

**Performance Tuning with Oracle Enterprise  
Manager  
Session # S300610**



*September 10, 2008  
Prepared by John Darrah  
DBA Knowledge, Inc.*

## Introduction

Oracle Enterprise Manager has had a long, sometimes not so glorious history with the Oracle DBA community. My first experience with the product was in 1998 and my immediate impression was that the product was of little use to a “serious” DBA. I further concluded that it was a waste of time to use OEM for performance tuning purposes. I spent the next 7 years learning every V\$ view and X\$ table, writing my own version of “Top Sessions”, and gleaning information from 10046 and 10053 trace files for particularly challenging SQL Tuning problems.

Over those years I formed the opinion that “real” DBAs use the command-line and hobbyists use GUIs like OEM. Then in 2005 I was hired by a large company that required each DBA to monitor as many as 50 Oracle Databases. The largest number of databases I managed prior to this job was 5. To further complicate things, a large number of the databases I was put in charge of managing were on Windows which rendered all of the shell scripts I had used for monitoring purposes over the years useless. One night as I was trying in vain to monitor all of these environments another DBA suggested I simply use Oracle Enterprise Manager, which was already deployed on all of the environments, to monitor and manage my environment. I was about to condescendingly explain my opinion on GUIs when I realized it was 8pm on a Tuesday night, the searing pain in my mid section probably was an ulcer, and I was likely going to lose my job if something didn’t change. Necessity being the mother of invention, I decided I had little to lose and logged into OEM.

Apparently, the developers at Oracle had been busy working on OEM during my 7 year hiatus. The product I saw did not resemble the product I remembered from 1998. After a week of using OEM 10G, I was converted. I could now easily monitor and manage multiple environments from one central point. Additionally, I could easily see the health of all the databases I was managing and quickly drill into databases that were performing poorly.

The performance aspects of OEM will be covered in this paper. I will take a top down approach detailing how OEM can be used to spot anomalies and how to quickly drill down into the data provided by OEM to find the root cause of performance problems. This paper will contrast OEM with some of the traditional performance tuning techniques.

## Getting a Top-Down View into Performance

One of the most powerful traits of OEM is its ability to gather data from many sources, aggregate it and present it summarized format. This allows a DBA to quickly ascertain the status of the group of environments and drill into any problem areas.

### Monitoring Performance with Groups

A very effective way to monitor and manage a large number of environments is through the use of Groups in OEM. OEM allows several targets to be grouped together under one name. This name acts as a logical container to display the metrics of several targets at once. For example, a DBA in charge of all the databases in the Florida datacenter may create a group Florida Databases. This is done by selecting the Groups sub-tab under the Targets tab in OEM. OEM will provide a list of every target it is currently monitoring as candidates for inclusion into the new group. Once all of the relevant targets have been added as members to the new group, the DBA can choose from a list of metrics what to display on a dashboard. Once the group is created, the DBA can select it from the “Groups” sub-tab under targets and launch a dashboard for that group. The dashboard will display high-level metrics for all targets in that group. In Figure 1 below we can quickly see that one host has crossed the critical threshold for the CPU Utilization metric, the same host has crossed the critical threshold for Disk I/Os per Second and a database has crossed the critical threshold for the Wait Time % metric. One of the challenges of diagnosing a performance problem is deciding where to start. From the dashboard it is clear that there is a CPU problem on one of the hosts in this group.

Type	Status	Alerts	Policy Violations	CPU Utilization (%)	Total Disk I/O Per Second	Memory Utilization (%)	Wait Time
Host	↑	35 10	6 3 0	●	●	●	-
Host	↑	0 2	11 2 0	●	●	●	-
Host	↑	0 0	19 3 0	●	●	●	-
Host	↑	1 9	12 2 0	●	●	●	-
Host	↑	0 1	5 3 0	●	●	●	-
Host	↑	0 0	6 3 0	●	●	●	-
Host	↑	5 2	7 3 0	●	●	●	-
Host	↑	2 2	6 3 0	●	●	●	-
Agent	↑	0 0	0 0 0	-	-	-	-
Listener	↑	0 0	2 2 0	-	-	-	-
Database Instance	↑	0 0	18 3 6	-	-	-	●
Database Instance	↑	0 1	7 2 1	-	-	-	●

Figure 1 Dashboard of the *Florida Databases* group

By clicking on the name of the host target, we can quickly drill into performance metrics for that host. Figure 2 illustrates the “Top Processes” page under the host performance tab. The screenshot shows an opmn process taking 75% of the CPU on the server. Further investigation revealed a port conflict between opmn and a listener. Using dashboards can greatly reduce the time it takes to identify root cause of a problem because so much information is immediately available to the DBA.

**Processes**  
Processes [355](#)

**Top 10 Processes**  
View By:

Process ID	Command	CPU Utilization (%)	CPU Total (seconds)
29853	/oracle/product/10g/opmn/bin/opmn -d	78.5	1,755,607
1443	/opt/VRTSob/bin/vxsvc -r /opt/VRTSob/config/Registry -e	5.8	14,899
631	/oracle/product/101/bin/tnslsnr LISTENER_openmisd -inherit	0.9	14,964
3	fsflush	0.8	14,519
3320	/var/opt/oracle/product/agent10g/bin/emagent	0.5	45,240
10403	ora_j000_	0.4	5,554
12082	vxconfigd -k	0.3	54,550
3282	/opt/VRTSvcs/bin/Application/ApplicationAgent -type Application	0.2	10,857
23046	oracle_ (LOCAL=NO)	0.2	731
25205	oracl_ (LOCAL=NO)	0.2	0

[Home](#) [Performance](#) [Administration](#) [Targets](#) [Configuration](#)

**Figure 2 Top Processes screen**

## Drilling into Cluster Performance

It can often be difficult to diagnose performance problems on RAC databases and clusters because there are several hosts involved. Comparing the metrics gathered across all nodes in a cluster is difficult and time consuming to do manually. OEM's ability to correlate and aggregate data across different targets is very helpful when diagnosing these types of problems. Figure 3 shows a three node RAC cluster. Hosts 1 and 2 are essentially idle while host 3 is averaging 75% CPU utilization with spikes to 100%. This particular cluster was frequently getting node evictions on host 3. Further investigation showed several single instance RAC databases had been installed on host3 giving it 3X more active instances than hosts 1 and 2.



Figure 3 Cluster Overview

Further investigation into the cluster metrics showed that one of the databases interconnects was going over the public Ethernet interface rather than the private interface. This miss configuration was clearly visible on the Cluster Metrics (Figure 4).

▼	DBa1	Cluster Database				
	DB1a1	Database Instance	bge257000	f10bb8-09	10.1.232.85	Public ✖

Figure 4 interconnect alert

## Performance Tuning on a 10g Database

The same drill-down method can be used to effectively tune a 10G database. OEM leverages the new features present in Oracle 10G such as AWR and ASH to help quickly diagnose performance problems<sup>1</sup>.

These features are all available whether or not OEM is in use but calling the packages that comprise the AWR and ADDM feature set from the command line is tedious and error prone. Just as in the examples above, OEM allows a DBA to view broad performance metrics and drill into areas of interest.

### The Performance Page

The performance page on a database target gives a dashboard view of the instance as a whole. The page contains graphs for server load average, active sessions, disk IO, transactions per second, and physical reads per second. Additionally, there are several links to other performance related pages. An easy to drill into Instance wide Wait Metrics is to click the “Wait Class” link in the legend of the “Active Sessions” graph. Clicking one of these links opens a page showing all SQL sessions that currently have wait events in the wait class listed. For example if a DBA wanted to drill into the SQL statements contributing to the “Other” wait class, he would simply click the “Other” link in the legend of the graph in Figure 5.

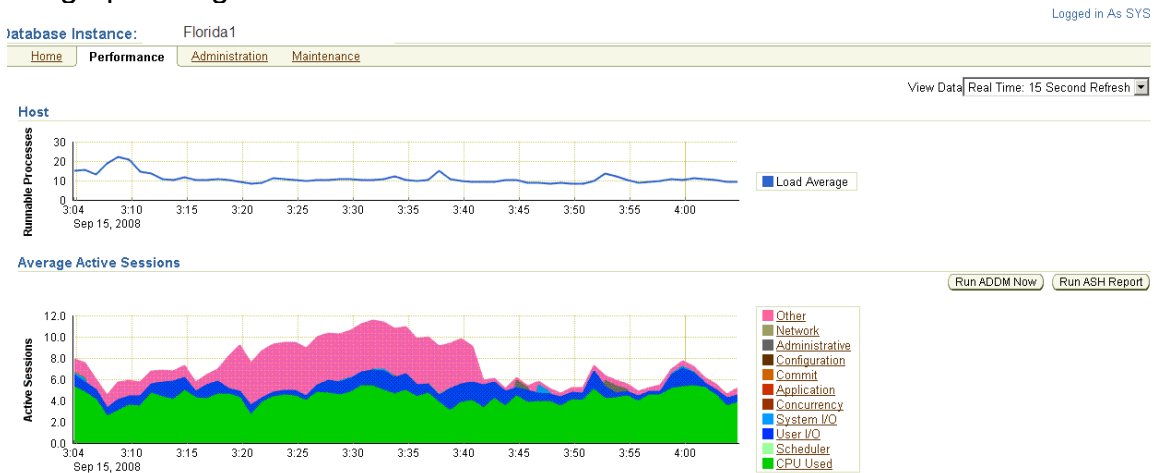


Figure 5 Performance Tab

<sup>1</sup> AWR and ASH require additional licensing of the Database Diagnostics and Tuning Pack. This pack must be licensed to use any of the AWR, ADDM, or ASH features regardless of whether or not they are used through OEM.

## Top Activity Page

Many times a DBA is asked to investigate the performance of a particular user session or SQL statement. A good starting point for this type of investigation is the “Top Activity” page. To view Top Activity, select the “Top Activity” link located on the database performance page. The default behavior of this page is to show the current top statements and session active in the instance. In Figure 6, there are 2 sessions accounting for roughly 37% of the overall activity on the database instance. By drilling into these statements it is possible for a DBA to see the statistics about the statement, its execution plan, what specific wait events have consumed the greatest percentage of time and what performance related recommendations Oracle recommends to improve the efficiency of the statement.

### Top Activity

Click on the band below the chart to change the time period for the detail section below.

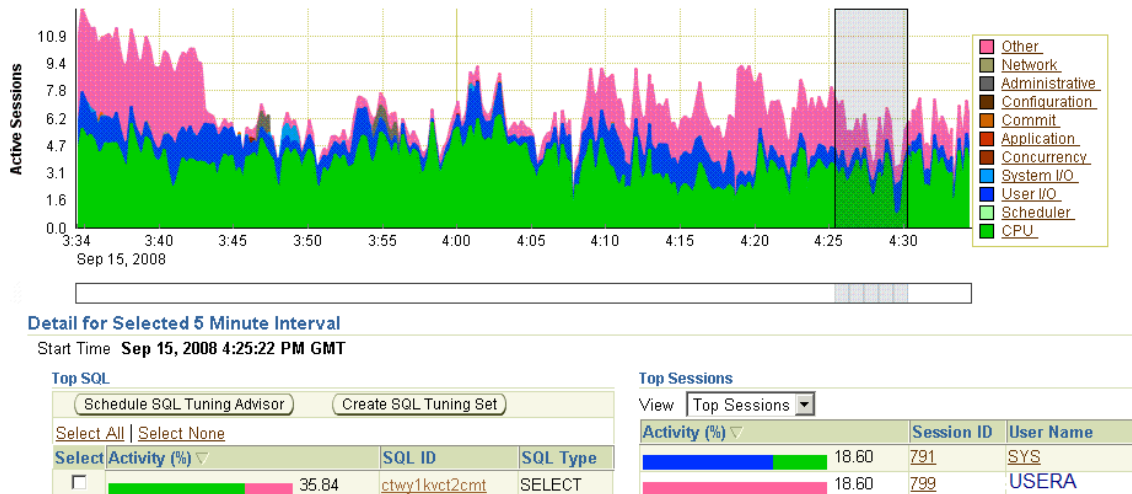


Figure 6 Top Activity

## SQL Profiles

One powerful addition to the Oracle 10g database is the SQL Profile. SQL Profiles are a continuation of Oracle’s SQL Outlines<sup>2</sup>. Both Profiles and outlines use hints to influence how a SQL plan will run. Profiles differ from outlines in that instead of essentially hard-coding the SQL plan for a particular statement, they provide additional information to the CBO in the form of hints that give more information about the data. For Example, a hint for an outline will direct the CBO to use a hash join every time the statement is run whereas a SQL Profile will inform the CBO that cardinality of a join is actually 5X greater than what that stored statistics show. Both of these hints will cause the same result (a hash join) but the difference is that while the outline method short circuits the CBO altogether and provides a hard-coded path, the profile is providing additional data to the CBO and still letting the CBO determine the best course of action.

<sup>2</sup> One big annoyance of OEM 10g is the absence of the outline editor that was present in the 9i version. An even bigger annoyance is that the 9i OEM outline Editor cannot be used to edit outlines on a 10g database requiring all outlines to be edited by hand post 9i.

## Performance Tuning on 8i and 9i Databases

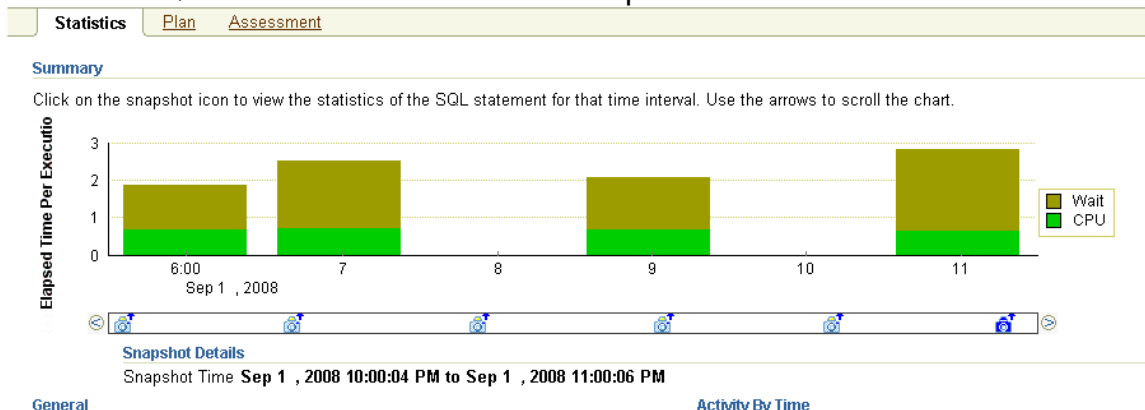
As nice as it would be to have every database on the latest release of Oracle, most environments must support legacy applications that cannot be upgraded for one reason or another. Although there is no AWR, ASH, or ADDM prior to 10g, OEM can still help diagnose performance problems in 8i and 9i databases. To maximize OEM's ability to drill into performance metrics, additional configuration is necessary. See Section 10.3 titled "Manually Configuring a Database Target for Complete Monitoring" in the Oracle Enterprise Manager Advanced Configuration Guide for detailed information on how to configure 8i and 9i databases. Once configured, DBAs can see historical data on instance activity, drill into specific SQL. To determine if an 8i or 9i database is configured to use statspack with OEM, inspect the diagnostic summary located on the main target page for the database. Figure 7 shows a target database that is properly configured. If the database is not properly configured, both the "Bad SQL" and "Top SQL Report" will show as being unavailable. To configure the database, follow the steps in section 10.3 of the Oracle Enterprise Manager Advanced Configuration Guide.

### Diagnostic Summary

Alert Log	<a href="#">Sep 15, 2008 7:34:52 PM</a>
Bad SQL	0
Top SQL Report	<a href="#">3</a>

Figure 7 The diagnostic summary of a properly configured database target.

If the database is properly configured, the statspack data can be queried similarly to how the AWR data is queried in a 10g database. Figure 8 shows a historical view of a SQL statement across several snapshots.



While SQL Profiles are not available, OEM can provide recommendations for specific SQL statements based on best practices. Take this information with a grain of salt, it is not as sophisticated as the SQL Tuning tasks in 10G.

## Contrasting OEM with Other Tuning Methods

It is clear that OEM has an array of data available for DBAs to use when performance tuning. While OEM is an excellent tool for performance tuning and one with which DBAs should become familiar, it should not be used in deference to all other tools at the DBAs disposal. Below are some other tools used for performance tuning. I'll contrast them with what is available in OEM.

### **SQL Trace**

SQL-Trace, sometimes referred to as a 10046 trace, (in reference to the event number that turns SQL tracing on and off), is a powerful utility for diagnosing individual statements. When used correctly and under the right conditions, there is, in my opinion, no better utility for tuning problem SQL. The problem lies in the first part of that last sentence. It can be exceptionally difficult to capture the right data in a trace. Consider the fairly common scenario of an N-tier application with several app servers each having their own pool of database connections, all of which connect to the database as the same user. Anyone who has tried to trace a user's actions in this situation knows that it can be nearly impossible to gather all of the SQL statements related to that user's actions.<sup>3</sup> In many cases, the information stored in statspack or the AWR can provide a root cause to a performance problem faster than it takes to execute a SQL trace.

### **DBA SQL\*Plus Scripts**

Any DBA worth his or her salt has at least a few tried and true SQL\*Plus scripts they carry from job to job for tuning and diagnostic purposes. Additionally most databases environments have some quirk or business function that requires customization to whatever out-of-box monitoring solution is being used in the environment. These scripts can be invaluable to troubleshooting and I would never recommend abandoning custom scripts altogether. I would recommend retiring scripts that have the same functionality as what comes standard with OEM. An example is the tablespace monitoring script that every DBA has written at some point in their career. Chances are good that the tablespace monitoring script supplied with OEM is more than sufficient to handle space monitoring in your environment. Retiring these scripts allows DBAs to spend time in other areas of their job instead of porting their scripts to the newest version of Oracle or rewriting to handle [insert new feature here].

---

<sup>3</sup> Oracle 10g has made this somewhat easier by enhancing Oracle's tracing capabilities and providing the trespess utility.

## **Other 3<sup>rd</sup> Party tools**

Over the years, a cottage industry has sprung up around Oracle database management and monitoring. These 3<sup>rd</sup> party applications arose in large part to fill the gaps in Oracle's Enterprise Manager product. Some of these gaps were perceived and some were real. With Oracle Enterprise manager 10GR2 most of these gaps have gone away.

Another complaint about OEM that led people to third party tools was OEM's inability to keep up with the latest features in the database. OEM would frequently lag behind command-line tools. One example of this was when Oracle introduced subpartitioning in Oracle release 8.1, OEM at the time had no ability to create a subpartitioned object requiring DBAs to use the command-line or third party tools that were more up to date with the latest database features. This is no longer the case with OEM 10G. Enterprise Manager is aware of and can manage just about every new feature in a 10G database.

One final benefit OEM has over third party tools is that it's free<sup>4</sup>

## **Conclusion**

Oracle Enterprise Manager has come a long way from the clunky application I abandoned in 1998. My hope is that reading this paper will prompt new DBAs to look have a closer look at what OEM has to offer and maybe convince some not so new DBAs to give OEM a second look.

I will end this paper with a suggestion; use OEM but continue to understand how to troubleshoot at the command-line. One pitfall DBAs risk when using OEM or any GUI tool is to become lazy and rely completely on that tool. Without understanding how OEM pulled the information or the rationale behind its suggestions, a DBA can do more harm than good. As advanced as Oracle Enterprise Manager is, it is still not a substitute for a DBA's knowledge and experience.

I am not suggesting reverse engineering the Enterprise manager code base or capturing and dissecting every SQL statement OEM runs to get its data, but a general understanding of the methods OEM employs and underlying objects (v\$ views DBA views) OEM queries is essential for every DBA.

---

<sup>4</sup> Provided a host is running Oracle Enterprise Edition, basic agent functionality (not including optional packs) is free. See Oracle's licensing documentation for more details.