

Performance Diagnosis Demystified: Best Practices for Oracle Database 10g

*An Oracle White Paper
September 2005*

Performance Diagnosis Demystified: Best Practices for Oracle Database 10g

Introduction	3
Performance Diagnosis in Oracle Database 10g	3
Performance Diagnosis – Life Before Oracle 10g	3
Intelligent Self-Management Infrastructure in Oracle 10g	4
Automatic Workload Repository	4
Active Session History (ASH)	5
OS Statistics	5
Database Metrics	6
Wait Classes.....	6
Time Model.....	6
Database Time	7
Automatic Maintenance Tasks	7
Advisory Framework	7
Alert Infrastructure	7
Performance Monitoring and Diagnostics Architecture: Oracle 10g....	8
Automatic Database Diagnostic Monitor (ADDM)	9
Performance Diagnosis: Usage Model	10
Proactive Diagnosis	10
Reactive Diagnosis.....	12
EM Performance Pages.....	13
ASH (Active Session History) Report.....	18
AWR Compare Periods Report.....	21
EM Memory Access Mode	24
Performance Diagnosis: Best Practices.....	26
Conclusion.....	30

Performance Diagnosis Demystified: Best Practices for Oracle Database 10g

INTRODUCTION

Database performance diagnosis and tuning are an important part of any DBA's daily job to ensure their business meets availability and performance service level agreements. The process of diagnosing and tuning of performance problems has traditionally been manual, and could take considerable time and effort depending on the nature of the problem and expertise of the DBA. Fortunately, Oracle Database 10g introduces a revolutionary set of diagnostic capabilities that make performance diagnosis quick and easy. Such capabilities significantly reduce the time and effort spent by DBAs, and in the process also lower the database management cost.

This paper discusses the performance monitoring and diagnostics architecture in Oracle Database 10g that simplifies performance diagnosis and tuning. Two key components, namely, Active Session History (ASH) and Automatic Database Diagnostic Monitor (ADDM) are discussed in detail. The performance diagnosis usage model is explained in terms of various proactive and reactive diagnostic solutions that can be employed by DBAs, including how each solution works, its most effective use, and when to use which solution. The usage model discussed covers advanced performance diagnostic techniques such as comparing current or historical performance to baselines, analyzing transient performance problems, and diagnosing hung databases. Finally, best practices from large real-world deployments that have successfully used these performance diagnostic solutions are discussed.

PERFORMANCE DIAGNOSIS IN ORACLE DATABASE 10G

In order to understand performance diagnosis and its underlying architecture in Oracle Database 10g, we briefly discuss the issues faced in this process prior to Oracle 10g.

Performance Diagnosis – Life Before Oracle 10g

Any database performance diagnosis and tuning methodology usually consists of three main phases, namely, Data Collection, Data Analysis, and Solution Implementation. The performance diagnosis and tuning prior to Oracle 10g was complex since the three phases involved were mostly manual and time consuming.

The “Data Collection” phase often required additional data capture which resulted in longer time to diagnose problems. Numerous statistics available through various tools and V\$ views resulted in data overload with no information on how to address the top problems and solutions. Also, there was no common infrastructure to tie all the statistics from different components of the database (e.g., SQL, Application, Memory, CPU, etc.) together in order to provide a holistic time-based view of the system. The manual nature of the process left the diagnosis to the value judgment of end-users. This often resulted in misguided tuning efforts and little in terms of guaranteed returns. As such, the performance diagnosis and tuning prior to Oracle 10g was tedious, error-prone, and often resulted in waste of time and effort.

In order to simplify and automate performance diagnosis (and other on-going system-management tasks), Oracle Database 10g introduced the Intelligent Self-Management Infrastructure. The Self-Management Infrastructure is part of the database kernel that provides the missing components and glue lacking in earlier releases to completely automate the three phases of performance diagnosis. In the next section, we briefly discuss the components of the Self-Management Infrastructure and their role in performance diagnosis.

Intelligent Self-Management Infrastructure in Oracle 10g

As mentioned earlier, the Self-Management Infrastructure is part of the database kernel and helps the database learn about its operational environment, use this intelligence to automatically remedy any potential problems, adapt to workload variations, and simplify routine administrative tasks. Unlike other products, Oracle’s Self-Management Infrastructure enables it to make self-managing decisions based on the environment it is operating in. The infrastructure includes key components to enable automatic performance. For a detailed discussion details on the Intelligent Self-Management Infrastructure, please refer to the white paper “Oracle Database 10g: Intelligent Self-Management Infrastructure” available on Oracle Technology Network ([OTN](#)) website.

Automatic Workload Repository

AWR is the foundation of the intelligent infrastructure that is built into every Oracle Database 10g database and stores a wealth of operational data and key vital statistics regarding database performance. This enables the database to be self-learning and self-managing. By default, every hour a snapshot of all workload and statistics information is taken and stored in the AWR. The data is retained for 7 days by default and both snapshot interval and retention settings are user configurable. AWR is installed out of the box, and the database automatically manages its space requirements by purging old data as needed. AWR captures all of the data previously captured by Statspack and much more. The data captured allows both system and user level analysis to be performed, reducing the requirement to repeat the workload in order to diagnose problems. Various optimizations have been done to ensure that the capture of data is performed

AWR automates the process of performance diagnosis data capture and lays the foundation for “self-management” of the database

efficiently to minimize the performance overhead, for e.g., SQL statements are captured by tracking the deltas of the data between snapshots only if they impact the system significantly by different dimensions. In summary, AWR automates the “Data Collection” phase of performance diagnosis with extremely low overhead and provides foundation for all self-management tasks including performance diagnosis. AWR infrastructure is covered in more detail in the parallel session titled “Oracle Database 10g Self-Management Framework Internals: Exploring the Automatic Workload Repository”.

ASH's intelligent sampling allows Oracle to gather fine-grain data at session and system level with low overhead. This minimizes the need to enable SQL_TRACE and replay workload for problem diagnosis

Active Session History (ASH)

DBAs are often faced with the difficult task of investigating why a user job or session is hung while the database is behaving normally. While AWR or STATSPACK snapshots provide statistics at instance level periodically, they fall short on providing fine-grain diagnostic information at process or session level. Using tracing facility such as SQL_TRACE provides too much detailed data and imposes significant performance overhead on the system. Also, tracing needs to be enabled manually, and DBAs need prior knowledge that the problem exists, which is seldom the case. Oracle Database 10g provides an elegant solution for capturing fine-grain information with minimal overhead by using intelligent sampling. Oracle samples all the ‘ACTIVE’ sessions (those in a database call) working in the database periodically into a circular buffer in memory called “ASH buffers”. The ASH buffers, by design can hold up to 1 hour of sampled fine-grain history of what is happening on the database. When under memory pressure or periodically, samples of ASH buffers are written to disk (AWR). Since only “ACTIVE” sessions rather than all the sessions on the database are captured, ASH data set is manageable. Further, since samples of ASH data rather than the entire ASH data are written to AWR, minimal performance overhead is incurred, while retaining the ability to perform accurate real-time and historical performance diagnosis. The ASH data that is within the AWR retention period can be analyzed by different dimensions (e.g., Time, SID, SQL_ID, Wait event, Program, Module, Action and Object, etc.) – this enables DBAs to answer various current and historical performance related questions with great accuracy. Also, since ASH data collection is always on and incurs almost negligible overhead⁷, DBAs generally no longer need to incur the cost, effort and performance penalty associated with repeating workloads and enabling SQL_TRACE to diagnose performance problems.

OS Statistics

The database statistics that were available prior to Oracle 10g lacked ability to detect hardware level resource contention (for e.g., paging, swapping, CPU contention) which made it difficult to diagnose performance problems. A new view

⁷ ASH data collection performance overhead was measured to be < 1%. The overall Self-Management Infrastructure overhead (including ASH data collection) was measured to be about 2%. For further details, please refer the white paper “Oracle Database 10g: Intelligent Self-Management Infrastructure” on [OTN](#) website.

V\$OSSTAT that captures machine level resource usage has been introduced in Oracle 10g to overcome this limitation.

Database Metrics

There are numerous statistics available in the database prior to Oracle10g, however, they not useful in performance diagnosis since they are cumulative in nature and fail to convey the rate at which they were accrued over a dimension such as time or transaction. Often it is important to know the rate of change of a statistic rather than the cumulated value since it helps determine if a problem exists on the system. In Oracle Database 10g, numerous database metrics are available that are pre-calculated and normalized by time and transaction. These metrics can be used for setting alerting thresholds as well as assessing if it impacts the system.

Wait Classes

There are over 800+ different wait events in Oracle Database 10g and it is hard for DBAs to diagnose performance problems since they have to understand what each wait event means and how to address it in order to tune their system. So while additional wait events have been added when there was an overlap in their usage to accurately identify problems, at the same time, they have also been categorized into 12 wait classes. This categorization enables mapping all events into well-defined solution spaces and performing high-level analysis. The 12 wait classes that all the wait events are mapped into are as follows: Application, Commit, Concurrency, Configuration, User IO, Cluster, Idle, Network, System IO, Administration, Other, and Scheduler.

Time Model

When tuning a database system, there are many components involved such as memory, CPU, IO and database operations. Each of the components has its own set of statistics and measurements making it hard to understand the quantitative impact of that component on the system. In order to look at the system as a whole, a common currency needs to be used. In Oracle 10g, “time” is the common currency used to assess the impact of each component on the system. Using “time” as the common currency also lets Oracle advisories perform sophisticated simulations to answer questions such as “What is the expected net improvement in performance in terms of time by moving 20% of the buffer cache memory to shared pool?” The time model statistics are available at system and session level through the V\$SYS_TIME_MODEL and V\$SESS_TIME_MODEL views. Using Time Model solves the problem of quantifying database operations (e.g, time for logons, soft parses, etc.) that was not possible in prior Oracle releases. Extensive code instrumentation was required to automatically track all operations in the database in order to make Time Model implementation successful.

Time Model data quantifies the cost of different operations on the database and enables performing component benefit analysis used for database self-management purposes

Database Time

Database time is the time model data from the perspective of the entire database. It represents the total time spent in database calls. It includes all the non-idle database wait times (e.g., IO, latches, enqueues, etc.), time spent running on CPU (e.g., parsing, fetching, etc.), and waiting on the run-queue for CPU. Then in order to tune an Oracle database for a given workload, the goal is to reduce the “DB Time” of the system. The reduction in “DB Time” is the measure of effectiveness of the tuning exercise.

Automatic Maintenance Tasks

AWR provides historical information of how the database is being used. Oracle Database 10g can identify and perform some routine maintenance tasks by analyzing the information in AWR. An Automated Task Infrastructure exists that enables Oracle to automatically perform those tasks. The Task Infrastructure includes a pre-defined nightly and weekend maintenance window, and can be customized if required. Some examples of maintenance tasks that are automated are optimizer statistics gathering, space reclamation advice, etc.

Advisory Framework

Oracle Database 10g includes a number of specialized advisors for different subsystems in the database to advise on how the operation of the sub-components can be further optimized. In order to ensure consistency and uniformity in the way various advisors function and interact with each other, an Advisory framework exists. The advisors are primarily used by the database itself to optimize its own performance, however DBAs can also invoke them to gain further insight into the working of a component. Examples of some advisors are SQL Tuning and Access Advisors, Memory Advisor, Segment Advisor, and Undo Advisor.

Alert Infrastructure

For problems that cannot be resolved automatically and require administrator notification, such as when running out of space, Oracle Database 10g provides a new infrastructure called Server-Generated Alerts. As the name suggests, Oracle Database 10g has the ability to monitor itself and send out alerts to the DBA of impending problems in an efficient and timely manner. The alerts are efficient since they use the push model (vs. pull by other tools) avoiding the need to frequently poll the database that might be expensive. The alert evaluation is timely, since database evaluates alert conditions when it performs its regular operations (for e.g., when evaluating space thresholds during segment allocation). The alerts produced can notify the DBAs of the problem and provide recommendation on how the problem being reported can be resolved. Oracle Database 10g Release 2 supports Adaptive Alert thresholds that minimize false alerts by helping set better thresholds based on workload patterns. For further information on Alert Infrastructure, please check the white paper, “The Self-Managing Database: Automatic-Health Monitoring and Alerting” on [OTN](#) website, and the parallel session, “Managing

Database Grids: New Features for Database Management in Oracle Enterprise Manager 10g”.

Performance Monitoring and Diagnostics Architecture: Oracle 10g

The Intelligent Self-Management Infrastructure described in the previous section provides the building blocks and glue for the performance monitoring and diagnostics architecture in Oracle Database 10g that is illustrated below (Figure 1).

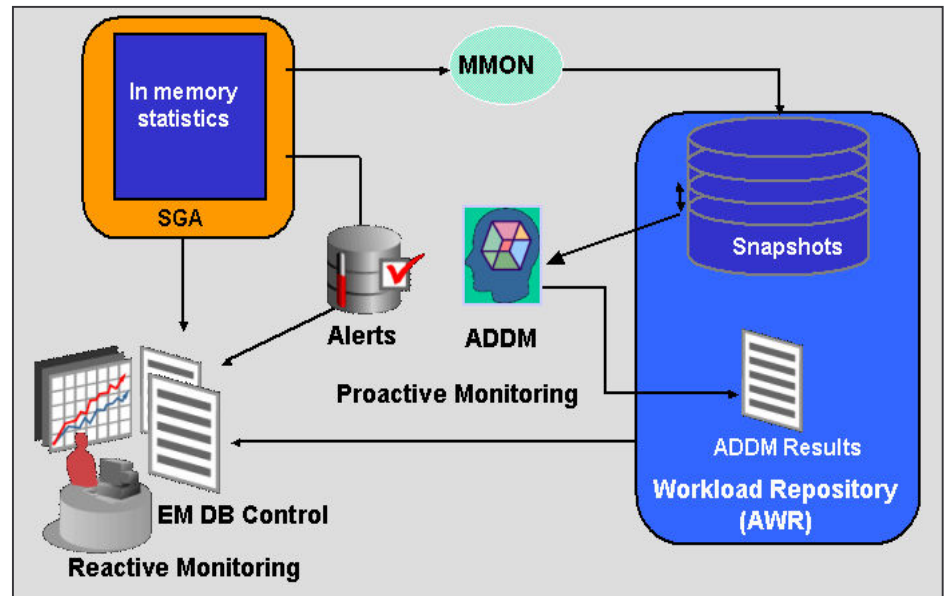


Figure 1: Performance Monitoring and Diagnostics Architecture in Oracle 10g

The architecture facilitates both allow both proactive and reactive monitoring and diagnostics. The proactive diagnostic component includes the Alert infrastructure discussed earlier, and Automated Performance Diagnostic infrastructure. In this paper we only explain the Automated Performance Diagnostic infrastructure. As part of proactive diagnostics, the database periodically examines itself, reports on any performance problems that were detected, and provides recommendations on them. It involves automatically collecting workload performance data, analyzing it, and providing quantified recommendations for the problems found. The key component of the performance monitoring and diagnostics architecture is the Automatic Database Diagnostic Monitor (ADDM) and is discussed in the next section.

The reactive diagnostic component includes EM Database Control (or Grid Control) Performance Page drill-downs that can be used when monitoring a problem that is currently happening, or requires some advanced analysis such as validating some tuning action and performing fine-grain analysis along a particular dimension of interest (session, time, SQL, module, etc.). As the database processes user requests, AWR infrastructure collects and maintains operational statistics for problem detection and self-tuning purposes in the background. AWR collects in-

memory statistics from SGA using the “MMON” background process and periodically persists them to disk (AWR). The in memory statistics include data sources like ASH, Time Model data, Database Metrics, TOP SQL statistics by different dimensions, etc. MMON also proactively checks if any alert thresholds have been exceeded and if so, raises an alert to the DBA using the Server-Generated Alert infrastructure. This enables DBAs to react and fix problems in a timely manner, thereby preventing further serious conditions in the database.

Automatic Database Diagnostic Monitor (ADDM)

ADDM automatically identifies performance bottlenecks, recommends corrective actions, and quantifies expected benefits

ADDM is a self-diagnostic engine built right into the database kernel that proactively examines the whole system from the data available in AWR. ADDM automatically identifies potential database bottlenecks, recommends corrective actions, and quantifies expected benefits. This is a revolutionary concept and ultimate solution in performance diagnosis and tuning, since the database can self-diagnose itself relieving DBAs of enormous effort and time spent. Since ADDM automatically runs after each AWR snapshot capture and completes in a few seconds, the health of the database is diagnosed proactively with negligible impact on the system. Also, Oracle Enterprise Manager presents the ADDM findings in an intuitive and convenient way that provides a guided step-by-step problem resolution to the DBAs. ADDM’s findings are stored in the AWR; this helps DBAs diagnose historical performance problems without further analysis.

ADDM takes a holistic approach to the performance on the system and suggests corrective action itself, or if necessary, recommends other Advisors such as SQL Tuning, Memory advisors, etc. ADDM analysis uses a top down approach that focuses on operations in the database consuming most time using the Time and Wait Model data. ADDM uses a sophisticated classification tree that maps symptoms (for e.g, latch contention) to root causes of the problem (in this case hard parsing) using ASH data. ADDM’s ability to detect root cause of problems rather than symptoms differentiates Oracle approach from any other tool or utility. This classification tree is based on several decades of performance expertise within Oracle and observations from real world. Also, ADDM can point to non-problem areas on the system, since it notes wait classes that are not significantly impacting the system as it walks down the classification tree. This enables administrator to focus their tuning efforts correctly so invaluable time and effort are saved. ADDM can diagnose most of the common problems seen in real world including those that involve components like Streams, RMAN, RAC, etc. For a list of some of the issues that ADDM can detect please refer to the white paper ‘The Self-Managing Database: Automatic Performance Diagnosis’ available on [OTN](#) website.

While ADDM offers the ultimate solution in performance diagnostics, sometimes it is necessary to drill down and perform advanced analysis such as when reacting to an alert or sudden CPU spike. All the performance diagnostics data sources are seamlessly integrated with Enterprise Manager (EM) Grid and Database Control interfaces. EM DB Control is the solution for managing a single database, while

EM Grid Control helps manage many databases and the entire Oracle eco-structure. EM DB Control is installed as part of the database install and provides out of the box solution for monitoring and managing a single database. The EM interface is easy to use, and intuitive, and with a few mouse clicks one can easily drill down to root causes of performance problems.

The next section discusses the usage model of the performance monitoring and diagnostics architecture discussed above.

PERFORMANCE DIAGNOSIS: USAGE MODEL

The performance diagnosis usage model closely follows the performance monitoring and diagnostics architecture and can be broadly categorized into two classes, namely, proactive diagnosis and reactive diagnosis. The next sections discuss the performance usage model in terms of proactive and reactive diagnosis.

Proactive Diagnosis

As a result of ADDM's proactive performance diagnosis, DBAs can now spend time in implementing the most effective solutions for their problems

As discussed earlier, ADDM provides the ultimate solution for performance diagnosis by automatically diagnosing performance problems. By using ADDM to diagnose problems quickly and accurately, DBAs can spend more time implementing the most effective solutions for their problems, which is the way it should be! When diagnosing performance problems, DBAs should first consult ADDM findings before performing further analysis. This is already a norm with DBAs at many customer sites like Dell, Qualcomm, Starwood Hotels, and Motorola, Inc. In fact, checking ADDM's findings to make sure database performance is normal is the top item on the everyday to-do list for DBAs at these sites.

ADDM's recommendations are available through the "Diagnostic Summary" link on the Database Home Page. Figure 2 highlights the ADDM Findings link.

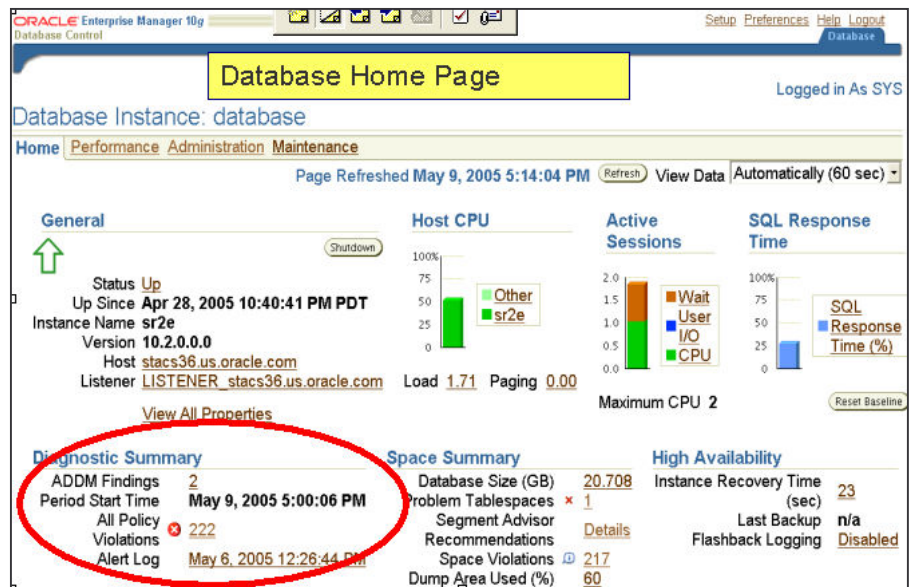


Figure 2: Database Home: ADDM Findings Link

By clicking on number next to the ADDM Findings link, you can get to the ADDM Findings as in Figure 3. The blue highlighted icon under the x-axis represents the ADDM findings for that period. Any of the checked icons can be selected to display ADDM findings for the corresponding period.

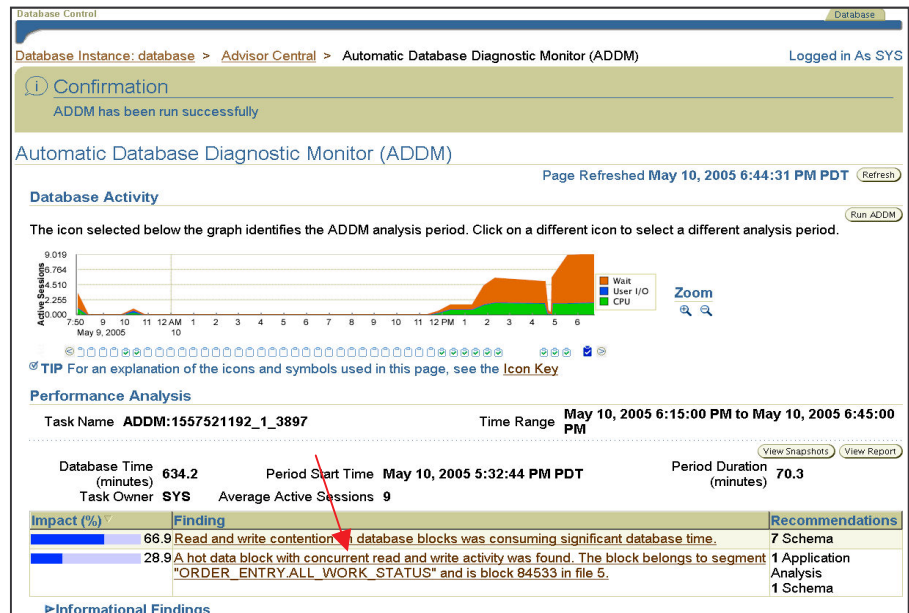


Figure 3: ADDM Findings

You can also use “zoom” feature to zoom in /out to the period of interest. The y-axis shows the active sessions and any change in the active sessions indicates a change in the database load on the system.

For each of the findings (in Figure 3), ADDM has shows the Impact % and Recommendations Category such as, Schema Analysis, Application Analysis, etc.

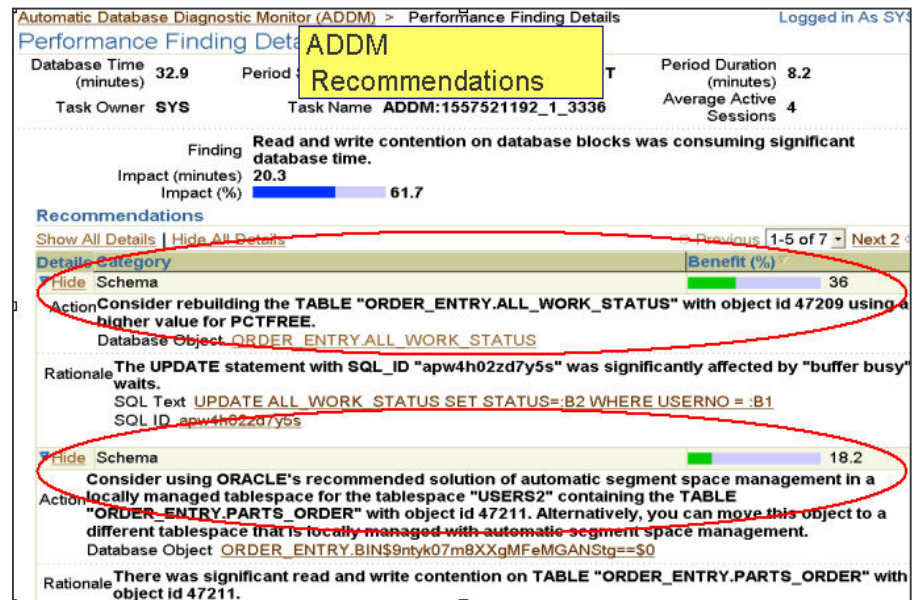


Figure 4: ADDM Recommendations

Clicking on the first ADDM finding in Figure 3 takes you to details as in Figure 4. ADDM automatically identifies an object causing excessive read/write contention (due to buffer busy wait events) and recommends higher PCTFREE setting. It also quantifies the impact of the finding on the system and the expected benefit when the change is implemented. It also gives the rationale for the finding, in this case the SQL_Id "apw4..." Also, for the second finding, note that ADDM recommends using ASSM tablespace to alleviate read/write contention problem.

ADDM can be also be run from command line interface using the Oracle supplied script, addmrpt.sql, available in \$ORACLE_HOME/rdbms/admin directory. ADDM by default runs on the last two snapshots in AWR, however, ADDM can also be run across a range of snapshots of interest if required.

Reactive Diagnosis

Proactive monitoring and diagnosis, and self-management of the database minimize the need for reactive diagnosis. However, there are few cases where DBA needs to manually intervene for problem diagnosis. This section discusses four advanced diagnostic solutions that exist in Oracle Database 10g, namely, EM Performance Pages, ASH Report, AWR Compare Periods Report, and EM Memory Access mode. An explanation of how each of these four diagnostic solutions works, what problem it solves, and it's most effective usage is provided. All the four diagnostic solutions are seamlessly integrated and available through the EM Database Pages.

EM Performance Pages

EM Interface let DBAs rapidly drill-down to the source of a performance problem and monitor in real-time

EM Performance Pages let DBAs quickly diagnose performance problems in real-time with a few mouse clicks. All the data sources, which ADDM uses, are exposed through the EM Performance Pages, so DBAs can rapidly drill down to the source of the problem. However, when diagnosing performance problems, DBA should first check the ADDM findings for the period of interest to see if it reports any findings and recommendations. As a second step, the DBAs can start from the EM Database Home page and use the Performance Page drill-downs to investigate further.

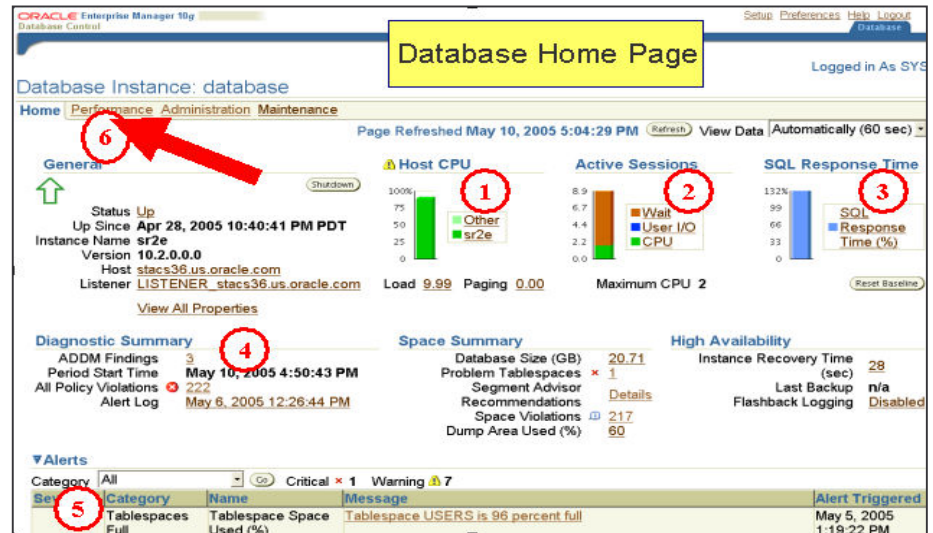


Figure 5: Database Home Page

The Database Home Page (Figure5) shows the current state of the database in terms of categories such as general information, availability, host CPU utilization, database load, alert and job status, space and other diagnostic summary information. This page is refreshed every few minutes. The relevant information for performance diagnosis on Database Home Page is explained further below:

1. Host CPU, in this graph, host CPU is being used at approximately 85% and all of it is being used by database instance, "sr2e". The warning threshold has been exceeded for host CPU utilization.
2. Active Sessions, there are 8.9 active sessions over the last sample; out of which more than 6 sessions on average are waiting for resources. Analogous to the load average or run-queue length that represents the demand for CPU at the Operating System level, Active Sessions represents the demand for database resources in terms of users consuming CPU, waiting for User IO, and other non-idle wait events.
3. SQL Response Time, the current response of the tracked set of SQL is 132% versus the baseline response of 100%. If the baseline and response time are

equal, the system is running normally. In this case, the system is performing more slowly than normal.

4. Diagnostics Summary, there are links to diagnostic (ADDM) findings and recommendations, policy violations detected on the system, and Alert Log errors
5. Alerts, there is a critical alert due to Tablespace USERS becoming full and seven other warning alerts
6. We click on the Performance tab to go to the EM Performance Page (Figure 6)

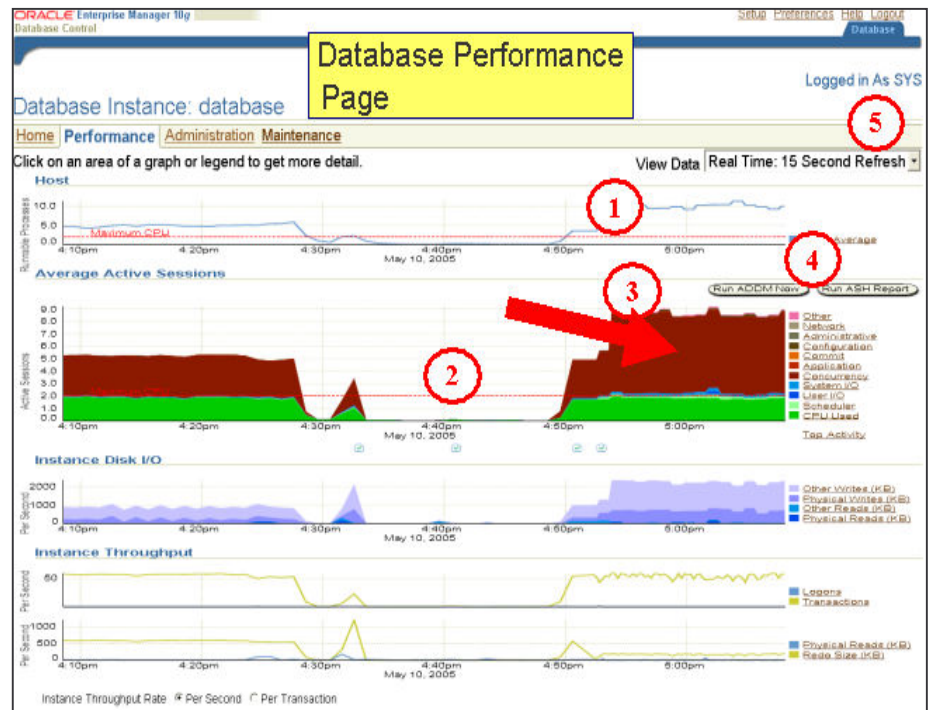


Figure 6: Database Performance Page

The Database Performance Page has 3 major sections, host information, database user activity, and throughput information. DBAs can first verify that ample host CPU exists before investigating the database. Then the database health can be assessed in terms of Active Sessions working in the database. The Active Sessions graph is rich in data. It shows how much CPU is being used and the non-idle waits incurred by the users instead of running on the CPU. If there are 10000 sessions connected and concurrently working on a system, only about 100 are active, and the Active Sessions graph shows the active users on the system. This data is refreshed every 15 seconds so DBAs can perform real-time analysis on the system. The graph is simple to use, the larger the block of color the worse the problem that can be attributed to the corresponding wait class. DBAs can click on the significant color, and that drills them into the Top Sessions and SQL related to that wait class. So in a few clicks DBAs can easily diagnose performance problems.

Finally, the throughput data can be used to correlate if it is impacted by machine or database resource contention. This is possible since all the three sections use the same time line.

Some of the important Performance page details (Figure 6) are described below:

1. Represents the demand for CPU (load average or run-queue length) at the Operating System level.
2. The Maximum CPU line is an important reference point. When the green “CPU Used” value reaches maximum CPU line, then the database instance is running at 100% CPU of the host machine.
3. All other values than “CPU Used” represent users waiting and contention for resources. In this case the biggest contention area is “Concurrency”. By either clicking on the colored area of the graph or on the legend, we can drill-down to another page to obtain more detailed information.
4. Run ADDM Now, Run ASH Report buttons:
 - a. Run ADDM Now: ADDM runs automatically after each snapshot is taken (every hour by default). The snapshot interval duration can be customized if required. Clicking on the “Run ADDM Now” will manually force a snapshot on the database and automatically trigger ADDM to perform analysis. Since ADDM runs periodically, it might be possible that the ADDM findings may lag by maximum of snapshot interval duration. However, DBAs can always invoke ADDM to perform instant analysis on the system rather than wait for the next snapshot to be taken.
 - b. Run ASH Report: ASH Report is useful in order to diagnose transient performance problems even up to a few minutes or perform advanced dimensional analysis (time, SQL_id, process, etc.). ASH Report has information that includes blocking session details, transaction ids, Top Sessions, SQL, Wait events and other such information aggregated by different dimensions that help diagnosing the source of the problem. ASH. ASH Report will be discussed further in the section “ASH Report”.
5. “View Data” drop down list can be selected from 15 sec, 60 sec or Manual Refresh, and Historical mode. When Historical mode is selected, the past performance of the system within the AWR retention period is shown

Figure 6 shows “Concurrency” as the significant wait class, we can drill-down by clicking on that color or the legend to display TOP SQL and sessions for that wait class as in Figure 7.

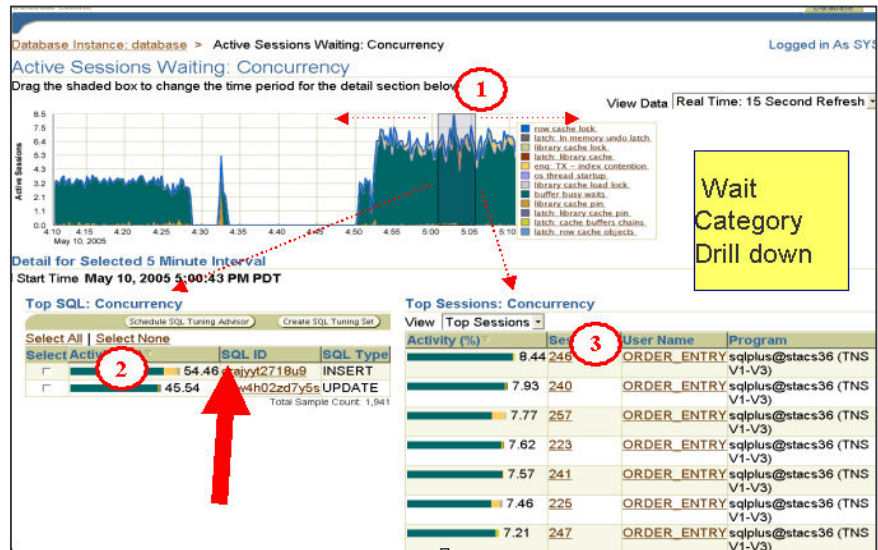


Figure 7: Active Sessions Waiting – Concurrency Wait Class

1. Active Sessions Waiting: Concurrency – gives details of waits in this group. The gray rectangle is a slider box that can be positioned over points of interests, changing the details in the Top SQL and Sessions bar charts on the lower half of the screen.
2. Top SQL by Wait Category– displays the SQL statements that were found waiting the most times during the selected interval. The idea is that if one or few statements are the majority then it should be looked into, which is the case, so we drill down on this statement.
3. Top Session by Wait Category – displays the sessions that were found waiting the most during an interval. In this case, the waits are fairly well balanced, but if one session stood out it should be looked at in more detail. The Top Sessions all appear to belong to same user but running multiple sessions.

Clicking on the Top SQL in Figure 7 shows the following screen (Figure 8).

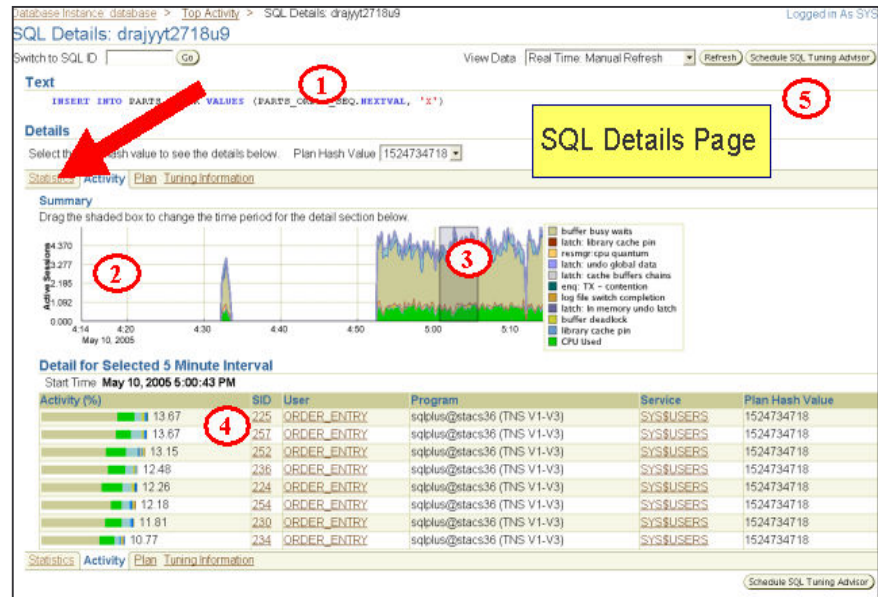


Figure 8: SQL Details Page: Activity

1. SQL Text – Activity tab, lists the text for this SQL statement
2. The number of active sessions that were running this query
3. The 5-minute duration that was selected and breakdown by wait category for this SQL statement.
4. Users executing the SQL statement and breakdown by wait category for period selected above.
5. “View Data” drop down menu can be selected among a) Manual Refresh b) Refresh every 15 seconds c) Historical. The Historical View shows the query performance and other statistics when the query ran in the past.

Clicking on the SQL Details Page, Statistics tab displays the page shown in Figure 9.

SQL Details Page, Statistics tab (Figure 9) provides a wealth of statistics for that particular SQL statement.

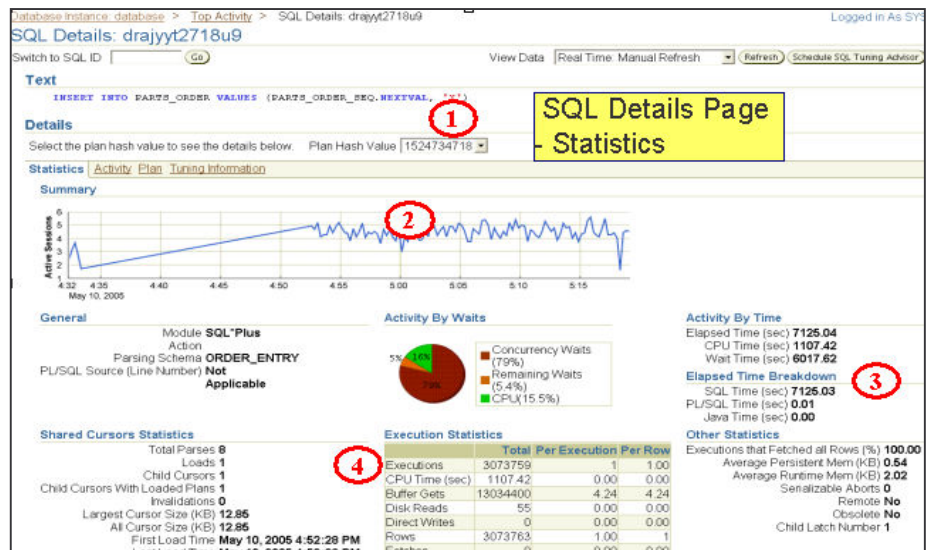


Figure 9: SQL Details - Statistics

1. Plan Hash Value for the SQL Statement that is being used.
2. Active Sessions (y-axis) that were running the SQL over time (x-axis).
3. Activity by Time - breakdown by Elapsed, CPU, Wait Times. Also break down by PL/SQL, SQL, and Java times.
4. Execution and other information about the SQL that could be useful in problem diagnosis.

ASH (Active Session History) Report

ASH Report can be used to perform targeted or transient performance analysis that is not possible with periodic analysis

ASH Report is useful to perform analysis of transient problems lasting up to a few minutes or targeted analysis over various dimensions (or their combinations), such as Time, SQL_ID, module, action, etc. Often DBAs want to understand the cause of a sudden hiccup or slow down lasting a few minutes in a particular session or the entire database. Due to the transient nature of the problem, periodic analysis of the database system might not reveal the cause of the slowdown. However since ASH report is based on the sampled history of the all events happening in the database, it can be easily used to understand the causes of transient or targeted performance problems.

ASH data that is captured few seconds for all the active sessions in the database is essentially a fact table with about 10 key dimensions such as Time, Sql_id, Program, Action, Module, File, Block, Session_id, Event, P1, P2, P3, etc. As such, dimensional statistical analysis can be performed on ASH data to answer detailed performance related questions. Unlike AWR/STATSPACK data, which represent exact counts of events and time, ASH data is sampled, and as such represents proxy for time. The more the number of ASH samples, the more accurate the results of ASH Report. The ASH report can be run transparently against both ASH data in memory and /or on disk (AWR) as long as data falls under the AWR retention period. The start and end times for the ASH report need not be tied to snapshot

boundaries, this makes it possible to focus ASH analysis on a very small period up to a few minutes. ASH report also facilitates blocker analysis by providing, transaction-ids, latch holders and other relevant information.

ASH report can be invoked from the EM Performance Page or from the Top Activity Page link on the Performance Page as shown below. It can also be run from the command line using the Oracle supplied script ashrpt.sql in \$ORACLE_HOME/rdbms/admin directory.

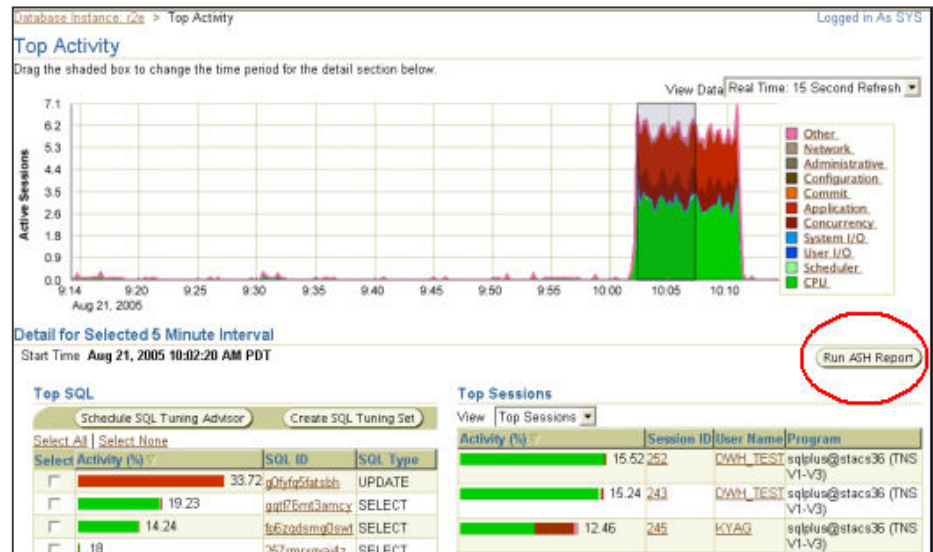


Figure 10: Top Activity Page: Run ASH Report

In the example below, we show how ASH Report was used on a large internal production system that is used globally to file and track defects across all software products in Oracle. A session was identified from the EM Performance Page/Top Activity as consistently consuming IO resources for more than 18 hours. The ASH Report was run with the problematic session as the target. It showed interesting dimensional data that helped identify the root cause of the problem.

Figures 11- 13 explain the ASH report that helped identify the problem in the above case.

Figure11:

- 1.Data Source used for the ASH Report (whether from memory – V\$ACTIVE_SESSION_HISTORY or AWR – DBA_HIST_ACTIVE_SESSION_HISTORY), in this case from AWR
2. Target for the ASH Report

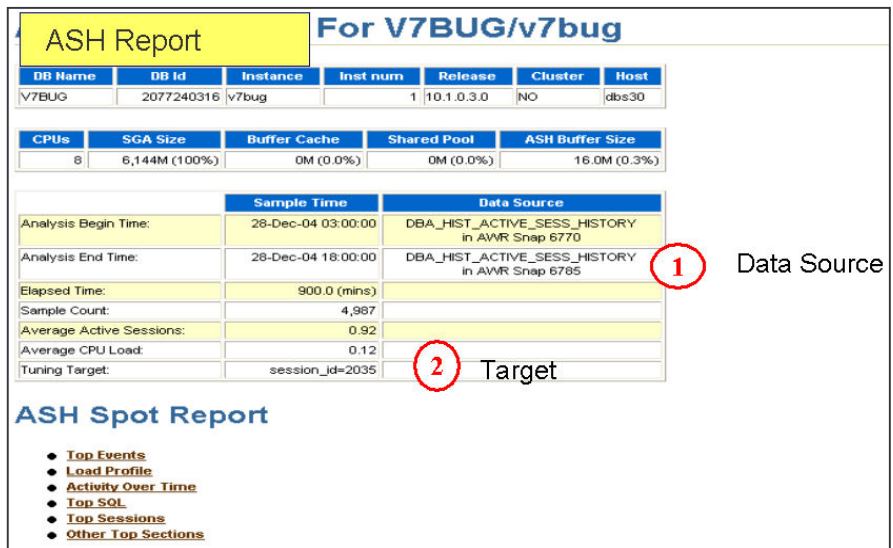


Figure 11: Top Activity Page: Run ASH Report

The report consists of data by key dimensions that help diagnose the problem. ASH dimensional data (Figure 12) revealed the “Module” causing 99.68% of the overall activity for that session. The Activity Over Time reveals the Top Events per each time slot. The analysis period is broken down into 11 smaller chunks called “slots” (in this case of 90 min), to understand the execution profile of the targeted session. The Activity Over Time consistently indicates a skew in “db file sequential read” wait as the TOP wait event.

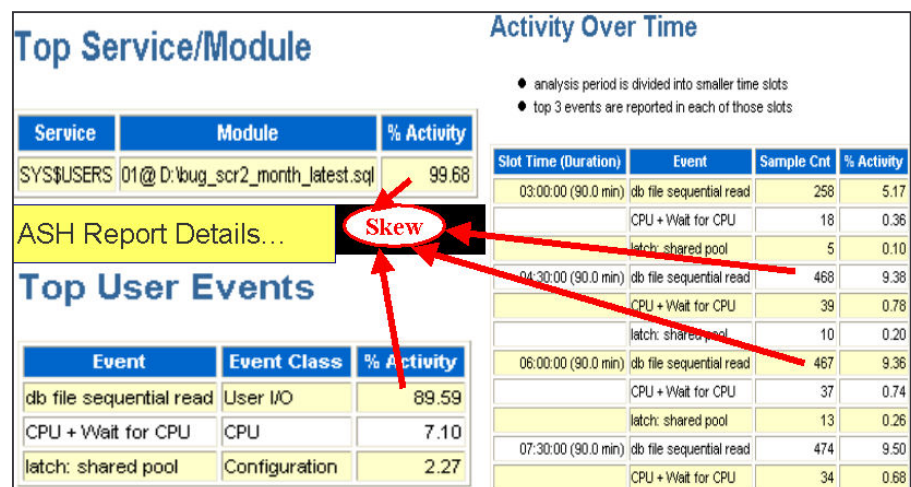


Figure 12: ASH Dimensional data

From the ASH dimensional data (Figure 13) listed in the report it becomes easy to infer that one particular SQL statement incurring “db file sequential read” as the cause of the problem

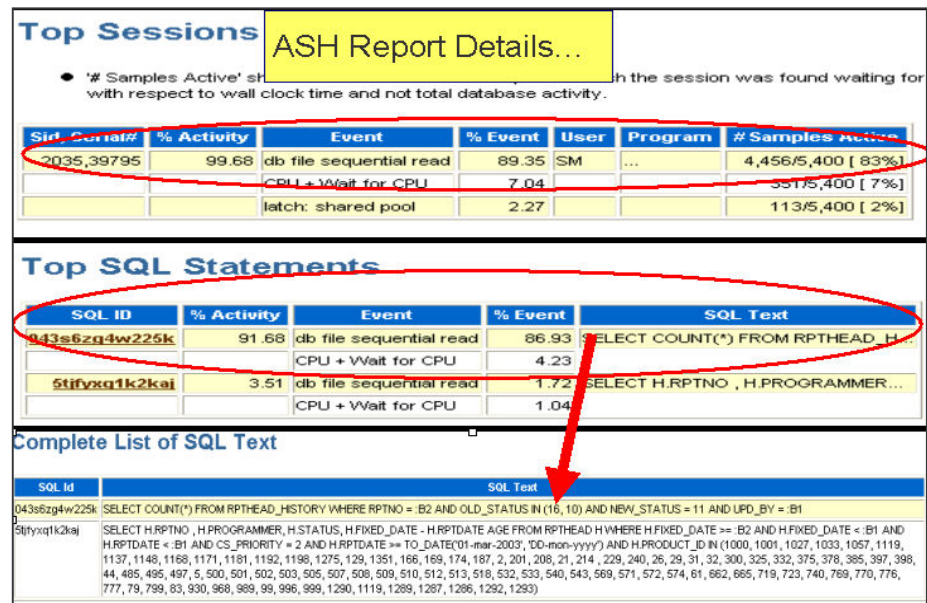


Figure 13: More ASH Dimensional Data and SQL text

Thus, using ASH report, we were able to identify the SQL statement as the root cause of the problem that ran in the past without needing to repeat the workload and enabling SQL tracing.

AWR Compare Periods Report

AWR Compare Periods Report takes two time periods as input and generates a report that compares those periods. The report not only contains detailed performance statistics and configuration settings from those time periods but also their comparison. This makes it easy to identify all the performance or database configuration attributes that changed between the two time periods.

Consider an example where an application workload is known to be stable for a given time of day, and the performance on Tuesday between 10:00 am and 11:00 am was poor. If you can generate a Compare Periods report with one period defined as Tuesday 10:00 am to 11:00 am and the other period defined as the previous day 10:00 am to 11:00 am, you are then able to identify configuration settings, workload profile, and statistics that were different between the two time periods. Based on the changes reported in the two time periods, you can quickly diagnose the cause of performance problems. AWR Compare Periods report can detect changes in key performance statistics such as SQL plan execution, Top Wait events, Top SQL, Top Objects by IO, Database Configuration settings, etc. The two time periods that are selected for comparison need not be of the same duration, since the reported statistics are normalized by “DB Time” for each time

AWR Compare Periods Report helps
diagnose problems caused due to
variations in workloads and configuration
settings

period. The Compare Periods Report is available for Oracle Database 10g, Release 2 and higher, and can currently be used to compare two time periods on the same database. An interesting use case for AWR Compare Periods Report is to compare similar workloads when upgrading to new patchset or minor releases (for e.g., hypothetically from Oracle releases 10.2.0.2 to 10.2.0.3, or 10.2.0.2 to 10.3.0.2). In this case AWR Compare Periods report can quickly narrow down configuration parameters, SQL plan regressions, etc., that have impacted performance. It is important to note that if the workloads differ significantly, there will be too many differences in the reported statistics rendering the report meaningless. DBAs can first investigate ADDM, AWR reports, and EM performance page drill downs when diagnosing problems, and then use AWR Compare Periods as an additional diagnostic technique to narrow down the problem.

AWR Compare Periods Report can be run from EM Interface or command line through the Oracle supplied script, awrddrpt.sql in \$ORACLE_HOME/rdbms/admin directory. However, the EM Interface makes it much easier to run and view the Compare Periods Report. In order to facilitate the comparison of the two time periods, AWR Preserved Snapshot Sets can be used. An AWR Preserved Snapshot Set (or baseline) represents a range of snapshots that is preserved beyond AWR retention period. It can be used for comparing similar workloads in future when performance problems occur or packaging the data to support for further diagnosis. By creating an AWR Preserved Snapshot Set for the period that represents the normal peak workload, and similarly creating another Preserved Snapshot Set when the performance is poor, we can quickly identify performance statistics and configuration settings that have changed significantly causing the performance to slow down.

The steps to generate an AWR Compare Periods are illustrated below. The “Manage Snapshots and Preserved Snapshot Sets” screen show below (Figure 14) can be accessed from EM Database Home Page -> Administration -> Statistics Management, Automatic Workload Repository.

Manage Snapshots and Preserved Snapshot Sets

Snapshots 1300
 Preserved Snapshot Sets 2
 Latest Snapshot Time Aug 27, 2005 2:40:26 PM
 Earliest Snapshot Time Aug 18, 2005 3:01:05 PM

Select Beginning Snapshot
 Go To Time [02766] [3:00 PM] [Go]

Compare Period Report Steps

1. Create Baseline (Good Perf.)
2. Create Another Baseline (Bad Perf.)
3. Run Compare Periods Report

Select ID	Capture Time	Collection Level
1276	Aug 27, 2005 10:41:01 AM	TYPICAL
1277	Aug 27, 2005 10:50:56 AM	TYPICAL
1278	Aug 27, 2005 11:01:02 AM	TYPICAL
1279	Aug 27, 2005 11:11:01 AM	TYPICAL

Preserved Snapshot Sets

Page Refreshed Aug 27, 2005 2:59:18 PM (Refresh)

Select Preserved Snapshot Set ID	Name	Beginning Snapshot ID	Beginning Snapshot Capture Time
8	BAD_PERF_BASELINE1	433	Aug 21, 2005 2:30:4 PM
9	GOOD_PERF_BASELINE2	569	Aug 22, 2005 1:10:58 PM

Actions

- Create SQL Tuning Set
- View Report
- Run ADDM
- Delete Preserved Snapshot Set
- Compare Periods

Figure 14: Creating Preserved Snapshot Sets and Generating AWR Compare Periods Report

A sample AWR Compare Periods Report is explained in Figures 15-17 below.

AWR Compare Report Details					End Snap Id	End Snap Time	Elapsed Time (min)	DB Time (min)	Avg Active Users
1st	569	22-Aug-05 13:10:58	575	22-Aug-05 14:10:49	59.85	127.80	2.14		
2nd	433	21-Aug-05 14:30:43	452	21-Aug-05 17:40:22	189.65	1,069.86	17.88		

Configuration Comparison			
	1st	2nd	%Diff
Buffer Cache:	172M	28M	-83.72
Std Block Size:	8K	8K	0.00
Shared Pool Size:	212M	100M	-52.83
Log Buffer:	6,996K	6,996K	0.00
SGA Target:	0	147M	100.00
PGA Aggregate Target:	26M	26M	0.00
Undo Management:	AUTO	AUTO	

Time duration can be different

Figure 15: AWR Compare Periods Report: Snapshot Time duration and Configuration comparison

Top 5 Timed Events					Top Wait Events Compared side-by-side				
1st					2nd				
Event	Waits	Time(s)	Percent Total DB Time	Wait Class	Event	Waits	Time(s)	Percent Total DB Time	Wait Class
CPU time		22,191.7	32.97		enq: TX - row lock contention	1,578	3,211.1	65.73	Application
enq: TX - row lock contention	10,634	21,127.4	31.39	Application	CPU time		1,244.4	25.47	
*latch: library cache	559,941	5,302.2	7.88	Concurrency	buffer busy waits	285	145.0	2.97	Concurrency
*latch: shared pool	376,303	3,184.8	4.73	Concurrency	*db file sequential read	25,916	22.0	.45	User I/O
buffer busy waits	1,647	737.3	1.10	Concurrency	*SQL*Net message to client	4,435,906	17.2	.35	Network
*SQL*Net message to client	5,984,422	48.2	.07	Network	*latch: shared pool	33	0.7	.01	Concurrency
*db file sequential read	2,285	0.7	.00	User I/O	*latch: library cache	182	0.4	.01	Concurrency

Figure 16: Top Wait events compared side-by-side

In Figure 16, in the first time period (baseline/preserved snapshot set for good performance) enq TX – row lock contention takes only 31.39% of Total DB Time,

Vs. 65.73% when performance was poor. Figure 17 highlights DB configuration settings that are different in the two time periods.

resource_manager_plan	PRI_SIM_PLAN	
resumable_timeout	0	
serial_reuse	disable	
service_names	r2e	
session_cached_cursors	20	
session_max_open_files	10	
sessions	280	
sga_max_size	629145600	
sga_target	146800640	0
shadow_core_dump	partial	
shared_memory_address	0	
shared_pool_reserved_size	4823449	2306867
shared_pool_size	12582912	222298112
shared_servers	0	
skip_unusable_indexes	TRUE	
sort_area_retained_size	0	
sort_area_size	1048576	
spfile	/scratch/pgongloo/10.2/r2e/dbs/s	
sql92_security	FALSE	

DB Configuration Settings Compared...

Figure 17: DB Configuration Settings Comparison

Figure 17 highlights DB configuration settings that were reported to be different.

EM Memory Access Mode enables diagnosing performance problems even when the database is hung by directly accessing the SGA of the instance

EM Memory Access Mode

Oracle Database Release 2 introduces functionality for diagnosing problems when the database hung or extremely slow. The new functionality introduced is often referred to as “Memory Access” or “Direct-Attach” mode supports collection of real-time performance data directly from the SGA. The EM interface integrates the Memory Access Mode along with ORADEBUG Hang Analysis, which is an utility used by support for understanding extremely slow moving or hung databases.

When a database is moving extremely slowly or hung, monitoring the database using SQL (called as “SQL Access” model) may not be feasible, since the SQL issued may require resources such as latches, locks, memory, CPU, etc. that are heavily contended. In fact, the issued SQL may cause additional load on the system. Since Memory Access Mode gathers the relevant data from the shared memory of the database instance, it can allow for finer granularity of sampling that might be required during such analysis without consuming additional or excessive resources on the host. EM Memory Access mode provides support for key high-level performance diagnostic V\$ views such as V\$SESSION, V\$SYSTAT, V\$SESSION_WAIT, and V\$SYSTEM_EVENT through the direct memory access method. SQL needs to be used for obtaining additional detailed drill-down information. “SQL Access” is the default access mode in EM. Since “SQL Access”

mode uses SQL for gathering performance data, the functionality provided by it is much richer. However, when a database is extremely slow or hung, DBAs can switch to “Memory Access” Mode for performance diagnosis.

There is one SGA collector per instance that is automatically started by the EM agent when it starts monitoring an instance. EM Memory Access mode is supported for both Oracle 9i and 10g databases.

The Memory Access mode is available from the “Related Links” section of all the EM Database related pages (see Figure 18 below), and also the Database Down page. The EM user interface for the Memory Access mode is similar to the SQL Access. The EM Performance Pages in Memory Access mode display information sampled at higher frequency than SQL Access, this potentially helps in diagnosing short duration events that might be missed otherwise.

Database Home Page: Memory Access Mode

Space Summary

Database Size (GB)	20.762	Instance Recovery Time (sec)	12
Problem Tablespaces	1	Last Backup	n/a
Segment Advisor	Details	Flashback Logging	Disabled
Recommendations	217		
Space Violations	61		
Dump Area Used (%)			

High Availability

Alerts

Category: Tablespaces Full Critical: 1 Warning: 5

Severity	Category	Name	Message	Alert Triggered
Critical	Tablespaces Full	Tablespace Space Used (%)	Tablespace USERS is 96 percent full	May 5, 2005 1:19:22 PM

Related Alerts

Job Activity

Jobs scheduled to start no more than 7 days ago

Scheduled Executions	0	Running Executions	0	Suspended Executions	0	Problem Executions	0
----------------------	---	--------------------	---	----------------------	---	--------------------	---

Related Links

Advisor Central	Alert History	Alert Log Content
All Metrics	Blackouts	SQL*Plus
Jobs	Manage Metrics	Table Baselines
Metric Collection Errors	Monitoring Configuration	Monitor in Memory Access Mode
Recovery Catalogs	SQL History	User Defined Metrics

Figure 18: EM Memory Access Mode Link

The ORADEBUG Hang Analysis utility is available for database releases Oracle 9i and higher to capture instantaneous system wait for graph. From the system wait for graph, both instance and cluster-wide analysis of waiting sessions can be performed. The EM Hang Analysis uses the same ORADEBUG utility and provides a graphical interface to display the blocking sessions and their details (Figure 19 below) that are essential in diagnosing a hung database.

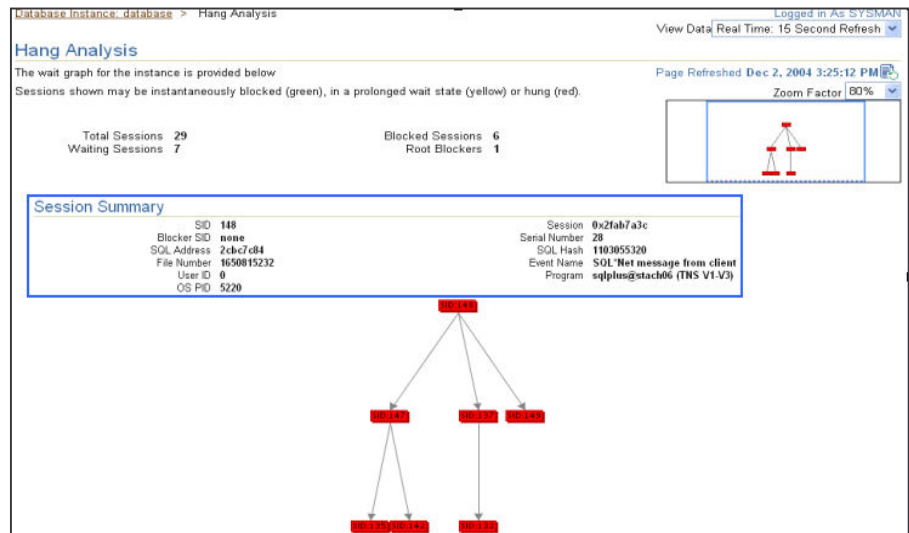


Figure 19: EM Hang Analysis

To summarize, EM provides SQL as well as Memory Access modes and, DBAs can choose the best option depending on the environment they are operating in without sacrificing the performance or the necessary drill-down information required for performance diagnosis of any situation.

PERFORMANCE DIAGNOSIS: BEST PRACTICES

Over the last two years, the Database Manageability Group in Oracle has worked with internal and external customers covering a diverse spectrum of workload (OLTP, DSS, Hybrid) and scale / type of business (Enterprise, Small and Medium Business, Independent Software Vendors). These customers are running Oracle Database 10g in production with databases sizes ranging from few terabytes to hundreds of gigabytes. This section discusses the best practices and lessons learnt in the process of deploying Oracle 10g.

- 1) **Recommended Performance Tuning Methodology:** As a best practice, when diagnosing performance problems, it's best to start from ADDM findings and investigate further. If there's a problem that's currently happening, you can manually force an ADDM run from the EM Performance Page. From our experience, ADDM was able to resolve majority of the performance problems that were encountered on the field. The reactive diagnostic solutions described in this paper can be explored for other few cases that involve advanced problem diagnosis.
- 2) **SYSAUX Sizing:** The minimum recommended setting for SYSAUX is 500 MB (typically for a small system, see best practice –3 also). DBCA and DBUA tools provide assistance on the setting of SYSAUX during creation or upgrade of the databases respectively. Also, the excessively growing occupants of SYSAUX can be monitored and moved to separate

tablespace if necessary. Another possible solution is to enable “autoextend” option on the SYSAUX tablespace if required.

- 3) **AWR Retention Period and Snapshot Interval Customization:** The default settings of 7 days AWR data retention and 60-minute snapshot interval are sufficient for most environments. However, DBAs can tailor the AWR settings to the environment the database is operating in. More frequent snapshots and longer AWR retention help better self-management, but at the cost of additional space required for AWR. Also, longer AWR retention helps database give better space related advice, such as with growth trends on segments. The best practice is that the AWR retention period should accommodate at least one complete business cycle (for e.g., month-end, quarter-end close, etc.).
- 4) **AWR Sizing:** AWR is one of the largest occupants of SYSAUX tablespace and requires a minimum of 300 MB. The general guidelines for AWR sizing with the default AWR settings are given in the table below. In order to obtain further AWR sizing and utilization information specific for your system, the scripts, *utlsyxsz.sql*, and *awrinfo.sql* in `$ORACLE_HOME/rdbms/admin` directory can be used.

<i>System</i>	Active Sessions	#CPUs	#User Objects	AWR Space Required
<i>Small</i>	10	2	500	0.5 GB
<i>Medium</i>	20	8	5000	2 GB
<i>Large</i>	100	32	50000	5 GB

To illustrate that the space consumed by AWR is negligible when compared to the vast benefit derived from it, a real-world data point is provided. The “GSI” system runs Oracle’s entire business and supports about 50K concurrent users. AWR space consumption on the system was about 13 GB with AWR retention set to 30 days and default snapshot interval of 60 minutes. Some detailed specifics on the GSI system are as follows: Oracle Database Release 10.1.0.3, 10000 concurrent users, 100 Average Active Sessions, 6 TB, 4-node Sun E25K, each node has 36 x 1.2 MHz dual-core CPUs (72 virtual CPUs) and 144 GB RAM, 60 Oracle Applications Modules are deployed on the system. Obviously, GSI system falls under “Large” configuration and the total AWR space consumed is negligible.

- 5) **STATISTICS_LEVEL Configuration Parameter Setting:** This parameter governs whether the self-management functionality of the database is enabled or disabled. It also controls various other statistics collection implemented in releases prior to Oracle 10g, such as Segment

Statistics, PGA Advisories, etc. The default setting of TYPICAL is the recommended setting for STATISTICS_LEVEL parameter, and ensures collection of all major statistics required for making intelligent self-management while ensuring minimal overhead on the system. When STATISTICS_LEVEL parameter is set to BASIC, the database can perform sub-optimally since the self-management infrastructure is turned off, and hence this setting is not recommended. When STATISTICS_LEVEL parameter is set to ALL, additional statistics such as timed OS, SQL plan, and latch statistics are collected. The additional statistics collected can be expensive on some platforms causing adverse impact on database performance, and hence the “ALL” setting is not recommended, except for advanced diagnosis in consultation with support. Please refer to the white paper “Oracle Database 10g: Intelligent Self-Management Infrastructure” available on [OTN](#) website for further information.

- 6) **AWR Vs STATSPACK:** If licensed for Diagnostic and Tuning Packs, and you have both AWR and STATSPACK snapshot infrastructure in place, you can disable STATSPACK snapshots. AWR infrastructure is a super set of STATSPACK and contains most of the information found in STATSPACK. If AWR and STATSPACK snapshots are running at the same time, they might cause excessive performance overhead on the system in some cases. If STATSPACK cannot be disabled for whatever reason, you can set STATSPACK and AWR snapshots to run at different times.
- 7) **Implementing ADDM or other Advisor Recommendations:** When implementing ADDM or other Advisor recommendations (such as SQL Tuning or Memory related), DBAs should look for persistent problems. Often, some DBAs try to tune their system for a one-time non-critical batch job by creating an index or adjusting the memory settings. Before implementing any ADDM recommendations, DBAs can ascertain if the problems are persistent and gauge for any trends by running ADDM reports for longer time periods representing peak load, for e.g., daily week days, 9.00 am to 12 noon for peak OTLP workload.
- 8) **Document Implemented Advisor Recommendations:** Since there could be multiple DBAs who’re managing several databases, they could be oblivious to the tuning actions made by each other. Hence, it is imperative to use a rigorous change control mechanism within the IT infrastructure. Besides, change control, there are few techniques that help preserve task data for future reference and these are discussed below.
 - a. **Preserving Advisory Task data:** The default task names for the advisors are system generated and not meaningful. It is often challenging to find the task known to have had generated some findings or advice after a few days. Using a meaningful task name

helps locate the task easily. Also, the default advisory task expiry is 30 days. So if a tuning action was implemented few months back, and if you wanted to find why a particular action was taken in the past, the task information would not be available. This can be solved either by changing the “default” expiration of the particular task, or for all the tasks to an appropriate value.

- b. **Preserving Raw AWR Snapshot data:** Based on the AWR retention period, the AWR raw data will eventually be purged. In order to avoid the AWR data being purged, one can create AWR Preserved Snapshot Sets for snapshot ranges of interest. This will help in situations such as ascertaining if a problem has occurred in the past, running AWR Compare Periods analysis, and packaging the AWR data to support for further diagnosis.
- 9) **Making Best Use of ASH data:** ASH data source can be used to get to an execution profile or “footprint” of a target such as session, module, SQL, etc. The ASH infrastructure is always on and incurs negligible overhead. Hence, DBAs can exploit ASH data in most cases of advanced problem diagnosis without replaying the workload or enabling SQL_TRACE. Also, the EM interface for database releases lower than Oracle 10g supports simulation of ASH data using V\$SESSION_WAIT view sampling. As such, the same technique that is used in ASH can also be exploited for pre-Oracle 10g databases.
- 10) **Making Best Use of SQL Tuning Set (STS):** A STS enables user to tune custom set of SQL statements easily by storing them persistently in the database. Also, a STS encompasses all the information needed to run the SQL statements, such as the execution context, bind variables, parsing user id, and other important execution statistics making it easy to input the statements to SQL Tuning and Access Advisors. From Oracle Database 10g Release 2 and higher, a STS can be transported to a different database for testing and analyzing SQL plan regressions. This enables DBAs to test performance of SQL statements before deploying them on new releases in production.
- a. **SQL Tuning and Access Advisors:** Please refer to the parallel session “Optimizing the Optimizer: Essential SQL Tuning Tips and Techniques”, Session Id: S997
- 11) **Maintenance Window Customization:** Oracle Database 10g uses a default maintenance window of 10 PM to 6 AM every day, and Saturday 12 AM to Monday 12 AM (48 hours) on weekends to perform routine administrative tasks such as gathering optimizer statistics and providing space reclamation advice on segments. However, if the maintenance window overlaps with other processing (e.g., nightly batch jobs) in the environment, the maintenance window or processing can be adjusted so

they can run at mutually exclusive or non-peak times as appropriate. This will ensure that there is no overlap or interfere with each other.

- 12) **Use SVG Plug-in for Mozilla /IE Browsers:** The SVG (Scalable Vector Graphics) plug-in renders better EM Pages, navigation, and interactivity in a browser. The plug-in for the appropriate browser can be downloaded from the following link:
<http://www.adobe.com/svg/viewer/install/main.html>

CONCLUSION

Database performance diagnosis and tuning is an important part DBA's role in ensuring uninterrupted business operations. Traditionally it has consumed significant time and effort with little in way of guaranteed results. The automatic diagnostic capabilities of ADDM and EM integration in Oracle Database 10g provide the DBA with the findings and recommendations so efforts are focused where they will result in most benefit in system throughput. Not only does Oracle Database 10g provide automated performance diagnosis that can be used in most situations, but also sophisticated solutions such as ASH and AWR Compare Periods Report, and EM Memory Access Mode that can be used for advanced problem diagnosis. The solutions provided in Oracle Database 10g enable problem diagnosis to be performed with significantly lower cost and effort, thereby making DBAs more productive and lower overall management costs for the business.



Performance Diagnosis Demystified: Best Practices for Oracle Database 10g
September 2005

Author: Prabhaker Gongloor

Contributing Authors: Graham Wood, Karl Dias

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:

Phone: +1.650.506.7000

Fax: +1.650.506.7200

oracle.com

Copyright © 2005, Oracle. All rights reserved.

This document is provided for information purposes only and the contents hereof are subject to change without notice.

This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission. Oracle, JD Edwards, PeopleSoft, and Retek are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.