dialog. Multiple users can now be created to facilitate both administrative and end-user access to performance data, and their passwords and permissions within Performance Analysis can be maintained quickly and easily.
Licensing options	Users can now license the following features individually: Performance Advisories, Oracle E- Business/Peoplesoft, and EMC Symmetrix.
Integration with Spotlight	Issues requiring the identification and diagnostic capabilities of Spotlight on Oracle can now be addressed with a single- click link.
Integration with Knowledge Expert	Quest's comprehensive Windows-based technical resource for Oracle administration and PL/SQL programming is now

	available with a single-click link.
Scalability tests (integration with Benchmark Factory)	Send groups of SQL statements directly to Benchmark Factory to create and execute scalability tests.
Standalone Product	The new standalone Performance Analysis now offers a comprehensive suite for diagnosing and tuning the Oracle instance and application workload. The separately purchased and installed toolset unleashes the full potential and power of Performance Analysis for Oracle; its advanced diagnostic features and specially- designed interfaces transform Oracle performance management into a faster, easier and more efficient experience.

Common Elements in the Console

The Performance Analysis two-pane display features a tree-node drilldown in the left-pane determining the display in the right-pane. The Performance Analysis window contains some or all the following visual elements:

- <u>Activity Timeline</u>
- Instance View
- Action Menu
- <u>Metric Totalings</u>
- Overview Tab
- Datafiles I/O Tab
- <u>Resource</u>
- <u>Resource Breakdown</u>
- <u>Time</u>
- <u>Toolbar</u>
- Display Features
- <u>Dimension Filtering</u>

Related Topics

The Scopus Console

Activity Timeline

The activity timeline is the principal display of Oracle activity over a specific time period. Its vertical calibration is based on an average of recorded baseline peaks of observed behavior. The dynamic representation displays the resource consumption in:

• a Resource View layered manner such that bottleneck occurrences can be readily identified as to their underlying <u>wait event category</u> causes and isolated for analysis.

-OR-

• a Baseline Range View manner against the upper and lower boundaries of the <u>baseline</u> range statistical deviation.

It is the principal Performance Analysis tool for monitoring Oracle instance activity, both current and historical. Use this view to look for trends in your database activity, to identify resource consumption, to trace activity, and to review previous activity.

Related Topics

<u>Time Navigation</u> <u>Understanding Activity Significance</u> <u>Dimension Navigation</u>

Instance View

The instance view is the default drilldown means of monitoring the entire real-time Oracle instance activity. The window offers both a ready and revealing overview and the ability to focus on critical resource usage as reflected in key <u>metric</u> values. Use the instance view to identify trends, bottlenecks, and anomalous resource usage.

The Performance Analysis instance view is a multi-faceted snapshot of Oracle activity. The Activity Timeline Drilldown and <u>Resource Breakdown</u> are the initial visual insights into system performance.

For RAC configurations, this **Cluster View** displays the composing instances.

Action Menu

The following table lists the commands on the **Action** menu and their uses. (The commands displayed are determined by context and cursor focus.)

Command	Use
Open in New Window	Open the New Window creation dialog wizard.
Refresh View List	Refreshes the <u>Business views</u> node for both this client and for system- wide changes.
New <u>Shortcut</u>	Create the mechanism to analyze the performance and resource consumption of specific dimensions of interest (over differing domains and filters across different time periods and granularity), both episodic and non-episodic.
New <u>User</u> <u>Defined</u> <u>Collection</u>	Create the mechanism by which Oracle resource consumption can be analyzed over selected domains and filters across different time periods.
Execute/Edit	Display the Summary window of the in-focus report.
Report	Open the Report Wizard dialog specification series.
Schedule	Opens the report <u>scheduling</u> properties dialog.
Remove Schedule	Deletes the specific report schedule currently active.
Preview	Generate window display of report.

Print	Print the in-focus saved report.		
Rename	Rename an existing customized template or history view.		
Delete	Delete an existing customized template or history view.		
Compare	An analytic baseline feature that facilitates graphical resource comparison by dimensions between two historical time frames on a relative or absolute basis. Each item activity is broken down into its components and compared with the second time frame. See <u>Compare</u> .		
Save Shortcut	Record the specified customized history view to be archived and retrievable.		
Roll Up	When the History instance view is at a <u>dimension</u> node depth level of four or greater, this function shifts the focus to the top level of this dimension.		
Show Top 25 Only	Limit the display to the 25 most resource-demanding domains (occurrences) of a specific dimension.		
View SQL	Display the statement syntax in [Quest] SQL View.		
Tune SQL	Open the SQL statement for SQL tuning (if installed).		
Explain Plan History	Compare any two forms of execution plan used historically by Oracle for executing a given SQL statement.		
	Opens all the SQL statements in the		

Index Analysis	node in SQL Tuning to be subjected to the SQL Tuning Lab index advisory process (if installed).
Scalability Test	Send groups of SQL statement directly to Benchmark Factory (if installed) to create and execute scalability tests for an automated load testing solution.
Copy SQL to Clipboard	Make the SQL text available for pasting into a Windows application.
Multi-line SQL Text	Expands the SQL display to a multi- line presentation so that the entire SQL is visible.
Summarize SQL Text	Shortens the SQL statements by replacing long SELECT and FROM clauses with ''
Kill Session	Launches TOAD to terminate the Oracle session.
Filter	Limits the dimension pane listing display to only those entries which meet a set of user-specified wait type numeric thresholds and SQL statement text conditions.
Drill Down	Performs a one-level drill down to the focused node member.
Enabled	Restart collection of a paused history view.
Disabled	Pause collection of a history view.

Metric Totalings

For representation of periodic data, Performance Analysis supports a variety of totaling methods. Select from among the possible methods in accordance with your summary needs.

Click this button	То	Functional in mode
	The button determines whether the numeric metric displays are denominated as rates or cumulative summaries.	
	Note that time graph displays always display the metrics as rate (per second), regardless of the button selection. Furthermore, in all windows, the TOP are selected using TOTAL. The fact that you select Rate, or per-interval or per-execution only affects the numbers displayed, not the selection of the records.	history and real-time
	Total - The default display. The metric values are the total over the selected period.Select this option if you are interested in the total period activity.	history and real-time
	Rate (Per Interval) - Rate per window-defined resolution (see time range), allowing for comparison and statistical (average and high values) calculation per interval.	history

	second option values yield values too low for ready comparison.	
	Rate (Per Second) - Metric parameters in the list pane are calculated as an average per second for the selected time frame, allowing for comparison and statistical (average and high values) calculation. All graphical displays are based on this calculation method. Select this option for ease of inter- period comparison.	history and real-time
	Per Execution - Metric parameters in the list pane are calculated as an average per SQL-statement execution. Select this option when interested in the impact of a SQL statement execution.	history
Note: The total selection affects only those values displayed in the metric		

Note: The total selection affects only those values displayed in the metric pane listings. The timeline graph statistics are always calculated and displayed on a per-second basis.

Overview Tab

The **Overview** tab and feature is the primary history and real-time tool for initial understanding of the system resource use patterns that you see in the = 4) BSPSPopupOnMouseOver(event)" CLASS=BSSCPopup id=A1>activity timeline graph. The tool lets you view the performance-oriented metrics, time correlated with any activity drilldown. The middle pane timeline graphical representation and explanation displays respond to the choice of resource focus in the Resource/Metric list pane. ToolTip hints display any additional available information. The display is composed of the resource timeline and expert metrics (in accordance with the selected resource).

The overview tool is a unified view of the entire range of activity projections: full instance, specific session, and any historical activity drilldown (for example, User JOHN executing a program on machine ATLAS).

The pane list metrics displayed in the window's lower third are determined by the chosen = 4) BSPSPopupOnMouseOver(event)" CLASS=BSSCPopup id=A3>Resource. The Resource chosen displays those metrics that have the greatest relevance and impact on that resource as it may affect throughput and response time. In turn, the chosen metric determines the text displayed in the middle third right side. For those metrics that can be represented as a percentage of a total, a horizontal bar presentation presents both its internal component portions (for example, in the *Oracle/System CPU Usage*, CPU consumption is composed of the instance background and foreground CPU and CPU used by all other processes) and their total.

The overview representation and values are determined by:

- <u>Dimension Navigation</u> parameters
- <u>Metric Totalings</u> For ease of use and comparability, differing totaling methods can be applied. Select from among the possible methods in accordance with your summary needs:
- Period **Total** (default) metric aggregation within one column for the entire time range

- Rate (Per Interval) summarizes the data for averages and totals with periods defined by the Data Interval
- Rate (Per Second)
- Per Execution of SQL statements

Note: For methods that entail both averages and totals, the horizontal bar presentation and ToolTip display reflect the column in focus.

Datafile I/O Tab

Use the **Datafile I/O** tab (visible for Workload or I/O resource specification) to display a lower pane listing of the data files, in the current <u>dimension drilldown</u> context, on which the most I/O resource-demanding data activities occurred.

For systems based on EMC storage arrays, right-click on a file of interest to initiate the Storage Analysis for EMC physical layout and I/O analysis for the corresponding data file.

Resource

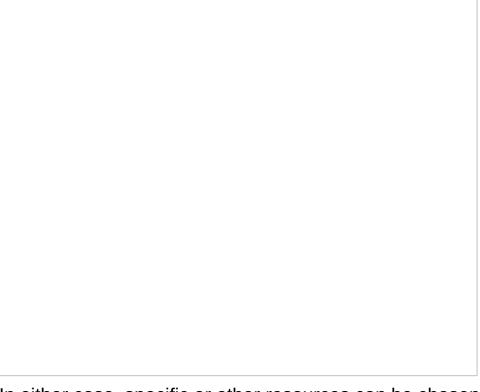
Resource usage is the processing activity measured through the total <u>Oracle Wait Event Categories</u> and denominated in time duration and percentage of the total. All displayed metrics are a reflection of the chosen resource.

Resource Breakdown

The Resource Breakdown pie chart and drill-down dialog pane listing display the relative Oracle wait event usage over the accompanying displayed activity timeline. In this manner, you get an overall period measurement of resource usage, abstracting from peak and null periods.

When you drill down into the resource breakdown pie chart, the focus and display are dependent on your access path.

- Clicking on **Wait Events...** displays the dialog in accordance with the default Resource selection.
- Clicking on a specific slice of the total pie chart (or double-clicking on a resource event category within the entire pane-listed Workload) opens the dialog box focused on the chosen wait event category.



In either case, specific or other resources can be chosen through the Resource drop-down listing from within the **Wait Events...** dialog. The two data columns are defined as follows:

• Percent of Total - The portion of the entire wait event activity (as a

percentage of the entire activity)

 Duration - Depending on the <u>= 4</u>) <u>BSPSPopupOnMouseOver(event)</u>"<u>CLASS=BSSCPopup</u> <u>id=A1>totaling method</u> in effect, this value represents the time (in seconds) consumed by this event per time unit specified.

Several Oracle wait events facilitate further breakdown to their internal composing parameters (those with the icon (\Box) at their left). For example, I/O events can be broken into the actual files on which the I/O operation has been performed.

For the key selected events, the text pane in the lower half of the window displays the actual Oracle wait event parameters of the selected resource. The parameters appear in accordance with the click-on-line focus for those events with an icon to the left, as in the preceding display.

Resource	Event Name	Parameter Breakdown
	direct path read temp	
	direct path write temp	
	write complete waits	
	db file sequential read	
I/O	db file scattered read	File Name
	db file single write	
	direct path read	

	direct path write		
	direct path read (lob)		
	direct path write (lob)		
	buffer busy waits		
	free buffer waits		
Buffer	local write wait	File Name	
	buffer busy global cache		
	buffer busy global CR		
Lock	global cache or disk request	File Name	
	latch activity		
	latch free		
	buffer latch		
Latch	wait list latch activity	Latch Name	
	wait list latch free		
	buffer read retry		
	DFS db file lock		
	global cache		

	open s	
	global cache open x	
	global cache null to s	
	global cache null to x	
	global cache s to x	
	global cache cr request	
	global cache busy	
	global cache retry prepare	
	global cache lock busy	
	global cache lock cleanup	
	global cache lock null to s	
	global cache lock null to x	
	global cache lock open s	
Other	global cache lock open ss	File Name
	global cache lock open x	
	global cache lock s to x	

global cache multiple locks	
global cache retry request	
global cache cr disk request	
global cache current request	
global cache cr multi block	
gc current multi block	
gc buffer busy	
gc current request	
gc cr request	
gc cr disk request	
gc cr multi block request	
gc block recovery request	

Time

Activity in Performance Analysis historical inquiry is recorded and reported in specific units of time. This feature facilitates the focus on data by frames (ranges). Time ranges consist of aggregated Data Intervals. Select and specify time ranges from values in the drop-down listing.

Related Topics

Real-Time: <u>Time Ranges</u> History:<u>Time Navigation</u>

Toolbar

The Performance Analysis toolbar is located at the top of the main window. The toolbar provides you with quick access to commonly used commands and functions. Click a button on the toolbar to carry out the command. To see a description of each button, rest your mouse pointer over the button. Information about the button is displayed in a ToolTip.

The following are common Performance Analysis buttons.

Click this button	То	Functional in mode
	Return to the last view you browsed. (Alt + Left)	history
	Advance to the next view. This command is only available if you are browsing a previous view. (Alt + Right)	history
	Click the button to immediately refresh the entire window with the most recent sampled data.	

The drop- down listing sets the automatic window refresh interval. Select <i>None</i> to disable automatic refresh.	real-time
Configure time of interest and manage the time bookmarks	history
The button determines whether the numeric metric displays are denominated as rates or cumulative summaries. (Time graph displays always display the metrics as rate (per	history and real-time

	second), regardless of the button selection.) For summary type detail, see <u>Metric</u> <u>Totalings</u>	
	Toggle display of <u>change</u> <u>tracking</u> points.	history
	Launch the <u>Compare</u> <u>tool</u>	history and home
<u>a</u>	Print an Executive Summary or Analysis Report of this window	home
	Toggle the present reporting period among the most recent:	
<u>11/13/2005 3:14 PM - 11/13/2005 4:14 PM</u>	 5 minutes 15 minutes 1 hour 	home

 6 hours 12 hours 24 hours 	
Specify the default time frame <u>here</u> .	

Display Features

Unit Display

Display the denomination unit of each columnar display of metrics and wait states (except in the <u>Overview</u> tab) by resting the cursor on the title bar. Units are dependent on the current <u>metric totaling</u> method.

Splitter/Closer

Display or hide graphic elements and legends by clicking the horizontal or vertical split/close buttons:

Dimension Filtering

Performance Analysis offers a filtering feature to limit display in the right pane History mode listing to only those dimensions matching a set of user-defined expression conditions.

When the history mode left tree-node focus is on a dimension of one of its constituting elements, the right lower-third pane listing is headed by the linked: "No filter applied: <u>Click to create...</u>". When clicked, the **Define Filter** dialog is displayed, wherein you define a set of filter expressions to limit display.

The available **Field**s are:

- Currently in-focus dimension node
- Displayed performance metrics

Note: Metrics can be added or deleted by <u>enabling the grids</u> <u>customization</u> and then right-click in the pane listing and selecting **Select Metric...** from the displayed shortcut menu.

The available **Conditions** are:

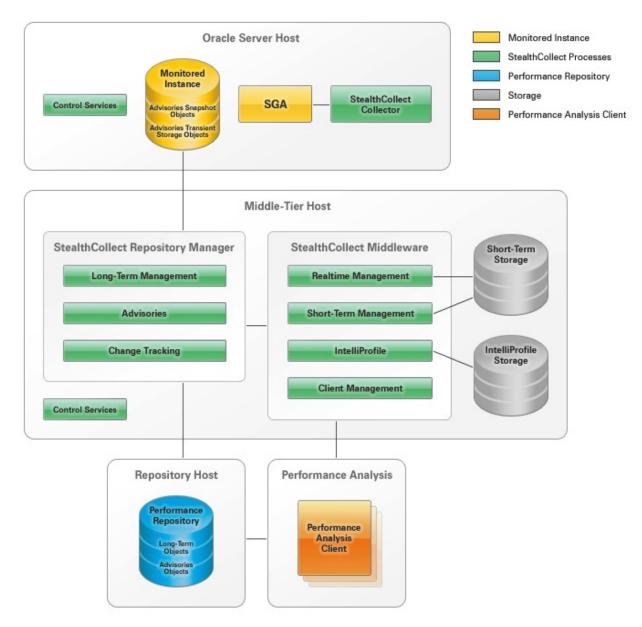
- contains, does not contain, begins with, ends with and is (for the current node)
- =,<>,>,<=,<,>= (for metrics)

The acceptable Values are:

- any set of ASCII characters (for the current node)
- any number (for metrics)

Agent

Performance Analysis is comprised of the client (GUI), database Advisories Objects (tuning), and the (StealthCollect) agent (back-end). The following is a simplified view of the Performance Analysis architecture:



The StealthCollect Agent (comprising the "Middle-Tier Host" section above) consists of the following:

- quest_launcher Starts, connects to, and stops processes and also manages user authentication.
- Collector Traces instance and sessions activity by rapidly (up to 20 times per second) sampling the Oracle instance's SGA. In this manner, no overhead is imposed, unlike querying Oracle's dynamic performance views (V\$). The collector resides on the same host as the monitored Oracle instance.
- Middleware Receives the SGA samples from the collector. The middleware is responsible for handling performance data and managing the interaction between the program's disparate applications across any Oracle-supporting platform. **Note**: The middleware can reside on any host; it is highly recommended to use a host other than that of the monitored Oracle instance. The middleware contains five key elements:
- Online Processor Translates the SGA samples into dynamic instance state snapshots for accumulation and diagnosis.
- Historical Processor Applies OLAP algorithms to create a shortterm database to answer short-term (up to three months) historical queries. The database uses standard operating system files that require disk space according to volume.
- Advisories Harness the Performance Analysis knowledgebase into a working "detect, diagnose and resolve" paradigm for problem resolution
- IntelliProfile artificial intelligence engine Generates normative baseline projections for exception-monitoring evaluation of current and historical behavior.
- Repository Manager Loads and aggregates historical data into the long-term performance repository. This process is active only if the long-term repository option is enabled. The performance repository can reside on any network host.
- Monitored Schema Objects Support the Advisories feature above with tables and procedures installed in the monitored Oracle instance.
- Performance Repository (optional) Stores long-term StealthCollect performance data. This is an additional, Oracle instance dedicated

to Performance Analysis.

For the latest information about supported platforms and versions, please consult the <u>PAO Supported platforms</u> document:

Quest software periodically extends the support for new database and platform versions. For registered Quest SupportLink users, click <u>here</u> to be directed to Quest's website for information on, and to download the latest Performance Analysis updates.

For additional technical information, view the <u>PA_Agent_Installer_help.pdf</u> file (also available on the installation CD).

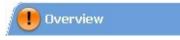
Related Topics

<u>The Performance Analysis Console</u> <u>History Mode</u> <u>Real-Time Mode</u> <u>Historical Diagnostic Reports</u>

Home Page

The Performance Analysis Home Page offers at-a-glance monitoring of primary <u>system health measurement</u> measurements, and provides the starting point for drilldown analysis in one easy to understand digital dashboard. This view offers both user-configurable, alarm-enabled normative baseline database advisory and applied knowledgebase toolsets to facilitate optimal user monitoring and tuning efforts; Users' attentions are intuitively focused on those instance and system aspects most in need of attention.

The **Home** page is divided into four distinct areas:



- The upper pane's two components display overall activity:
- The **Workload** bar graph displays the total current instance activity relative to a peak periods and is t a measure of instance resource capacity. Use this bar graph to assess overall database health from the relative value within the <u>baseline</u> range, relative to the total workload experienced, and by analyzing any observed performance <u>anomalies</u>.
- The **Instance Workload** timeline representation (in either the Baseline or <u>Workload</u> views) displays instance activity over the most recent time period. Use this timeline to identify the specific period of time when notable behavior occurred, either through the occurrence of specific wait events or by reviewing instance activity relative to the baseline.
- The lower pane's two components display individual dimension period bar graph measurements:
- The Oracle **Resource Breakdown** <u>wait event</u> consumption display helps users understand recent workload activity and whether any specific resource's consumption rate is problematic based on baseline, advisory and anomaly readings.
- The System Utilization **CPU**, **Memory** and **Disk** capacity usage indicators. Use this display of actual resource utilization to understand your system's throughput as a percentage of the

actual total available (capacity) through absolute, baseline and anomaly readings.

Performance Advisories Detect, diagnose, and resolve issues with one click. Performance Advisories identify and explain significant performance issues affecting database operation.

One click accesses both Quest's contextsensitive knowledgebase expertise and intelligently suggested solutions. Non-optimal or non-industry "best practices" database configuration and operational conditions are identified at the touch of a button, specific to the Oracle version being monitored.

@ E-Business

Key Oracle E-Business operational areas, including application performance and security characteristics, can be analyzed through a one-click application or health review.

Color Alerts

Performance Analysis presents two sets of clickable color coding to denote performance areas requiring further analysis.

- Performance Anomalies—Alert users to potential instance (and system) deviations from the normative behavior and/or configured thresholds.
- Advisories—Recommend improvements to both current and future instance performance behavior.

These ready-to-use color-coded alerts have been incorporated into Performance Analysis as follows:

Performance Anomalies—Home Page

Baseline Home page alerts direct your attention to any performance metrics indicating a potential problem during the reporting period. Click the resource bar displaying a performance anomaly to review the potential problem and optionally follow the hyperlink provided to drilldown for further investigation. There are two types of Performance anomaly alerts:

- Observed behavior is above the normative baseline range
- Metric values have exceeded predetermined threshold levels
- Performance or resource consumption is below or within the normative baseline range.

Performance or resource consumption has exceeded the baseline range's warning threshold and drill-down investigation of this parameter may be warranted.

Performance or resource consumption has exceeded the baseline range's critical threshold, likely requiring immediate attention.

This measurement exceeded the <u>Home Page's upper threshold</u>, likely requiring immediate attention.

Note: The overall instance alert level is determined by the most severe individual component.

Advisories

For resources exceeding their associated <u>performance advisory</u> <u>threshold</u>, a clickable warning flag is displayed. Clicking the flag identifies the issue and its severity and provides a hyperlink to an advisories page for reviewing specific timeframes when the problem was observed, as well as a suggested <u>action plan</u> solution.

The metric is behaving normally. Green advisories are displayed by default, and must be <u>activated</u>.

The metric (or workload resource) has exceeded the <u>configured</u> warning threshold. This may be causing performance degradation and warrants reviewing the affected periods for a more detailed understanding.

The metric (or workload resource) has exceeded the configured critical threshold. This is likely causing performance degradation and warrants reviewing the affected periods for a more detailed understanding.

Note: The overall instance alert level is determined by the most severe individual component

Overview Tab

The Performance Analysis Overview tab is the Home mode default view. Prioritized performance alerts, current and <u>baseline</u> normalized system resource utilization, key Oracle performance advisories, and industry best practices are presented side-by-side in a clean, intuitive format. <u>Common</u> <u>graphical elements</u> provide a consistent and intuitive representation of performance deviation and degradation in both relative and absolute terms.

The Overview tab is a real-time monitoring, alerting and tuning digital dashboard. Use it as a reference point and springboard to identify and address which outstanding issues the instance currently faces. Because of its rich analysis and alerting features, the Overview tab immediately pinpoints any resource contention or capacity limitations in need of immediate attention. By increasing real-time service delivery, problem-solving time is greatly reduced.

The user-friendly context-sensitive display and navigation tools give a complete performance report for the specified time frame—the time frame can be <u>configured</u> to show any timeframe from the last 5 minutes to the last 24 hours. Each indicator relates to the entire specified timeframe. This information can be seamlessly incorporated into each user's work flow and methodology.

Measuring System Health

The **Workload** gauge is a measurement of total instance activity; the gauge level indicates the proximity of instance activity to that of peak activity. Use it to compare the total session activity during the specified timeframe with that during peak periods.

Note: This measurement is intended to provide an at-a-glance understanding of instance behavior relative to an observed norm and not a measurement of system capacity.

Current activity can easily be determined to be typical, or relatively high or low by comparing it to the baseline range. The clickable <u>color coded</u> alerting feature both details workload metrics and serves as a launch point for further investigation of database behavior. The **Instance Workload** activity timeline displays Oracle instance activity (system workload) over the specified recent timeframe, measured by the average number of active sessions. You may toggle between baseline and <u>resources</u> views to gain a better understanding of database activity trends and levels, and of resource consumption.

Oracle Performance displays both the numeric and analogous resource distribution as a percentage of workload activity (displaying the same data as the <u>Real Time Resource Breakdown</u> pie chart). Use both the baseline comparison feature (like the Workload) and <u>performance</u> advisory flags to display resource metric usage details and to launch further investigations. Understand the composition of the present instance activity at-a-glance, and determine specifically which, if any, resource is creating a problem based both on the normative baseline and accepted industry standards.

System Utilization displays system and host parameter resource itemization. As with Oracle Performance, both the baseline comparison and performance advisory flags display resource metric usage details and serve as launch points for further investigation. High system utilization values do indicate full capacity has been reached.

Related Topics Best Practice Analysis E-Business Performance Analysis Console

Performance Advisories

Performance Advisories provide clear, explicit description of performance degradations affecting the database instance. Developed by Oracle performance tuning experts, these advisories:

- Detect and diagnose performance degradations and provide steps to resolve them
- Incorporate wait event analysis to focus only on significant issues affecting database performance
- Use fuzzy evaluation techniques to combine relevant metrics and convey the appropriate priority of each issue relative to the database workload and to other problems evident within the specified timeframe
- Provide clear and detailed explanations of performance degradations offering just-in-time education, and extensive background tuning information to assist DBAs in solving the problem at hand and in avoiding similar problems in the future

Advisories are presented in either a default ordered and categorized tuning plan or as an action plan that lists first the issues impacting database performance the most.

Also presented are any significant deviations in database workload and resource service times that may indicate performance anomalies that may merit further review or investigation.

Advisories are evaluated at 15 minute (optional), 1 hour and 6 hour intervals and considers database performance averages over the duration of the interval. Single degradation incidents that last for 15 minutes or less will show a greater priority in a 15 minute period than they might in a 6 hour interval. Use of three evaluation intervals balances the desire to identify single incident short duration degradations with the cost of repository storage. Frequent or chronic degradations will be evident in any of the intervals. Deviation advisories are evaluated at only 1 and 6 hour intervals.

Note: Advisory report exclude activity occurring outside of 15 minute

intervals not completely contained within the timeframe. For example, if the report timeframe starts at 12:03 PM and ends at 1:03 PM, then activity occurring during the interval 12:00 PM – 12:15 PM would not be included.

Related Topics

Action Plan Advisories Listing Advisories Report Performance Analysis Console Performance Analysis Configuration - Home

Action Plan

The action plan displays areas where there may be an opportunity to increase the overall performance of your database given some tuning effort. Advisories are sorted in order of priority and consider the number of periods in which they occurred for the selected timeframe. Click an advisory title to view detailed analysis results, a complete description of the recommended action to be taken, and some background information on the applicable tuning area.

Related Topics

Performance Analysis Configuration - Home

Advisories Listing

The Database Performance advisories are tasks grouped as follows:

Optimize the Application Workload		
	SQL statements may be using literal values when they could be using bind variables.	
	A SQL statement may contain variables that change from execution to execution. These variables are typically parameters to the SQL statement. They define the rows to be processed or new values to be inserted or updated. These variables can be specified either as literals or as bind variables. Using literals to retrieve details for employee 1234 could use the following SQL statement: SELECT * FROM employees WHERE employee_id=1234	
Bind Variables	A more efficient and scalable approach is to define the employee_id as a bind variable:	
	SELECT * FROM employees WHERE employee_id=:empid	
	The value of the host variable empid will be supplied to the SQL statement when it is executed. When a SQL statement uses bind variables, it is not necessary to re-parse the statement every time the argument changes. When the SQL statement is executed in another session, the matching SQL will be found in the shared pool and parsing will be avoided.	
	Determine if cursors are being reused inefficiently. When executing a SQL statement, the Oracle	
0	RDBMS assigns a private work area for that statement. This work area contains information about the SQL statement and the set of data	
Cursor		

an ap stater can ir may r	ed or affected by that statement. When plication submits the same SQL nent multiple times, reusing the cursor nprove database efficiency. Applications not be set up or designed properly to take ntage of cursor reuse.
red Pool Size would (lock) threat (PX) locate large unava the qu pool. Perfo	mine if overall database performance benefit from resizing the shared pool. hared pool area of the SGA contains two ry areas: the library cache and the hary cache. It also contains control ures such as NLS loadable objects, on and process context areas, enqueue resources, etc. Additionally, multi- ded server (MTS) and parallel execution are optional components that prefer to e their respective memory areas within the pool when defined. When a large pool is allable, the session memory of MTS and arery buffers of PX will reside in the shared rmance can improve when the shared s sized properly to avoid reparsing and ding.
uce Contention	
red Pool Latch Oracl	mine if significant shared pool latch ntion exists. ed pool latch contention can occur when hared pool is sized too large to be lively managed. This problem is less non starting with Oracle version 8.1.6 Oracle implemented improved shared nanagement architecture. In previous e versions, the shared pool management the to avoid shared pool latch contention
red Pool Latch red Pool Latch	pool when defined. When a large p ailable, the session memory of MTS arry buffers of PX will reside in the mance can improve when the shar s sized properly to avoid reparsing ling. mine if significant shared pool latch ntion exists. ed pool latch contention can occur v hared pool is sized too large to be ively managed. This problem is less non starting with Oracle version 8.1. Oracle implemented improved sha nanagement architecture. In previo

	The overhead to track the memory chunks in a large shared pool impacted the speed at which processes were able to obtain a needed chunk of memory.
Library Cache Latch	Determine if significant library cache latch contention exists. The library cache latches protect the cached SQL statements and the cached object definitions held in the shared pool library cache. A library cache latch must be obtained before Oracle can add a new statement to the library cache. During a parse request, Oracle searches the library cache for a matching statement. When one is not found, Oracle will acquire the library cache latch, parse the SQL statement, and insert the new SQL. Contention for the library cache latch occurs when an application generates numerous unique, unsharable SQL (when literals have been used instead of bind variables). When library cache latch contention exists, improve the use of bind variables within your application. Misses on this latch can indicate that your application is parsing SQL at a high rate and may be suffering from excessive parse CPU overhead.
Redo Allocation Latch	Determine if significant redo allocation latch contention exists. Redo allocation latches are used to control the allocation of space for redo entries in the redo log buffer. When redo information is generated, it is written to the log buffer. The redo allocation latch protects against simultaneous writes to the same log buffer memory. There is only one redo allocation latch per instance and only one process can allocate space in the log buffer at a time.

Redo Copy Latch	Determine if significant redo copy latch contention exists When redo information is generated, it is first written to the log buffer before being written to disk. When the LOG_SIMULTANEOUS_COPIES (renamed _LOG_SIMULTANEOUS_COPIES in 8i) parameter is set to a value greater than 0, Oracle uses redo copy latches to reduce contention for the redo allocation latch. The redo allocation latch is used to prevent another process from writing in the same location of the log buffer. There is only one redo allocation latch per instance and only one process can allocate space in the log buffer at a time.
Cache Buffers Chains (CBC) Latch	Determine if significant cache buffers chains latch contention exists. Cache buffers chains (CBC) latches enforce synchronized updates to data in the Oracle buffer cache. Data blocks in the buffer cache are managed in singularly linked lists of data block headers (sometimes referred to as buckets). Data block headers contain information about the data block including the memory address of the block in the buffer cache. To provide efficient data block access, Oracle computes a hash algorithm on the dat block address (DBA - an encoding of file and block number) to identify the cache buffers chain (one of many) that may contain the block. The chain is then scanned to find the corresponding header. When the header is located, the buffer cache address is available and the block can be retrieved. This process is much faster than scanning the entire buffer cache for the requested block.

	While Oracle scans and updates a cache buffers chain, a CBC latch protects the chain to ensure that block headers are not being added or removed from the list simultaneously by other users. Optimally, other users should not have to wait long to access the same cache buffers chains latch.
CacheBuffers LRU Chain Latch	Determine if significant cache buffers LRU chain latch contention exists. The cache buffers LRU chain latch is held when updates are made to the LRU (Least Recently Used) lists. Two LRU lists are used to track database blocks. The first list, the LRUW, tracks "dirty" (modified blocks not yet written to disk) block buffers. The second list, the LRU, tracks "clean" (unmodified) block buffers. Both lists track the least recently used blocks. When a block is used or freed, it goes to the top of the list, and the blocks below are moved down the list. After the dirty blocks on the LRUW reach a certain level, they are written to disk and placed on the LRU list containing clean blocks. When a clean block is needed, the LRU list is searched. When a clean block is modified, it is placed on the LRUW list for writing by the DB writer. Cache buffers LRU chain latch contention is caused when the LRU and LRUW lists are updated and the chains of buffers in each hash bucket are too numerous to be scanned quickly. Contention is also caused when the volume of transactions overloads the block tracking on the LRU lists. The solution to either of these cases is to allocate the resources needed to update the LRU lists quickly.
	Determine if significant lock contention exists.

Locking	Lock (enqueue) waits occur when a process must wait to update data when another process is already updating it. Locks are mechanisms used to provide ordered access when updating data. Oracle also uses locks for maintaining internal data structures and processes may have to wait for access to these internal locks as well (for instance, the Space Transaction enqueue). When the database is well tuned and the application design sound, lock waits should be negligible.
Tables Experience Lock Waits	Determine if transaction lock waits for tables were observed. The purpose of a lock within the database is to protect a database resource (in this case a table) that may be needed for a relatively long time as compared to latches which are expected to be held only briefly. The Oracle database locking mechanism allows multiple sessions to wait in line for a given table if that table is currently locked by another session in a mode that is incompatible with the lock being requested. When entire tables are being locked unnecessarily, contention on such tables may result.
Space Transaction Waits	Determine if significant space transaction lock wait exists. If so, identify users that have permanent tablespaces defined for use as temporary storage. When Oracle modifies the segments allocated to objects, it grabs and holds the space transaction lock until the modification completes. Excessive segment modification is especially noticeable when permanent tablespaces are assigned for use as temporary

storage, significant disk sort operations are occurring in those tablespaces, and the default size for the tablespace is set too low. It may also occur when rollback segments are growing and shrinking excessively.

Oracle addressed this issue with the introduction of tablespaces of type TEMPORARY and further improved the situation with the introduction of Temporary tablespaces in version 8i.

Assigned Temporary Tablespace

Each database user is assigned a tablespace to use for temporary storage operations. Temporary storage is necessary when performing large sorts, large hash joins, and other operations. However, sort operations are typically the primary use for temporary storage. When sort operations are too large to be done entirely in memory, temporary disk storage is utilized to store intermediate sort results. Sort operations are necessary for such purposes as completing ORDER BY or GROUP BY operations, sort-merge join operations, and for index creation.

When creating a tablespace, it can be designated for use as temporary-only storage. By doing this, Oracle more efficiently manages the utilized space for temporary purposes. Temporary tablespaces should always be specified when assigning tablespaces to users for temporary space usage.

Determine if there are enough redo log disks defined and if they are optimally sized. When a transaction is committed, a physical write to the redo log file must occur. The write must complete before the COMMIT returns

Redo Log Disk Configuration	control to the user and therefore, the writing of the redo log imposes a limit to throughput of update intensive applications. Redo log I/O is optimized when the log is located on a dedicated device where there is no contention with other processes. By not having other processes using the disk device, the disk head maintains its position while awaiting the next write request. Thus, the disk does not need to take the time to "seek" before performing the next write and thus write times are minimized.
Number of Redo Logs	Determine if the database has a sufficient number of redo logs. A log switch checkpoint must complete before a redo log group can be used. This checkpoint ensures that all transactional information in the redo log buffer has been written to disk. When in ARCHIVELOG mode, redo log groups must be archived before they can be reused. When the archiving process is sufficiently slow (usually due to disk device contention) and the archiver has not yet completed archiving a redo log group, the database will halt and transactions will wait for the archive process to complete before allowing the LGWR (log writer) process to re-use a redo log group. The database cannot be modified before the checkpoint or the archive processes have completed. Avoiding these waits is therefore very important to ensure adequate throughput and response time. The best way to guarantee that you do not experience waits for checkpoints or for archiving is to ensure that there are enough redo log groups. This allows checkpoints and archiving to complete before the logs are

	scheduled for reuse.
Redo Log Size	 Determine if redo logs are sized too small. When a redo log is full, Oracle must perform a "log switch." Redo logs should be large enough to avoid too frequent log switches and any corresponding log switch waits. Log switch waits can occur under the following conditions: The log switches are too frequent. The log switch checkpoint is not complete. The database is in ARCHIVELOG mode and the log being switched to is not yet archived.
	Redo logs should be sufficiently large (and numerous) to avoid these conditions.
Redo Log Buffer Size	Determine if resizing the redo buffer log will improve performance. The redo log buffer stores changes that will eventually be written to the redo log files. It buffers these writes so that the user session can continue working without requiring a disk I/O. When a COMMIT is issued, changes in the redo log are written to a disk by the log writer before the COMMIT is completed. When the log buffer reaches 1/3rd full, the log is written to disk even without a COMMIT. The redo log buffer should be sized properly to minimize the time waiting for writes to the log file to complete.
	Determine if sequence generators have an optimal cache size. Oracle sequence number generators utilize a built-in caching mechanism to gain their

Sequence	Cache
Size	

substantial efficiency over an approach of storing and retrieving sequence values from a user table. This is because reading and updating a sequence value in memory is much faster than updating one stored on disk. Oracle sequence generators are cached in the shared memory area (the SGA). By default, 20 numbers are cached. Once the 20 cached numbers are used by the application, Oracle caches another 20 values and updates Oracles internal sequence table. When sequence numbers are allocated at a high rate, performance can be improved by increasing sequence cache size. A larger cache size reduces the number of required read/writes to the sequence table.

Increasing the cache size does not consume additional memory in the SGA. However, it does increase the number of sequence numbers that can be lost if the cache is flushed or the database shutdown. Most applications do not require a complete set of contiguous numbers and therefore, lost sequence numbers do not pose a problem. If a specific application does require complete accounting for a set of sequence numbers, take care in designing the application code to avoid loss of numbers, including when transactions are rolled back.

Determine if database buffers are experiencing contention.

Buffer busy waits happen when multiple sessions want concurrent access to a block in the buffer cache. The modifying session marks the blocks header with a flag letting other users know a change is taking place and to wait. This temporary blocking ensures the reader has a

	 coherent image of the block (with either the before or after image). The two main cases where this wait can occur are: Another session is reading the block into the buffer Another session holds the buffer in an incompatible mode
	While the block is being changed, the block is marked unreadable. The changes should last under a few hundredths of a second. A disk read should be under 10 to 20 milliseconds and a block modification should be under one millisecond. Many buffer busy waits need to occur before performance degradation is noticeable.
Buffer Busy Wait	However, in a problem situation, there is usually a hot block, such as the first block on the free list of a table with high concurrent inserts. Users insert into that block at the same time, until the block exceeds PCTFREE, then users insert into the next free block on the list, and so on. Description of Freelists
	Freelists are structures which are included within each table. They list the blocks that are free to be used for inserts. These structures are maintained in the table header block (unless multiple FREELIST_GROUPS are defined). Multiple freelists can be configured with the FREELISTS clause in the CREATE TABLE and (in 8.1.6) the ALTER TABLE statements. Configuring multiple freelists is an important optimization for tables that are going to be subjected to high rates of concurrent inserts.

	A buffer busy wait occurs when an oracle session wants to modify or pin a buffer (copy of a database block in the SGA) but cannot because another session holds the buffer in an incompatible mode. The V\$WAITSTAT virtual table records the number and duration of "buffer busy" waits, broken down by the class of buffer for which waits are occurring. When there are insufficient freelists, multiple sessions are allocated the same block in which to perform their insert When they both insert into this data block concurrently, one session performs the insert and the other experiences a "buffer busy wait" on "data block".
Rollback Segments	Determine if sufficient rollback segments are defined. When Oracle transactions make database changes, the information required to undo those changes is written to a rollback segment. Oracle ensures that the available rollback segments are evenly used. However, when too many simultaneously active database transactions are using the same rollback segment, the generation of rollback information can be inefficient.
	Determine if the current workload across the interconnect in a Real Application Cluster (RAC) environment is an indicator of any network latency. RAC uses cluster interconnects to transfer blocks between the nodes participating in the cluster. A block transfer occurs when a user session connected to one instance requests a block being held in the cache of another instance. This feature of RAC to transfer information from the cache of one instance to

Cluster Interconnect Latency	the cache of the requesting instance is called cache fusion. The performance of the cluster interconnects is crucial to the performance of the RAC cluster and more specifically to the movement of cached data between instances. Its performance is measured by determining the average time a block takes to reach the requested node i.e., from the moment that a block was requested by an instance to the moment the requesting instance received the block. As in any application or database instance, occasional spikes of user activity are expected. However, when the average of such spikes remains high for an extended period of time, it could indicate a correctable performance degradation of the database cluster.
Minimize Physical Di	sk I/O
Buffer Pools	Determine if resizing the buffer cache will improve performance. Reading a data block from a physical disk device is thousands of times slower than reading it from memory. A disk read is avoided when a data block is already in the buffer cache. Avoiding disk reads in this manner is one of the most effective optimizations that can be made to an Oracle database. Allocating more memory to the buffer cache improves the chance that a data block will be in memory when requested a second time.
	Determine if adjusting the sort area size may improve performance Increasing the sort area size allocates more memory for sort operations and thus can

Sort Area Size	need to use sorts. The s the SORT parameter. Sort operation commonly r memory sol sort operation allocated sol performed us storage. Other	erall performance by re slower disk storage to sort area size is specifi AREA_SIZE configuration ons in Oracle are of tw referred to as either dis rts. When the data volu- on is small enough to ort memory, the sort ca using only memory and herwise, temporary dis red to perform the sort.	o perfor ed by s tion vo type sk sorts ume of fit in the an be d no dis sk stora	rm setting s and s or the sk ge
	Determine if resizing the PGA will improve performance. Oracle allocates a private process memory area for each server process called the Program Global Area or PGA. This memory will be allocated whether the type of connection is dedicated or shared. The contents and location of the memory will vary depending upon the type of connection. The table below lists the two main components and their corresponding memory locations for dedicated and shared servers.			
	PGA Contents	Description	Memory location for	Memory location for shared
	Session memory	Holds sessions logon information and other details regarding session.	PGA	SGA
PGA Size	SQL Execution memory	Data such as bind information and runtime memory structures	PGA	Part in SGA and PGA
	Persistent area	Holds the cursor information that persists across multiple executions of the same statement such as, bind details, data type conversion, etc. The persistent area is deallocated when the cursor is closed.	PGA	SGA
	Runtime area	This area contains information	PGA	PGA for

Optimize Physical Di	used while a SQL statement is being executed. Its size depends on the type and complexity of SQL statement as well as the number and size of rows being processed. A large portion of this memory is dedicated as the working area for operations such as sort, hash-join, bitmap-merge etc.DML/DDL SGA for queriesBy adjusting the memory allocated to the PGA, performance may improve by reducing the time needed to recreate memory structures needed to complete frequent database operations.DML/DDL SGA for queries
Datafile I/O Load	Determine if I/O load is balanced across
	datafile disk devices.
	Datafiles should have similar I/O service times. When datafiles have very different I/O service times, it indicates that the I/O load across datafile disk devices may not be balanced. Following are common reasons why a device/filesystem may have longer service times than another:
Balance	 It may be physically slower (IO per second rating or average service time rating)
	 rating or average service time rating). It may be servicing simultaneous requests from multiple processes and/or datafiles (contention). It may not be using caching (or better caching algorithms).
	 Other devices may be using RAID or other advanced technology disk architectures.
	Determine if a sufficient number of database writer processes or slaves are configured. The database writer process (DBWR) is the only process that writes modified database blocks from the buffer cach to the database

Database Writer Processes	files. The database writer works asynchronously. This means that a user process never needs to wait for the database writer to finish writing. However, if the DBWR falls behind sufficiently, then the buffer cache will fill up with "dirty" blocks and waits will occur while user processes try to read new blocks into the cache. These bottlenecks can be eliminated by configuring multiple database writers/slaves or by enabling asynchronous I/O.
Number of Datafile Disk Devices	 Determine if sufficient disk devices are allocated for database files. The amount of physical (non-cached) I/O that can be performed by the database server is constrained by the following factors: The number and type of disk devices allocated to the database The I/O rate each disk can sustain The degree to which database I/O is evenly spread across disks The degree to which database I/O is evenly spread over time Contention for database disk devices by non-Oracle processes
Review Performance	Deviations
CPU Usage Deviation	Determine if CPU utilization has varied significantly from the typical utilization for the given period. Variances should be reviewed to ensure that they do not indicate the occurrence of a

	problematic change in database performance.
CPU Wait Deviation	Determine if CPU wait time has varied significantly from the typical wait time for the given period. Variances should be reviewed to ensure that they do not indicate the occurrence of a problematic change in database performance
I/O Wait Deviation	Determine if I/O wait time has varied significantly from the typical wait time for the given period. Variances should be reviewed to ensure that they do not indicate the occurrence of a problematic change in database performance
Latch Wait Deviation	Determine if latch wait time has varied significantly from the typical wait time for the given period. Variances should be reviewed to ensure that they do not indicate the occurrence of a problematic change in database performance
Lock Wait Deviation	Determine if lock wait time has varied significantly from the typical wait time for the given period. Variances should be reviewed to ensure that they do not indicate the occurrence of a problematic change in database performance
Other Deviation	Determine if other wait time has varied significantly from the typical wait time for the given period. Other wait is a catch bucket for resource wait items that do not fit within the other listed categories. Variances should be reviewed to ensure that they do not indicate the occurrence of a problematic change in database performance.

Workload Deviation	Determine if database workload has varied significantly from the typical workload for the given period.
	Variances should be reviewed to ensure that they do not indicate the occurrence of a problematic change in database performance.

Best Practice Analysis - Overview

The Best Practices analysis lists database configuration and operational conditions that are not optimal or that violate industry "best practices" for specific Oracle versions. Database administrators may find that following these "best practice" recommendations may improve the health and maintainability of the database.

The best practice categories include the following:

- **Consistency** report checks best practices related to rollback/UNDO and Redo logs within the database.
- **Data Dictionary** report checks for Data Dictionary best practice violations. This includes checking for existence of SYS schema object statistics and exception scenarios related to the SYS.DUAL table and its public synonym.
- Database report checks overall database best practice violations.
- **Instance** report checks initialization parameters and other instancelevel best practices. Bringing atypical parameter settings to a DBAs attention allows for their quick review for appropriateness. Database upgrades, instance-level debugging, parameter mismatches, and simply "forgetting" to undo temporarily-set instance parameters frequently causes unnecessary performance overhead.
- **Object** report checks best practices related to tablespaces, tables, indexes, synonyms, and other object types to ensure the best possible environment for optimal performance, system, and application maintenance. Whereas some object checks identify issues that may lead to improved performance, other checks simply identify issues which may lead to reduced DBA or application programmer maintenance requirements.
- **Security** report check security related configuration of the database including a check for default and easily guessable passwords.
- **User** report checks best practices related to Users (schemas), Profiles, and Roles

Detailed results and recommendations are displayed from the analysis by clicking on the category titles.

These Best Practices help you to stay on top of the new Oracle features introduced for each Oracle release. Most database administrators will find that applying these best practices will improve the relative health and maintainability of their databases and database applications.

Related Topics

E-Business Analysis

E-Business Analysis

The E-Business analysis reports on key Oracle E-Business operational areas including application performance and security characteristics and can be used for a general E-Business health check review.

These key areas include the following:

- <u>Application Site Profile</u>
- Invalid Objects
- <u>Security Overview</u>
- Audit Space Management Optimization (Purge)
- <u>Application Performance</u>

Application Site Profile

This report provides a high-level overview of the installed application.

This overview includes the following:

- Installed version
- Application Attributes
- Critical Site Level Profile Options
- Server Summary
- Installed Product Summary

The information in this report is used as the starting point of any application analysis or debugging effort. The version number and optimizer mode are used to determine the level of object statistics that should be collected.

The following topics are included in this report:

- Oracle E-Business Suite Environment
- Optimizer mode
- Profile Options

Invalid Objects

This report details the numbers of invalid objects related to the Oracle Ebusiness and registered custom applications installation.

A registered application is any application that is registered through the Oracle E-Business application.

It is important to know whether there are any objects that have become invalid over the past 1 - 2 months, as these objects may be used as part of a business process that will fail if and when the object is used. This may be a major problem if the result of encountering the object causes any delay and forms part of the highly visible month end-processing schedule.

An invalid object is any database object Function, View, Stored Procedure, etc. that has been marked (internally) with a status of Invalid. In an Oracle E-business environment there are over 45,000 objects, many of which are interrelated and some of which are dependent on shared modules (not fully installed). So it is common for there to be a number of invalid objects that will never be "fixed". There are many causes of objects invalidation. The main cause is version patching.

The following topics are included in this report:

- Identifying New Invalid Objects
- Findings
- Recommendation

Security Overview

Many users from different organizations and possibly even users outside your corporate firewall may use the Oracle E-Business Suite. For this reason, maintaining application security is essential. This report provides a high-level starting point to identify security issues within the application.

An application user is any user that has been registered in the application and has not been end dated. New users are often added, but the user maintenance process generally falls down when the user leaves the organization and they are not removed from the application (end dated).

This report provides an overview of the user activity and identifies all the users who are candidates for removal (end dating). This report also identifies the users that have Systems Administrator or Application

Developer responsibilities (this help you to locate users who should not have these responsibilities). Finally this report will identify common security issues that are easy to remedy.

The following topics are included in this report:

- Aged Users
- Accounts with Special responsibilities
- File Permissions
- Database account with special privileges
- Database Account access
- Password complexity
- Recommendations
- Monitoring
- Notes

Audit Space Management Optimization (Purge)

The purpose of this report is to identify high priority purge programs which the Applications System Administrator should run to ensure a proper working environment.

This report checks to see whether specific Oracle supplied purge programs should be run. It checks to see if these Oracle purge programs are scheduled and then determines the volume of data that can potentially be purged based on best practices. This report also lists some key indicators about the range of data that can be purged.

The Applications Administrator should also review the purge programs available in the various functional areas installed.

The purge programs listed below represent a small portion of the purge programs available in the Oracle E-Business Suite.

The following topics are included in this report:

- Concurrent Request and Concurrent Queue information
- Sign on Audit information
- Temporary Table information
- Workflow Runtime Data

- Generic File Manager information
- Purge Debug log
- Order management Message information
- Recommendations
- Monitoring

Application Performance

This report details the top batch processes in the Oracle E-Business Suite Environment. These programs should be examined for possible fixes either in the form of Oracle performance patches, custom programming fixes or modification of business practices around these areas.

The concurrent programs running in the Oracle E-Business Suite environment can have a significant impact of the performance and/or perception of performance of the environment. Understanding what programs are running the most and if these patterns change is a necessary part of maintaining the Oracle E-Business Suite environment. This report provides a view of the top consumers of system resources. It also shows the administrator the distribution of concurrent jobs over time to help the administrator balance schedule processes to balance the system load over the day and weeks.

Quest Software's Spotlight on Oracle Applications can be used with these reports to more closely examine these jobs and their impact. Spotlight on Oracle when used with Spotlight on Oracle databases and SQL Lab is a complete solution from Quest Software to identify and resolve performance problem with the Oracle E-Business Suite. Additionally, Quest Software's Foglight can be used to proactively monitor specific programs and assist with the tuning effort.

The following topics are included in this report:

- Top 50 concurrent programs by runtime
- Program schedule distribution
- Concurrent Manager tuning
- Sleep times

- Database initialization parameters for Oracle Applications version 11i
- Statistics
- Resolving overloaded server problems
- Monitoring

Related Topics
Best Practices Analysis

Real-Time Mode

The real-time mode is the Performance Analysis tool for real-time and recent (up to the last 60 minutes) monitoring of both instance and session activity. Use this tool to trace session activities down to the statement resolution. You can identify excessive resource consuming sessions and determine which SQL statements caused this excessive consumption. Powered by the StealthCollect technology, this tool offers resource-light real-time monitoring in a friendly graphical interface.

The customizable Time and <u>Resource</u> specifications are a powerful and user-friendly feature for DBAs. For analysis of your Oracle system's current performance, you can obtain instant drilldown and metric data for active sessions, <u>Top Sessions</u> and <u>Current Sessions</u>.

Performance Analysis Real-Time mode has two nodes:

- <u>Real-Time Instance View</u> for rapid and comprehensive analysis of the current Oracle instance resource usage.
- <u>Current Sessions</u> for real-time display and drilldown into all active sessions.

Time Ranges

Use the **Time Range** selector to display and select from the available time ranges (last 1, 5, 15, 60 minutes). The automatic refresh interval is set in the <u>toolbar</u>. Changing the time range will affect all Real-Time window data. For the Current View, the time range is always that of the last minute.

Instance View

This window facilitates monitoring of current instance activity.

The graphs in the upper portion of the window display the entire <u>instance</u> <u>activity</u> layered by active wait states and a pie chart <u>resource breakdown</u> display.

Use the middle tabs to toggle between <u>Overview</u> and <u>Top Sessions</u> panes. When the selected resource is *Lock*, the <u>Blocking Locks</u> tab is available.

Overview tab

The **Overview** tab and feature is the primary history and real-time tool for initial understanding of the system resource use patterns that you see in the = 4) BSPSPopupOnMouseOver(event)" CLASS=BSSCPopup id=A1>activity timeline graph. The tool lets you view the performance-oriented metrics, time correlated with any activity drilldown. The middle pane timeline graphical representation and explanation displays respond to the choice of resource focus in the Resource/Metric list pane. ToolTip hints display any additional available information. The display is composed of the resource timeline and expert metrics (in accordance with the selected resource).

The overview tool is a unified view of the entire range of activity projections: full instance, specific session, and any historical activity drilldown (for example, User JOHN executing a program on machine ATLAS).

The pane list metrics displayed in the window's lower third are determined by the chosen = 4) BSPSPopupOnMouseOver(event)" CLASS=BSSCPopup id=A3>Resource. The Resource chosen displays those metrics that have the greatest relevance and impact on that resource as it may affect throughput and response time. In turn, the chosen metric determines the text displayed in the middle third right side. For those metrics that can be represented as a percentage of a total, a horizontal bar presentation presents both its internal component portions (for example, in the *Oracle/System CPU Usage*, CPU consumption is composed of the instance background and foreground CPU and CPU used by all other processes) and their total.

The overview representation and values are determined by <u>Metric</u> <u>Totalings</u>. For ease of use and comparability, differing totaling methods can be applied. Select from among the possible methods in accordance with your summary needs:

- Period **Total** (default) metric aggregation within one column for the entire time range
- Rate (Per Second)

Note: For methods that entail both averages and totals, the horizontal bar presentation and ToolTip display reflect the column in focus.

Top Sessions Tab

The **Top Sessions** tab is a pane display that lists the top resource consuming sessions per time range. Both resource and time range can be specified. The sessions are sorted in a descending resource consuming order (according to the specified Resource). Because the display includes all activity up to one hour ago, sessions that were active in this period but are no longer connected (dead), are also displayed.

The resources displayed include an initial general set and an additional listing of metrics specific to the chosen <u>Resource</u>. The following table contains a description of the general columns displayed in the Top Sessions grid.

Column	Description		
SID	Session identifier for the session.		
Serial	Serial number of the session. SIDs can be reused after the session disconnects, but the combination of SID and Serial number is always unique.		
SQL Executions Started	Number of SQL statement executions that ended during the current interval.		
Logon Time	The date and time that the user logged on to the database.		
DB User	Oracle user used by the program to connect to the Oracle instance.		
SQL Executions Ended	Number of SQL statement executions that ended during the current interval.		
Status	Denotes if the session is active or inactive for sessions that are currently connected or dead for sessions that logged out during the specified time frame and are no longer connected.		
Logout	The date and time when the session was logged		

Time out.

Double-click a session entry <u>to drilldown into the session</u> and trace its activity.

In addition to the <u>Common Elements in the Performance Analysis</u> <u>Console</u>, right-clicking the list pane display area in the Top Sessions tab displays a pop-up menu with the following options:

Option	Description	
View SQL	Display the active statement syntax in [Quest] SQL View.	
Tune SQL	Opens SQL Tuning and commences analysis of the chosen SQL statement.	
Copy SQL to Clipboard	Copies the SQL Text for pasting into a Windows application for further analysis.	
Multi-line SQL Text	Expands the SQL display to a multi- line presentation so that the entire SQL is visible.	
Summarize SQL Text	Shortens the SQL statements by replacing long SELECT and FROM clauses with ''	
Kill Session	Launches TOAD to terminate the Oracle session.	

Related Topics

Real-Time Instance View

Datafile I/O Tab

Use the **Datafile I/O** tab (visible for Workload or I/O resource specification) to display a lower pane listing of the data files on which the most I/O resource-demanding data activities occurred.

For systems based on EMC storage arrays, right-click a file of interest to initiate the Storage Analysis for EMC physical layout and I/O analysis for the corresponding data <u>file</u>.

Blocking Locks Tab

In real-time mode, with the Resource focus on Lock waits, the **Blocking Locks** tab displays all locks that have occurred and are occurring in the current period. This tool facilitates an at-a-glance, in-depth viewing of system locking activity. Use it to identify sessions causing excessive lock problems.

The default blockers display identifies the blocking sessions (denoted with **•**). The displayed blockers attributes are:

Blocker	Attribute
SID	SID of the blocking session. Click the SID to drilldown into the blocking session and <u>trace its activity</u> .
Serial	Serial number identifying the blocking session.
DB User	Oracle user under which the session is running.

Expanding the blocking session by clicking on the \square icon displays the set of individual locks created by the blocking session. Only locks which actually resulted in a block are included. The lock icon (\square) indicates a currently active lock. The displayed metrics consist of:

Column	Description		
SID	SID of the blocked session. Click the SID to drilldown into the blocked session and <u>trace</u> <u>its activity</u> . In the case of an active lock, the actual blocked statement can be discovered by looking at the current active cursor in the Open Cursors tab.		
Serial	Serial number identifying the blocked session.		
	Oracle user under which the session is		

DB User	running.
Locked Server PID	Process ID of the blocked session server process.
Lock Type	Indicates the Oracle lock type (for example, TX or TM).
Mode Wanted	Mode in which the blocked session has requested the lock.
Object Wanted	Database object on which the lock took place.
Lock Time	Date and time when the blocking began.
Duration in Time Range	Amount of time the lock took place within the displayed period (denominated in accordance with the <u>totaling method</u> in effect.

Current Sessions Node

The Real-Time mode Current Sessions node traces real-time instance and session activity sorted in accordance with chosen <u>Resource</u> and <u>Toolbar</u> settings. The Time range selector is unavailable, as all displays are fixed on real-time presentation.

Current Instance Activity

The = 4) BSPSPopupOnMouseOver(event)" CLASS=BSSCPopup id=A1>Activity Timeline dynamic representation displays the instance resource consumption in a layered manner, and the <u>Resource</u> <u>Breakdown</u> in pie-chart fashion, both, as of the last minute.

Activity Summary Panes

The following two panes are displayed/hidden by clicking on the \mathbb{H}/\mathbb{H} icon.

The **Oracle Sessions Summary** pane displays a current reading of the number of existing Oracle sessions, and their grouping as to whether active or idle. Active sessions are further classified as to their *current* <u>wait</u> <u>state</u> usage. These readings indicate how your Oracle sessions are employing system resources.

The **System Resource Usage** pane displays the CPU and Memory resources as follows:

- **CPU** Total percentage of CPU being used, divided into the portion used by this Oracle instance and the portion used by all other activities.
- **Memory** Server computer RAM, divided into the amount used by this Oracle instance, the amount used by all other activities and the remaining idle capacity.

Current Session Pane Listing

The list pane displays a *current* snapshot of *all* presently live sessions sorted in accordance with the = 4) BSPSPopupOnMouseOver(event)"

<u>CLASS=BSSCPopup id=A2>resource</u> selection.

Select <u>individual current sessions</u> () for further analysis and display by double-clicking the session of interest. The selected session then joins the left node listing, where it remains until up-to-an-hour after the session expires (that is, becomes inactive) or the user disconnects from Performance Analysis.

Use this feature to easily identify the most resource-demanding sessions currently affecting your system.

Pop-up Menu	Description		
View SQL	Display the statement syntax in [Quest] SQL View.		
Tune SQL	Opens SQL Tuning and commences analysis of the chosen SQL statement.		
Copy SQL to Clipboard	Make the SQL text available for pasting into a Windows application.		
Multi-line SQL Text	Expands the SQL display to a multi-line presentation so that the entire SQL is visible.		
Summarize SQL Text	Shortens the SQL statements by replacing long SELECT and FROM clauses with ''		
Kill Session	Launches TOAD to terminate the Oracle session.		
Show Background Sessions	Determines whether or not to display the Oracle background sessions.		

Right clicking within this area displays the following pop-up menu:

Related Topic

Real-Time Instance View

Individual Session Nodes

This feature's purpose is to trace the activity of a currently running session and its resource consumption up to the last hour. In this window, the Time Selector again becomes active. The Individual Session window is divided into three panes.

Session Activity Timeline

The upper-third graphically display an <u>Activity Timeline</u> and <u>Resource</u> <u>Breakdown</u> pie chart for the chosen session.

Session Detail Panes

The following two panes are displayed/hidden by clicking on the \mathbb{H}/\mathbb{H} icon.

The **Oracle Session Details** pane displays the Oracle session properties on a dynamic basis.

The "Waiting For" parameter is the actual wait event on which the session is presently waiting.

The **System Process Details** pane displays operating system attributes of the session's server process.

In a MTS environment, the Server PID attribute dynamically changes to reflect the operating system Process ID of the shared server that is currently serving the session.

Session Analysis Tools

The lower-third session tabs display facilitates focusing on specific session parameters:

- **Session Overview** tab Total Oracle resource usage and breakdown for the session in the activity timeline period with panelisted metrics, corresponding to the selected time frame.
- **Session Statistics** tab- Displays basic Oracle or extended statistics in a classical manner for the session, organized by Oracle groupings. The displayed **Value** is a function of the <u>totaling method</u>

in effect.

The first drop-down listing allows for a choice of either:

- Oracle Detailed Statistics: standard V\$SESSTAT statistics, grouped by class.
- Oracle Extended Statistics: calculated expert statistics.

The second drop-down listing totals either for:

- Selected Time Range: The metric display reflects only the currently displayed time range activity.
- **Since** (session) **Logon:** Reflects the entire session activity from its logon time, regardless of the selected time range.
- **Open Cursors** tab- Addresses the question of "What are the statements currently executing by the session?" Listing and details of all cursors currently opened by the session. The selection changes to reflect the currently active cursor. The displayed statement statistics reflect the chosen resource. All displayed statistics are cumulative since the beginning of the statement execution, regardless of the selected time frame.

Right-click within the **Open Cursor** display area to reveal the following pop-up menu:

Pop-up Menu	Description
Pin Cursor	If set, causes the active cursor to remain selected even when not active. Used for tracing a specific cursor.
Multi-line SQL Text	Expands the SQL display to a multi-line presentation so that the entire SQL is visible.
Summarize SQL Text	Shortens the SQL statements by replacing long SELECT and FROM clauses with ''

Tune SQL...Opens SQL Tuning and commences analysis of
the chosen SQL statement.

Use the <u>= 4) BSPSPopupOnMouseOver(event)" CLASS=BSSCPopup</u> id=A1>Statement Level Tabs for additional cursor analysis by clicking the cursor of interest.

• **Traced SQL** tab- Addresses the question "What were the most recent statements executed by this session?" The display offers a trace of all the recently executed statements. Similar display to that of the Open Cursors, but includes statements that have also already completed. Each statement refers to a specific execution

Use the <u>= 4) BSPSPopupOnMouseOver(event)" CLASS=BSSCPopup</u> id=A2>Statement Level Tabs for additional statement analysis.

• **SQL Summary** tab- Addresses the question "What is the impact of a specific SQL on real-time session activity?" Statements with identical SQL text are aggregated into a single entry. Similar to Open Cursors. The SQL Executions value indicates the number of distinct executions of the specified syntax as reflected in each summary line. This feature is useful when a session is rapidly executing SQLs (usually within a loop) with different bind variables and you are interested in the entire impact of the SQL rather than a single execution.

Use the <u>= 4) BSPSPopupOnMouseOver(event)" CLASS=BSSCPopup</u> id=A3>Statement Level Tabs for analyzing the SQL activity (the analysis tool will refer to the entire SQL grouping).

Session Locks tab- Listing of the locks (and their corresponding sessions) that are blocking (
 or □ when expanded) or blocked by (□ or without icon) this session during the course of its statement executions. An active lock that is currently blocking the session is indicated with an (□) icon.

Column Description

SID	 SID of the blocking/ed session. Click the SID to drilldown into the blocking/ed session and trace its activity. In the case of an active lock, the actual blocking/ed statement can be discovered by looking at the current active cursor in the Open Cursors tab 	
Serial	Serial number identifying the blocking/ed session	
DB User	Oracle user under which the session is running	
Server PID	Process ID of the blocked session server process	
Object Wanted	Database object on which the lock took place	
Lock Time	Date and time when the blocking/ed began	
Duration in Time Range	Amount of time the lock took place within the displayed period (denominated in accordance with the <u>totaling method</u> in effect	

• **Datafile I/O** tab- Display a lower pane listing of the files on which the session performed most of the I/O.

Note: For storage systems based on EMC arrays, right-click a file of interest to analyze the file within Storage Analysis for EMC.

Related Topic

Real-Time Instance View

Statement Level Tabs

The following parameter sets describe statement-level properties:

- Activity tab- Full statement syntax; The <u>Resource Breakdown</u> pie chart reflects all activity from its start to end or to the present if the statement is still active.
- **Statistics** tab- The drop down listing (of cumulative statistics since the beginning of statement execution or to the present if the statement is still active) allows for a choice of either:
- **Oracle Detailed Statistics**: standard V\$SESSTAT statistics, grouped by class.
- **Oracle Extended Statistics**: calculated expert statistics
- **Blocking Locks** tab- Listing of the locks (and their corresponding sessions) that are blocking (or blocked) the statement during the course of its execution. An active lock that is currently blocking the statement is indicated with an (□) icon.

Column	Description
SID	SID of the blocking session. Click the SID to drilldown into the blocking session and trace its activity. In the case of an active lock, the actual blocking statement can be discovered by looking at the current active cursor in the Open Cursors tab.
Serial	Serial number identifying the blocking session
DB User	Oracle user under which the session is running
Server PID	Process ID of the blocked session server process
Object Wanted	Database object on which the lock took place
Lock Time	Date and time when the blocking began

	Amount of time the lock took place within		
Duration in	the displayed period (denominated in		
Time Range	accordance with the <u>totaling method</u> in		
	effect		

• **Datafile I/O** tab- Display a lower pane listing of the files on which the statement performed most of the I/O.

Note: For storage systems based on EMC arrays, right-click a file of interest to analyze the file within Storage Analysis for EMC.

History Mode

The Historical Activity window is the Performance Analysis historical diagnostic activity component.

Performance Analysis introduces OLAP concepts for application performance analysis. The tool facilitates an iteration of detect, diagnose and resolve through examination of the activity dimension of interest.

The data that is displayed reflects the collection settings you specified at installation. The opening displayed History View (and the default view for most view drill downs) is as follows:

- The left-pane <u>dimension</u> navigation tree. The drilldown feature provides access to any of the key dimensions associated with Oracle databases activity, the tree node dimension display determines what subset of activity is displayed.
- The upper portion graphically displays the Oracle database wait state instance resource usage in either a <u>baseline</u> normative view or a dynamic <u>activity timeline</u> layered representation. Use the sliding magnification scale: Q— To magnify the metric scale range. The period activity is summarized in a <u>Resource Breakdown</u> pie chart. Configurable by **Resource** and **Time** parameters, these graphical representations provide you with an intuitive introductory picture and overview tool to the system Oracle activity at a glance.
- The lower <u>Overview</u>, = 4) <u>BSPSPopupOnMouseOver(event)</u>" <u>CLASS=BSSCPopup id=A2>Time Breakdown</u>, <u>Change Tracking</u> and <u>Datafile I/O</u> tabs display explanatory parameters of the system Oracle activity for purposes of greater understanding of the behavior of interest. The display includes a list pane of Oracle and system statistics whose activity you can individually time-line display and focus on through the assorted tab modalities.

Historical data exists from the most recent minute. A wide variety of means exist to view and navigate through the collected historical data, using dimension drill downs, filters, wait state selections, period analyses, and change occurrences.

Related Topics

Dimension Navigation

Time Navigation

= 4) BSPSPopupOnMouseOver(event)" CLASS=BSSCPopup

id=A3>Resource Navigation

Common Elements in the Performance Analysis Console

Dimension Navigation

The History mode tree Instance View drilldown is a unique feature that facilitates iterative (up to three levels) access to any of the key dimensions associated with Oracle databases activity, based on the OLAP multidimensional model. Domain nodes offer a hierarchical view of all types of Oracle activity characteristics.

The Tree node dimension display determines what subset of activity is displayed. Iterative drilldown into domains of interest facilitates increasingly refined focus and diagnosis until the core cause of the performance problem is identified. The problem can then be resolved with appropriate Quest tools.

The root node to which the display opens is the entire instance activity. By expanding the tree, you change the application focus to that of the selected dimension into which you have drilled. The tree-pattern iterative drilldown is a two-node process where the first node is the dimension type selection and the second is the dimension value.

For example, from the overall instance activity view, we seek to identify the most active DB User. Expanding the instance view node reveals the list of available dimensions. Clicking on the **DB User** (_____) node displays the top most active (in accordance with the selected resource) database users. Clicking on the first user will focus the entire window on this user's activity. We can now identify the most demanding SQL statement that this specific user has executed by expanding the user node and then clicking on the **SQL statement** dimension node. This will display the topmost active SQL statements executed by this user. In this manner, you can iteratively drilldown into any Oracle dimension of interest for a complete understanding of the causes of its behavior.

Performance Analysis highlights application behavior, usage patterns and service levels, as well as revealing performance problems and helping users to drill down to the problem root causes quickly and efficiently.

Among the more practical monitoring and administrative scenarios where Performance Analysis dimensions act as the key insight are the following:

• Find the topmost SQLs executed by a specific user.

- Find the most offensive users executing a specific program. For those of interest, identify the SQLs causing the resource usage.
- Within your Oracle Application, find the most active ORAAPPS users running a specific form.

To alter the display or order of Dimensions on the tree node, see <u>Performance Analysis Configuration</u>.

Default Oracle Dimensions:

- Instance (for RAC configurations) SID identifying the cluster member instance.
- SQL Statements The executed SQL commands.

Note: Right-clicking tree-node or pane-listed SQL statements enables activation of **Explain Plan History** to compare any two forms of execution plan used historically by Oracle for executing a given SQL statement.

• Programs - Name of the programs connecting to Oracle and executing the SQL statements.

Note: If Oracle fails to report a valid program name and the program is running on the database host, the program will be named according to the running program executable.

- OS Users Operating system users running the client program.
- DB Users Oracle users used by the program to connect to the Oracle instance.
- Client Machines The machines on which the client executable (connected to Oracle) is running.
- Actions Optional value in the V\$SESSION.ACTION column during SQL execution.
- Modules Optional value in the V\$SESSION.MODULE column during SQL execution.
- Clients Info Optional value in the V\$SESSION.CLIENT_INFO column during SQL execution.
- Command Types Executed SQL command type (for example

INSERT, SELECT, and so on).

• Sessions - Identifying the executing session (available for <u>User</u> <u>Defined Collections</u> only).

Note: When data received from Oracle contains empty values, the " <EMPTY>" designation will appear in the dimension tree of the history dimension navigation. (For example Oracle background processes which have no specific database user will appear as empty in the user node.) Drilling down within this domain will reveal its identifiable components. **Note**: Metrics can be added or deleted from the right pane listing by <u>enabling the grids customization</u> and then right-click in the pane listing and selecting **Select Metric...** from the displayed shortcut menu.

Related Topic

ERP Extensions

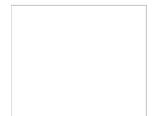
Time Navigation

Focus on historical periods of interest through time navigation. The vertical calibration is based on an average of recorded <u>baseline</u> peaks of observed behavior. Navigate in historical time within and between specific ranges by means of either:

- Time range drop down listing The drop-down time listing facilitates accessing time ranges relative to the present or defining customized time ranges as a baseline or bookmark for observance and return viewing. Unlike the default drop-down range specifications, which are relative to the moment of the query activation, time frames created from the editor are fixed. They can serve as persistent bookmarks for future reference and comparison.
- Activity Timeline drilldown The activity timeline is the principal display of Oracle activity.

To zoom (drilldown) into a period of interest

• Click and drag the cursor (\Box)over the period:



The period of interest can be expanded in an iterative process in increasing measures of granularity:

To zoom (shrink) out from a period of interest

 <shift> + click the cursor (
) at the point in time you want to be the center of a larger period.

Note: For drilldowns where period expansion requires display of data (in accordance with the displayed data interval) to granularity greater than that stored by StealthCollect, only the (most recent) periods for which data of the displayed granularity exists will be displayed. Decrease the data interval granularity to allow full period data display.

- Activity timeline scrollbar Horizontally scroll the entire range of data recorded in StealthCollect agent and repositories as displayed in the activity timeline. The scrollbar at the bottom of this view lets you perform the following tasks:
 - Scroll through your selected time period.
 - Page through the entire recorded history.
 - Select a time period to display.

To rapidly create a customized time range,

- 1. Click the **Time range** drop-down listing.
- 2. Click the **Other...** entry to display the **Edit Time range** dialog.
- 3. Use the **From** and **To** entries to specify a period of interest.
- 4. Clicking **OK** navigates you to the period.
- 5. Save the Range permanently by selecting the **Save as Permanent** checkbox and optionally supplying a **Time Range Name** other than the default range settings.

Alternatively, access the time range creation feature by clicking **Time range** (

Edit buttons to display the **Edit Time Range** dialog for creation of new time ranges or deletion or modification of existing time ranges.

In both the predefined and customizable **Relative Time Range**, both Last and Previous time units (relative to the present moment) can be specified. "Last" refers to the immediately preceding, while "Previous" refers to the most recent whole unit. Thus, at 10:20 A.M. on Wednesday, August 18, the *last* hour, 7 days, and 30 days refer to: 9:20—10:20; 10:20 August 11—10:20 August 18; and 10:20 July 19—10:20 August 18, respectively, while the *previous* hour, day, week and month refer to: 9:00—10:00; August 17; August 8 (Sunday)—August 14; and July, respectively.

Related Topic

Understanding Time Ranges and Data Intervals

Understanding Time Ranges and Data Intervals

The time range in Performance Analysis history mode is built on a pyramidal model of granularity, such that more recent data is available in smaller discrete time units (that is, higher granularity) than data from the more distant past. For example, in the following default configuration:

Time Range (Most real- time)	Default Data Interval	Time Range (Most real- time)	Default Data Interval
Last Hour Last 3 Hours	Seconds	Last 7 Days Previous Week	6 hours
Last 6 Hours	Minutes	Last 30 Days Previous Month Last 90 Days Previous 3 Months	Days
Last 24 Hours Yesterday	Hours	Last Year	Weeks
		All Available Time	Months

Note: Yesterday, Previous Week, and Previous Month are not time periods relative to the moment of definition; rather they represent, respectively, the previous calendar day (from 00:00 to 23:59), week (Sunday through Saturday), and month (first to month-end).

The data intervals default display is a system-generated optimal resolution in accordance with the displayed range and data availability. StealthCollect supports two models of historical collections (as determined at installation):

- Short term only Data is stored only up to 90 days. All data in all resolutions is held in data files within the StealthCollect middleware.
- Long term Data can be stored for an unlimited time period. Data intervals of one and 15 minutes are stored in files because of their quantity. Longer time intervals are stored in the repository, a restriction that limits the period for which this data can be aggregated and archived. The list of available intervals for each time range is derived from the historical collection configuration. The time pyramid configuration (initially set at the time of StealthCollect installation) determines the duration for which each data interval is stored.

Timeline Graph and Data Unavailability

In the above example, Performance Analysis may be unable to display data for a specific period (resulting in a blank display), for one of the following reasons:

- Historical data for the period was not collected because either the database or the StealthCollect agent was down. Verify which one by examining the value of the <u>= 4</u>) BSPSPopupOnMouseOver(event)" class=BSSCPopup id=a2>DB Availability instance level statistic.
- <u>Data Intervals</u>. The requested data intervals cannot be entirely satisfied with the selected data interval. In such cases, grayed out intervals can appear in either or both the display beginning and end.

This occurs principally when using dynamic (for example, last day, last week, last month) time frames.

Understanding Activity Significance

The Performance Analysis Resource Breakdown timeline provides you with a powerful and intuitive tool to rapidly isolate offending resource utilization, relative to entire instance and category totals. Use the <u>Activity</u> <u>Timeline Scaling Options</u> configuration in the Display Options <u>History tab</u> to focus the display:

• Select the **Scale graph to instance** check box if you want the display to retain the instance vertical scaling. Thus, any drilldown activity display is proportional to the total instance activity, granting you an immediate sense of its significance. You can manually refocus by using the vertical magnifier to the right of the timeline display.

If cleared, the display scale is that of the dimension subset of which the present drilldown is a part. (For example, the vertical scale could be a specific SQL statement and the activity representation, that of a specific database users portion within it.)

• Select the **Focus on target activity** checkbox if you want the view to automatically refocus as you drill down within dimensions. You can manually refocus by using the vertical magnifier to the right of the timeline display.

If cleared, the scale reverts to that of the instance (if selected, as above) or that of the dimension in which this drill down is a part.

• Select the **Show total in graph** check box to display a dotted line representing total instance activity (if selected, as above) or that of the dimension in which this drill down is a part.

ERP Extensions

In systems where ERP applications use Oracle instances as their database, Performance Analysis enables domain specification relevant to them. Performance Analysis customizes itself to reflect the domain-specific settings present in the system. Presently, Performance Analysis supports <u>PeopleSoft</u> and <u>Oracle Applications</u>.

ERP application identification occurs only at the time of StealthCollect installation.

Because most application servers facilitate pooling mechanisms, the basic Oracle dimensions lose their descriptive relevance as all connections opened within the connection pool are created using the same operating system user, database user, program and host. Thus, you can use the added ERP-specific dimensions to investigate the data using the ERP characteristics rather than the underlying Oracle dimensions.

Note: In clustered application servers, the Oracle Client Machine dimension can be used to identify the cluster member.

When connecting with Performance Analysis to an ERP database, the <u>dimension list</u> is extended to include the supported ERP systems. Standard application features and flow are then available for the ERP dimensions.

Oracle Applications

In addition to standard Oracle dimensions, Performance Analysis details the following Oracle Applications dimensions:

- OraApps User Name used to log into Oracle Applications.
- OraApps Module Oracle Applications grouping of windows and forms that constitute a domain application solution. The displayed value is the Oracle Applications module name.

Note: If the StealthCollect agent has not been configured to report the Oracle Applications module name, the module ID will be displayed.

Note: To let the StealthCollect agent automatically report the Oracle Application module name, logon to the Oracle Applications database instance as the APPS account and issue the following SQL command:

grant select on fnd_application_vl to public;

Note: To translate the existing module IDs into module names, issue the following query against the Oracle Applications scheme:

SELECT application_name ,
application_short_name
FROM apps.fnd_application_vl
WHERE application_id= <MODULE#>

- OraApps Responsibility Oracle Applications user-assigned role.
- OraApps Form Oracle Applications form executing the SQL command.
- OraApps Block Name of the block (form subset), which executed the command.

Related Topics Configuring Oracle Applications

PeopleSoft Dimensions

In addition to standard Oracle dimensions, Performance Analysis details the following PeopleSoft dimensions:

- PeopleSoft User Name used to log into PeopleSoft.
- PeopleSoft Client OS User Operating system user from which the client logs into PeopleSoft.
- PeopleSoft Client Machine Machine running the PeopleSoft client software.
- PeopleSoft Domain PeopleSoft server domain to which the PeopleSoft client is connected.
- PeopleSoft Programs PeopleSoft module being executed.

Analysis Tools

The principal means to display the variety of historical data available from Performance Analysis is by means of the available tabs, as follows:

Tab	Description
<u>Advisories</u>	Access here the Performance Advisories in a tree-node pane listing format. Advisory priorities are displayed in both summary and individual episode groupings.
<u>Overview</u>	Instance activity, as per the selected filter and as a function of time, facilitating a display of bottlenecks.
<u>Highlights</u>	A primary intelligent interpretation of your instance performance.
<u>Time</u> <u>Breakdown</u>	Resource activity in increasingly granular calendric time units.
<u>Tops</u>	The most resource demanding elements for the specified dimension, according to the selected resource (available when dimension nodes within the Instance View are in the active focus).
Datafile I/O	Files on which the most I/O resource- demanding data activities occurred.
<u>Change</u> <u>Tracking</u>	Changes in environments and activity correlated with Oracle's activity.
<u>Blocking</u> Locks	All locks occurring in the specified time period, the objects for which the wait was incurred and the lock duration (available when the Lock Resource is chosen).

Advisories Tab

Access database <u>Performance Advisories</u> in a tree-view listing format. Advisory priorities are displayed in both summary and individual episode groupings and are ranked by **Priority.** Expanding the listings to individual episodes displays the <u>severity color level</u> on the <u>activity timeline</u> both as a background and as a top color guideline.

Performance Advisories provide clear, explicit description of performance degradations affecting the database instance. Developed by Oracle performance tuning experts, these advisories:

- Detect and diagnose performance degradations and provide steps to their resolution
- Incorporate wait event analysis to focus only on significant issues affecting database performance
- Use fuzzy evaluation techniques to combine relevant metrics and convey the appropriate priority of each issue relative to the database workload and to other problems evident within the specified timeframe
- Provide clear and detailed explanations of performance degradations offering just-in-time education, and extensive background tuning information to assist DBAs in solving the problem at hand and in avoiding similar problems in the future

Related Topics

Performance Advisories

Overview tab

The **Overview** tab and feature is the primary history and real-time tool for initial understanding of the system resource use patterns that you see in the = 4) BSPSPopupOnMouseOver(event)" CLASS=BSSCPopup id=A1>activity timeline graph. The tool lets you view the performance-oriented metrics, time correlated with any activity drilldown. The middle pane timeline graphical representation and explanation displays respond to the choice of resource focus in the Resource/Metric list pane. ToolTip hints display any additional available information. The display is composed of the resource timeline and expert metrics (in accordance with the selected resource).

The overview tool is a unified view of the entire range of activity projections: full instance, specific session, and any historical activity drilldown (for example, User JOHN executing a program on machine ATLAS).

The pane list metrics displayed in the window's lower third are determined by the chosen = 4) BSPSPopupOnMouseOver(event)" CLASS=BSSCPopup id=A3>Resource. The Resource chosen displays those metrics that have the greatest relevance and impact on that resource as it may affect throughput and response time. In turn, the chosen metric determines the text displayed in the middle third right side. For those metrics that can be represented as a percentage of a total, a horizontal bar presentation presents both its internal component portions (for example, in the *Oracle/System CPU Usage*, CPU consumption is composed of the instance background and foreground CPU and CPU used by all other processes) and their total.

The overview representation and values are determined by:

- <u>Dimension Navigation</u> parameters
- <u>Metric Totalings</u> For ease of use and comparability, differing totaling methods can be applied. Select from among the possible methods in accordance with your summary needs:
- Period **Total** (default) metric aggregation within one column for the entire time range

- Rate (Per Interval) summarizes the data for averages and totals with periods defined by the Data Interval
- Rate (Per Second)
- **Per Execution** of SQL statements

Note: For methods that entail both averages and totals, the horizontal bar presentation and ToolTip display reflect the column in focus.

Related Topic

Overview Tab - Common Console Elements

Highlights Tab

Use the **Highlights** tab as an effective insight into your historical instance activity. The window features an intelligent rule-based engine that automatically identifies:

- Performance anomalies
- Resource bottlenecks
- Peak Activities

The Highlights feature performs a primary intelligent interpretation of your instance performance, saving the DBA precious time as the performance analysis is completed automatically. In addition to the performance findings, hot-links are provided to grant you one-click access to the issues requiring further attention.

The Highlights tab is available only for the historical instance view.

Performance Highlights

Performance Highlights displays instance level anomalies as measured in exceptional and significant metric values. For example, CPU and memory usage.

The textual row listings entries consist of:

- Performance metrics listings
- Explanations of the egregious behavior
- Hyper-links to the relevant time period
- (Quantitative metric values)

Additional information is available by clicking \blacksquare .

Right-click a specific rule implementation to display the <u>Performance</u> <u>Highlights</u> tab for configuration, with the focus on that rule.

Activity Highlights

Use the **Activity Highlights** pane listing to identify situations in which an individual or small activity subset (characterized by dimension values) is

the primary source of an activity (resource) workload, likely to be exceptional and significant behavior.

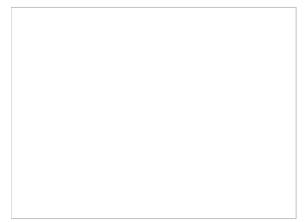
Clicking on the dimension hyper-link will navigate you to the consuming activity.

Configure the rule engine in the <u>Activity Highlights</u> tab.

Time Breakdown tab

Use this tab to display, in a pane-listed manner, resource activity in increasingly granular calendric time units determined by the <u>data interval</u> in effect. This feature is particularly useful in comparative analysis of periodic system performance. The selected resource (<u>Oracle wait state</u>) determines which metrics are shown in the pane listing display and the <u>totaling method</u> in effect determines the metric values.

Right-clicking within the pane listing display area brings up a dialog with the following options:



- Collapse Tree (with a sub-menu dialog to specify the available aggregation and sub-division display options) Report, and <u>= 4</u>)
 <u>BSPSPopupOnMouseOver(event)</u>" <u>CLASS=BSSCPopup</u>
 <u>id=A1>Compare</u>. As with all time-based parameters, available resolutions are a function of the specified time pyramid. For system default time units, you can iteratively drill down into the displayed values (and collapse to a higher level) by clicking on the plus (minus) icon at the left margin.
- Reports launched from a Time Breakdown period pane listing by a right-click initiate the Report feature with the pane in focus as the Specified Time Range.
- Comparisons launched from a Time Breakdown period pane listing by a right-click initiate the Compare feature with the focused line as the baseline and comparison.

Related Topic <u>History Mode</u>

Tops Tab

In accordance with the dimension filter, the most resource demanding elements for the specified dimension are displayed and sorted, according to the selected resource. (Available when dimension nodes within the Instance View are in the active focus.)

The Tops graphical display contains the overall Resource activity in accordance with the defining Dimension Filter. Clicking on a dimension value within the Tops pane listing displays its portion of the total resource activity in the graphic display above it. Thus you can readily see its usage as a portion of the total, over time. In this manner you can determine more fully this element's behavior pattern over the entire period of interest, including mutual effects of several entities on total activity.

The combination of top pane-listings of the **entire** period and a graphical view **over** the period lets you distinguish between elements whose resource usage is monotonous and elements with anomalous behavior (which are more likely to be causing bottlenecks).

To investigate behavior other than the default resource usage, use the drop-down listing in the graphical portion to focus on the different metrics. The metrics are those which are relevant to the displayed resource. The change of metrics does not alter the pane element listing, which are dependent solely on the resource selection.

Double clicking a tops entry drills down into it and causes it to be a focus of activity.

Note: The displayed dimension listing of top resource consuming entries is sorted based on the selected <u>Resource</u>. Thus the pane listing will differ for a given period, depending on the Resource selection in effect.

To configure the default number of Tops displays, see <u>Performance</u> <u>Analysis Configuration</u>.

SQL Filtering

Performance Analysis offers a filtering feature to limit display in the right pane listing to only those dimensions matching a set of user-defined expression conditions. When the left tree-node focus is on a dimension, use the <u>Click to create...</u> hyperlink to launch the **Filter Definition** dialog wherein you can construct, modify and delete a filter set composed of three part expressions (predicates):

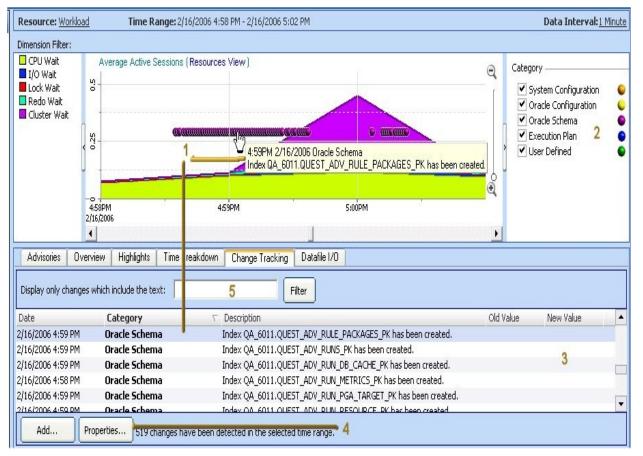
- **Field** The current dimension selected or any of the displayed Oracle wait event performance metrics
- Condition
- Dimension contains, does not contain, begins with, ends with, or is
- Oracle wait event metric =, <>, >=, >, <=, or <
- Value a valid character string or numeric value for the specified field

Related Topic History Mode

Change Tracking tab

The change tracking tool is an integrated monitoring feature available only if the repository is installed. It periodically tracks changes in environments and activity that can potentially influence system performance and enables the user to correlate between occurrences of changes and Oracle's activity and behavior patterns. Use the Filter feature to refine the set of displayed change tracking occurrences.

In the following graphic, access the five callout text messages, aligned with the number pointers, by hovering the mouse over the area of interest:



Note: Accessing change tracking from within a specific historical SQL dimension drilldown limits the displayed occurrences to only those relevant to that SQL. For example, you will be able to see only changes in tables, indexes and execution plan of that statement.

Oracle change categories:

System Configuration

Hardware and operating system configuration (for example, OS global parameters, disks and network interfaces (NIC); amount of RAM; CPU count; device installations and swap space allocation).

Oracle Configuration

Instance-level configuration: **init.ora** parameters; files (data, log and control) location, size and status; rollback segments; temporary table space.

Oracle Schema

Any schema object: Tables and indexes and their partitions and subpartitions; clusters; constraints; views; materialized views; packages and types and their bodies; functions; procedures; and triggers (for example, altering a table; altering or rebuilding an index).

Execution Plan

StealthCollect captures SQL statements whose execution plans have changed. When an execution plan has (unintentionally) changed, the outcome can result in SQL performance degradation. Therefore it is essential you investigate the change, by clicking **Properties** to graphically display the execution plan before and after the change, to verify that the nature and outcome of the change improve SQL performance.

User Defined

Changes inserted by user. Use this feature to document occurrences of interest. For example, the System Administrator installed an operating system patch. Recording this activity can assist in determining whether a change in performance can be directly associated with this event.

Create user-defined change-tracking points by clicking Add and

specifying the time and description in the subsequently displayed **Add Change** dialog.

Related Topic History Mode

Blocking Locks Tab

When the Lock Resource is chosen, the **Blocking Locks** tab is visible. At the instance view level, the resulting pane listed entries comprise all locks occurring in the specified time period, the objects for which the wait was incurred and the lock duration (in accordance with the = 4) <u>BSPSPopupOnMouseOver(event)'' class=BSSCPopup id=A1>totaling</u> <u>method</u>).

At the level of a drilldown into a specific Dimension, you can see the objects on which the current activity has been locked. Expanding the object displays the list of activities (characterized by their dimensions) that created the lock on that specific object and therefore blocked the current activity. Duration represents the time period for which this lock occurred. Clicking on a dimension entry facilitates a roll-up of the tree node with the selected dimension in focus, so that you can initiate an investigation of the locking source.

Related Topic

Lock Wait Category Blocking Locks Tab - Recent

Execution Plan History

Display in a dialog any two forms of an execution plan used historically by Oracle for executing a given SQL statement.

To access the Execution Plan History dialog

From a Change Tracking > Execution Plan entry click the History button or by right-clicking a SQL statement of interest in either:

- Left node pane listing
- Tops tab pane listing

Compare

Use the Performance Analysis Compare where differences in period activity illustrate underlying performance and monitoring issues. It helps you determine whether a comparison occurrence is an isolated incident or a sign of a potentially significant performance problem. The comparison can be of: whole instances or selected dimension breakdowns (such as user or SQL). Use Compare to address questions such as:

- What caused a specific activity?
- What were the resource and load demands of today's instance activity compared with that of a previous day's?
- Is an instance imposing different load levels now than previously?
- How do we explain the difference in a SQL performance compared with a previous period?
- What are the differences in program characteristics over two periods that caused different performance?

After identifying the different resource usage, you can use compare to identify what caused this difference: Usage pattern? Different SQL behavior? Environmental problem?

Overall Activity

The upper third **Overall Activity** panes graphically displays the Workload and SQL Executions and itemizes those wait events whose activity is significantly (\blacksquare) or moderately (\blacksquare) different between the two comparisons. Expanding a specific wait event displays graphically its occurrence over the source (base) and target (observed). (Workloads which are similar over the source and target are designated with a yellow icon (\diamond)).

Use this section to understand which wait states are most resourcedemanding and likely causing the difference in resource usage or whether the difference in behavior can be explained by the difference in quantity of SQL executions. **Note**: The graphical comparisons are on an absolute time basis, with the period commencements aligned.

Activity Highlights

For the wait event in focus, the middle **Activity Highlights** section displays the dimensions (of up to the top five) which were significant elements of the difference. Expanding the individual lines displays the dimension members which caused the difference and the composing metrics whose differences exceed the specified threshold.

Use the set of performance related metrics (defined by the chosen resource) to help you explain the difference in resource consumption; for example, a rise in I/O Wait might be explained by a rise in the quantity of physical reads.

All the **Comparison Results** <u>thresholds</u> and <u>metric totaling</u> methods are configurable.

Environment

The **Environment** section displays key <u>metric</u> values as occurred in both source and target, regardless of their absolute or relative value or difference. The displayed metrics reflect total instance or system activity, which are considered as an environmental factor that might affect the application behavior. Use this section to understand whether system or instance level problems are externally affecting your application so as to be the root cause of a problem (rather than the application itself). For example, an increase in the non-Oracle CPU consumption might explain an increase in an Oracle CPU wait state (as external applications consume more CPU, Oracle must wait more for CPU to become available).

Related Topics

= 4) BSPSPopupOnMouseOver(event)" CLASS=BSSCPopup id=A2>Running the comparison feature History Mode.

Running Comparisons

The comparison definition window is context sensitive with the time ranges and dimensions reflecting the settings in effect when the comparison was launched.

To run comparisons

1. Click the compare icon (\Box)

-OR-

From within the left History mode dimension tree node, right click a node and select **Compare**. The source dimension will reflect the tree node detail and the source time range will be that of the displayed Time range.

-OR-

From within the right Time Breakdown tab text pane, right click a line and select **Compare**.

-OR-

From within the Reports mode, with the left-pane focust on Compare, click **Run Compare report** in the right pane.

- 1. The source time frame and object activity are displayed in upper portion of the displayed **Compare Parameters** dialog left pane and reflect the tree node detail and the source time range from which the comparison was launched.
- 2. In the middle **Compare Options** portion of the displayed dialog left pane, click **Start a new comparison** to define the comparison input parameters. In each definition option, a set of windows allows you to customize the comparison, before clicking **Compare** to launch the feature:
 - Same object activity within a different time frame

- Different object activity within the same time frame
- Different object activity within a different time frame
- Modify source settings

To run a different comparison

• As per step 2 above, in the middle **Compare Options** portion of the displayed dialog left pane, click **Start a new comparison** to define the comparison input parameters.

To save an existing comparison

• Click the **Save compare settings as...** in the middle **Compare Options** portion of the displayed dialog left pane to display a dialog in which you enter a name under which the specific parameter specifications are to be saved.

To rerun a saved comparison

• From within the **Saved Comparisons** portion of the displayed dialog left pane, click the name of the saved comparison.

To remove a saved comparison

• From within the **Saved Comparisons** portion of the displayed dialog left pane, click **More Comparisons** and in the resulting displayed **Saved Comparisons** dialog, click **Remove this comparison from the list**.

To display the comparison in a report

• Click Report .

Business Views

Views are user-defined historical projections used for a more detailed, refined, and convenient presentation.

Use the views feature to rapidly and powerfully identify specific historical behavior of interest, both continuous and episodic. Views have the inherent advantage of greater flexibility and comprehensiveness in their ability to include multiple dimensions and domains for an at-a-glance understanding of pre-defined Oracle resources usages of interest.

If Benchmark Factory is installed, you can now submit groups of SQL statement directly for creation and execution of scalability tests.

In addition to dynamic Instance View drilldowns, with Performance Analysis you can create the following:

- <u>Shortcuts</u> are to be used like activity bookmarks, for one-click recall and continued observance of a specific activity defined by a dimension set of interest.
- <u>User Defined Collections</u> are shortcuts powered by a dedicated data collection in the back end to supply better resolution, more data, and more refined granularity than the system's default configuration.

Shortcuts

In its simplest usage, views are permanent shortcuts (
) to tree-view dimension node drilldowns, imposing no additional system resource requirements. Use shortcuts to gain rapid and convenient bookmark-like access to specific filter definitions.

Shortcuts definitions are saved on the user client computer and are therefore available only to the defining user.

With shortcuts you can preserve tree-navigation views of interest as you work.

- From within the application flow, as you encounter a drilldown of interest that you want to monitor over time.
- Or you can manually formulate more complex dimension combinations (for example, the resource consumption of a "cost unit" consisting of three specific users running any of four specific programs). Such complex combinations are unavailable by only using the dimension tree-node drilldown.

The Quest **Benchmark Factory** load testing process view is included by default. Use Benchmark Factory to generate system stress through automated testing, virtual users and agents by means of industry standard benchmarks to measure system capacity and performance bottlenecks. The Benchmark Factory view exposes the load generated by Benchmark Factory.

Related Topic To create a new shortcut

Creating a Shortcut

To open the Shortcut Editor dialog from the Performance Analysis tree view

- 1. From within the History node, right-click the **Instance View** to display the Performance Analysis action item menu.
- 2. Select Shortcut | New.

Alternatively, create (Save) a shortcut from an existing dimension tree domain drill-down node by right-clicking on the domain view of interest:

The **Shortcut Editor** dialog opens with the current drill-down dimension nodes already specified.

User Defined Collections (UDCs)

Like <u>Shortcuts</u>, User Defined Collections () are a mechanism by which Oracle resource consumption can be analyzed over selected domains and filters across different time periods. Unlike shortcuts, UDCs involve an additional collection of historical data, requiring middleware CPU and storage resources. Requesting inclusion of only those dimensions and time periods of interest minimizes unnecessary resource usage. Because of this additional resource consumption it imposes, you are advised to limit the quantity of concurrent UDCs to only those that are critical. Similarly, you are well advised to delete UDCs once you have completed your usage of them.

Data for UDC is kept within middleware files and not in the Performance Repository.

The default history collection addresses the entire system activity and therefore records activity only at a level of significance to the entire system-wide picture, so some of the activity is left out. Use User Defined Collections when you have identified a specific activity of interest and the granularity level provided in the default system collection is insufficient. By creating a UDC, you can record and monitor activity of interest at any level of granularity.

Added UDCs appear as additional Views tree nodes and are navigated in the same manner as the instance. The initial UDC node view is that of the total activity recorded by the specified dimension aggregation filter when the UDC was created and drilldowns reflect the dimension subsets.

After creation, you can only alter the history collection and storage attributes. For more information, see <u>Configuring User Defined</u> <u>Collections</u>.

To delete your User Defined Collections

• Right-click the UDC tree node to display the pop-up menu. Select **Delete**.

-OR-

From the Toolbar, click **Configure | StealthCollect**. From within the User Defined Collections node, with the cursor focus on the UDC to

be eliminated, click **Delete**.

In addition to Delete, from within **Configure | StealthCollect | User Defined Collection,** you can also:

- **Clear** your UDC of data. Perform this action to reset the existing content of the collection in situations where the data is no longer valid or of interest.
- **Stop/Start** the UDC from continued and additional activity. Use this action when you seek to preserve the collected data but have no further or a renewed interest in any further collection.

Related Topic

To access the New User Defined Collection wizard

Creating a User Defined Collection

The UDC definition wizard entails the following defining windows:

- <u>Identification</u> Record here the referential Name and description parameters.
- <u>Dimension</u> Limit the set of historical data to be stored and retrieved by clearing from check-box selection those dimensions that are not of interest for this UDC. After the setup is executed, modifying the dimension may cause data loss.
- <u>Filters</u> Limit the set for which historical dimension data is to be collected to that which includes up to three domain instances.
- <u>Scheduling</u> Specify here the time range, frequency and collection duration of historical data to be gathered. Limit the time range to only the period of interest because, as with dimensions, the time range size determines the extent of middleware resource requirements.
- <u>History Model</u> UDCs also require data storage by StealthCollect in both middleware data files and the performance repository, with data granularity a function of allocated disk space. Define here the general level of collection or use the <u>Custom option</u> to configure the data collection pyramid. Separately, accept the default data directory or specify an alternative location.
- <u>Summary</u> Review the UDC parameter settings displayed here as subsequent changes may cause data loss.

To access the defining History Configuration dialog feature

• (From the Performance Analysis tree view) Right-click in the left pane History mode and select **New User Defined Collection** in the displayed action item menu.

User Defined Collection (UDC) Creation Wizard

Use User Defined Collections when you have identified a specific activity of interest and the granularity level provided in the default system collection is insufficient.

The usefulness of UDCs as a performance monitoring tool is a function of how well the administrator knows the system's Oracle resources and (potential) bottlenecks. The ease and rapidity with which problematic behavior is identified and isolated depends on how well the critical dimensions are chosen.

For example:

- A user reports performance problems every Thursday between 13:00 and 16:00. Use the UDC to fully monitor the user's activity down to the statement activity level.
- Trace a known problematic SQL statement through an orientation of mutual effect of the SQL in response to tuning efforts (for example, index optimization or I/O optimization) and in this manner identify the specific effects and trends.

Related Topic
User Defined Collection

User Defined Collection (UDC) Wizard - Dimension

Use this step to define the dimensions of interest for which data is to be collected. For this specific UDC, limit the dimensions to only those relevant for the purpose of this collection.

Limitation of requested dimensions (and confining the collection periods) to only those of interest is the principal means to minimize the size of the storage files required by the UDC.

Available Oracle Dimensions:

- SQL Statement The executed SQL command.
- Program Name of the program connecting to Oracle and executing the SQL statements.
- DB User Oracle user used by the program to connect to the Oracle instance.
- OS User Operating system user running the client program.
- Client Machine The machine on which the client executable (connected to Oracle) is running.
- Command Types Executed SQL command type (for example INSERT, SELECT, and so on).
- Module Optional value in the V\$SESSION.MODULE column during SQL execution.
- Action Optional value in the V\$SESSION.ACTION column during SQL execution.
- Client Info Optional value in the V\$SESSION.CLIENT_INFO column during SQL execution.
- Session [SID, Serial]: DB RunID Identifying the executing session.

Note: The identifying session format is as follows: [SID, Serial]: DB RunID.

The SID and Serial combination identifies the session. Historically speaking this combination is not unique because the combination can reappear when a database has been restarted. The DB RunID uniquely identifies periods that occur between a database start and shutdown, therefore making this combination of SID, Serial and DB RunID a historical unique identifier for the session.

The dimension list will be automatically expanded in accordance with any site ERP configurations. For more information, see <u>ERP Extensions</u>.

Caution: Once the UDC has been created, the dimension list cannot be modified.

Related Topic
User Defined Collection

User Defined Collection (UDC) Wizard - Filters

Filtering each selected dimension will cause the collection to record only that activity which matches at least one of the selected values per each dimension (thereby further minimizing the size of the collected data). In this step you define which instance activity subset you want the collection to record. The filter feature works as follows:

Within each dimension for which filtered values are chosen, the activity's corresponding dimension value must be contained within the list. This condition must be met by each and every dimension for which a filter list value has been defined (Otherwise the activity is filtered out).

For example:

• When **DB Users** John_Doe and Rebecca_Smith run the **Programs** monthly payroll (monroll.exe) and invoice accrual (invaccr.exe), they occasionally encounter performance problems. You could create a UDC for these two users and two programs by including only DB User dimension values John_Doe and Rebecca_Smith and program dimension values **monroll.exe** and **invaccr.exe**.

The activity **Search** entry box specification serves a multiple role. Search entries can be used for either:

- a dimension set initial limiting string. For example, specification of "DROP" as a **Command Types** filter will display only and all those commands beginning with "DROP" (for example, DROP TABLE, DROP INDEX, etc..) for your individual (>) or group (>>) item selection.
- values not available through the displayed drop-down listing. Enter the value in the Search box and click ">" to add it to the included filter set.

Note: If no filters are specified, the UDC view will be that of the entire instance.

Related Topic

User Defined Collection

User Defined Collection (UDC) Wizard - Scheduling

This step lets you control the exact date and time at which the collection takes place. In order for activity to be recorded it should match both the dimension filtering and scheduling criteria.

Collection Period

The Collection Period defines the entire period where data collection occurs. Data will not be collected for any period prior to the actual starting date of the UDC.

For the Following Days

For the Following Days lets you set for which days within the week or month data collection is to occur. Use **Weekly** or **Monthly** scheduling in accordance with the underlying activity of interest.

Note: For selection of Monthly dates, the Days entry box allows specific days in a comma separated string, or ranges spanning a "-" for example, "2,16" for the collection to occur on the 2nd and 16th of each month or "10-15" for the collection to occur each month from the 10th through the 15th.

Collection hours within each day

Collection hours within each day defines the starting time and duration for the collection within each calendar day.

Related Topic

User Defined Collection

User Defined Collection (UDC) Wizard - History Model

This window determines the quantity and granularity of data stored for historical analysis.

Select the history level most appropriate for your system characteristics and needs. The data model selected determines the short term and long term space requirements.

Level	Select When
Entry	Uses less disk space. Top offenders still visible. High resolution data is saved for a short period only.
Standard	Suitable for most collections. Optimal combination of accuracy, granularity and space consumption.
High Volume	For collections of intensive activity (thousands of SQL executions per minute) and for which most of the activity is to be recorded. Offers high granularity at a cost of large disk space requirement.
Custom	Advanced Users only. Tailor the aggregation options to meet your needs. See <u>Custom</u> <u>Configuration</u> .

Disk Space

Short term historical data is stored in files (in the middleware). The entire space is claimed at the time of installation. The **Data directory** (located

on the middleware host) contains the history data files. This space is claimed during collection creation and therefore the directory should have sufficient free space to hold the indicated amount of required data.

When the amount of disk free space is lower than the estimated required amount, the free space indicator is marked in red and you must either select a less demanding history model (or manually customize) or find a sufficient data directory location. Either manually enter the data directory, or use the browse (___) option.

Until sufficient space is found on the data directory, you are prevented from proceeding with the collection creation.

User Defined Collection (UDC) Wizard - History Model Custom Configuration

Time Pyramid tab

The purpose of the time pyramid tab is to give an optimized storage and performance and rapid access to the data of interest. The logic of this model is based on the premise that the closer the data is to the present, the more granular (i.e. measured in finer time units) you want it to be. Conversely, the further it is from the present, the less refined the desired time unit of measurement, illustrated as follows:

Placing the cursor focus on a **Resolution** line lets you change its value and (in the sliding window box below) that of the **Keep For storage** period. Any changes initiated will be immediately reflected in the **Est. Server Space** and **Required Disk Space** values.

Reset to enables resetting of the entire configuration to the selected model.

Collection Settings tab

Specify in this tab's **SQL collection** settings:

- The quantity of top dimension occurrences and occurrence combinations. The overall tradeoff is between the number (depth) of collected occurrences and impact on space. StealthCollect data is, in fact, stored in a number of aggregations. The value for **items** setting has a greatest impact on top level dimension comprehensiveness (and therefore accuracy). The **item combinations** setting more directly affects second and third level dimension drilldowns.
- Select the check box **Remove literals in SQL** to replace every literal encountered in SQL statements with a bind variable. This option is useful if the application generates dynamic SQL with embedded

literal values and you want to aggregate the activity for these similar statements. This option is not selected by default.

• Select the **Ignore statements** if you want the collection to ignore statements whose activity was less than the Active Time (in seconds) that you specify here.

Use the Wait Events and Locks settings to:

- Enable/disable detailed wait events and locks collection.
- Define the threshold (as a percentage of the statement's entire active time) for the wait events and locks to exclude a statement from inclusion.

Disk Space Allocation tab

For advanced users:

• Use this option to manually override the system-determined UDC disk space requirement by specifying an alternative disk allocation. Your files will be permanently relocated to the new directory.

Note: Manually decreasing this disk allocation might cause a loss of data.

• Override the suggested default **Data directory.** The collection data files will be placed in the specified directory. Use this option if the default directory does not have sufficient free space.

User Defined Collection (UDC) Wizard - Summary

To display the UDC configuration settings

• Click **Create** to create the specified UDC.

Historical Diagnostics Reports

The variety of report types assists DBAs/Managers in understanding their system's behavior. These reports supply supporting documentation of instance activity. The Performance Analysis reports mode is an integral tool in tuning and monitoring Oracle database instances.

Use the supplied predefined read-only <u>templates</u> or create your own reports and save the template definition for subsequent <u>scheduled</u> executions. Output is available in a variety of formats, including window display, html, RTF and pdf file for effortless periodic delivery, distribution and archiving. Accepting the default-specified "Output to Performance Analysis" facilitates archiving and display within Performance Analysis.

Note: For reports to be run as scheduled, the Windows task schedule systems service must be running and the client must be active and have a network connection.

With the left-pane tree-node focus on a specific report, display its saved editions in a browsing mode in the lower-right pane by clicking on the specific execution reference above. Report output format is a first page of graphical representation of the data, headings, and definitions; subsequent pages of numeric data points; and a final legend page, if warranted.

To launch the report creation wizard

- 1. Right click within the:
 - Historical tree node
 - pane-listed Tops tab
 - pane listed Time Breakdown tab

-OR-

or

- : '\003cp' CLASS=StepParaTight style=;">Click:
 - Actions | Reports in the Toolbar

- **Reports** in the Performance Analysis navigation bar
- 2. Select one of the seven available types:
 - Executive Workload Summary The Executive Workload Summary is a one-click report-generating feature designed to deliver the critical <u>Home Page</u> displayed parameters in a useful configurable format to key decision-makers.
 - <u>Activity</u> The Activity report traces the activity and impact of a certain Oracle entity in a specific time frame. The report is modeled on the Performance Analysis historical diagnostics windows and gives an overview of behavior of a specific Oracle activity over time.
 - <u>Throughput</u> The Throughput report shows the throughput of a specific resource as a result of the load placed upon the resource by a given Oracle entity. The most common Template defaults: Upper is response time and lower is load (example CPU usage compared with SQL executions) so that variable functions can be compared one to another.
 - <u>Tops</u> The Tops report displays the most offensive Oracle activities (dimensions) for a given time frame, like that of the Performance Analysis' historical tops view.
 - <u>Trend</u> The Trend report graphs performance of specified top activities over an entire time period divided into customizable time frames. The purpose of this report is to expose performance changes occurring over extended periods of time.
 - <u>Time Breakdown</u> The Time breakdown report displays the top activities in each customizable time frame within a given period.
 - or

: '\003cp' CLASS=StepParaTight style=;">Change Tracking -The Change Tracking report display changes in environments and activity that can potentially influence system performance.

• Compare - The Compare report mechanism displays differences in period activity illustrate underlying performance and monitoring issues.

Report Templates

Performance Analysis allows you to select from among six types of reports. From within each group, you can customize a report (by changing any of its arguments and display formats) and save it as a report template.

Report templates provide a ready mechanism for an automatic creation and distribution of usage reporting of general interest. For example, a daily report of intensive resource users can be automatically published on an intranet site for wide dissemination, awaiting your morning arrival. In this manner, customized reports can fulfill key DBA monitoring and reporting responsibilities. Use the supplied templates or create your own template-based report.

Templates are organized within the <u>seven report groupings</u>. From the Reports navigation tree, expand a report group node to display the supplied and user-customized templates associated with that group:

- **Report Name**—A user-customized or default-supplied name for referencing and clarification. Mandatory for saving, this report will be added to and stored with the existing list under this name.
- **Description**—An optional user-customized (or default-supplied) description for referencing and clarification. Useful when saving the report template.
- Scheduled to run at—The upcoming scheduled running of the report.
- Last Run—The most recent execution of the report.
- Last Error—Errors generated at the time of report creation.

Those templates which have been scheduled, appear with a small clock on the icon (\Box). For these, the **Next Execution** field will display the upcoming run.

For your convenience, preset supplied templates offer one-click creation of the most common DBA reporting tasks. <u>Schedule</u> the templates for their periodic running and distribution of reports.

To create and save templates

- 1. Click the report type of interest from the available set of six report groupings (for example: Throughput).
- 2. Double-click the **Add new report** icon on the right pane. The Reports wizard is displayed.
- 3. Follow the wizard instructions to customize the report as per your needs.
- 4. In the final **Summary** window, enter the mandatory **Report name** identifying the report and optional **Description** for referencing and clarification. Click **Save** to store the template in Performance Analysis.

Viewing template executions

When focused on a specific report template, the display on the right lists the report occurrences in the upper pane, represented by its execution date and time. click an occurrence to preview:

Scheduling

A key component of the report functionality is the ability to schedule periodic report running. Any preset or customized template can be scheduled by means of the **scheduling properties** dialog.

To schedule template executions

Click in the Reports wizard Summary window

-OR-

• Right-click a report of interest in the tree node and select Schedule from the pop-up menu.

Note: For reports to be run as scheduled, the Windows task schedule systems service must be running and the client must be active and have a network connection.

Define the schedule by means of its two composing tabs: Schedule and Delivery Options:

Schedule tab

Divided into two sections, use the upper half to schedule periodicity (daily, weekly, monthly or one-time only) and commencement (Start time and Start date) and the lower half for (first-time) User Information definition.

In the lower half, a valid Windows operating system user ID and password must be specified. This account is used by the Windows Scheduler to perform the actual task of report creation. If no account

information exists, click (or through Display Options

configuration: Scheduler tab) to display the Set Scheduler Account dialog, wherein you supply the valid Windows operating system user ID and password. If yours is a domain account, the user identification must be prefixed with the domain identification and "\". (For example, user John Smith in the PRODUCTION environment might have

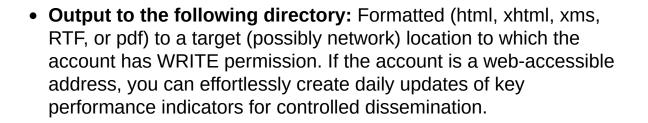
a user identification of: PROD\Jsmith)

Note: Once you enter this data, it updates all existing schedules to that account information and any subsequently created schedules (for this and all other Oracle instances accessed from this client) will also have that same account. If your network user/password is changed, you must reenter the User Information for your schedules to work.

Delivery Options tab

Use these checkboxes to set the report delivery destinations:

• Output to Performance Analysis—In order for you to make use of the Performance Analysis tree node archival and retrieval functionality and to list each report occurrence in the right pane, you must select this checkbox. Only by selecting this checkbox can you preserve customized runnings, like that of the customized template "Weekly Activity", in the following illustration:



 Output by Email to the Following Addresses: Enter here valid email addresses, semi-colon delimited, of intended report recipients. The reports (formatted as above) will be sent as attachments in the specified format, no validity or receipt checks will be performed and the recipients will be unable to reply to the sender. Click Set SMTP Properties to define the SMTP address (Simple Mail Transfer Protocol), the email sender address, and as necessary, the email account information (username and password). Click Send test mail to verify that the SMTP properties are specified correctly.

Executive Workload Summary

The Executive Summary is a one-click report-generating feature designed to deliver the critical <u>Home Page</u> displayed parameters in a useful configurable format to key decision-makers.

To generate a report from the Home Page View

The report is based on the Home Page displayed Time Range. Complete the report definition wizard (**Summary** window) as follows:

- Create a textual:
- Report name An optional user-input for referencing and clarification. Mandatory for saving, this report will be added to and stored with the existing list under this name.
- **Report Description -** An optional user-input for referencing and clarification. Useful when saving the report template.
- **File name** Name of the report file template. A recommended default is suggested.
- Click **Save** to save the template for subsequent executions.
- Use **Schedule** to set the periodic <u>Schedule and Delivery</u> <u>Options</u> for this report's execution and archiving.
- Use **Preferences** to select:
- your <u>totaling</u> method where the report data is to be displayed
- If SQL texts are to be printed with full colors.

Click **Back** to access **Time Range** where you select a **Predefined Time Range** of a system default or a user-defined <u>permanent</u> range; or define one here by means of a **Specific Time Range**. Also, specify the <u>data</u> <u>interval</u> **Resolution**.

Click **Back** again to access **Metrics** where you select from among the potential metric set that will be appended to the report. Each selected metric entry will include: observed behavior, the <u>baseline</u> range minimum and maximum and a graphical display of the deviation

The report may be either: previewed on the window, printed, or stored in Performance Analysis.

To generate a report from the Reports node and mode

Define Metrics and Time Range, as above.

Complete the report definition wizard (**Summary** window), as above.

Activity

The Activity report traces the activity and impact of a certain Oracle entity in a specific time frame. The report is modeled on the Performance Analysis historical diagnostics windows and gives an overview of behavior of a specific Oracle activity over time.

The Activity Reports wizard has four defining windows:

- Dimensions Include in your report template the entire Instance (or Cluster) Level activity or a subset (Specify Activity Filter) of historical (up to three) Dimension data.
- 2. Metrics
- Select Metrics for Resource: Select here the resource on whose components you want to focus. The chosen metric serves as the initial set of composing metrics to be pane listed time unit in each reporting period. (and as the initial choice for the **Display Timeline Graph** metric).
- Select from among the potential metric subset those that will appear in the table for each time unit in the reporting period.
- 3. Time Range Select here a Predefined Time Range of a system default or a user-defined <u>permanent</u> range; or define one here by means of a **Specific Time Range**. Also, specify the <u>data interval</u> **Resolution**.
- 4. Summary
 - Create a textual:
 - **Report Description -** An optional user-input for referencing and clarification. Useful when saving the report template.
 - **Report name -** An optional user-input for referencing and clarification. Mandatory for saving, this report will be added to and stored with the existing list under this name.
 - **File name** Name of the report file template. A recommended default is suggested.
 - Click **Save** to save the template for subsequent executions.

- Use **Preferences** to select
 - your <u>totaling method</u> where the report data is to be displayed
 - If SQL texts are to be printed with full colors.
- Use **Schedule** to set the periodic <u>Schedule and Delivery Options</u> for this report's execution and archiving.

The report may be either: previewed on the window, printed, or stored in Performance Analysis.

Throughput

The Throughput report shows the throughput of a specific resource as a result of the load placed upon the resource by a given Oracle entity. The most common Template defaults: Upper is response time and lower is load (example CPU usage compared with SQL executions) so that variable functions can be compared one to another.

The Throughput Reports wizard has four defining windows:

- Dimensions Include in your report template the entire Instance Level activity or a subset (Specify Activity Filter) of historical (up to three) Dimension data.
- 2. Metrics
- Select Metrics for Resource: Select here the resource on whose components you want to focus. The chosen metric serves as the initial set of composing metrics to be pane listed time unit in each reporting period. (and as the initial choice for Upper Graph Metric).
- Select from among the potential metric subset the Lower Graph Metric.
- Select from among the potential metric subset those that will appear in the table for each time unit in the reporting period.
- 3. Time Range Select here a **Predefined Time Range** of a system default or a user-defined <u>permanent</u> range; or define one here by means of a **Specific Time Range**. Also, specify the <u>data interval</u> **Resolution**.
- 4. Summary
- Create a textual:
 - **Report Description -** An optional user-input for referencing and clarification. Useful when saving the report template.
 - **Report name -** An optional user-input for referencing and clarification. Mandatory for saving, this report will be added to and stored with the existing list under this name.
 - File name Name of the report file template. A recommended

default is suggested.

- Click **Save** to save the template for subsequent executions.
 - Use **Schedule** to set the periodic <u>Schedule and Delivery</u> <u>Options</u>for this report's execution and archiving.
- Use **Preferences** to select:
 - your <u>totaling method</u> where the report data is to be displayed
 - If SQL texts are to be printed with full colors.
- Use **Schedule** to set the periodic <u>Schedule and Delivery</u> <u>Options</u> for this report's execution and archiving.

• The report may be either: previewed on the window, printed, or stored in Performance Analysis.

Tops

The Tops report displays the most offensive Oracle activities (dimensions) for a given time frame, like that of the Performance Analysis' historical tops view.

The Tops Reports wizard has four defining windows:

- Dimensions Include in your report template the entire Instance Level activity or a subset (Specify Activity Filter) of historical (up to three) Dimension data.
- 2. Metrics
 - Select Metrics for Resource: Select here the resource for display as the upper graph. The most resource demanding elements for the specified dimension are displayed and sorted, according to the selected resource. The chosen resource determines which metrics are available for the other presentations in the report.
- Show Top <number> <dimensions>: Determines the number of top dimension activities to be displayed. For example, display top five SQL statements.
- Sorted By:Determines the metric by the top results will be sorted (though not selected as the selection is according to the selected resource). The default is always according to the selected resource.
- Select from among the potential metric subset those that will appear in the table for each time unit in the reporting period.
- 3. Time Range Select here a **Predefined Time Range** of a system default or a user-defined <u>permanent</u> range; or define one here by means of a **Specific Time Range**. Also, specify the <u>data Interval</u> resolution.
- 4. Summary
 - Create a textual:
 - **Report Description -** An optional user-input for referencing and clarification. Useful when saving the report template.
 - Report name An optional user-input for referencing and

clarification. Mandatory for saving, this report will be added to and stored with the existing list under this name.

- File name Name of the report file template. A recommended default is suggested.
- Click **Save** to save the template for subsequent executions.
- Use **Preferences** to select
 - your <u>totaling</u> method where the report data is to be displayed
 - if you want to expand the dimension column width or replace the dimensions value with the legend (useful when dimension values are long, as in SQL statements)
 - if SQL texts are to be printed with full colors.
- Use **Schedule** to set the periodic <u>Schedule and Delivery Options</u> for this report's execution and archiving.

The report may be either: previewed on the window, printed, or stored in Performance Analysis.

Trend

The Trend report displays top aggregated dimension and metric resource usage over time.

The Trend Reports wizard has four defining windows:

- Dimensions Include in your report template the entire Instance Level activity or a subset (Specify Activity Filter) of historical (up to three) Dimension data.
- 2. Metrics
 - Select Metrics for Resource: Select here the resource on whose components you want to focus. The chosen metric serves as the initial set of composing metrics to be pane listed time unit in each reporting period.
- Show Top <number> <dimensions>: Determines the number of top dimension activities to be displayed. For example, display top five SQL statements.
- **Sorted By**: Determines the metric by which the top results will be sorted (though not selected as the selection is according to the selected resource). The default is always according to the selected resource.
- Select from among the potential metric subset those that will appear in the table for each time unit in the reporting period.
- 3. Time Range Select here a **Predefined Time Range** of a system default or a user-defined <u>permanent</u> range; or define one here by means of a **Specific Time Range**. Also, specify the <u>data interval</u> **Resolution**.
- 4. Summary
- Create a textual:
 - **Report Description -** An optional user-input for referencing and clarification. Useful when saving the report template.
 - **Report name -** An optional user-input for referencing and clarification. Mandatory for saving, this report will be added to

and stored with the existing list under this name.

- File name Name of the report file template. A recommended default is suggested.
- Click **Save** to save the template for subsequent executions.
- Use **Preferences** to select:
 - your totaling method where the report data is to be displayed
 - your graph style, representing the time period graphs as either stacked or side-by-side entries
 - if you want to expand the dimension column width or replace the dimensions value with the legend (useful when dimension values are long, as in SQL statements)
 - if SQL texts are to be printed with full colors.
- Use **Schedule** to set the periodic <u>Schedule and Delivery Options</u> for this report's execution and archiving.

The report may be either: previewed on the window, printed, or stored in Performance Analysis.

Time Breakdown

The Time breakdown report displays the top activities in each customizable time frame within a given period. The Time Breakdown Reports wizard has four defining windows:

- Dimensions Include in your report template the entire Instance Level activity or a subset (Specify Activity Filter) of historical (up to three) Dimension data.
- 2. Metrics
- Select Metrics for Resource: Select here the resource on whose components you want to focus. The chosen metric serves as the initial set of composing metrics to be pane listed time unit in each reporting period.
- Show Top <number> <dimensions>: Determines the number of top dimension activities to be displayed. For example, display top five SQL statements.
- **Sorted By**: Determines the metric by which the top results will be sorted (though not selected as the selection is according to the selected resource). The default is always according to the selected resource.
- Select from among the potential metric subset those that will appear in the table for each time unit in the reporting period.
- 3. Time Range Select here a **Predefined Time Range** of a system default or a user-defined <u>permanent</u> range; or define one here by means of a **Specific Time Range**. Also, specify the <u>data interval</u> **Resolution**.
- 4. Summary
 - Create a textual:
 - **Report Description -** An optional user-input for referencing and clarification. Useful when saving the report template.
 - **Report name** An optional user-input for referencing and clarification. Mandatory for saving, this report will be added to and stored with the existing list under this name.

- File name Name of the report file template. A recommended default is suggested.
- Click **Save** to save the template for subsequent executions.
- Use **Preferences** to select:
 - your totaling method where the report data is to be displayed
 - if you want to expand the dimension column width or replace the dimensions value with the legend (useful when dimension values are long, as in SQL statements)
 - if SQL texts are to be printed with full colors.
- Use **Schedule** to set the periodic <u>Schedule and Delivery Options</u> for this report's execution and archiving.

The report may be either: previewed on the window, printed, or stored in Performance Analysis.

Change Tracking Report

For instances with an installed repository, change tracking reports display changes in environments and activity that can potentially influence system performance and enables the user to view correlation between occurrences of changes and Oracle's activity and behavior patterns. The Change Tracking Reports wizard has three defining windows:

- 1. Change Types Specify here the change tracking events to be included in the report
- 2. Time Range Select here a **Predefined Time Range** of a system default or a user-defined <u>permanent</u> range; or define one here by means of a **Specific Time Range**. Also, specify the <u>data interval</u> **Resolution**.
- 3. Summary
 - Create a textual:
 - **Report Description -** An optional user-input for referencing and clarification. Useful when saving the report template.
 - **Report name -** An optional user-input for referencing and clarification. Mandatory for saving, this report will be added to and stored with the existing list under this name.
 - **File name** Name of the report file template. A recommended default is suggested.
 - Click **Save** to save the template for subsequent executions.
 - Use **Schedule** to set the periodic <u>Schedule and Delivery</u> <u>Options</u> for this report's execution and archiving.
 - Use **Preferences** to select:
 - your <u>totaling</u> method where the report data is to be displayed
 - If SQL texts are to be printed with full colors.

The report may be either: previewed on the window, printed, or stored in Performance Analysis.

Advisories Report

Access the **Publish Advisories** window from the **Home** mode printer icon (
.

Use this functionality to create PDF reports consisting of either the most severe recent episodes or the most recent **Database Performance**: **Reduce Contention** and **Optimize Physical Disk I/O** analyses.

Customize the output as to:

- File Name and location
- Layout Paper size and orientation
- Advisory Selection Either the most severe recent episodes or the most recent Database Performance: Reduce Contention and Optimize Physical Disk I/O analyses.
- **Post-Publish Actions** View and/or Print the created PDF file.

Related Topics

Advisories Listing Performance Advisories Performance Analysis Console Performance Analysis Configuration - Home

StealthCollect Agent

Click **Agent Options** (⁽⁾) to display the **StealthCollect Agent Status** tab

-OR-

From start | Programs | Quest Software | Performance Analysis | Agent Administration | Agent Administration

The **Instance Information** (SID and Oracle version) and overall status indicator are displayed above the **Agents Information** tabs.

Use **Shutdown StealthCollect** to shutdown all agent processes on the server. The processes will be automatically restarted upon any subsequent connection attempt.

Collector tab

The <u>Collector</u> process attaches to Oracle's shared memory and directly reads Oracle operational data. Use the **Status** indicator to verify that the collector process is active.

Attribute	Description
Host	The monitored host on which the collector is installed.
Operating System	Operating system type and version of the monitored host.
Location	The server path where the agent binaries are installed.
Agent Version	The StealthCollect agent version.
	Running Error
Status	Indicates the collector status. (Normally Running). In case of an

	Error status, click Error to display the full error text and status.
Running as group	Indicates the process run-time OS privileges.

Middleware tab

The StealthCollect <u>Middleware</u> is responsible for aggregating and storing all the performance data for both real-time and short-term historical queries. Use the **Status** indicator to verify that the middleware process is active.

• If historical collection is configured for this instance, use **History Collection** to stop/start historical data collection.

Attribute	Description		
Host	The monitored host on which the middleware is installed.		
Operating System	Operating system type and version of the monitored host.		
Location	The server path where the agent binaries are installed.		
Agent Version	The StealthCollect Agent version.		
Status	Indicates the middleware status. (Normally Running). In case of an Error status, click Error to display the full error text and status.		
	Not installed - Historical Collection has not been configured for this instance.		
History	Started - Historical collection is active.		
Collection	Stopped - Historical collection is		

paused.
Error - Historical is paused due to a
problem Click Error to display the

problem. Click **Error** to display the full error text and status.

Performance Repository

The <u>Repository Manager</u> is responsible for populating the long-term Performance Repository and change tracking. Use the status indicators to verify that the data collections, change tracking and advisories status are active.

- If the long-term repository is configured for this instance, use **Stop Repository Manager** to Stop/Start repository loading.
- Click **Load History Collection Now** to initiate the history collection process.
- Click **Track Changes Now** to override the change tracking schedule, and immediately perform the change tracking.

Attribute	Description
Host	The monitored host on which the middleware is installed.
Instance	The specified instance on the monitored host.
Database Schema Version	The internal repository schema version.
	Not installed - The long-term historical collection has not been configured for this instance.
	Idle - Running but currently inactive and waiting for next scheduled execution.
Long-term History Collection	OK - The long-term historical collection (repository loading) is

	active.	
	Error - The long-term historical collection (repository loading) is paused due to a problem. Click Error to display the full error text and status.	
Change Tracking	Not installed - Change tracking has not been configured for this instance.	
	Idle - Running but currently inactive and waiting for next scheduled execution.	
	OK - Change tracking is active.	
	Error - The change tracking feature is paused due to a problem. Click Error to display the full error text and status.	
If the reposito	ory was not created with the initial ager	nt setup, vou r

If the repository was not created with the initial agent setup, you may <u>add</u> <u>it</u> by rerunning the initial Performance Analysis for Oracle Installation wizard.

History Collection

StealthCollect rapidly collects and aggregates performance data for historical analysis. It lets you configure the collection and aggregation in accordance with your monitoring and analysis needs. Configure an existing historical collection, using the following four displayed tabs:

Caution: Historical collection customization should be performed only by advanced users. Misconfiguration may damage the quality of your historical analysis.

Oracle Dimensions

Select/clear the <u>dimension</u> checkboxes as per your desired collection set.

Collection Settings tab

Specify in this tab's SQL collection settings:

- The value for the collected dimension setting has a greatest impact on top level dimension comprehensiveness (and therefore accuracy). The item combinations setting more directly affects second and third level dimension drilldowns. The overall tradeoff is between the number (depth) of collected occurrences and impact on space.
- If the **Remove literals in SQL** is checked, the agent replaces constants in SQLs with bind variables, so that statements that differ only in constants are treated as the same. Activating this option is useful where bind variables are not used.
- The **Ignore statements** checkbox lets you ignore statements active for less than a designated threshold period.

Time Pyramid tab

All layers must be specified with a positive value range and every layer should be retained for a minimum of twice the next layer resolution (for example one day must be kept for at least two weeks; one minute must be kept for at least 30 minutes). The purpose of the time pyramid tab is to give an optimized storage and performance and rapid access to the data of interest. The logic of this model is based on the premise that the closer the data is to the present, the more granular (that is, measured in finer time units) you want it to be. Conversely, the further it is from the present, the less refined the desired time unit of measurement, illustrated as follows:

Placing the cursor focus on a Resolution line lets you change the storage duration in the sliding window box below and view the impact on the respective middleware and repository space layers. You can also view the impact on the overall **Estimated** storage periods.

Middleware Disk Space

Short term historical data is stored in files (in the middleware host). The estimated required entire space is Allocated (designated) at the time of installation. The data directory (located on the middleware host) contains the short-term history log files. This space is allocated during installation and therefore the directory should have sufficient free space to hold the indicated amount of required data. The **Free** space is the available unused space in the specified directory. If the free space is less than the allocated, it appears in red and you must either select a less demanding history model (or manually customize) or find a sufficient data directory location.

Performance Repository Space

(For long-term installations with Performance Repository.) These tablespaces constitute the long-term historical data. Estimated size for the repository schema is the size of the tables or indexes (respectively data or index tablespaces) required to host the entire defined historical data. As the Performance repository can house historical data for several years, you are not required to allocate the entire free space at installation.

The **Allocated** space determines the already allocated schema size, expanding in accordance with repository size growth. If the Free space is less than that needed to allow the schema to expand to that of the total Estimated size, it appears in orange, indicating that more space should be added to the tablespaces as historical data is added and the schema expands. (You may proceed without adjustments, though you will have to add additional space in the future.)

Space Allocation Override

Use this feature to manually override the system-determined **Allocated** middleware space requirements (to a maximum of 20 GB).

You can also specify an alternative **Data directory.**

To select the data directory location

- 1. Either accept the provided default, manually enter a data directory, or use the browse () optionUntil sufficient space is found for the data directory, you are prevented from continuing with the installation.
- 2. Click Next.

User Defined Collections (UDCs)

<u>User Defined Collections</u> are a feature by which Oracle resource consumption can be analyzed over selected domains and filters across different time periods. This tab provides the following options:

- New... Launch the User Defined Collections creation wizard.
- Stop/Start Pause/Recommence data collection for the User Defined Collections on the selected collection.
- Delete Delete the selected collection.
- Clear Erase the selected collection contents.

To view and configure existing User Defined Collections

• Double click the User Defined Collections pane listing of interest to display the defining set-up tabs:

Identification tab

Displays the UDC name (read only) and description.

Oracle Dimensions tab

(For viewing only) Displays those <u>dimensions</u> which are enabled in this collection.

Activity Filter tab

(For viewing only) - <u>Activity filters</u> are a means to focus on and limit collections of occurrences of activities of interest. Activities are defined by their corresponding dimension values.

Collection Settings tab

Fine tune SQL <u>Collection Settings</u> and Wait and Lock event granularity rules, and rule exceptions.

Time Pyramid tab

Optimize storage and performance and gain rapid access to the data of interest through the <u>Time Pyramid</u> settings.

Scheduling tab

<u>Scheduling</u> (for viewing only) - Displays the exact date and time at which the collection takes place.

Space Allocation Override tab

Use this option to manually <u>override</u> the system-determined UDC disk space requirement by specifying an alternative disk allocation.

Agent Options

The StealthCollect collector process is responsible for sampling both the memory and the underlying operating system. The light-weight sampling combined with the high-frequency capacity is the StealthCollect technological core. The sampling frequency has a tradeoff of data accuracy and application overhead. Higher sampling frequency yields more accurate data, at a cost of increased overhead on the monitored host.

To control the memory/OS sampling frequency

The SGA Sampling Rate is modified as follows:

- **Fixed** A Fixed number of samples are performed each second, regardless of the overhead imposed on the server.
- Automatic The agent automatically adjusts the sampling frequency such that the imposed overhead does not exceed the given threshold (as measured in % Server CPU usage); Yet this limitation itself does not lower the sampling frequency below that specified in the second criteria of the minimal acceptable (as measured in

samples per second). Clicking on either button lets you define each criteria with a more detailed resolution, consisting of the memory sampling and OS sampling components.

The StealthCollect agent features a low-rate **Dictionary Sampling** for supplementary data. Adjust the collection and refresh sampling frequency (measured in hours), as desired. Click **Refresh Now** to initiate an immediate collection of dictionary information

Performance Repository

Identification

The <u>Repository Manager</u> is responsible for populating the long-term Performance Repository and change tracking. Use the status indicators to verify that the data collections and change tracking are active.

Attribute	Description
Host	The monitored host on which the repository is installed.
Instance	The specified instance on the monitored host.
Port	The TCP/IP port through which the repository instance is monitored.
Version	The StealthCollect Repository Manager version.
Oracle Repository account	Username: The user under which the repository was accessed.

Change Tracking

The change tracking tool is an integrated monitoring feature available only if the repository is installed. It periodically tracks changes in environments and activity that can potentially influence system performance and enables the user to view correlation between occurrences of changes and Oracle's activity and behavior patterns.

The change tracking process is performed within the StealthCollect Repository manager.

To enable/disable Change Tracking

• Select/clear the **Enable change tracking** checkbox to enable/disable <u>change tracking</u>.

To set the retention period for change tracking data

• Set the period for which discovered change occurrences are stored in the Performance Repository by specifying the expiration period in the **Expire changes after** box (measured in days). Change items dated beyond the threshold are automatically aged out.

To configure the tracked change types

The inclusiveness of the tracked <u>change types</u> determines the monitored Oracle instance overhead and repository space requirements. Therefore, track only those change types which are relevant to your system.

• Select/clear the change types from their associated checkboxes for the inclusion/exclusion of the change type.

To schedule the Change Tracking

Change tracking is a periodic process executed at specified intervals.

- 1. Specify which days of the week, which exact time (hh:mm:ss) within these days the change tracking process is to be executed. By default, the process is executed daily at midnight.
- 2. Enable multiple daily change tracking in the **times per day** box.

Maintenance

The Performance Repository is designed to hold a large volume of data. The Performance Repository has a built-in feature for analyzing its objects and rebuilding its indexes to ensure its robustness and performance quality. The maintenance procedures automatically are executed in the background in accordance with your configuration.

To customize Repository maintenance procedure

- 1. Select/clear the **Analyze tables** checkbox for inclusion/exclusion of this repository maintenance procedure.
- 2. Select/clear the **Index rebuild** checkbox for inclusion/exclusion of this repository maintenance procedure.
- 3. Set the **performance statistics** and **Change Analysis** loading time

(in seconds) thresholds which when exceeded trigger performance of the repository maintenance procedures.

- 4. Specify the **minimal interval** (in hours) between each execution of maintenance procedures.
- 5. Click Perform Now.

Instance List

The instances (and their hosts) served by the specified performance repository are listed here, in addition to the repository space allocation totals.

Advisories

<u>Performance Advisories</u> provide explicit and clear description of performance degradations affecting the database instance. Developed by Oracle performance tuning experts, these advisories:

- Detect and diagnose performance degradations and provide paths to their resolution
- Incorporate wait event analysis to focus only on significant issues affecting database performance
- Use fuzzy evaluation techniques to combine relevant metrics and convey appropriate priority of each issue relative to the database workload in a given period and to other problems evident within the same period
- Provide clear and detailed explanation of performance degradations offering just-in-time education, extensive background tuning information, and knowledge to assist DBAs with solving the problem at hand and avoid similar problems in the future

Advisories are presented in either a default ordered and categorized tuning plan or as an <u>action plan</u> that lists first the issues impacting database performance the most.

General Instance Profile Advisories Best Practices E-Business

General

The advisories status denotes the advisories condition, as follows:

- OK
- Init(ializing)
- Idle
- Working
- Stopped
- In Progress
- Error

If the advisories are configured for this instance, use **Pause/Resume** to Stop/Start the Performance Advisory feature

In the **15 Minutes Collection** choose between the 15 minute and 1 hour collection frequencies based on tradeoff between timeliness requirements and space considerations

Instance Profile

The database instance profile settings define critical system hardware and software components, as follows:

- Available RAM in MB
- Transaction Type
- Datafile disk devices
- Datafile RAID status
- Dedicated redo devices
- <u>Alternated redo devices</u>
- Dedicated archive devices

Available RAM in MB

Specify the amount of physical memory (RAM) available for use by this database on the host computer. Only specify the amount of memory designated for use by this database taking into consideration memory used by the operating system and other application software hosted on this computer. Include memory available for both Oracle's SGA and user/background processes

To view available memory on a Windows NT/2000/XP computer:

- 1. Press Ctrl-Alt-Delete. The Windows Security window is displayed. Click **Task Manager**.
- 2. Select the Performance tab. Physical memory is listed in the Physical Memory group. Memory is displayed in kilobytes. Kilobytes can be converted to megabytes by dividing by 1024.

To view available memory on a UNIX computer, consult the operating system documentation for instructions or the system administrator.

Transaction Type

Database transaction type can be generally separated into the following types:

- On-line Transaction Processing (OLTP) These applications involve many users who are updating, inserting, and running numerous quick select statements. These statements access data through an index and return single row results or small result sets. The SQL statement response time is often sub-second. Examples of these kinds of applications are rental car reservation systems, order entry applications, airline reservation systems, and banking applications.
- Decision Support System (DSS) Decision support systems typically run adhoc or report generation SQL to mine data for trends, produce summary data, and perform general analysis of data. These types of queries can often take seconds or minutes to run. DSS queries don't always use indexes. When a full data set must be scanned to perform calculations, full table scans occur. Examples of DSS systems are those that look for market trends or monthly/quarterly/annual reporting applications.
- **Mixed or Hybrid** Mixed or hybrid systems are used both for online transaction processing and decision support. Tuning performance of these systems is more difficult because the two usage types often benefit from opposite tuning approaches. For example, to improve screen refresh response time, OLTP applications benefit from tuning actions that quickly return a query's first rows at the expense of response time for the total result set. On the other hand, DSS applications, in trying to maximize throughput, benefit from tuning actions that minimize time to return the entire result set at the expense of response time to return the query's first few rows.

Datafile disk devices

To find the number of disks in Windows:

- 1. Open the Control Panel.
- 2. Double-click Administrative Tools, then double-click Computer Management.
- 3. Expand the Storage node and select Disk Management.
- 4. The disk device information for the machine is displayed.

To find the number of disks in UNIX:

On UNIX, the underlying disk devices can be seen by using the following commands:

- mount
- df
- iostat

These commands only give partial information about the disk devices available on the system. For example, the command mount does not list disks that are used as raw devices.

The more physical disks that are available to store datafiles, the more efficiently datafile access can be configured to avoid contention and I/O bottlenecks.

Understanding I/O problems

The performance of many software applications is inherently limited by disk input/output (I/O). Often, CPU activity must be suspended while I/O activity completes. Such an application is said to be I/O bound. Placing datafiles on multiple disk devices decreases the chance for I/O contention.

What is disk contention?

Disk contention occurs when multiple processes try to access the same disk simultaneously. Disks have limits on both the number of accesses and the amount of data they can transfer per second. When these limits are exceeded, processes will wait to access the disk.

Datafile RAID status

Is RAID striping in use?

RAID (Redundant Array of Independent Disks) arrays are a popular way of providing fault tolerant, high performance disk configurations. There are a number of RAID levels available. This question seeks to establish whether a RAID configuration is in use, and if so, does the RAID level implement striping. Striping is a proven method for increasing I/O performance.

There are three common levels of RAID provided by storage vendors. Other levels typically implement differing configurations of mirroring, striping, and parity.

RAID 0

RAID 0 implements disk striping. In striped configurations, a logical disk is constructed from multiple physical disks. The data contained on the logical disk is spread evenly across the physical disks and hence random I/Os are also likely to be spread evenly. There is no redundancy built into this configuration. When a disk fails it need to be recovered from a backup.

RAID 1

RAID 1 implements disk mirroring. In mirrored configurations, a logical disk is comprised of at least two physical disks, each holding the exact same data as the others. In the event that one physical disk fails, processing can continue using the other physical disk. Writes are processed in parallel so there should be no degradation of write performance. Multiple disks are available for reads so there can be an improvement in read performance.

RAID 5

In RAID 5, a logical disk is comprised of multiple physical disks. Data is arranged across the physical devices in a similar way to disk striping (RAID 0). However, a certain proportion of the data on the physical devices is parity data. This parity data contains enough information to derive data on other disks should a single physical device fail.

RAID 0+1

It's common to combine RAID 0 and RAID 1. Such striped and mirrored configurations offer protection against hardware failure together with the

spread of I/O load.

RAID levels 2, 3, 4, 6, 7, 10, and 53 are other RAID configurations that implement some form of striping.

Performance implications of RAID

Both RAID 0 and RAID 5 improve the performance of concurrent random reads by spreading the load across multiple devices. However, RAID 5 tends to degrade write I/O since both the source block and the parity block must be read and then updated.

Dedicated redo devices

Are the redo log files located on a disk device dedicated only to the redo log files?

To view the redo log file locations on disk, run the following sql query as system, sys or a sysdba:

GROUP#	<pre>* from v\$logfile STATUS MEMBER \ \\ \ \ \ \</pre>
	-
1	
-	D:\ORACLE\ORADATA\XXXX\RED004.LOG
2	
	D:\ORACLE\ORADATA\XXXX\RED003.LOG
3	
	D:\ORACLE\ORADATA\XXXX\RED002.LOG

The log writer (LGWR) process writes sequentially to the redo log files. LGWR writes all change vectors to the redo log files. A change vector is a physical representation of the data changed in an Oracle data block. The actual data blocks changed by an update, insert, or delete are written out to disk sometime before or immediately after the user runs a commit command. When a user commits, the redo logs on the disk must contain a representation of all the changes they made in the database. This guarantees that the database can be recovered when there is a database crash. A crash can cause loss of data stored in memory, such as changed data blocks that have not yet been written to disk. After every commit, the user waits for LGWR to respond with a confirmation that the change vectors were written to the log files. The speed at which the LGWR can write to the redo log files has a direct impact on user response time when data changes are made.

LGWR writes most efficiently to the redo log files when LGWR is the only process writing to the disk. Contention occurs when the disk is used for other file I/O while the LGWR is writing to the redo logs. Contention causes slower redo log writes and thus causes slower response time in completing a user commit.

The fastest redo log file write times are achieved on raw devices.

Alternated redo devices

For databases running in archive log mode.

Are the redo log files in the database set up on different disks?

The archiver process (ARCH) should be reading from a different disk than that which is written to by the log writer process (LGWR).

The redo log files can be queried from the view **v\$logfile** as in:

```
SQL>select * from v$logfile
GROUP# STATUS MEMBER
--
1
D:\ORACLE\ORADATA\XXXX\RED004.LOG
2
D:\ORACLE\ORADATA\XXXX\RED003.LOG
3D:\ORACLE\ORADATA\XXXX\RED002.LOG
```

From this view you can see whether the log files are on different partitions. Different partitions could be on the same disk or disk device. To determine whether different partitions are on the same device, you will need more information about the underlying disk device system.

The archive log file location can be determined by using the following command:

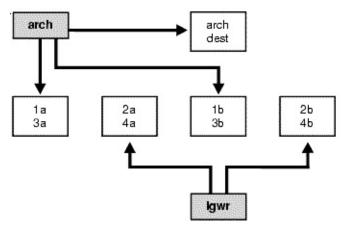
SQL>archive log list	
Database log mode	Archivelog
Automatic archival	Enabled
Archive destination	D:\ORACLE\arch
Oldest online log sequence	188082
Current log sequence	188084

The information above shows that all of the redo logs and the archive log are on the same device. This configuration could cause contention.

What are "alternated redo devices"?

An Oracle database has a minimum of 2 redo log files. It is recommended that these two files be on separate disks. When there are more than two redo log files, the 3rd redo log file can be put on the same disk as the first log file and the 4th log file on the same disk as the 2nd log file. This is called **alternating redo log files**.

The only process writing to a redo log file is the log writer process, known as LGWR. LGWR only writes to one of the redo log files at a time. Thus LGWR writes efficiently to the redo log files whether they are on one disk or separate disks*. LGWR writes to the available redo log file using the round robin method. LGWR writes to one redo log file until filled and then switches to the next redo log file, eventually coming back to the first redo log file.



Contention can occur in the redo log archive process (ARCH) if the archiver process reads from a log file located on the same disk as one

currently being written by the log writer process. ARCH reads the redo log files after LGWR has filled them. After filling a redo log file, the LGWR switches to the next redo log file. When this next redo log file is on a separate disk from the last redo log file, no contention occurs (as long as the archiver finishes reading the log file before the log writter switches back to a log file residing on the same disk).

Higher archiver throughput can be achieved by implementing larger numbers of dedicated redo devices and alternating the redo logs in round robin fashion. The greater the number of redo log devices, the higher possible throughput from multiple archiver processes. In general, a two disk configuration for alternating log files should be sufficient.

*In some less common cases, using multiple disks for a redo log file by using striping can be beneficial.

Dedicated archive devices

To determine the devices to which the archive process is writing, run the command archive log list in sqlplus (or svrmgr) as internal or sysdba.

Example:

sql>connect internal	
connected.	
sql>archive log list	
Database log mode	Archivelog
Automatic archival	Enabled
Archive destination	
D:\Oracle\Ora81\RDB	MS
Oldest online log sequence	3
Current log sequence	6

Compare 'Archive destination' with the other files in the database. The other files can be found with the following sql commands:

select name from v\$datafile; select name from v\$tempfile; select member from v\$logfile;

```
select value from v$parameter where
name='control_files';
```

Do any other database files use the same disk device? Do any other applications other than the database use the same disk device?

The archiver process (ARCH) backs up redo log files after they have been filled by the log writer process (LGWR). When ARCH is unable to keep up with the amount of redo generated by LGWR, contention can occur when LGWR starts writing to a new redo log located on a disk currently being read by the archiver process. To quickly complete the archive process and avoid contention, the archiver process should write archived redo log files to a disk that is only used for this purpose.

Advisories

Disable collection of advisories whose monitoring value is insignificant.

Best Practices

Disable/Enable the Best Practice feature.

Schedule the days on which collection of best practices data is to occur.

E-Business

Disable/Enable the E-Business feature.

Schedule the days on which collection of e-business data is to occur.

Home Page

The Performance Analysis <u>Home</u> mode (page) is the both the primary <u>system health measurement</u> functionality and the starting point for drilldown analysis through its user-configurable: alarming, normative baseline, advisory and applied knowledgebase toolsets. Its digital dashboard at-a-glance comprehensiveness facilitates optimal user monitoring and tuning effort by intuitively focusing administrative attention on instance and system aspects most in need of attention.

Designate here the <u>resource</u> consumption thresholds for insignificant (green) and critical (red) levels relative to the total resource availability. Levels below the insignificant threshold or above the critical threshold are color-indicated.

IntelliProfile (Baseline)

The baseline algorithm parses sampled data into pre-defined time units (for example, day of the week) and distinct episodes (for example, systematically recurring spikes in CPU consumption). These episodes form the basis on which a normative distribution of instance behavior is projected, allowing for both exception monitoring and performance diagnostics. In this manner, the dynamically-created statistically-expected range of resource consumption for both overall workload and individual metrics lend themselves to configurable high and low metric values. The IntelliProfile engine learns the instance "signature" behavior characteristics via sophisticated algorithms to identify the various cyclical and periodic behavior and to neutralize and minimize the random noise distortion (for example, an employee absent due to illness). Its adaptive element grows "smarter" and more accurate with increases in the sampling period. Armed with this knowledge, IntelliProfile can project the acceptable/normal ranges of activity per each point in time.

General

Status - Enable/Disable the baseline functionality. When the adaptive engine is disabled, baseline data collection stops until re-enabled, but data collected until that moment remains available for the algorithm to process

IntelliProfile

Intelliprofile is a "smart" functionality which continually learns the specific instance "signature" (and in the process builds the baseline) either automatically (recommended) or via a specified sample.

The baseline advisory calculating algorithm parses sampled data into defined (for example, day of the week) episodic units. These units form the basis on which a normative distribution of temporal behavior is projected. Specify here whether the baseline ranges are to be derived from the entire period of collected data, or from a sample week, or a setof-days subset.

Excluded Periods

Intelliprofile can exclude abnormal periods whose data would distort calculation. Use the listing to specify periods to be excluded.

The Intelliprofile learning process is, by default, scheduled to occur in the early hours of the morning. As the specific instance "signature" understanding grows over time, the re-learning process is scheduled to occur at increasing intervals.

Related Topics
Baseline
Baseline
Baseline
Baseline

Support

To access the Support Tools Utility

Agent Options | Support

-OR-

start | Programs | Quest Software | Performance Analysis | Agent Administration | Utilities

The Quest agent Support Tools dialog offers utilities to configure and manage the agent settings and a means to generate a support bundle package of files required by Quest Support for diagnosing and troubleshooting your environment.

User Manager

This utility is designed to allow users with the Administrator role to grant or deny users access to the agent and change passwords or roles. Other users are limited to changing their own passwords.

For security purposes, Performance Analysis enables Administrators to define and maintain a list of users authorized to use the client software to access each Oracle instance. When a user attempts to use the client software to access a specific Oracle instance, the user is prompted to specify a login name and password that appears on the list of authorized users for the Oracle instance.

Verify Accounts

This utility verifies that the Oracle and agent accounts credentials supplied at the Performance Analysis (StealthCollect) agent installation stage is valid. This tool is required in environments in which passwords are periodically updated.

Support Bundle

StealthCollect has a built-in support utility that retrieves troubleshooting

information from the system if you encounter a problem with it. When turning to support, you will be requested to generate and send a support bundle, through either of the following methods.

To fully service troubleshooting and diagnosis requests, use this utility to generate a package containing the requisite files for forwarding onto Quest support. In the displayed **Support Bundle** tab, select the SGA Monitor (Collector), Middleware, and Performance Analysis (GUI) checkboxes as warranted and navigate to the directory in which the bundle is to be stored.

Adding the Performance Repository

The value and power of Performance Analysis is enhanced when configured with a performance repository. If the repository was not created with the initial agent setup, you may add it by reconfiguring the Performance Analysis agent.

To subsequently add the Long Term Performance Repository

- 1. Launch the **Performance Analysis Agent for Oracle Installer** (available from the Performance Analysis Local Server Installation <u>website</u>.).
- 2. On the **Introduction** window, select the **Manage Agent Components** option.
- 3. On the subsequent Manage Agent Components window, select the **Install repository support for an existing agent installation without a repository** option.
- 4. Continue through the remaining wizard windows to complete the performance repository configuration.

Related Topics Performance Repository

Starting and Stopping the Agent

The quest <u>launcher</u> is the Performance Analysis agent component used to connect to the other quest processes on the Oracle server host or on the midde-tier host.

If the quest_launcher is not running on the Oracle server host, Performance Analysis will not be able to display the instance activity and will display an error message when trying to connect to the Performance Analysis agent.

In order to run the Quest launcher, perform the following steps:

•Verify that the connection to the agent services is operative (for example, that no firewall inhibits client-agent communication), by running the following command from the client PC command line:

quest_launcher.exe -command -status -host
<host> -port <port>

Output will be either unsuccessful (if agent is down, or if there is a firewall in between):

ERROR: ERR_CONNECTION_REFUSED - Connect to launcher daemon failed

or successful, as per the following sample:

PRODUCT	VERSION	PROCESS	
	INSTANCE	STATUS PID	
AgentManager	2.0	guard	
	common	RUNNING	
214926			
AgentManager	2.0	installer	
CO	mmon	READY	
AgentManager	2.0	guardcmd	
C	ommon	READY	
StealthCollect	3.0	quest_sc_collecto	r
morpheus-OR	A817 RUNNIN	G 31516	
StealthCollect	3.0	quest_sc_mw	
morp	heus-ORA817	RUNNING 58406	
StealthCollect	3.0	quest_sc_rep	

morpheus-ORA817 RUNNING 215068 StealthCollect 3.0 quest_sc_mw_g morpheus-ORA817 RUNNING 18392 StealthCollect quest_sc_rep_g 3.0 morpheus-ORA817 RUNNING 211088 StealthCollect 3.0 quest_sc_os_monitor common READY StealthCollect guest_sc_admin 3.0 READY common

•On a Unix/Linux platforms

• To verify the agent status, run the quest_launcher from the client as follows:

quest_launcher.exe -command -host <host> -port <port> -user quest
-pass quest -status

Output will be either unsuccessful (if agent is down): ERROR: ERR_CONNECTION_REFUSED - Connect to launcher daemon failed

or successful, as per the following sample:

PRODUCT	VERS	ION PROCESS	5 IN	STANCE	STATUS	PID
AgentManager	2.0	guard	common	RUNNIN	G 2384	
AgentManager	2.0	installer-0	common	RUNNI	IG (NOCONI	F) 192
AgentManager	2.0	installer	common	READY		
AgentManager	2.0	guardcmd	common	READ	Y	
SC_MSSQL	1.0	quest_sc_os_m	ionitor com	mon RE	EADY	
SC_MSSQL	1.0	quest_sc_admi	n commo	on REA	DY	

- To start the launcher process: <Agent installation directory>/bin/quest_launcher_daemon start (For starting the launcher process at system boot, see here.)
- To stop (kill) the launcher process: <Agent installation directory>/bin/quest_launcher_daemon stop

On Windows platforms

- To start, stop or check the agent status:
 - 1. Open the **Services** window as follows: from within **start** menu | select **Run**, enter: **services.msc**.

- 2. From within the displayed Services window, navigate to the installed agent QAM Launcher port <port>.
- 3. Select either **Start** or **Stop**, as appropriate. The service Status is displayed in the status column, as follows:

Action <u>V</u> iew		😫] 🕨			· · · · ·	
Tree	Name 🔺	Description	Status	Startup Type	Log On As	
Services (Local)	NET Runtime Optim	Microsoft		Manual	Local System	
ofte management of the second s	🆓 Alerter	Notifies sel		Disabled	Local Service	
	Application Layer G	Provides s	Started	Manual	Local Service	
	Application Manage	Provides s		Manual	Local System	
	ASP.NET State Serv	Provides s		Manual	Network S	
	Automatic Updates	Enables th	Started	Automatic	Local System	
	Background Intellig	Transfers f	Started	Automatic	Local System	
	🖏 ClipBook	Enables Cli		Disabled	Local System	
	COM+ Event System	Supports S	Started	Manual	Local System	
	QAM Launcher 3399			Manual	LocalSystem	
	QAM Launcher 3566	Start	Ŋ	Manual	LocalSystem	
	QAM Launcher 3567	Stop		Manual	LocalSystem	
	Windows Image Ac.	Pause	tarted	Automatic	LocalSystem	
	🦓 Windows Installer 🕛	Resume		Manual	LocalSystem	
	Windows Managem	Restart		Manual	LocalSystem	
	🦓 Windows Managem / —			Manual	LocalSystem	1
Start service QAM Launcher 3566 on Local Computer		All Tasks				
		Refresh				
		Properties				
		Help				

Display Options

Click **Display Options** (**D**) to display the **<u>Home</u>** tab.

This utility lets you configure the display and behavior of Performance Analysis. Configurations defined here affect only your client computer.

All changes made in this utility influence only display and not collection of data, which are principally defined in the <u>StealthCollect administration</u>.

See also:

Home Real-Time History General Change Tracking Compare Performance Highlights Activity Highlights Scheduler Baseline Support

Home

Use the **Home** configuration tab to set the default time frame display length and to enable display of only advisory exceptions, or of all advisory category metrics.

- **Default time frame** Determines the period (Last 5 minutes, 15 minutes, 1 hour, 6 hours, 12 hours or 24 hours) which is to be displayed upon initial client connection to the monitored instance.
- **Performance Advisories** Grouped into performance categories, the <u>advisory metrics</u> collectively summarize the database instance health.
- **Default Selection**—Specify whether the default display is to be the <u>Action Plan</u> (a prioritized listing of all advisories which may improve the performance of your database) or <u>Database</u> <u>Performance</u> (a 26-advisory report which provides explicit and clear description of performance degradations affecting the database instance).
- Select the checkbox to limit the display to only those advisories whose value indicates their likely contribution to recorded performance degradation

Real-Time

Use the **Recent** configuration tab to set the refresh and display options for the real-time instance view window.

• **Refresh Window Interval** - Determines the default window refresh interval (seconds), when set to Recent.

Top Sessions window

• Number of top sessions - Maximum number of top sessions to display in the <u>Top Sessions</u> tab.

Refreshing options

Lets you minimize the application automatic refreshing overhead in specific circumstances.

- **Refresh data when the application is inactive** (select/clear) -When the application is not the desktop active window, automatic refreshing will/will not occur.
- **Refresh data when the application is minimized** (select/clear) -When the application is in minimized mode, automatic refreshing will/will not occur

History

Use the **History** configuration tab to set the display and scaling options for the history instance view screen.

- Number of top values to display Determines the default amount of top items that are retrieved when the navigation tree is expanded.
- Select Collection Specifies the set from which you may select dimensions for display. Select a User Defined Collection rather than the default History if it includes dimensions which, because of resource consumption constraints, consists of a larger set.
- Selected <u>dimensions</u> Determines the dimensions for which historical processes and queries can be performed and their order appearance in the navigation tree. The set of available dimensions is specified during StealthCollect installation and cannot be subsequently changed.
- Activity Timeline Scaling Options The three checkboxes determine the scaling and display of the historical <u>activity timeline</u> graph.
- Select the **Scale graph to instance** check box if you want the display to retain the instance vertical scaling. Thus, any drilldown activity display is proportional to the total instance activity, granting you an immediate sense of its significance.

If cleared, the display scale is that of the dimension subset of which the present drilldown is a part. (For example, the vertical scale could be a specific SQL statement and the activity representation, that of a specific database user's portion within it.)

• Select the **Focus on target activity** checkbox if you want the scale to recalibrate as you drill down within dimensions.

If cleared, the scale reverts to that of the instance (if selected, as above) or that of the dimension in which this drill down is a part.

• Select the **Show total in graph** checkbox to display a dotted line representing total instance activity.

General

Use the **General** configuration tab to modify Performance Analysis appearance.

SQL Statement Appearance

Sets the default appearance for SQL statements across the entire application.

- Show Hash Value Display the hash value generated when the statement was fired. (Not displayed in the Real Time tab | Current Sessions | Specific Session | Activity pane)
- Show SQL Text
- Show SQL statements summarized (select/clear) Shortens the SQL statements by replacing long SELECT and FROM clauses with (...).
- Show SQL statements in multiple lines (select/clear) Expands the SQL display to a multi-line presentation so that the entire SQL is visible.

Chart Appearance

Specifies whether to display empty series and how the chart contours are to be represented.

- **Hide empty series** Suppresses display of wait events with no measurable activity.
- **Contour visible** Displays a marked contour at the edge of the activity timeline series representation.

Contour Color (when Contour visible selected)

Specifies whether the border (contour) color line of the activity timeline is to be represented as:

- Black
- 65% Fill color

Color Mode

Specifies the palette-type of color of the activity timeline representation, whether:

- Pastel (pale)
- Vibrant (bright)

Application Mode

Specifies whether the Oracle Wait Event Categories display is to be:

- Simple
- Expert (detailed)

Refreshing Options

Specify if data is also to be refreshed when the application is inactive or minimized.

Change Tracking

Use the **Change Tracking** configuration tab to define the set of configuration and schema parameters to be monitored and displayed.

Select changes to trace - For databases with repository enabled, select/clear <u>changes to track</u> on the displayed checkbox tree node entries. The set of possible changes is specified during StealthCollect installation and can be subsequently changed. Enable only those changes of interest so as to minimize both visual window clutter and system resources required for processing the data.

Compare

Use the **Compare** configuration as follows:

- In the **Resources** tab, set the Resource thresholds severity levels and significance thresholds for resource usage reporting in the comparison **Overall Activity** view.
- In the Activity Highlights tab, set the minimum inclusion Threshold for comparison percentage resource difference in the Activity Highlights view. Set the Grouping and Group size values as per your desired Activity Highlights view group size focus.

Performance Highlights

Use this tab to configure the rule-based engine used by the <u>Highlights</u> tab.

The left pane displays the available rules for each resource. For each Resource, specify which rules you do not want tested by clearing their checkboxes. Clicking on a rule will display in the right pane the list of the composing necessary conditions.

Use the right pane to customize the rule thresholds.

This tab is context-sensitive accessible directly through a right-click a specific rule implementation in the Highlights tab.

Activity Highlights

Use the **Activity** rules to specify and configure dimension nodes (singly and in groups) whose exceptional and significant behavior or distribution is of interest. The default setting is to include all rules.

For each dimension, specify which rules you do not want tested by clearing their checkboxes. Alter the **Threshold** percentage and **Group size** values as wanted. Set the **Grouping** sliding low–high setting according to your preferred group size focus.

Scheduler

Before reports can be run (or to change the network user), specify here the Windows operating system user account to be used by the Windows Scheduler to perform the actual task of report creation:

Click **Edit Account** to display the **Set Scheduler Account** dialog, wherein you supply the valid Windows operating system user ID and password. If yours is a domain account, the user identification must be prefixed with the domain identification and "\". (For example, user Paul Leno in the PRODUCTION environment might have a user identification of: PROD\PLeno)

Note: Once you enter this data, it updates all existing schedules to that account information and any subsequently created schedules (for this and all other Oracle instances accessed from this client) will also have that same account. If your network user/password is changed, you must reenter the User Information for your schedules to work.

Baseline

Use the **Baseline** configuration tab to enable or disenable the baseline display and to set the speed with which the IntelliProfile-generated baseline ranges are calculated and displayed.

- Enable baseline display Use this checkbox to enable or disable the baseline display on this Performance Analysis client. This option will only affect the graphical representation. Use the Agent IntelliProfile configuration to mange the IntelliProfile baseline engine.
- Animate baseline-to-resource display switch Use this checkbox to specify if the transition in the timeline background view between the baseline range and resources is to fade in-and-out (animate) or to occur abruptly.
- Minimal baseline period for knowledgebase generation Use this setting to determine the minimal baseline "learning" period before the baseline is first displayed. Baseline accuracy is dependent upon the amount of historical activity gathered for the baseline calculation and is improved as the historical sampling period is increased. For optimal baseline information, Quest recommends a minimal "learning" period of 24 hours before initial baseline presentation.

Related Topics

Baseline Baseline—agent configuration

Support

Use the **Support** tab to enable several support features, for use under the direction of Quest technical support.

Protocol Log

- Log Server (select/clear)
- Log Client (select/clear)
- Log Questions (select/clear)
- Log Answers (select/clear)

Repository Protocol Log

- Log Server (select/clear)
- Log Client (select/clear)
- Log Activity (select/clear)
- Log Questions (select/clear)
- Log Answers (select/clear)

Advanced Settings - Enable grids customization

Use this checkbox to enable activation of the Metric Selection dialog box. Once activated, you can right-click in the resource metric toolbar of the history and real-time mode lower pane listing and add to or subtract metrics from the default listing of displayed metrics, including their source (either system or Oracle) and total value.

Oracle Wait Event Categories

Wait events are statistics that are incremented by a server process/thread to indicate that it had to wait for an event to complete before being able to continue processing. Wait event data reveals various symptoms of problems that might be degrading performance.

Simple and **Expert** detailed wait events are displayed below, grouped by categories.

Note: Toggle display of Expert/Simple wait event categories in the <u>Application mode</u> of the Performance Analysis Configuration General tab.

All Wait states measurements are in seconds.

Overall CPU	The amount of time Oracle spent both waiting for CPU and using CPU during current interval.
CPU Usage	 When SQL statements and other types of calls are made to Oracle, an amount of CPU time is necessary to process the call. Average calls require a small amount of CPU time. However, a SQL statement involving a large amount of data or a runaway query can potentially consume a large amount of CPU time, reducing CPU time available for other processing. CPU utilization is a key operating system statistic in the tuning process. Excessive CPU could be caused by an inadequately-sized system, by untuned SQL statements, or by inefficient application programs.
	Wait Time until the CPU resource is available. Time spent by the session waiting in the system's run queue to be granted for CPU cycles. The

CPU Wait	amount of time is dependent upon the number of concurrent processes and threads requesting CPU time. The metric value should be inspected in conjunction with the value of the "Run Queue Length" metric.
I/O Wait	Time spent waiting for disk input/output operations to complete. Input/output (I/O) is one of the most expensive operations in a database system. SQL statements that are I/O intensive can monopolize memory and disk use and cause other database operations to compete for these resources. Generally, I/O Wait is caused by poorly-tuned SQLs or applications which generate a significant amount of logical I/O translating into excessive physical disk usage. In this case, SQL/Application tuning can reduce the logical I/O- induced load. However, it could also be caused by poorly- configured disks or storage sub-
	systems.
Datafile Write	The DBWR is performing a write to files and blocks (for example, flushing dirty data buffers from the SGA to the data files).
	The db file writes are not experienced by user sessions.
Multi	The session is reading multiple data blocks from the data files into the SGA buffer. Multi-Block Read waits usually reflects
Block	

Read	full table or index scans. Generally full scans of large segments should be avoided, although they are sometimes unavoidable, as in data warehouse environments.
Single Block Read	The session is reading a single block from a data file into the SGA buffer. Large values can be acceptable, reflecting a high-transaction, well- tuned system with significant index utilization. However, a large number of waits here could also indicate poor joining orders of tables, or unselective indexing.
Direct Path I/O	The session has issued asynchronous I/O requests that bypass the buffer cache and is waiting for them to complete. Direct Path I/O usually reflects disk sorts and bulk loads.
Control File I/O	The session has issued I/O requests to control file blocks, and is waiting for all of the reads and writes to complete.
DB File Parallel Read	The session has issued multiple I/O requests in parallel to read blocks from data files into memory and is waiting for all requests to complete. This may occur during recovery or during regular activity when a session batches many single block I/O requests together and issues them in parallel.
External File I/O	Session waits because of an I/O request on an external large object (LOB), such as a BFILE.

Miscellaneous I/O	Sporadic I/O activity, such as opening a data file. Usually negligible.
Network Wait	 Time spent waiting for messages to be sent or received over the network interface. Network performance is measured in number (per second) of packets sent and received. SQL*Net is the Oracle network interface that allows Oracle tools running on network workstations and servers to access, modify, share, and store data on other servers. SQL*Net is considered part of the program interface in network communications. Network statistics can be used in much the same way as disk statistics to determine if a network or network interface is overloaded or not performing optimally. Excessive network wait can be caused by either: excessive network usage originating in the application physical problems, identifiable by network errors and network collisions
SQL*Net	The wait time it takes for messages, data or break reset to be transmitted over the network interface to the client.
DB	The wait time it takes for messages, data or break reset to be transmitted

Link	over the network interface to another server process on a remote database.
Buffer Wait	Buffer waits occur because of a contention in accessing database block buffers. For example, when a session wants to access a data block that is in an incompatible mode (buffer busy) or wants to read a data block into the buffer, but it has no available space (free buffer).
Buffer Busy	The session wants to access a data block that is either:
	 currently not in memory, but another process has already issued an I/O request to read the block into memory. in memory but in an incompatible mode (current versus consistent,
	for example). The session will wait until the block becomes available. Buffer busy waits should not be greater than 1 percent of the entire activity.
Free Buffer	The session needs a free buffer so it can bring a data block into the buffer cache and is waiting for a buffer that is not dirty to become available. This can occur if DBWR is not writing dirty buffers to disk fast enough. This wait usually indicates a non- optimal physical I/O configuration. Free buffer waits can be minimized by increasing the checkpoint frequency,

	using more DBWR processes, or increasing the number of physical disks.
Other DB Buffer	Sporadic buffer waits caused by buffer contention, such as buffer write wait. Usually negligible.
	Lock wait occur when a session attempts to acquire a lock that is already held by another session. A session will be blocked until the blocking session releases the lock. Locks are designed to ensure data integrity by limiting simultaneous data access.
Lock Wait	Multi-user database locking generally consists of two levels: exclusive locks and share locks. You want to watch out for exclusive locks (that is, TX) as they prohibit resource sharing. For example, the first transaction that exclusively locks a resource is the only one that can alter the resource (except for the DBA) until the exclusive lock is released. Share locks, unlike exclusive locks, allow a resource to be shared. Deadlocking is commonly seen in multi-user systems. It typically occurs when all the hung users are waiting to access a table that another user has locked. This situation causes a deadlock, because each user (transaction) is waiting for resources to be freed by the other user (the blocker). Often, many developers attempt to update the same table and

	many users attempting to update or select from the same table. Most locking issues are application- specific and can be addressed by tuning the concurrency logic in the application.
Latch Wait	Latch waits are the waits that occur when a session needs to acquire a latch that is held by another session. Latches are simple, low-level serialization mechanisms to protect shared SGA data structures and shared code segments from simultaneous session access. They are similar in purpose to locks: Latches protect internal memory structures while locks protect data structures. They are designed to be very quickly acquired and freed. The implementation of latches is operating system dependent, particularly in regard to whether a process will wait for a latch and for how long. For example, latches protect the list of users currently accessing the database and protect the data structures describing the blocks in the buffer cache. A server or background process acquires a latch for a very short time while manipulating or looking at one of these structures. Rarely is latch contention tunable by reconfiguring the instance. Rather, latch contention usually is resolved through application changes. Latch

	waits can be eliminated by either tuning the instance or the application. Tuning latch contention is dependent on the type of latch wait encountered.
Redo Wait	 All DML activity occurring in the database is recorded in the redo log. Redo waits include the contention resulting from: User sessions writing redo entries to the redo log buffer. LGWR writing the log buffer contents into the active redo log file. ARCH archiving redo logs to the destination location.
Archival	Any activity which generates LGWR for archiving the redo logs to the target location. Additionally, user sessions can also experience archival waits when the log that the LGWR will be switching into has not yet been archived and the session is issuing DMLs.
Log Buffer	Waits that occur when copying redo entries into the log buffer. A frequent cause of log buffer wait is if redo entries are written into the log buffer faster than LGWR can write it to the redo logs, and user sessions must therefore wait because of log write latency.
Log Switch	The session waits for a redo log

	switch to be completed.
Log Write	 Waiting for the write (physical I/O) to a logfile to complete. LGWR experience log write waits when it periodically flushes the contents of the log buffer into the active log file. User sessions experience log write waits when, upon commits, it waits for the LGWR to flush its redo information to the redo log file. The user session will post the LGWR to write the log buffer to the redo log file.
Cluster Wait	Waits arising from cluster-related activity contention. For example, cache synchronization and global locks.
Global Locks (RAC)	Global Cache Service (GCS) Operations tracks the locations, modes, and roles of data blocks. The GCS therefore also manages the access privileges of (and imposes waits on) various instances in relation to resources. Oracle uses the GCS for cache coherency when the current version of a data block is in one instance's buffer cache and another instance requests that block for modification. Apart from the regular locking at the instance level, Oracle maintains another lock structure and a set of lock escalation rules to manage data blocks between instances. This lock structure is a three character representation consisting of a Role, Mode and indicator:

	 Roles could be either NULL, Shared or Exclusive Modes could be either Local or Global The indicator represented by a 0 or 1 is an internal mechanism to track if the block found in the cache of an instance is either a current image or a previous image.
Cluster Coordination	Resource coordination within Real Application Clusters occurs at both an instance level and at a cluster database level. Waits under this category refer to activities that are of a global nature, such as waits encountered while movement of blocks between instances participating in the cluster or waits encountered by cluster specific background processes.
Network/	Apart from the regular network activity between the client and the database servers, in a RAC environment Oracle uses a private interconnect for transfer of data between instances to satisfy user requests. Performance of the interconnect is critical to the overall performance of the database server. This categorization represents waits encountered with respect to the interconnect activity. Significant network interconnect wait may indicate that excessive cache

Interconnect	 transfer because of a non-optimal application cluster affinity or a too-slow network link. In general, the network interconnect is responsible for: Monitoring Health, Status, and Synchronize messages Transporting Distributed lock manager (DLM) messages Accessing remote file systems Moving application-specific traffic Providing cluster alias routing
Other Wait	Total time spent waiting for miscellaneous operations to complete. None of the underlying operations (wait events) can be classified into any other wait categories. Usually negligible.
Backup/ Recovery	When archiving is not performed fast enough, database operations can be suspended temporarily whenever the log writer is forced to wait for an inactive group to become available for reuse.
	Waits experienced by sessions when exceeding thresholds/quotas permitted them by the resource manager. Resource Manager (10g and later) can be used to provide more fine- grained control of resource allocations to Services in both RAC and single instance environments. These two facilities are integrated so that

Resource Manager	Services can be mapped to Resource Manager consumer groups. Resource Manager waits are thus controls placed on Service workloads to limit the execution times of long running operations, the number of active sessions that are allowed to execute concurrently, the resources consumed by long running resource intensive sessions, and other resource usages.
Miscellaneous Other Waits	Assorted independent waits. Usually negligible.
Non- [Resource] Activity	Instance activity generated by resources other than the selected Oracle wait event.

Metric Listing

Unless stated otherwise, time and resource usage metrics are denominated in seconds and bytes, respectively.

ABCDEFGILMNOPRSTU

Agent Down	The amount of time that the StealthCollect agent was inactive during the current interval.
Active Time	Sum of all the active waits, equal to the session total activity within the current interval.
Archival Wait	Any activity which generates LGWR for archiving the redo logs to the target location. Additionally, user sessions can also experience archival waits when the log that the LGWR will be switching into has not yet been archived and the session is issuing DMLs.
Average Collisions per Packet	The number of collisions (number of times when two machines send packets at the same time on the network and the packets "collide" so that both packets need to be retransmitted) occurred during the current interval divided by the number of packets transmitted during the current interval.
	The average time spent (ms) a resource was locked (held in contention). Lock waits occur when a session attempts to acquire a lock that is already held by another session. One session will be blocked

Average Lock Duration	until the blocking session (the first to acquire a lock on the resource) releases the lock. High values of lock duration may indicate aggressive lock incidence which may be improved upon tuning. The metric is calculated as follows: (Lock Wait) 100 * Enqueue Waits
Average Packet Size	Average packet size sent by the system on all network interfaces.
Average Read Time	The average time (ms) spent reading a data block from the disk into the database buffer cache. On most disks this metric should not exceed 20 milliseconds and readings above 40 milliseconds may indicate either a device bottleneck, or poorly configured disks or storage sub- systems. The metric is calculated as follows: <u>100 * (Single block read + Multi block read + Data file parallel read)</u> (Physical reads - Physical reads direct)
	The average time spent (ms) writing a redo log entry to the log files. User sessions experience log write waits when, upon commits, an individual session waits for the LGWR to flush

Average Redo Write Time	its redo information to the redo log file. Unlike most other Oracle write I/Os, Oracle sessions must wait for redo log writes to complete before they can continue processing. On most disks this metric should not exceed 20 milliseconds and readings above 40 milliseconds may indicate log files are not placed on devices with sufficient write speeds. The metric is calculated as follows: $\frac{(Log Writes)}{100*}$ Redo Writes
Average SQL Duration	Average duration of the SQL statements, executed during the current interval
Average SQL Response Time	The gross response time for a given SQL cursor; response time is measured beginning with cursor creation and continuing until the cursor is closed. This value tends to be higher than the value of average Active Time Per Execution, since Active Time measures the net SQL activity; net SQL activity ignores any pauses in SQL execution such as sporadic application processing between distinct Fetch operations.
	The average time spent (ms) writing a data block from the database buffer to the disk. The DBWR writes data blocks to datafiles (for example, flushing dirty data buffers from the SGA to the datafiles).

Average Write Time	On most disks this metric should not exceed 20 milliseconds and readings above 40 milliseconds may indicate either a device bottleneck, or poorly configured disks or storage sub-systems. The metric is calculated as follows: (Datafile write)
	¹⁰⁰ * Physical writes - Physical writes direct
Backup/ Recovery Wait	When archiving is not performed fast enough, database operations can be suspended temporarily whenever the log writer is forced to wait for an inactive group to become available for reuse.
Baseline	IntelliProfile is the new adaptive configurable (both agent and display) baseline profiling technology introduced in the Performance Analysis agent. IntelliProfile learns the instance signature behavior characteristics via sophisticated algorithms to identify the various cyclical and periodic behaviors and to neutralize the random noise distortion (for example, an employee absent due to illness). Its adaptive element grows smarter and more accurate with increases in the sampling period. Armed with this knowledge, IntelliProfile can project the acceptable/normal ranges of activity per each point in time.
	This ratio measures the amount of DML work that each transaction

Block Changes Per Call (Blocks)	performs by computing the average blocks changed per single user call. Creating or dropping indexes affects this value, because changes to index blocks increment it. The ratio is calculated as follows: <u>DB Block Changes</u> User Calls
Block Reads	This rate determines the block get rate, which is a basic measure of the rate at which the application system references the database. Change in metric size may indicate that the database workload has materially increased or that the referenced segment sizes increased.
Buffer Busy Wait	 The session wants to access a data block that is either: currently not in memory, but another process has already issued an I/O request to read the block into memory. in memory but in an incompatible mode (current versus consistent, for example).
Buffer Wait	Buffer waits occur because of a contention in accessing database block buffers. For example, when a session wants to access a data block that is in an incompatible mode (buffer busy) or wants to read a data block into the buffer, but it has no available space (free buffer).

Buffer Wait as % of Active Time	Buffer waits occur because of a contention in accessing database block buffers. For example, when a session wants to access a data block that is in an incompatible mode (buffer busy) or wants to read a data block into the buffer, but it has no available space (free buffer). The ratio is calculated as follows: $\frac{Buffer Wait}{Active time} * 100$
Bytes Sent via SQL NET to Client	Total number of bytes sent to the client from the foreground processes.
Bytes Sent via SQL NET to Dblink	Total number of bytes sent over a database link.
Cache Hit Ratio (%cache hits)	The effectiveness of the buffer cache, computed by comparing the number of block requests, which found the block in the buffer cache with the entire block requests (including requests which performed disk reads).
	 Acceptable values: in excess of 70%
	This ratio is calculated as follows:
	1 <u>Physical Reads - Physical</u> - <u>Reads Direct</u> - Logical Reads
	This rate measures the workload being placed on the instance from all work sources. It should be noted, however, that this rate might not be directly

Call Rate	comparable across application system version changes where row-at-a-time loop constructs have been re-coded as set operations or vice versa. Use of an array interface will also affect this rate. The rate is calculated as follows: <i>Recursive Calls</i> + <i>User Calls</i>
Chained Row Ratio	 This ratio measures the percentage of chained or migrated rows encountered during a fetch as a percentage of the entire fetched rows. This ratio should be very close to zero, except in applications handling LONG columns. If this ratio increases over time, usually PCTFREE has been set too low for the LONG containing tables. The ratio is calculated as follows: <u>Table Fetch Continued Row</u> Table Fetch by RowID + Table Scan Rows Gotten
Cluster Coordination Wait	Resource coordination within Real Application Clusters occurs at both an instance level and at a cluster database level. Waits under this category refer to activities that of a global nature such as waits encountered while movement of blocks between instances participating in the cluster or waits encountered by cluster specific background processes.
	Common cluster-wide membership

Cluster Wait	node view encompassing cluster resources (such as interconnect hardware, shared disks, and Oracle instances)
Consistent Change Ratio	This ratio measures the extent to which applications must exercise the read consistency mechanism. It is important to realize that the query processing parts of the UPDATE and DELETE operations are subject to read consistency. The ratio is calculated as follows: <u>Consistent Changes</u> Consistent Gets
Consistent Changes	Number of times a user process has applied rollback entries to perform a consistent read on the block. Workloads that produce a great deal of consistent changes can consume a great deal of resources. The value of this statistic should be small in relation to the "consistent gets" statistic.
Consistent Reads	The number of data block reads in CONSISTENT mode (i.e. consistent reads). A CONSISTENT read retrieves the block version as at the time the query has started. Consistent mode reads are usually associated with query activity. Also known as: consistent gets
Control File I/O Wait	The session has issued I/O requests to control file blocks, and is waiting for all of the reads and writes to complete.

CPU Usage	When SQL statements and other types of calls are made to Oracle, an amount of CPU time is necessary to process the call. Average calls require a small amount of CPU time. However, a SQL statement involving a large amount of data or a runaway query can potentially consume a large amount of CPU time, reducing CPU time available for other processing. CPU utilization is a key operating system statistic in the tuning process. Excessive CPU could be caused by an inadequately-sized system, by untuned SQL statements, or by inefficient application programs.
CPU Wait	Wait Time until the CPU resource is available. Time spent by the session waiting in the system's run queue to be granted for CPU cycles. The amount of time is dependent upon the number of concurrent processes and threads requesting CPU time. The metric value should be inspected in conjunction with the value of the "Run Queue Length" metric.
Current Reads	The number of data block reads in CURRENT mode (i.e. current reads). A CURRENT read retrieves the latest version of the block as at the time of reading. Current mode reads are usually associated with DML activity. AKA: db block gets.
	The DBWR is performing a write to files and blocks (for example, flushing

Datafile Write Wait	dirty data buffers from the SGA to the data files). The db file writes are not experienced by user sessions.
DB Availability	The relative amount of time that the Oracle instance was running during the current interval. (The percentage of time during which the database was available.) If the instance is down, no data collection occurs. Thus, if this metric value is less than the interval length, the value of this statistic reflects only the active portion of the interval. The three states are as follows: • <i>UP</i> - Database is running • <i>DOWN</i> - Database is shutdown • <i>UNKNOWN</i> - StealthCollect agent is down and therefore the database state cannot be determined
DB Block Changes	The number of database blocks in memory (SGA) that were modified by the session as a result of DML activity. Such changes generate redo log entries and hence become permanent changes to the database if the transaction is committed. Because each logical change to a block typically involves more than one physical change to the block, the value of this statistic will always be considerably greater than the number of blocks dirtied in the cache.
	The session has issued multiple I/O

Data File Parallel Read Wait	requests in parallel to read blocks from data files into memory and is waiting for all requests to complete. This may occur during recovery or during regular activity when a session batches many single block I/O requests together and issues them in parallel.
DB Link Wait	The wait time it takes for messages, data or break reset to be transmitted over the network interface to another server process on a remote database.
DB Unavailable	 The relative amount of time that the Oracle instance was unavailable during the current interval. (The percentage of time during which the database was unavailable.) If the instance is down, no data collection occurs. Thus, if this metric value is less than the interval length, the value of this statistic reflects only the inactive portion of the interval. The three states are as follows: <i>UP</i> - Database is running <i>DOWN</i> - Database is shutdown <i>UNKNOWN</i> - StealthCollect agent is down and therefore the database state cannot be determined
Degree of Parallelism	The total number of server processes (master and slaves) assigned to handle a single operation.
Direct Path I/O	The session has issued asynchronous I/O requests that bypass the buffer cache and is waiting for them to

Wait	complete.
	Direct Path I/O usually reflects disk sorts and bulk loads.
Disk Utilization	The percentage of time the busiest device spent serving system-wide I/O requests. This metric serves as a measure for the system I/O load. High values may indicate device bottleneck.
Elapsed Time	The amount of time (in microseconds) used by the active cursor for parsing/executing/fetching.
Enqueue Requests	Total number of table or row locks obtained by the session.
Enqueue Waits	Enqueue waits occur when a session waits to obtain a lock. In most cases, this occurs because of a lock on a table or row that the waiting session needs to lock or modify. In some circumstances, the lock involved may be an Oracle internal lock. If the database is well tuned and the application design sound, enqueue waits should be negligible.
	Wait state out enqueue operations as a percentage of the entire enqueue operations.
Enqueue Waits Ratio (% enqueue blocked) (Lock)	 An enqueue operation (either get or convert) will enter wait state if the enqueue cannot be immediately granted.
	The ratio is calculated as follows: <u>Enqueue Waits</u> Enqueue Requests

Executions Ended	Number of statements whose activity finished during the current interval.
Executions Started	Number of statements issued during the current interval.
External File I/O Wait	Session waits because of an I/O request on an external large object (LOB), such as a BFILE.
Free Buffer Wait	The session needs a free buffer so it can bring a data block into the buffer cache and is waiting for a buffer that is not dirty to become available. This can occur if DBWR is not writing dirty buffers to disk fast enough. This wait usually indicates a non- optimal physical I/O configuration. Free buffer waits can be minimized by increasing the checkpoint frequency, using more DBWR processes, or increasing the number of physical disks.
	Global Cache Service (GCS) Operations tracks the locations, modes, and roles of data blocks. The GCS therefore also manages the access privileges of (and imposes waits on) various instances in relation to resources. Oracle uses the GCS for cache coherency when the current version of a data block is in one instance's buffer cache and another instance requests that block for modification. Apart from the regular locking at the instance level, Oracle maintains another lock structure and a set of lock

Global Locks (RAC)Wait	 escalation rules to manage data blocks between instances. This lock structure is a three character representation consisting of a Role, Mode and indicator: Roles could be either NULL, Shared or Exclusive Modes could be either Local or Global The indicator represented by a 0 or 1 is an internal mechanism to track if the block found in the cache of an instance is either a current image or a previous image.
I/O Wait	Time spent waiting for disk input/output operations to complete. Input/output (I/O) is one of the most expensive operations in a database system. SQL statements that are I/O intensive can monopolize memory and disk use and cause other database operations to compete for these resources. Generally, I/O Wait is caused by poorly-tuned SQLs or applications which generate a significant amount of logical I/O translating into excessive physical disk usage. In this case, SQL/Application tuning can reduce the logical I/O- induced load. However, it could also be caused by poorly- configured disks or storage sub- systems.

Idle	Idle events are events that spend their time idle or waiting for an action. The time keeps incrementing for the event until an action occurs. An action can be a database process, user or application process. These wait events can be ignored most of the time.
Latch Misses Ratio	The latch miss ratio defines the number of times a process obtained a willing-to-wait latch vs. missing the attempt. Latches protect the many memory structures in Oracle's SGA. They ensure that one and only one process at a time runs or modifies any memory structure at the same instant. Latches are much more restrictive than locks, which at least allow for some collective user interaction. They have no queuing mechanism, and therefore, you either get the latch or you are forced to continually retry. A high value for this ratio indicates a latching problem, whereas a low value is generally good. However, as the data is rolled up over all latches, a low latch miss ratio can artificially mask a low get rate on a specific latch. If the latch miss ratio exceeds 1%, you should take action to resolve the amount of latch contention occurring. Latch waits are the waits that occur
	when a session needs to acquire a latch that is held by another session. Latches are simple, low-level serialization mechanisms to protect

Latch Wait	shared SGA data structures and shared code segments from simultaneous session access. They are similar in purpose to locks: Latches protect internal memory structures while locks protect data structures. They are designed to be very quickly acquired and freed. The implementation of latches is operating system dependent, particularly in regard to whether a process will wait for a latch and for how long. For example, latches protect the list of users currently accessing the database and protect the data structures describing the blocks in the buffer cache. A server or background process acquires a latch for a very short time while manipulating or looking at one of these structures. Rarely is latch contention tunable by reconfiguring the instance. Rather, latch contention usually is resolved through application changes. Latch waits can be eliminated by either tuning the instance or the application. Tuning latch contention is dependent
	on the type of latch wait encountered.
	Latch waits are the waits that occur when a session needs to acquire a latch that is held by another session. Latches are simple, low-level serialization mechanisms to protect shared SGA data structures and shared code segments from

Latch Wait as % of Active Time	simultaneous session access. They are similar in purpose to locks: Latches protect internal memory structures while locks protect data structures. They are designed to be very quickly acquired and freed. The implementation of latches is operating system dependent, particularly in regard to whether a process will wait for a latch and for how long. For example, latches protect the list of users currently accessing the database and protect the data structures describing the blocks in the buffer cache. A server or background process acquires a latch for a very short time while manipulating or looking at one of these structures.
	Rarely is latch contention tunable by reconfiguring the instance. Rather, latch contention usually is resolved through application changes. Latch waits can be eliminated by either tuning the instance or the application. Tuning latch contention is dependent on the type of latch wait encountered. The ratio is calculated as follows: Latch Wait Active time * 100
	Lock wait occur when a session attempts to acquire a lock that is already held by another session. A session will be blocked until the blocking session releases the lock.

Lock Wait	Locks are designed to ensure data integrity by limiting simultaneous data access. Multi-user database locking generally consists of two levels: exclusive locks and share locks. You want to watch out for exclusive locks (that is, TX) as they prohibit resource sharing. For example, the first transaction that exclusively locks a resource is the only one that can alter the resource (except for the DBA) until the exclusive lock is released. Share locks, unlike exclusive locks, allow a resource to be shared. Deadlocking is commonly seen in multi-user systems. It typically occurs when all the hung users are waiting to access a table that another user has locked. This situation causes a deadlock, because each user (transaction) is waiting for resources to be freed by the other user (the blocker). Often, many developers attempt to update the same table and many users attempting to update or select from the same table. Most locking issues are application- specific and can be addressed by tuning the concurrency logic in the application.
	Lock wait occur when a session attempts to acquire a lock that is already held by another session. A session will be blocked until the blocking session releases the lock. Locks are designed to ensure data

Lock Wait as % of Active Time	integrity by limiting simultaneous data access. Multi-user database locking generally consists of two levels: exclusive locks and share locks. You want to watch out for exclusive locks (that is, TX) as they prohibit resource sharing. For example, the first transaction that exclusively locks a resource is the only one that can alter the resource (except for the DBA) until the exclusive lock is released. Share locks, unlike exclusive locks, allow a resource to be shared. Deadlocking is commonly seen in multi-user systems. It typically occurs when all the hung users are waiting to access a table that another user has locked. This situation causes a deadlock, because each user (transaction) is waiting for resources to be freed by the other user (the blocker). Often, many developers attempt to update the same table and many users attempting to update or select from the same table. Most locking issues are application- specific and can be addressed by tuning the concurrency logic in the application. The ratio is calculated as follows: Lock Wait Active time * 100
	Waits that occur when copying redo entries into the log buffer. A frequent cause of log buffer wait is if

Log Buffer Wait	redo entries are written into the log buffer faster than LGWR can write it to the redo logs, and user sessions must therefore wait because of log write latency.
Log Switch Wait	The session waits for a redo log switch to be completed.
	This ratio measures the LGWR waits for log switches, relative to total instance activity. A log switch occurs when LGWR stops writing to one online redo log group and starts writing to another. By default, a log switch occurs automatically when the current online redo log file group becomes full.
Log Switch Wait Time	Log Switch Wait time portion should not be greater than five percent of the total workload activity, however it can become critical under the following circumstances: The next redo log to be used is not available because the checkpoint started when the log switch is not complete, or the redo log has not yet been archived. The ratio is calculated as follows: Log Switch Active time
	Waiting for the write (physical I/O) to a logfile to complete. LGWR experience log write waits when it periodically flushes the contents of the log buffer

Log Write Wait	into the active log file. User sessions experience log write waits when, upon commits, it waits for the LGWR to flush its redo information to the redo log file. The user session will post the LGWR to write the log buffer to the redo log file.
Logical Reads	The number of data blocks read by Oracle, which are already present in the buffer cache, thus requiring no I/O read operations. (The total of "Current Reads" plus "Consistent Reads"). AKA: session logical reads.
Machine RAM	Amount of machine RAM available.
Memory Utilization	The amount of physical memory consumed by the various processes. High memory utilization along with high swapping and paging rates indicates that the amount of physical RAM should be increased.
Miscellaneous I/O Wait	Sporadic I/O activity, such as opening a data file. Usually negligible.
Miscellaneous Other Waits	Assorted independent waits. Usually negligible.
Multi-Block Read Wait	The session is reading multiple data blocks from the data files into the SGA buffer. Multi-Block Read waits usually reflects full table or index scans. Generally full scans of large segments should be avoided, although they are sometimes unavoidable, as in data warehouse environments.

Network/ Interconnect Wait (Cluster)	 Apart from the regular network activity between the client and the database servers, in a RAC environment Oracle uses a private interconnect for transfer of data between instances to satisfy user requests. Performance of the interconnect is critical to the overall performance of the database server. This categorization represents waits encountered with respect to the interconnect activity. Significant network interconnect wait may indicate that excessive cache transfer because of a non-optimal application cluster affinity or a too-slow network link. In general, the network interconnect is responsible for: Monitoring Health, Status, and Synchronize messages Transporting Distributed lock manager (DLM) messages Accessing remote file systems Moving application-specific traffic Providing cluster alias routing
Network Collisions	Number of times when two computers send packets at the same time on the network and the packets "collide" so that both packets must be re- transmitted.
Network Errors	Number of errors reported on the various network interfaces.

Network Wait	 Time spent waiting for messages to be sent or received over the network interface. Network performance is measured in number (per second) of packets sent and received. SQL*Net is the Oracle network interface that allows Oracle tools running on network workstations and servers to access, modify, share, and store data on other servers. SQL*Net is considered part of the program interface in network communications. Network statistics can be used in much the same way as disk statistics to determine if a network or network interface is overloaded or not performing optimally. Excessive network wait can be caused by either: excessive network usage originating in the application physical problems, identifiable by network errors and network collisions
Non-Oracle CPU Usage	Overall CPU consumption not associated with the monitored Oracle instance (background and foreground processes).
Non-Oracle Resident Memory	Amount of physical RAM consumed by processes other than that of the monitored Oracle instance. The statistic is calculated as follows: System Resident Memory Usage -

	Resident SGA Size - Session Resident Memory Usage
Oracle BG CPU Usage	Amount of memory consumed by the monitored Oracle instance background processes, both physical and swap memory.
Oracle BG Memory Usage	Amount of physical memory consumed by Oracle background processes.
Oracle BG Resident Memory	Amount of physical memory consumed by Oracle background processes.
Oracle CPU Usage	Overall monitored Oracle instance processes CPU usage.
Oracle Data I/O Operations	Total number of I/O operations (both physical and logical) performed by the Oracle session.
Oracle FG CPU Usage	CPU time consumed by the monitored Oracle instance user (session) processes.
Oracle FG Memory Usage	Amount of physical memory consumed by the monitored Oracle instance foreground (session) processes.
Oracle FG Resident Memory	Amount of physical memory consumed by the monitored Oracle instance foreground (session) processes.
Oracle Physical I/O Operations	Total number of physical I/O operations (read, writes and redo) performed by the monitored Oracle instance processes. Note: A physical I/O issued by Oracle can be satisfied by the operating system's file cache.

Oracle Resident Memory Usage	The amount of physical memory consumed by the various monitored Oracle instance processes (including the SGA).
	The amount of CPU consumed by the entire system detailed into monitored Oracle instance activity and all other activities.
Oracle/System CPU Usage	 High values indicate that the monitored Oracle instance is a major CPU consumer and is likely causing bottlenecks. It is thus the object of tuning efforts. Low values indicate other system activity as the major CPU consumer and the problem solution likely resides with other applications or at the operating system level.
	The ratio of monitored Oracle instance I/O operations to total system I/O operations. This ratio ignores logical I/O operations performed by Oracle (only operations which were translated to actual system calls are included). It should be used when identifying specific disk I/O problems.
Oracle/System I/O Ratio	 High values indicate that Oracle is a major I/O activity source and is likely causing bottlenecks. It is thus the object of tuning efforts. Low values indicate other system activity as the major I/O source

and the problem solution likely resides with other applications or at the operating system level.The ratio is calculated as follows: Physical Reads + Physical Writes + Redo Writes System Logical Reads + System Logical WritesOther WaitTotal time spent waiting for miscellaneous operations to complete. None of the underlying operations (wait events) can be classified into any other wait categories. Usually negligible.Other DB BufferSporadic buffer waits caused by buffer contention, such as buffer write wait. Usually negligible.Other Wait as % of Active TimeTotal time spent waiting for miscellaneous operations to complete. None of the underlying operations (wait events) can be classified into any other wait categories. Usually negligible.Other DB BufferTotal time spent waiting for miscellaneous operations to complete. None of the underlying operations (wait events) can be classified into any other wait categories. Usually negligible. The ratio is calculated as follows: Other Wait as follows: Other Wait wait categories. Usually negligible. The ratio is calculated as follows: Other Wait * 100Other DB Buffer WaitSporadic buffer waits caused by buffer contention, such as buffer write wait. Usually negligible.Other DB Buffer WaitSporadic buffer waits caused by buffer contention, such as buffer write wait. Usually negligible.Other DB Buffer WaitSporadic buffer waits caused by buffer contention, such as buffer write wait. Usually negligible.Other DB Buffer WaitActivity attributable to dimension		
Physical Reads + Physical Writes + Redo Writes System Logical Reads + System Logical WritesOther WaitTotal time spent waiting for miscellaneous operations to complete. None of the underlying operations (wait events) can be classified into any other wait categories. Usually negligible.Other DB BufferSporadic buffer waits caused by buffer contention, such as buffer write wait. Usually negligible.Other Wait as % of Active TimeTotal time spent waiting for miscellaneous operations to complete. None of the underlying operations (wait events) can be classified into any other wait categories. Usually negligible.Other DB Buffer wait categories.Total time spent waiting for miscellaneous operations to complete. None of the underlying operations (wait events) can be classified into any other wait categories. Usually negligible.Other Wait as % of Active TimeSporadic buffer waits caused by buffer contention, such as buffer write wait. Usually negligible.Other DB Buffer WaitSporadic buffer waits caused by buffer contention, such as buffer write wait. Usually negligible.Other DB Buffer WaitSporadic buffer waits caused by buffer contention, such as buffer write wait. Usually negligible.		resides with other applications or
Other Waitmiscellaneous operations to complete. None of the underlying operations (wait events) can be classified into any other wait categories. Usually negligible.Other DB BufferSporadic buffer waits caused by buffer contention, such as buffer write wait. Usually negligible.Other DB BufferTotal time spent waiting for miscellaneous operations to complete. None of the underlying operations (wait events) can be classified into any other wait categories. Usually negligible.Other Wait as % of Active TimeTotal time spent waiting for miscellaneous operations to complete. None of the underlying operations (wait events) can be classified into any other wait categories. Usually negligible. The ratio is calculated as follows: Other Wait * 100 Active timeOther DB Buffer WaitSporadic buffer waits caused by buffer contention, such as buffer write wait. Usually negligible.		<u>Physical Reads + Physical Writes +</u> <u>Redo Writes</u> System Logical Reads + System
Other DB Buffercontention, such as buffer write wait. Usually negligible.Other Wait as % of Active TimeTotal time spent waiting for miscellaneous operations to complete. None of the underlying operations (wait events) can be classified into any other wait categories. Usually negligible. 	Other Wait	miscellaneous operations to complete. None of the underlying operations (wait events) can be classified into any other
Other Wait as % of Active Timemiscellaneous operations to complete. None of the underlying operations (wait events) can be classified into any other wait categories. Usually negligible. The ratio is calculated as follows: Other Wait 	Other DB Buffer	contention, such as buffer write wait.
Wait Contention, such as buffer write wait. Usually negligible.		miscellaneous operations to complete. None of the underlying operations (wait events) can be classified into any other wait categories. Usually negligible. The ratio is calculated as follows: <u>Other Wait</u> * 100
Other Activity attributable to dimension		contention, such as buffer write wait.
[dimension] Activity attributable to dimension entities other than the selected one.	Other [dimension]	Activity attributable to dimension entities other than the selected one.
Overall CPU The amount of time Oracle spent both waiting for CPU and using CPU during current interval.	Overall CPU	waiting for CPU and using CPU during
Total number of nackets transmitted		Total number of packets transmitted

Packets	and received by the operating system on the various network interfaces. Packets are blocks of information sent over the network each time a connection or data transfer is requested. The information contained in packets depends on the type of packet: connect, accept, redirect, data, and so on.
Packets In	Number of packets received by all network interfaces.
Packets Out	Number of packets sent on all network interfaces.
Page In	Total number of page-in operations performed by the operating system.
Page Out	Total number of page-out operations performed by the operating system.
Paging	Total number of page operations (Page-in and Page-outs) performed by the operating system.
Parse Count	Parse Count indicates the number of times a SQL statement was "hard" parsed. Hard parses (as opposed to soft parses which are satisfied by using already-cached objects) are real parse calls and, as such, are a very expensive operation in terms of memory use, because they require Oracle to allocate a work heap and other memory structures and then build a parse tree.
Physical Memory	Amount of physical memory consumed by entire operating system processes (including Oracle) as a percentage of

Utilization	the total available.
Physical Reads	The number of data blocks read by Oracle, which are not present in the buffer cache, thus requiring I/O read operations, i.e. the total number of data blocks read from disk.
Physical Reads Direct	Number of reads directly from disk into the PGA, bypassing the buffer cache. Generally, these are reads from temporary segments.
Physical Writes	Total number of data blocks written to disk. This metric represents the rate at which the monitored Oracle instance generates system I/O calls.
Physical Writes Direct	The number of writes directly written to disk bypassing the buffer cache (as in a direct load operation).
Recursive Calls	Number of recursive calls generated at both the user and system level. When Oracle needs to make a change to the tables maintained for internal processing, it internally generates an internal SQL statement, which in turn generates a recursive call. They occur because of cache misses and dynamic storage extension. If the dictionary data is found in cache, a recursive call is not made and the data is read from cache directly. A high value for recursive calls is caused by improper cursor usage, excessive dynamic space management actions, and or excessive statement re-parses. You need to

	determine the cause and correct it By either relinking applications to hold cursors, use proper space management techniques (proper storage and sizing) or ensure repeat queries are placed in packages for proper reuse.
Redo Entries	The number of redo records created (copied into the redo log buffer).
Redo Log Space Requests	Number of times the active log file is full and the session must wait for a log switch to be performed. Once the log switch completes, the redo log entries can be allocated, and the session resumed. Log files that are small in relation to the size of the SGA or the commit rate of the workload can cause performance problems.
Redo Log Space Wait Ratio (% requests)	This ratio measures the redo memory allocation. If it is greater than 1/5000, then the redo log buffer should be increased until the redo log space wait ratio stops failing. The ratio is calculated as follows: <u>Redo Entries</u> Redo Log Space Requests
Redo Wait	All DML activity occurring in the database is recorded in the redo log. Redo waits include the contention resulting from: User sessions writing redo entries to the redo log buffer. LGWR writing the log buffer contents into the active redo log file.

	ARCH archiving redo logs to the destination location.
Redo Writes	Total number of write operations issued by LGWR to the redo log files.
Resident SGA Ratio	 This ratio measures the relative SGA portion, which is RAM resident. This ratio should be close to 100%; otherwise it indicates that the SGA fragments are swapped out by the operating system. Setting LOCK_SGA=TRUE will ensure that the entire SGA is locked into the physical memory.
	The ratio is calculated as follows: <u>Resident SGA Size</u> Resident SGA Size + Swap SGA Size
Resident SGA Size	Amount of SGA resident in physical memory. In a properly working environment, all SGA is RAM resident. When portions of SGA are being swapped, Oracle performance is significantly degraded, as previously cached data now requires I/O to be accessed. Specify LOCK_SGA=TRUE to ensure that the entire SGA is RAM resident.
	Waits experienced by sessions when exceeding thresholds/quotas permitted them by the resource manager. Resource Manager (10g and later) can be used to provide more fine-grained control of resource allocations to Services in both RAC and single

Resource Manager Wait	instance environments. These two facilities are integrated so that Services can be mapped to Resource Manager consumer groups. Resource Manager waits are thus controls placed on Service workloads to limit the execution times of long running operations, the number of active sessions that are allowed to execute concurrently, the resources consumed by long running resource intensive sessions, and other resource usages.
Row Source Ratio (% scans)	This ratio measures the percentage of the total rows retrieved, which came from full table scans as a percentage of total retrieved rows. • Significant values may indicate non-optimal queries. The ratio is calculated as follows: <u>Table Scan Rows</u> <u>Gotten</u> Table Scan Rows Gotten + Table Fetch by RowID
Rows Processed	The total number of rows returned by your SELECT or modified by your INSERT, UPDATE, or DELETE statement.
Run Queue	System average run queue. The CPU run queue is a holding area for threads and processes that require the CPU when the CPU is busy serving other processes. The run queue length is an indicator of whether the system has

Length	 sufficient CPU resources for all the processes it executes. High values along with high CPU utilization, indicates that the system requires faster or more CPUs to handle the given load.
Samples	Number of Collector SGA samples performed during the interval.
Seconds per second	Seconds per second is a measure of activity occurring in the database. It can also be described as the total amount of active time consumed by all sessions during a given second. Example: If during a given second, two sessions were active throughout the entire second, the total active time would be 2 seconds, therefore displayed as 2 seconds per second. If one session was active for only half a second, while the second was active throughout the entire second, the total active time would be 1.5 seconds, therefore displayed as 1.5 seconds per second . Note : This figure also equals to the average number of active sessions within a given second.
Session Logons	Number of sessions that performed a logon during the current interval.
Sessions Ended	Number of sessions logged out during the current interval.
Session	Entire session process memory

Memory Usage (MB)	consumption, both physical and swap.
Session PGA Memory (MB)	Amount of PGA memory allocated for the session.
Session Resident Memory Usage	Amount of physical memory consumed by the session process.
Sessions Running	Number of sessions that were logged on during the current interval.
Session Swap Memory Usage (MB)	Amount of swap memory consumed by the session.
Session UGA Memory	Amount of UGA memory allocated for the session.
SGA Size	Amount of memory allocated for SGA of the monitored Oracle instance. Not a V\$SESSTAT statistic.
SGA Swap Size	Amount of swapped SGA. In a properly working environment, this metric's value should be zero. When portions of SGA are being swapped, Oracle performance is significantly degraded, as previously cached data now requires I/O to be accessed. Specify LOCK_SGA=TRUE to ensure that the entire SGA is RAM resident.
Single Block Read Wait	The session is reading a single block from a data file into the SGA buffer. Large values can be acceptable, reflecting a high-transaction, well-tuned system with significant index utilization. However, a large number of waits here could also indicate poor joining orders of tables, or unselective indexing.

Sort Overflow Ratio (% disk scans)	This ratio measures the portion of sorts that could not fit into a session's sort area and therefore used temporary segments. High values might indicate that the value of SORT_AREA_SIZE parameter (in the initialization files) should be increased to allow more memory space for sort operations. However, if the application requires sorting of vast number of rows, the use of disk sorting and temporary segments might not be avoided. The ratio is calculated as follows: Sorts to Disk Sorts to Disk + Sorts in Memory
Sort Rows	Sort rows is the total number of rows sorted. AKA: sorts (rows)
Sorts in Memory	Number of sort operations that were performed completely in memory and did not require any disk writes. The sort operation can take place in memory if the sorted data can be fit into the memory block specified by the SORT_AREA_SIZE parameter. AKA: sorts (memory)
Sorts to Disk	Sorts that require creation of temporary segments on disk to store the intermediate sort results. This occurs if the data being sorted cannot be fit into memory block specified by the SORT_AREA_SIZE parameter. Obviously, disk sorts are several times slower than memory sorts and hence high values indicate that SORT_AREA_SIZE parameter should

	be increased. AKA: sorts (disk)
SQL*Net Wait	The wait time it takes for messages, data or break reset to be transmitted over the network interface to the client.
SQL NET Roundtrips To/From Client	Total number of SQL*Net messages sent to and received from the client.
SQL Executions	Number of statements executed during the current interval.
System Read/Write Operations	Number of system-wide physical I/O operations (both read and write operations). Physical operations require disk access.
Table Fetch By RowID	Number of rows that are fetched using a ROWID (usually recovered from an index lookup). Generally, table scans indicate either non-optimal queries or lack of proper indexing. Therefore, this metric should increase (while the value of TABLE SCANS ROWS GOTTEN metric decreases) as queries are optimized and schemas are tuned using indexes.
Table Fetch Continued Row	Number of times a chained or migrated row is encountered during a fetch. Retrieving rows that span more than one block increase the logical I/O by a factor that corresponds to the number of blocks that need to be accessed, a problem which can be eliminated by exporting and re-importing. To avoid or minimize the problem of chained or migrated rows, evaluate the settings for the storage parameters PCTFREE and PCTUSED in relation to

	the average row length. This problem cannot be avoided if rows are larger than database blocks (for example, if the LONG datatype is used and the rows are extremely large).
Table Scans Long Tables	Number of times Oracle has completely scanned a long table. A table is defined as long if: The table's high water mark is greater than five blocks or 2% of the db block buffer cache, or The table's high water mark is higher than CACHE_SIZE_THRESHOLD (v8.0 or earlier), or The table is defined without the CACHE option set (v8i and later). Generally, an excess of long table scans may indicate non-optimal table organization.
Table Scans RowID Ranges	During parallel query, the number of table scans conducted by the query slaves with the specified ROWID ranges.
Table Scans Rows Gotten	The number of rows scanned during full table scans. High values may indicate non-optimal queries and/or the absence of proper indexing.
Table Scans Short Tables	Number of times Oracle has completely scanned a short table. A table is defined as short if: The table's high water mark is less than five blocks or 2% of the db block buffer cache, or The table's high water mark is less than CACHE_SIZE_THRESHOLD

	(v8.0 or earlier), or The table is defined with the CACHE option set (v8i and later).
Total CPU Usage	Overall operating system CPU Usage (including Oracle).
Total Free Memory	Amount of free RAM in the system.
Total I/O Write Rate	Overall amount of I/O write operations performed by the entire operating system per time unit.
Total Kernel CPU Usage	Percentage of CPU time consumed by the operating system's processes (kernel mode activities).
Total Logical Reads	Number of system-wide logical reads. The metric counts both physical I/O reads which require disk access and reads which were satisfied entirely by the operating system's file cache.
Total Logical Writes	Number of system-wide logical writes. The metric counts both Physical I/O writes that require disk access and writes performed entirely in the operating system's file cache.
Total Memory Usage	Amount of memory consumed by entire operating system processes (including Oracle), both physical and swap.
Total Physical I/O Operations	Total number of disk operations (both read and write) performed by the operating system. This metric does not include operations that were satisfied using the file system cache.
	Number of system-wide physical

Total Physical Reads	reads. The metric counts Physical I/O reads that require disk access. Physical reads occur when the requested data is not present in the file system cache.
Total Physical Writes	Number of system-wide physical writes. The metric counts Physical I/O writes that require disk access.
Total User CPU Usage	Percentage of CPU time consumed by the operating system's processes (user mode activities).
Total Resident Memory Usage	Amount of physical memory consumed by entire operating system processes (including Oracle).
Total I/O Reads	Overall amount of I/O read operations performed by the entire operating system.
Transaction Rollback Ratio	The user rollback ratio indicates the rate at which application transactions are failing. Rolling back a transaction uses significant resources, and indicates that all the resources expended in executing the transactions have been wasted. The ratio is calculated as follows: User Rollbacks User Commits + User Rollbacks
User Calls	Number of user calls such as logon, parse, fetch, or execute. This metric value represents the rate at which the user application addresses Oracle.
	This rate indicates how well the application is managing its context areas. It measures the average

User Calls Per Parse	amount of calls in relation to statement parses. If it changes, then application change is the most likely explanation; but it may also indicate that usage patterns are changing and users are moving from one module to another, either more frequently or less frequently. An increasing value of this metric indicates that the effectiveness of the shared pool has improved, as fewer parses are needed. The ratio is calculated as follows: User Calls Parse Count
User Commits	Number of transactions committed during the current interval. Once a transaction is committed, the redo generated by that transaction reflects the changes made to database blocks and they must be written to disk.
User Rollbacks	Number of transactions that were rolled back. Rollbacks occur either because of manually issuing a ROLLBACK command or in the event of an error.
	Workload is also referred to as Active Time. Workload is a measurement of the Oracle instance's net processing time, measured in seconds. Workload is calculated by aggregating the net activity (non-idle time) for all Oracle processes. Performance Analysis considers an

Oracle process active when it is either:

- Consuming CPU
- Queued by the operating system, waiting for CPU to become available
- Marked by Oracle as active with a non-idle wait event

The Workload indication is used in various contexts to reflect application load; such contexts can be the overall Oracle instance load, individual session activity, program load over time, etc. and its interpretation may vary among the various totaling methods:

- <u>Total</u> Workload is measured in seconds. This representation can be used to compare loads.
- Average Workload <u>Per Second</u>— Workload is measured by Average Active Sessions. This representation reflects the average number of sessions that were active during a specified timeframe. This is the default representation for timeline graphs as it best reflects the session load on the instance.

For example:

 A value of 5 Average Active Sessions for the Oracle instances Workload indicates that during the specified timeframe, an average of 5 sessions were actively processing. Such an average

	can reflect:
Workload	 a. 10 sessions, each processing for only 50% of the time b. 20 sessions, each processing for only 25% of the time, or
	 c. 5 sessions, each processing throughout the specified timeframe.
	 2. A value of 0.75 Average Active Sessions for a specific session indicates that during the specified timeframe, the session was active 75% of the time. Values greater than 1 indicate that the session has been parallelized (parallel query) and is using more than a single query slave. 3. A value of 3 Average Active Sessions for a specific SQL Statement indicates that during the specified timeframe, an average of 3 sessions ran the SQL statement for the entire duration. Such an average can reflect: a. 6 sessions, each running the SQL statement for 50% of the specified timeframe, b. 12 sessions, each running the statement for 25% of the specified

timeframe, or

- c. 3 sessions, each running the SQL statement throughout the entire specified timeframe.
- Average Workload Per Execution— Workload is measured in Seconds Per Execution; Seconds Per Execution measures the average net active time for a particular SQL statement. This representation is useful for measuring or comparing the duration and impact of SQL statements. Note that this metric represents the average net processing time and therefore differs from the <u>Average SQL Duration</u> metric that measures the average elapsed SQL time.



Performance Analysis offers a rich set of sophisticated and user-friendly features that enable DBAs to rapidly and effectively focus their attention on application and database related performance issues, solve problems quickly and improve overall application responsiveness throughout the application lifecycle.

The following library of animated demonstrations shows how commonly occurring use cases can be resolved employing Performance Analysis.

DBA Use Case	Solution Methodology	Demo
Identifying Performance Deviation	Using the Review Performance Deviations hyperlink, we investigate causes and solutions to wait contention	<u>View</u> demo
Identifying real- time problematic session/statement consuming I/O resources	Using the Real-time Top Sessions tab, we can review the most I/O resource consuming sessions/statements.	<u>View</u> demo
Understanding an impact of schema changes on performance	Using the Change Tracking Tab in History mode, we can identify the database schema changes causing the execution plan change and its resulting performance degradation.	<u>View</u> demo

Diagnosing and resolving real time lock contention	Using the Real-time Blocking locks tab, we can review the sessions/statements generating the lock contention.	<u>View</u> demo
Diagnosing redo log contention	Using the Advisories hyperlink, the cause of contending redo log activity can be identified and addressed.	<u>View</u> demo
Shared pool latch contention	Using the Performance Advisories hyperlink, we investigate causes and solutions to shared pool latch contention	<u>View</u> demo
Determining origin of the CPU load	We review the system utilization metrics to determine the actual (Oracle or non-Oracle) source of CPU load.	<u>View</u> demo
Diagnosing Baseline deviations	Reconcile recent instance performance with any environmental updates using the Review Performance Deviation tab content.	<u>View</u> website
Analyzing resource metrics	Determining the reasons for resource load deviation using	View

deviation	analysis of instance performance metrics.	website
Optimizing buffer cache size	Review diagnosis and suggested Action Plan for optimizing buffer cache size when performance degradation is caused by sub- optimal buffer cache settings.	<u>View</u> website
Diagnosing and resolving buffer busy wait	Using the context- sensitive Performance Advisory analysis for diagnosing of buffer busy wait event and as a guide for remedial action.	<u>View</u> website
Diagnosing non- applicative (internal) locks	Using the Performance Advisories, we can determine the source of the non- applicative (internal) lock contention and review the recommended solutions.	<u>View</u> website
Diagnosing and	Using the context- sensitive Performance Advisory analysis to optimize redo log	View

resolving redo log contention	configuration to address the performance degradation caused by sub-optimal settings.	<u>website</u>
Identifying application component causing high redo load	Drill down to the most frequently executed statement causing redo log contention to understand the sudden increase in workload.	<u>View</u> website
Resolving shared pool latch contention	Using the context- sensitive Performance Advisory analysis for diagnosing the cause of shared pool latch contention and as a guide for optimal database settings and recommendations for application tuning.	<u>View</u> website
Diagnosing latch contention	Using the Real-time Top Sessions tab, we can review the statements/sessions generating the most intensive latch contention.	<u>View</u> website
	Using the Real-time Top Sessions tab,	

Diagnosing high network utilization	we can review the most network resource consuming sessions/statements	<u>View</u> website
Diagnosing and resolving resource and workload distribution across RAC nodes	Using both real-time and history cluster representation facilitates monitoring and diagnosing of RAC workload and resource distribution for appropriate tuning.	<u>View</u> website
Diagnosing RAC nodes workload balance by means of compare utility	Using the compare mechanism to isolate the load- imbalance causing resource.	<u>View</u> website
Analyzing instance workload against baseline using Compare utility	Using the compare mechanism to detail a specific time period's workload with that of the baseline.	<u>View</u> website
Using the Filter dimension feature to customize data display	Apply the dimension filtering feature to display only the most relevant data.	<u>View</u> website
Generating Executive Workload Summary report	For supervisory responsibilities, this mechanism provides a ready ability to access and distribute an immediate entire instance workload	<u>View</u> website

	scorecard.	
Analyzing Oracle Application resources usage (Historical)	Using the ERP functionality to determine which Oracle application dimensions were the greatest consumers of resources.	<u>View</u> website
Analyzing PeopleSoft resources usage (Historical)	Using the ERP functionality to determine which PeopleSoft dimensions were the greatest consumers of resources.	<u>View</u> website
Displaying Oracle Application session details (Real-time)	Using the ERP functionality, to understand currently running Oracle Application session details: application user, application client machine, module, etc.	<u>View</u> website
Displaying PeopleSoft session details (Real-time)	Using the ERP functionality, to understand currently running PeopleSoft session details: application user, application client machine, module, etc.	<u>View</u> website

Resolving Locks

Performance Analysis provides users the ability to review and investigate locking and blocking behavior for a monitored instance. Lock investigation and analysis can be performed both in real-time using the Recent View, and for past locks using the History View. For instance, if a database administrator is alerted to locking problems on the server, he can immediately open a Performance Analysis console, and using the Recent View, begin to investigate the cause of the blocking. Once the problem is remedied, the DBA can use the History View to return to the point in time when the blocking behavior was identified to see what factors contributed to the problem.

Below are step-by-step instructions on how to review locking behavior using Performance Analysis.

Real-Time Analysis

- 1. Click the **Real-Time** mode, to display the **Recent** tree **Instance View** node,
- 2. Directly to the right of the treeview, above the graphical workload timeline, change the **Resource** value to **Lock**:



- 3. Click the <u>Blocking Locks</u> tab to view the locking map. Use the displayed data to reveal lock and session parameters
- 4. Review the blocking locks. The sessions that are causing blocking problems are displayed in the list.
- Click the plus sign (

 to the left of the blocking session to display the locking map and reveal the BLOCKED processes (individual locks created by the blocking session). Only locks which actually resulted in a block are included. The lock icon (□) indicates a currently active

lock.

- 6. Click on one of the numeric hyperlinks presented to view the specific session's activity.
- 7. From within the **Recent** tree **Current Sessions** node, directly to the right of the treeview, above the graphical workload timeline, change the **Resource** value to **Lock**.
- 8. Individual user sessions are listed at the bottom of the screen. The Lock Wait column represents the cumulative time a given session has waited for a locked resource. Double-click one of the sessions showing lock activity.
- After double-clicking a blocked/locked session, it is added to the left treeview under Current Sessions. You are now viewing information about the BLOCKED/LOCKED session. Click the right-most tab, Session Locks to display both blocking and expired locks.

Historical Behavior

- 1. A lock wait alarm (♥) is generated either from the Home mode Lock <u>Oracle Performance</u> Resource Breakdown or from the Performance Advisories Locking metric.
- 2. A one-click drilldown opens the **History** mode period with offending period and lock resource in focus.
- 3. Click the <u>Blocking Locks</u> tab to display detail about the sessions blocking this session. Just like the Recent view, the data displayed applies to the BLOCKED/LOCKED session. By default, Performance Analysis groups this information by the locked object.
- 4. Use the information provided under the Blocking Locks tab to begin investigating the SQL statements executing under those dimensions during the same time period. This helps you determine the cause of the locking. You can drill-down through several dimensions to collect useful information and further narrow your search, for key parameters, such as each session's client machine, the executing program, the blocked SQL Statement, and the context information for the blocked session.

Home Page Common Graphical Elements

The graphical elements comprising the Home Page view are designed to facilitate a rapid and profound understanding of your Oracle instance's current performance, and to provide intuitive drill-down for root-cause problem solving:

- <u>Color-coded advisories</u> provide at-a-glance health measurement thresholds of metrics, advisories or total instance resource consumption and a hyperlink launch pad for further analysis.
- <u>Baseline</u> performance guidance in the Overview mode provides a normative range of workload and metric consumption
- Use the displayed buttons to access configuration of the Performance Analysis <u>client</u> and <u>agent</u>.
- Use the context-sensitive icon <u>toolbar</u> to configure the display and navigate among your browsed views.

Related Topics

Common Elements in the Drilldown Console

Toolbar

The context-sensitive Performance Analysis toolbar is located at the top of the main window. The toolbar provides you with quick access to commonly used commands and functions. Click an icon on the toolbar to carry out the command. To see a description of each icon, rest your mouse pointer over it. Information about the button is displayed in a ToolTip.

The following are common Performance Analysis buttons.

Click this icon	То	
昌	Print an <u>Executive</u> <u>Summary</u> or Analysis Report of this window	
國	Send an email of the window contents	
	Launch the <u>Compare</u> <u>tool</u>	
	Toggle display between workload breakdown of <u>resources</u> and <u>baseline</u> .	
	Toggle the present reporting period among the most recent:	
11/13/2005 3:14 PM - 11/13/2005 4:14 PM	 5 minutes 15 minutes 1 hour 6 hours 12 hours 24 hours 	
	Specify the default time frame <u>here</u> .	

Advisories Listing

The Database Performance advisories are tasks grouped as follows:

Optimize the Application	on Workload				
	This rule determines if bind variables are being used.				
	A SQL statement may contain variables that change from execution to execution. These variables are typically parameters to the SQL statement. They define the rows to be processed or new values to be inserted or updated. These variables can be specified either as literals or as bind variables. Using literals to retrieve details for employee 1234 could use the following SQL statement:				
	SELECT * FROM employees WHERE				
	employee_id=1234				
Bind Variables	A more efficient and scalable approach is to define the employee_id as a bind variable:				
	SELECT * FROM employees WHERE				
	employee_id=:empid				
	The value of the host variable empid will be supplied to the SQL statement when it is executed. When a SQL statement uses bind variables, it is not necessary to re-parse the statement every time the argument changes. When the SQL statement is executed in another session, the matching SQL will be found in the shared pool and parsing will be avoided.				
	Determine if cursors are being reused inefficiently.				
Cursor Management	When executing a SQL statement, the Oracle RDBMS assigns a private work area for that statement. This work area contains information about the SQL statement and the set of data returned or affected by that statement. The cursor is a mechanism by which that work area can be named and the information within it manipulated.				
	In its simplest form, a cursor can be thought of as a pointer into a table in the database.				
	Determine if overall database performance would benefit from resizing the shared pool.				
	The shared pool area of the SGA contains two primary areas: the library cache and the dictionary cache. It also contains control structures such as NLS loadable objects, session and process context areas, enqueue (lock) resources, etc. Additionally, multi-threaded server (MTS) and parallel execution (PX) are optional components that prefer to locate their respective memory areas within the large pool when defined. When a large pool is unavailable, the session memory of MTS and the query buffers of PX will reside in the shared pool.				
Shared Pool Size	The SGA is generally categorized into fixed and variable regions. The fixed region is usually quite small, less than 60KB. It primarily contains latch control structures. The variable region is where we find the shared pool. The shared pool's components grow and shrink based upon workload. Such components include the shared SQL area, the PL/SQL area, the dictionary cache, and control structures for latches, locks, and the current character set. These				

requires additional memory, another area must relinquish it. Performance degrades when the shared pool is undersized and such components compete for memory. Reduce Contention Determine if significant shared pool latch contention exists. Shared Pool Latch Shared pool latch contention can occur when the shared pool is sized too large to be effectively managed. This problem is less common staring with Oracle versions 0.1.6 when Oracle implemented improved shared pool management dia Unite to avoid shared pool latch contention. The overhead to track the memory chunks in a large shared pool inpacted the speed at which processes were able to obtain a needed chunk of memory. Library Cache Latch Determine if significant library cache latch contention exists. The library cache latches protect the cached SQL statements and the cached obtain a needed chunk of memory. Determine if significant library cache latch contention exists. Library Cache Latch Determine if significant reduce the library cache latch nust be obtained before Oracle can add a new statement to the library cache. During a parse request, Oracle scarbes the library cache latch, parse the SQL statement, and insead of bind variables. When library cache latch cours when an application sinstead of bind variables. When library cache latch cours when an application latch contention exists. Redo Allocation Redo allocation latches are used to control the allocation of space for redo entries in the redo log buffer. When redo information is garsing SQL at a high rate and may be suffering from excessive parse CPU overhead. Lis written to be log buffer there or splice here and allocation latch, performs the copy, and finaly releases the redo copy latc				
Shared Pool Latch Determine if significant shared pool latch contention exists. Shared Pool Latch Shared pool latch contention can occur when the shared pool management architecture. In previous Oracle versions, the shared pool management architecture. In previous Oracle versions, the shared pool management architecture. In previous Oracle versions, the shared pool management architecture. In previous Oracle versions, the shared pool management architecture. In uper shared pool impacted the speed at which processes were able to obtain a needed chunk of memory. Library Cache Latch Determine if significant library cache latch contention exists. The library cache latches protect the cached SQL statements and the cached object definitions held in the shared pool library cache. A library cache latch must be obtained before Oracle can add a new statement to the library cache. During a parse request, Oracle searchs the library cache latch, must be obtained before Oracle will acquire the library cache latch, parse the SQL statement, and insert the new SQL. Contention for the library cache latch contention exists. Redo Allocation Determine if significant redo allocation latch contention exists. Redo allocation latches are used to control the allocation is parsing SQL at high rate and may be suffering from vecessive parse CPU overhead SQL at high rate and may be suffering from vecessive parse CPU overhead Noracle versions 7.3 and 8.0, prior to writing a log buffer entry, Oracle comparse the redo allocation latch period space for redo antices in the redo allocation latch periods parse the redo allocation latch periods of the submas an edo copy latch, then obains the redo allocation latch, periods wa		degrades when the shared pool is undersized and such components compete for		
Shared Pool Latch Shared pool latch contention can occur when the shared pool is sized to large to be effectively managed. This problem is less common starting with Oracle version 8.1.6 when Oracle implemented improved shared pool management architecture. In previous Oracle versions, the shared pool management afcitecture, In previous Oracle versions, the shared pool amagement on the overhead to track the memory chunks in a large shared pool impacted the speed at which processes were able to obtain a needed chunk of memory. Library Cache Latch Determine if significant library cache latch contention exists. The library cache latches protect the cached SQL statements and the cached object definitions held in the shared pool library cache. A library cache latch must be obtained before Oracle can add a new statement to the library cache. During a parse request, Oracle searches the library cache latch, must be obtained before Oracle will acquire the library cache latch, parse the SQL statement, and insert the new SQL. Contention of the library cache latch, parse the SQL statement, and insert the new SQL. Contention of the library cache latch cache latch occurs when an application generates numerous unique, unsharable SQL (when literals have been used instead of bind variables). When library cache latch cache latdh oracle within your application. Misses on this latch can indicate that your application is parsing SQL at a high rate and may be suffering from excessive parse CPU overhead. Redo Allocation Determine if significant redo allocation latch protects against simultaneous writes to the same log buffer metry. Oracle compares the redo allocation latch protects against simultaneous writes to the same log buffer. The redo allocation latch protects against simultaneous writes to the same log buffer metry. There is only one redo allocation lat	Reduce Contention			
Redo Allocation In The library cache latches protect the cached SQL statements and the cached object definitions held in the shared pool library cache. A library cache latch must be obtained before Oracle can add a new statement to the library cache. During a parse request, Oracle searches the library cache for a matching statement. When one is not found, Oracle will acquire the library cache latch, parse the SQL statement, and insert the new SQL. Contention for the library cache latch, parse the SQL statement, and insert the new SQL. Contention for the library cache latch occurs when an application generates numerous unique, unsharable SQL (when literals have been used instead of bind variables). When library cache latch contention exists, improve the use of bind variables. When library cache latch contention exists, improve the use of bind variables within your application. Misses on this latch can indicate that your application is parsing SQL at a high rate and may be suffering from excessive parse CPU overhead. Redo Allocation Determine if significant redo allocation latch contention exists. Redo Allocation latches are used to control the allocation of space for redo entries in the redo log buffer. The redo allocation latch protects against simultaneous writes to the same log buffer memory. There is only one redo allocation latch per terists in the redo only one process can allocate space in the log buffer at a time. In Oracle versions 7.3 and 8.0, prior to writing a log buffer entry, Oracle compares the redo entry size to the LOG_SMALL_ENTRY_MAX_SIZE parameter. When the amount of redo is smaller than this parameter (defaults to 80 bytes), the redo copy takes place while holding the redo allocation latch. When the redo entry size is larger than this parameter (defaults to 80 bytes), the redo allocation latch, performs the coy, and finally releases	Shared Pool Latch	Shared pool latch contention can occur when the shared pool is sized too large to be effectively managed. This problem is less common starting with Oracle version 8.1.6 when Oracle implemented improved shared pool management architecture. In previous Oracle versions, the shared pool management did little to avoid shared pool latch contention. The overhead to track the memory chunks in a large shared pool impacted the speed at which processes were able		
Redo Allocation LatchRedo allocation latches are used to control the allocation of space for redo entries in the redo log buffer. When redo information is generated, it is written to the log buffer. The redo allocation latch protects against simultaneous writes to the same log buffer memory. There is only one redo allocation latch per instance and only one process can allocate space in the log buffer at a time.Redo Allocation LatchIn Oracle versions 7.3 and 8.0, prior to writing a log buffer entry, Oracle compares the redo entry size to the LOG_SMALL_ENTRY_MAX_SIZE parameter. When the amount of redo is smaller than this parameter (defaults to 80 bytes), the redo copy takes place while holding the redo allocation latch. When the erdo allocation latch, performs the copy, and finally releases the redo copy latch. In 8i, Oracle removed the log_small_entry_max_size and always uses redo copy latches by default.Determine if significant redo copy latch contention existsWhen redo information is generated, it is first written to the log buffer before being written to disk. When the LOG_SIMULTANEOUS_COPIES (renamed _LOG_SIMULTANEOUS_COPIES in 8i) parameter is set to a value greater than 0, Oracle uses redo copy latches to reduce contention for the redo 	Library Cache Latch	The library cache latches protect the cached SQL statements and the cached object definitions held in the shared pool library cache. A library cache latch must be obtained before Oracle can add a new statement to the library cache. During a parse request, Oracle searches the library cache for a matching statement. When one is not found, Oracle will acquire the library cache latch, parse the SQL statement, and insert the new SQL. Contention for the library cache latch occurs when an application generates numerous unique, unsharable SQL (when literals have been used instead of bind variables). When library cache latch contention exists, improve the use of bind variables within your application. Misses on this latch can indicate that your application is parsing		
When redo information is generated, it is first written to the log buffer before being written to disk. When the LOG_SIMULTANEOUS_COPIES (renamed _LOG_SIMULTANEOUS_COPIES in 8i) parameter is set to a value greater than 0, Oracle uses redo copy latches to reduce contention for the redo allocation latch. The redo allocation latch is used to prevent another process from writing in the same location of the log buffer. There is only one redo allocation latch per instance and only one process can allocate space in the log buffer at a time.Redo Copy LatchIn Oracle versions 7.3 and 8.0, prior to writing a log buffer entry, Oracle		Redo allocation latches are used to control the allocation of space for redo entries in the redo log buffer. When redo information is generated, it is written to the log buffer. The redo allocation latch protects against simultaneous writes to the same log buffer memory. There is only one redo allocation latch per instance and only one process can allocate space in the log buffer at a time. In Oracle versions 7.3 and 8.0, prior to writing a log buffer entry, Oracle compares the redo entry size to the LOG_SMALL_ENTRY_MAX_SIZE parameter. When the amount of redo is smaller than this parameter (defaults to 80 bytes), the redo copy takes place while holding the redo allocation latch. When the redo entry size is larger than this parameter, Oracle first obtains a redo copy latch, then obtains the redo allocation latch, allocates the space, releases the redo allocation latch, performs the copy, and finally releases the redo copy latch. In 8i, Oracle removed the log_small_entry_max_size and		
COMPARES DE LEOD ENTRY SIZE TO THE LAACE SIVIALLE ENTRY IVIAA SIZE	Redo Copy Latch	When redo information is generated, it is first written to the log buffer before being written to disk. When the LOG_SIMULTANEOUS_COPIES (renamed _LOG_SIMULTANEOUS_COPIES in 8i) parameter is set to a value greater than 0, Oracle uses redo copy latches to reduce contention for the redo allocation latch. The redo allocation latch is used to prevent another process from writing in the same location of the log buffer. There is only one redo allocation latch per instance and only one process can allocate space in the log buffer at a time.		

	(defaults to 80 bytes), the redo copy takes place while holding the redo allocation latch. When the redo entry size is larger than this parameter, Oracle first obtains a redo copy latch, then obtains the redo allocation latch, allocates the space, releases the redo allocation latch, performs the copy, and finally releases the redo copy latch. In 8i, Oracle removed the log_small_entry_max_size parameter and always uses redo copy latches by default.
Cache Buffers Chains (CBC) Latch	Determine if significant cache buffers chains latch contention exists. Cache buffers chains (CBC) latches enforce synchronized updates to data in the Oracle buffer cache. Data blocks in the buffer cache are managed in singularly linked lists of data block headers (sometimes referred to as buckets). Data block headers contain information about the data block including the memory address of the block in the buffer cache. To provide efficient data block access, Oracle computes a hash algorithm on the dat block address (DBA - an encoding of file and block number) to identify the cache buffers chain (one of many) that may contain the block. The chain is then scanned to find the corresponding header. When the header is located, the buffer cache address is available and the block can be retrieved. This process is much faster than scanning the entire buffer cache for the requested block. When a block header is not found in its corresponding cache buffers chain, Oracle reads the data block from the disk, copies it to the buffer cache, and adds its header to the appropriate cache buffers chain. While Oracle scans and updates a cache buffers chain, a CBC latch protects the chain to ensure that block headers are not being added or removed from the list simultaneously by other users. When the scan process begins, the Oracle process "grabs" and holds the CBC latch for the corresponding cache buffers chain and thus prevents other users from changing block headers in all chains protected by the held latch. While the latch is held, other users wanting to scan a cache buffers chain protected by the same latch will wait until it is released. The fewer the number of block headers maintained in a cache buffers chain, the faster the search for a buffer header can be completed, and the less time the latch will be held. Optimally, other users should not have to wait long to access the same cache buffers chains latch.
CacheBuffers LRU Chain Latch	Determine if significant cache buffers LRU chain latch contention exists. The cache buffers LRU chain latch is held when updates are made to the LRU (Least Recently Used) lists. Two LRU lists are used to track database blocks. The first list, the LRUW, tracks "dirty" (modified blocks not yet written to disk) block buffers. The second list, the LRU, tracks "clean" (unmodified) block buffers. Both lists track the least recently used blocks. When a block is used or freed, it goes to the top of the list, and the blocks below are moved down the list. After the dirty blocks on the LRUW reach a certain level, they are written to disk and placed on the LRU list containing clean block. When a clean block is needed, the LRU list is searched. When a clean block is modified, it is placed on the LRUW list for writing by the DB writer. Cache buffers LRU chain latch contention is caused when the LRU and LRUW lists are updated and the chains of buffers in each hash bucket are too numerous to be scanned quickly. Contention is also caused when the volume of transactions overloads the block tracking on the LRU lists. The solution to either of these cases is to allocate the resources needed to update the LRU lists quickly. Because latches are used to protect the buffer cache, very high rates of logical I/O - possibley due to untuned SQL - will increase the rate and duration of latch holds and increase the probability that latch contention will occur.
	This rule determines if significant lock contention exists.

Locking	Lock (enqueue) waits occur when a process must wait to update data when another process is already updating it. Locks are mechanisms used to provide ordered access when updating data. Oracle also uses locks for maintaining internal data structures and processes may have to wait for access to these internal locks as well (for instance, the Space Transaction enqueue). When the database is well tuned and the application design sound, lock waits should be negligible.
Tables Experience Lock Waits	Determine if transaction lock waits for one or more tables were observed. The purpose of a lock within the database is to protect a database resource (in this case a table) that may be needed for a relatively long time compared to latches, which we expect to be held only briefly. The Oracle database locking mechanism allows multiple sessions to wait in line for a given table if that table is currently locked by another session in a mode that is incompatible with the
	lock being requested. What is meant by incompatible? For example, a share lock request for a table that already has another share lock on it will likely result in a compatible locking situation, and therefore no locking delays will occur. If, however, a session requested an exclusive lock on a row, another exclusive lock request on that same row will likely have to wait, because the two locks are incompatible.
	Non-Indexed Foreign Keys
	Foreign key constraints are used to maintain referential integrity between tables related by a common column or columns. For example, an EMPLOYEE table (child) has a foreign key constraint defined on the DEPTNO column. The foreign key ensures that the DEPTNO value in the EMPLOYEE table matches a value in the DEPTNO column of the DEPARTMENT table (parent).
	Violations of referential integrity are prevented by making sure that no uncommitted inserts or updates to child tables could be affected by the update or deletion of the foreign key in the parent table (DEPTNO in the DEPARTMENT table). When the child table foreign key is indexed, the index can be checked to determine if any child table rows are affected. When there is no index on the child table foreign key, the transaction updating or deleting the foreign key in the parent table locks the entire child table. The lock on the child table will block any other transactions on the child table attempting to insert, update, or delete until the parent table transaction completes.
	In Oracle 9i, the problem of child table locking due to un-indexed foreign keys still exists but has been reduced through downgrading the exclusive lock to a shared lock after the update completes. When the transaction completes, the shared lock is released. This reduces the duration that the exclusive lock is held.
Space Transaction Waits	Determine if significant space transaction lock wait exists. If so, identify users that have permanent tablespaces defined for use as temporary storage.
	When Oracle modifies the segments allocated to objects, it grabs and holds the space transaction lock until the modification completes. Excessive segment modification is especially noticeable when permanent tablespaces are assigned for use as temporary storage, significant disk sort operations are occurring in those tablespaces, and the default size for the tablespace is set too low. It may also occur when rollback segments are growing and shrinking excessively. Oracle addressed this issue with the introduction of tablespaces of type
	TEMPORARY and further improved the situation with the introduction of Temporary tablespaces in version 8i.

	Assigned Temporary Tablespace
	Each database user is assigned a tablespace to use for temporary storage operations. Temporary storage is necessary when performing large sorts, large hash joins, and other operations. However, sort operations are typically the primary use for temporary storage. When sort operations are too large to be done entirely in memory, temporary disk storage is utilized to store intermediate sort results. Sort operations are necessary for such purposes as completing ORDER BY or GROUP BY operations, sort-merge join operations, and for index creation.
	When creating a tablespace, it can be designated for use as temporary-only storage. By doing this, Oracle more efficiently manages the utilized space for temporary purposes. Temporary tablespaces should always be specified when assigning tablespaces to users for temporary space usage.
	Determine if there are enough redo log disks defined and if they are optimally sized.
Redo Log Disk Configuration	When a transaction is committed, a physical write to the redo log file must occur. The write must complete before the COMMIT returns control to the user and therefore, the writing of the redo log imposes a limit to throughput of update intensive applications. Redo log I/O is optimized when the log is located on a dedicated device where there is no contention with other processes. By not having other processes using the disk device, the disk head maintains its position while awaiting the next write request. Thus, the disk does not need to take the time to "seek" before performing the next write and thus write times are minimized.
	Determine if the database has a sufficient number of redo logs.
	A log switch checkpoint must complete before a redo log group can be used. This checkpoint ensures that all transactional information in the redo log buffer has been written to disk.
Number of Redo Logs	When in ARCHIVELOG mode, redo log groups must be archived before they can be reused. When the archiving process is sufficiently slow (usually due to disk device contention) and the archiver has not yet completed archiving a redo log group, the database will halt and transactions will wait for the archive process to complete before allowing the LGWR (log writer) process to re-use a redo log group.
	The database cannot be modified before the checkpoint or the archive processes have completed. Avoiding these waits is therefore very important to ensure adequate throughput and response time.
	The best way to guarantee that you do not experience waits for checkpoints or for archiving is to ensure that there are enough redo log groups. This allows checkpoints and archiving to complete before the logs are scheduled for reuse.
	Determine if redo logs are sized too small.
	When a redo log is full, Oracle must perform a "log switch." Redo logs should be large enough to avoid too frequent log switches and any corresponding log switch waits.
	Log switch waits can occur under the following conditions:
Redo Log Size	 The log switches are too frequent. The log switch checkpoint is not complete.

	• The database is in ARCHIVELOG mode and the log being switched to
	is not yet archived.
	Redo logs should be sufficiently large (and numerous) to avoid these conditions.
	Determine if resizing the redo buffer log will improve performance.
Redo Log Buffer Size	The redo log buffer stores changes that will eventually be written to the redo log files. It buffers these writes so that the user session can continue working without requiring a disk I/O. When a COMMIT is issued, changes in the redo log are written to a disk by the log writer before the COMMIT is completed. When the log buffer reaches 1/3rd full, the log is written to disk even without a COMMIT.
	Determine if sequence generators have an optimal cache size.
Sequence Cache Size	Oracle sequence number generators utilize a built-in caching mechanism to gain their substantial efficiency over an approach of storing and retrieving sequence values from a user table. This is because reading and updating a sequence value in memory is much faster than updating one stored on disk. Oracle sequence generators are cached in the shared memory area (the SGA). By default, 20 numbers are cached. Once the 20 cached numbers are used by the application, Oracle caches another 20 values and updates Oracle's internal sequence table. When sequence numbers are allocated at a high rate, performance can be improved by increasing sequence cache size. A larger cache size reduces the number of required read/writes to the sequence table. Increasing the cache size does not consume additional memory in the SGA. However, it does increase the number of sequence numbers that can be "lost" if the cache is flushed or the database shutdown. Most applications do not require a complete set of contiguous numbers and therefore, "lost" sequence numbers do not pose a problem. If a specific application does require complete accounting for a set of sequence numbers, take care in designing the application code to avoid "loss" of numbers, including when transactions are rolled back. A complete discussion of such design is beyond the scope of this topic.
	Determine if database buffers are experiencing contention.
	Buffer busy waits happen when multiple sessions want concurrent access to a block in the buffer cache. The modifying session marks the block's header with a flag letting other users know a change is taking place and to wait. This temporary blocking ensures the reader has a coherent image of the block (with either the before or after image). The two main cases where this wait can occur are:
	• Another session is reading the block into the buffer
	• Another session holds the buffer in an incompatible mode
	While the block is being changed, the block is marked unreadable. The changes should last under a few hundredths of a second. A disk read should be under 10 to 20 milliseconds and a block modification should be under one millisecond. Many buffer busy waits need to occur before performance degradation is noticeable.
Buffer Busy Wait	However, in a problem situation, there is usually a hot block, such as the first block on the free list of a table with high concurrent inserts. Users insert into that block at the same time, until the block exceeds PCTFREE, then users insert

	into the next free block on the list, and so on.
	Description of Freelists
	Freelists are structures which are included within each table. They list the blocks that are free to be used for inserts. These structures are maintained in the table header block (unless multiple FREELIST_GROUPS are defined). Multiple freelists can be configured with the FREELISTS clause in the CREATE TABLE and (in 8.1.6) the ALTER TABLE statements. Configuring multiple freelists is an important optimization for tables that are going to be subjected to high rates of concurrent inserts.
	A buffer busy wait occurs when an oracle session wants to modify or pin a buffer (copy of a database block in the SGA) but cannot because another session holds the buffer in an incompatible mode. The V\$WAITSTAT virtual table records the number and duration of "buffer busy" waits, broken down by the class of buffer for which waits are occurring.
	When there are insufficient freelists, multiple sessions are allocated the same block in which to perform their insert When they both insert into this data block concurrently, one session performs the insert and the other experiences a "buffer busy wait" on "data block".
	Determine if sufficient rollback segments are defined.
Rollback Segments	When Oracle transactions make database changes, the information required to undo those changes is written to a rollback segment. Oracle ensures that the available rollback segments are evenly used. However, when too many simultaneously active database transactions are using the same rollback segment, the generation of rollback information can be inefficient.
	Determine if the current workload across the interconnect in a Real Application Cluster (RAC) environment is an indicator of any network latency.
	RAC uses cluster interconnects to transfer blocks between the nodes participating in the cluster. A block transfer occurs when a user session connected to one instance requests a block being held in the cache of another instance. This feature of RAC to transfer information from the cache of one instance to the cache of the requesting instance is called cache fusion.
	Oracle introduced Cache Fusion with Oracle 9i. Prior to Oracle 9i under Oracle Parallel Server, data was shared between users by forcing the instance holding the data to first write it to disk so that the requesting instance could then read it. With Cache Fusion, when users on one instance request data held in cache on another instance, the holding instance transfers the data across a cluster interconnect and avoids any writes and reads to disk. Disk I/O is significantly slower than cache transfers via the cluster interconnects.
	The performance of the cluster interconnects is crucial to the performance of the RAC cluster and more specifically to the movement of cached data between instances. Its performance is measured by determining the average time a block takes to reach the requested node i.e., from the moment that a block was requested by an instance to the moment the requesting instance received the block. As in any application or database instance, occasional spikes of user activity are expected. However, when the average of such spikes remains high for an extended period of time, it could indicate a correctable performance degradation of the database cluster.
	While high average GCS CR block receive time indicates possible interconnect performance issues, tuning and monitoring the following areas may help improve the performance of the cluster interconnects.

NOTE: On certain platforms such as SUN, enabling the

	CLUSTER_INTERCONNECT parameter could disable the high availability feature of RAC.
	DISCLAIMER: Hidden parameters (those that begin with an underscore) such as _LM_DLMD_PRCS should be modified with caution and only after first consulting with Oracle Support. It should be noted that Oracle could change or remove such parameters without notice in subsequent releases.
	While single instance performance remains important, in a RAC or Oracle clustered environment, applications and users share multiple instances connected to the same physical copy of the database and the cluster interconnects plays a very critical role in the performance of these systems.
Minimize Physical Disk	
	Determine if resizing the buffer cache will improve performance.
	Reading a data block from a physical disk device is thousands of times slower than reading it from memory. A disk read is avoided when a data block is already in the buffer cache. Avoiding disk reads in this manner is one of the most effective optimizations that can be made to an Oracle database. Allocating more memory to the buffer cache improves the chance that a data block will be in memory when requested a second time. An often used measure of evaluating whether the buffer cache is sized too small is the buffer cache "hit ratio." It expresses how often a requested block is found in the buffer cache without requiring a disk read. The buffer cache "hit ratio" is calculated by dividing the number of reads that did not require a disk read by the total number of reads requested. Often, when the hit ratio is low, or the absolute number of disk reads is too high, increasing the size of the buffer cache can significantly improve disk performance.
	Interpreting the Buffer Cache Hit Ratio While a low buffer cache hit ratio can indicate a need to increase the cache size, there are other factors to consider:
	• Repeated scanning of large tables lowers the buffer cache hit ratio.
	Increasing the cache size here will have little effect on the hit ratio
Buffer Pools	because these blocks are unlikely to be reused.
Builer Foois	• A high cache hit ratio can conceal inefficient SQL - when the
	application frequently scans an unselective index it is more likely to
	find these blocks in the cache and the hit ratio will be inflated.
	• When adding multiple buffer pools to Oracle, consider the reasons for
	allocating objects to each pool. The KEEP pool is for small tables that
	are frequently scanned and should have a high hit ratio. The
	RECYCLE pool is allocated for large infrequently scanned tables and
	prevents these tables from occupying unnecessary space in the cache.
	The DEFAULT pool is for the tables that are subject to index-lookup.
	Oracle introduced a buffer cache advisory in 9i. The Oracle buffer cache advisory option is used to evaluate the need to adjust buffer cache sizes. In order to utilize this option, it must be enabled. When enabled, the Oracle

	statistics that are used to estimate the effects of resizing the buffer cache are displayed.
	The estimated physical reads per buffer cache size are populated in the V\$DB_CACHE_ADVICE view. This view contains 20 rows for each pool present in the buffer cache. Each row represents a possible cache size and includes the estimated physical reads for that size. These estimates are based on the database workload from the time the advice is turned on.
	Determine if adjusting the sort area size may improve performance
	Increasing the sort area size allocates more memory for sort operations and thus can improve overall performance by reducing the need to use slower disk storage to perform sorts. The sort area size is specified by setting the SORT_AREA_SIZE configuration parameter. Sort operations in Oracle are of two types and commonly referred to as either disk sorts or memory sorts. SORT_AREA_SIZE is an Oracle configuration parameter that specifies the memory allocated for sort operations and effectively determines which type of sort is performed. When the data volume of the sort operation is small enough to fit in the allocated sort memory, the sort can be performed using only memory and no disk storage. Otherwise,
	temporary disk storage must be used to perform the sort. SORT_AREA_SIZE defines the maximum size, in bytes, of the memory that can be allocated to each user session for sorting. The allocation of sort memory only occurs when the session first performs a sort. The size of memory allocated is the amount of memory sufficient to perform the sort up to the SORT_AREA_SIZE maximum. SQL statements that use the ORDER BY or GROUP BY clauses and index creation are examples of activities that require sort operations.
Sort Area Size	The value of SORT_AREA_SIZE is limited by the physical memory of the computer running the Oracle database. When the machine on which the database resides has an abundance of memory, the value of this configuration parameter can be increased beyond the default.
	Note: When using the cost-based optimizer, it is possible that increasing the sort area size may result in decreased performance contrary to the usual affect of a performance increase. The cost-based optimizer considers the size of the sort area when deciding which join operation to use for joining two tables. Increasing the sort area size may cause the optimizer to select a sort-merge operation over a nested loops operation when it believes the presence of more sort area space will result in better performance. On some few occasions, this optimizer choice may cause reduced performance. This adverse affect can typically be corrected by adding hints to the one or two SQL statements that exhibit this affect. Although this performance effect is unlikely to occur in most databases, always test performance modifications in a test environment to ensure they have the desired effect in the environment.
	Oracle 9i introduced new automatic memory management for work area memory allocation. Automatic memory management can be set on or off. When set on, you no longer have to set values for the various *_area_size parameters (sort_area_size, hash_area_size, bitmap_merge_area_size and create_bitmap_area_size). Oracle allocates appropriate work areas and keeps total memory allocation within the amount specified by the pga_aggregate_target parameter. To enable automatic work area memory allocation, set the pga_aggregate_target parameter greater than 0 and set the workarea_size_policy parameter to "AUTO".
	Determine if resizing the PGA will improve performance.

Automatic SQL Execution Management Concepts

Oracle allocates a private process memory area for each server process called the Program Global Area or PGA. This memory will be allocated whether the type of connection is dedicated or shared. The contents and location of the memory will vary depending upon the type of connection. The table below lists the two main components and their corresponding memory locations for dedicated and shared servers.

PGA Contents	Description	Memory location for dedicated servers	Memory location for shared server
Session memory	Holds session's logon information and other details regarding session.	PGA	SGA
SQL Execution memory	Data such as bind information and runtime memory structures	PGA	Part in SGA an PGA
Persistent area	Holds the cursor information that persists across multiple executions of the same statement such as, bind details, data type conversion, etc. The persistent area is deallocated when the cursor is closed.	PGA	SGA
Runtime area	This area contains information used while a SQL statement is being executed. Its size depends on the type and complexity of SQL statement as well as the number and size of rows being processed. A large portion of this memory is dedicated as the working area for operations such as sort, hash-join, bitmap-merge etc.	PGA	PGA for DML/DDL SGA for querie

From this table you can see that SQL execution memory is the most important component in the PGA. The size will vary depending upon the complexity and data size of the SQL being executed. Obviously the DBA would be more inclined towards tuning this component to reduce response time because of the complex SQL operations like sorting, hash-joins etc. Before Oracle9i a DBA had to depend upon initialization parameters such as SORT_AREA_SIZE,

PGA Size

HASH_AREA_SIZE etc. to adjust fixed sizes. Of course these parameters are not adjusted based on the load of the system.

The new SQL memory management introduced in 9i will solve these issues. A DBA can initially start the instance with a predefined PGA target size using the initialization parameter PGA_AGGREGATE_TARGET. Oracle will allocate appropriate run time memory for the SQL operators keeping this target size as its high water mark. By this Oracle avoids the abnormal growth of the total PGA size. In addition, Oracle adapts the SQL operation's size according to the current load and complexity of the instance. Unfortunately this new SQL memory management is available only for dedicated servers, so DBAs should still depend on parameters like SORT_AREA_SIZE, HASH_AREA_SIZE etc., for shared servers.

How does SQL memory management affect the performance? Generally speaking, a larger working area can significantly improve the performance of SQL operations and therefore reduce the response time of queries. Ideally, the size of the work area should be big enough so that it can accommodate all input data and auxiliary structures needed by the operation. Oracle allocates three types of memory sizes for SQL operations.

		Cache size	It is the amount of memory required to run the
			entire SQL operation in memory.
			Cache size optimizes performance.
		One-pass	It is the amount of memory required to prevent
		size	more than one pass through the input data.
			Response time increases if only part of input data
			can be accommodated.
		Multi-pass	It is the amount of memory allocated for a small
		size	part of the input data necessitating multiple passes
			over input data.
			Results in a drastic increase in response time.
		Using PGA_A	GGREGATE_TARGET as the high water mark for the
		total PGA size,	Oracle allocates cache, one-pass, or multi-pass sizes
		for the PGA. I	f load increases, Oracle will initially try to allocate the
		one-pass size r	ather than the cache size. For these size calculations
		Oracle uses a "	Feedback loop mechanism" to maintain total memory
		allocations wel	l below the PGA target size.
Optimize Physical Disk	: I/O		
	Determine if I/O load is balanced across datafile disk devices.		

Datafiles should have similar I/O service times. When datafiles have very different I/O service times, it indicates that the I/O load across datafile disk devices may not be balanced. Following are common reasons why a device/filesystem may have longer service times than another:

- It may be physically slower (IO per second rating or average service time rating).
- It may be servicing simultaneous requests from multiple processes

and/or datafiles (contention).

- It may not be using caching (or better caching algorithms).
- Other devices may be using RAID or other advanced technology disk architectures.

What is RAID?

	RAID (Redundant Array of Independent Disks) arrays are an increasingly popular way of delivering fault tolerant, high performance disk configurations. There are a number of factors to take into consideration when deciding upon a RAID configuration and RAID level to implement. There are three common levels of RAID provided by storage vendors:
Datafile I/O Load Balance	RAID 0 Sometimes referred to as "striping" disks. In this configuration, a logical disk is constructed from multiple physical disks. The data contained on the logical disk is spread evenly across the physical disk and hence random I/Os are also likely to be spread evenly. There is no redundancy built into this configuration so if a disk fails it will have to be recovered from a backup.
	RAID 1 Referred to as disk "mirroring". In this configuration a logical disk is comprised of two physical disks. In the event that one physical disk fails, processing can continue using the other physical disk. Each disk contains identical data, and writes are processed in parallel so there should be no negative effects on write performance. Two disks are available for reads so there can be an improvement in read performance.
	RAID 5 In this configuration, a logical disk is comprised of multiple physical disks. Data is arranged across the physical devices in a similar way to disk striping (RAID 0). However a certain portion of the data on the physical devices is parity data. This parity data contains enough information to derive data on other disks should a single physical device fail.
	It's common to combine RAID 0 and RAID 1. Such striped and mirrored configurations offer protection against hardware failure together with spread of I/O load.
	Performance implications of RAID
	Both RAID 0 and RAID 5 improve the performance of concurrent random reads by spreading the load across multiple devices. However, RAID 5 tends to degrade write I/O performance since both the source block and the parity block must be read and then updated.
	Determine if a sufficient number of database writer processes or slaves are configured.
	The database writer process (DBWR) is the only process that writes modified database blocks from the buffer cach to the database files. The database writer works asynchronously. This means that a user process never needs to wait for the database writer to finish writing. However, if the DBWR falls behind sufficiently, then the buffer cache will fill up with "dirty" blocks and waits will occur while user processes try to read new blocks into the cache. These bottlenecks can be eliminated by configuring multiple database writers/slaves or by enabling asynchronous I/O.
	Keeping the database writer optimized is critical to maintaining database throughput. The best way to optimize database writer throughput is to spread I/O across multiple disk devices and allow the database writer to write to these disk devices in parallel.
B	Parallel database file writes can be achieved using the following methods:
Database Writer Processes	• Multiple database writers can be configured using the DB_WRITERS

configuration parameter in Oracle7, or the
DB_WRITER_PROCESSES or DBWR_IO_SLAVES parameters in
Oracle8+.
• Operating system asynchronous I/O or list I/O can be enabled. This
will allow the database writer to issue write requests against multiple
disk devices simultaneously.
Operating system asynchronous I/O performs more efficiently than multiple database writers. However, asynchronous I/O may not be available on all platforms or may require special measures. In some operating systems, asynchronous I/O may require building the database on raw devices. In large systems undergoing heavy write activity, multiple database writers can improve performance even when asynchronous I/O is in use.
Determine if sufficient disk devices are allocated for database files.
The amount of physical (non-cached) I/O that can be performed by the database server is constrained by the following factors:
• The number and type of disk devices allocated to the database
• The I/O rate each disk can sustain
• The degree to which database I/O is evenly spread across disks
• The degree to which database I/O is evenly spread over time
• Contention for database disk devices by non-Oracle processes
For example, suppose there are 3 disk devices, each capable of maintaining a sustained I/O rate of 100 I/Os per second. When the I/Os are spread evenly across these devices, they are able to sustain a physical I/O rate of 300 I/Os per second. link IOOPT">Click Here for ideas that may reduce I/O load.
In most cases, reducing the number of physical I/O requests (by tuning the application or configuring a larger buffer cache) is preferable to adding disk devices. Adding devices can only spread existing I/O load, whereas tuning SQL or application code can eliminate unnecessary or reduce excessive I/O. Memory access is significantly faster than physical I/O; note that memory access times are measured in nanoseconds while disk access times are measured in milliseconds. link BCOV>Click Here for ideas on reducing I/O load by tuning the buffer cache.
Datafile devices are assumed to be dedicated to this Oracle instance. Oracle collects statistics for the I/O requests that it makes. Operating system utilities such as iostat and sar can measure all activity against disk devices.
Devices dedicated to Oracle include drive controllers as well as disk drives. Ensure the operating system and other applications are not causing device contention. A thorough understanding of the network topology and contention points may assist tuning.

Recent Storage Technology Advancements

Since it is not possible to determine the physical characteristics of the disk devices during analysis, it is important to understand the database storage topology. Without such understanding, it is difficult to plan or properly tune

Number of Datafile Disk Devices	 device utilization. There have been several recent advancements in storage technology that make higher I/O rates possible. A basic understanding of disk device concepts is required before moving existing datafiles or adding more devices to spread existing or anticipated load. A common question when planning for storage is whether more smaller disks is better than fewer larger disks. The correct answer is: it depends. Cost and throughput are usually deciding factors. As storage capacity has increased significantly, the average access time has not. Calculations should be made regarding cost vs. performance of the usable portions of the disk. See the ZBR discussion later in this topic. Also consider disk controllers, RAID, and other redundancy technologies into the overall plan. Sixteen 9 GB disks in the environment may or may not have better cost/throughput than two 72 GB disks. Disk drives are commonly being grouped together in what are called storage systems or network appliances. These technologies include RAID arrays, SAN, NAS, and hybrids thereof. Specific enhancements have been made to each of these technologies to support database systems. Many SAN and NAS products available today incorporate RAID technology, yet there can be RAID without SAN or NAS. In general, SAN is for block-based and transaction-based applications. NAS is more geared toward file-based systems. Talk with the system administrator or hardware vendor to decide whether NAS or SAN is right for the business needs. Enhancements have been designed into dedicated SAN/NAS hardware with database features in mind such as parallel server, backup/recovery, replication, and VLDB. Another benefit of these technologies is that they offload mirroring (for example the redo logs) to the hardware – further increasing performance. Caching data at the operating system (OS) or storage device-level for the purposes of decreasing read I/O service time may not help much in an Oracle environment. This is because Oracle
	On the other hand, when a storage device caches write requests, it improves the performance of the database writer (DBWR) process(es) by allowing write requests to complete sooner. Some hardware configurations result in very low write times. For example, a single DBWR with an advanced disk write-cache can easily show write times of less than 1 millisecond per write request.
	A logical volume manager (LVM) is disk management software that groups arbitrary physical disks into virtual disk volumes. Depending upon configuration, it can reduce overall access times by spreading load across multiple physical devices. Attempting to spread datafiles across LVM-managed disks without understanding how the disks are configured could prove counter- productive in I/O tuning efforts. Work with the system administrator to understand their configuration.
	For random I/O, disk seek/latency time is reduced and I/O throughput is increased for data stored on the outer-portion of a disk formatted using zone bit recording (ZBR) technology. Simply put, the outer-edge of a disk contains more "data" than the inner-edge. Most modern disks can be logically partitioned such that Oracle data files can be located in a partition that only spans the outer tracks of a disk. It is quite possible to receive better throughput from partitioning and only using the outer-most 25% of a 72 GB drive at 7200 RPM, rather than an 18 GB drive at 10,000 RPM.
	Finally, one additional technology that can add big value when implemented appropriately is solid state disks (SSDs). The primary advantages of SSDs are their ultra low latency servicing requests and their virtually nil contention, due to the lack of physical head movement. However, the per-GB cost of SSDs is

	relatively high compared to standard disks. Solid state disks are capable of 10,000 to 1,000,000+ I/Os per second. Some good uses for SSDs include:
	• Redo logs – by providing low latency writes and high-bandwidth reads
	(for archiving). Particularly useful in OPS/RAC environments where
	multiple servers can share the same SSD device. This makes cost
	justification easy – 1 SDD rather than 2 standard disks (excluding
	RAID protection) for each node in the RAC.
	• Temporary tablespaces – by providing low latency and high
	concurrency. Primarily used when sorting memory exceeds
	SORT_AREA_SIZE or for scratch space when using global temporary
	tables.
	• Undo/Rollback tablespaces – by providing low latency.
Review Performance D	eviations
CPU Usage Deviation	
CPU Wait Deviation	
I/O Wait Deviation	
Latch Wait Deviation	
Lock Wait Deviation	
Other Deviation	
Workload Deviation	