July 2006 Volume II Issue 3

ORAtips On Database

Oracle® Application Database Management, Part II: Navigating Oracle Applications Manager

By Natalka Roshak

Editor's Note: Continuing the Oracle Application Database Management series, Natalka Roshak expands her discussion of OAM focusing on other aspects of OAM's database monitoring capability. What follows is a useful overview and "how-to" for navigating through OAM, managing session management, and de-mystifying waits.

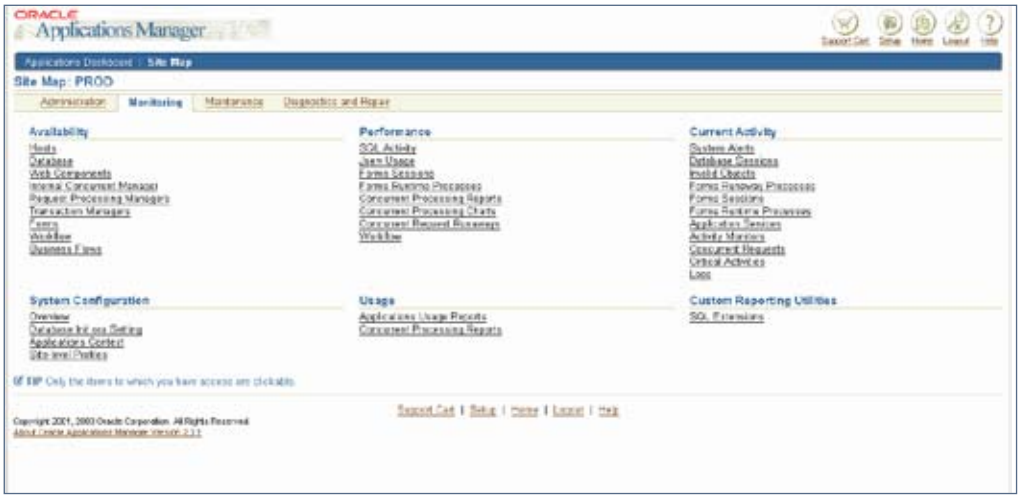

Figure 1: Main OAM Login Screen

OAM offers a broad spectrum of monitoring and analysis tools that provide the Apps DBA with a wealth of information about the database, the Web servers, application services, and concurrent requests.

Introduction

Oracle Applications Manager, or OAM, lets you monitor and administer components of your Oracle Applications instance. OAM offers a broad spectrum of monitoring and analysis tools that provide the Apps DBA with a wealth of information about the database, the Web servers, application services, and concurrent requests.

In the first article in this series, "Oracle Application Database Management, Part I: Take Database Monitoring in Oracle Applications Manager to the Next Level", we focused on one aspect of OAM's database monitoring capability, SQL Activity, and showed how to extend OAM's SQL Activity monitoring with custom SQL queries.

This article will drill down on other aspects of OAM's database monitoring capability. As in the first article, every SQL query presented here can be added to OAM by using the SQL Extensions page.

Getting Started with Oracle Applications Manager

If you haven't used Oracle Applications Manager before, there are two ways to get there. One way is to go to the Rapid Install portal page. Click on "Apps Logon Links" on the left-hand side. Choose "Oracle Applications Manager" and log in. The other way is to log in from the Sysadmin screen.

From the main screen , click on "Site Map" in the top navigation bar to take you to the Site Map (Figure 1).

From the Site Map, select the "Monitoring" tab to view OAM's monitoring capabilities (Figure 2).

OAM's database monitoring capabilities include:

- Database availability
  - Navigation: Database > Availability

ORAtips Journal

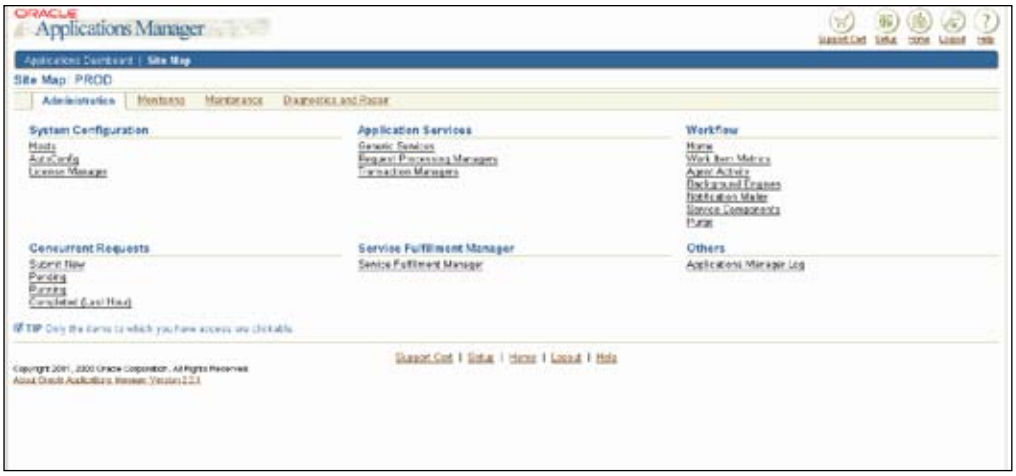

Figure 2: OAM Monitoring Screen

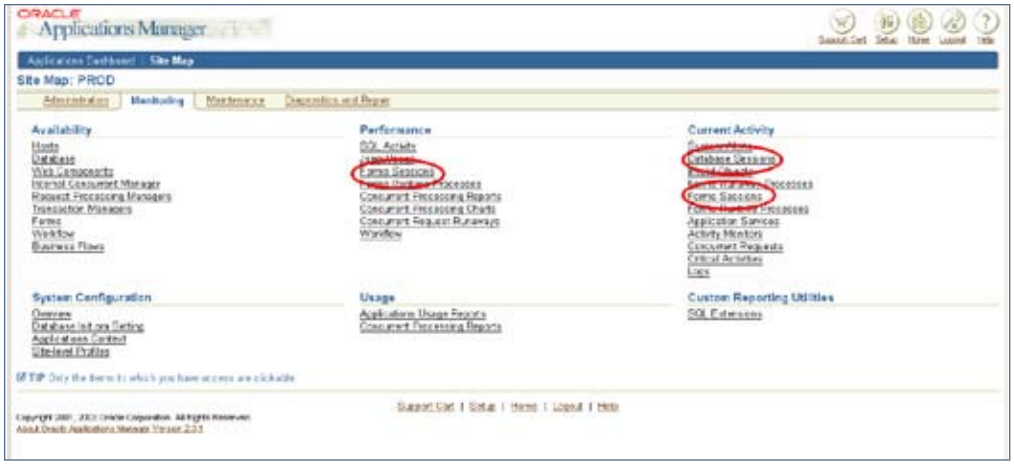

Figure 3: Places to Access Database Session Monitoring

- SQL activity
  - Navigation: Performance > SQL Activity
- Database and forms sessions
  - Navigation: Performance > Forms Sessions
  - Current Activity > Database Sessions
  - Current Activity > Forms Sessions
- Concurrent report requests
  - Navigation: Performance > Concurrent Processing Reports
- Runaway requests
  - Navigation: Performance > Concurrent Request Runaways
- Invalid database objects
  - Navigation: Current Activity > Invalid Objects
- Database logs
  - Navigation: Current Activity > Logs
- Your custom SQL reports
  - Navigation: Custom Reporting Utilities > SQL Extensions

The first article focused on the SQL Activity report, explaining and extending the information available in OAM. In this article, we will look at OAM's session and request monitoring capability.

Showing Database Session Information

There are a couple of places that you can monitor database and forms sessions:

Click on "Database Sessions" under Current Activity > Database Sessions (Figure 3) to bring up the Database Sessions screen as shown in Figure 4.

If you have set the profile option "Sign-On: Audit Level", then you can see similar information for the current forms sessions. Each open form has its own database session, which you can monitor using the navigation path Performance > Forms Sessions

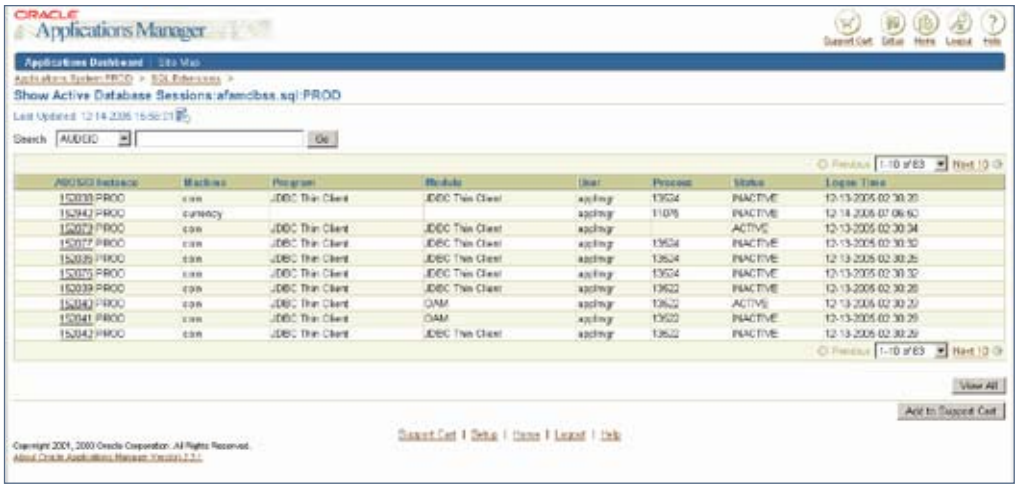

Figure 4: Session Monitoring Screen

or Current Activity > Forms Sessions.

Click on the AUDSID to bring up the Database Session Information screen, which has detailed information on that session as shown in Figure 5.

A great deal of information about the session you've clicked on is presented in this screen. What follows is a detailed explanation of the information that is presented, and how to dig down on the individual fields to get to even more in-depth information about your sessions.

Understanding OAM's Session Information

Under "Instance Attributes", OAM gives both a session ID (SID) and serial number. These two numbers together uniquely identify this session in the database. We use them to identify the session in Oracle's dynamic performance views, such as v$session and v$session_wait.

If you need to kill a client session, you can use the SID and serial number to do it. For example, to kill a client session with SID of 12 and Serial# of 3456, log in to the database as SYS, and issue an ALTER SYSTEM KILL SESSION command:

```
SYS@MYDB> alter system kill session '12,3456' ;
or
SYS@MYDB> alter system kill session '12,3456' immediate;
```

The ALTER SYSTEM KILL SESSION command:

- marks the session as killed,
- rolls back the session's transactions,
- releases the session's locks, and
- recovers the session's resources.

In some circumstances, Oracle is not able to roll back the transaction and release locks right away. For example, if the session is executing a remote transaction, then Oracle will mark the session as KILLED and wait for the remote transaction to complete. In this case,

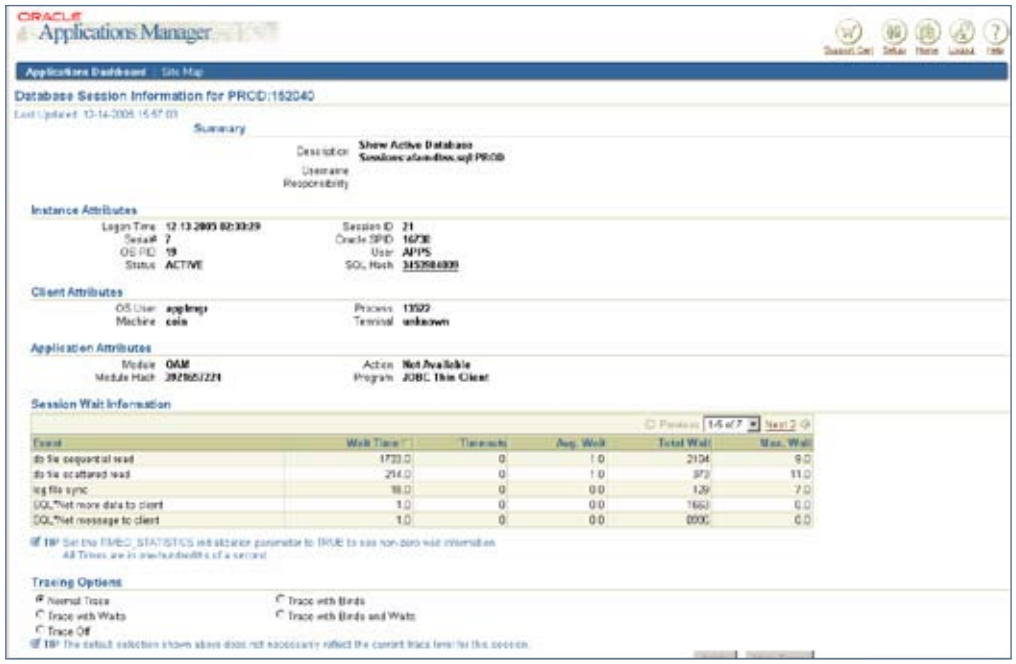

Figure 5: Session Details Screen

try killing the session with ALTER SYSTEM KILL SESSION ... IMMEDIATE.

Note: If your database is a RAC, you must make sure you're on the same instance as the session before issuing the KILL SESSION command. You can see which instance the session is on by looking at the INST_ID column of gv$session:

```
SYS@MYDB > select inst_id from
gv$session where sid=12 and 2 >
serial#=3456;
```

The next field under the "Instance Attributes" section is the OS PID. The OS PID refers to the session's Process IDentifier on the operating system. If you are running in dedicated server mode, then each client session corresponds to a unique process on the database server's operating system.

Knowing the session's PID is useful for two reasons. First, if you're unable to end a session using the following:

```
[oracle@oraserver ~]$ ps uwx
USER  PID %CPU %MEM  VSZ
START  TIME COMMAND
oracle 4924 0.0 0.6 2456076 21:58
0:00 oraclePROD (LOCAL=NO)
oracle 4989 0.0 0.3 2454512 22:01
0:00 oraclePROD (LOCAL=NO)
oracle 5021 0.7 1.8 2461236 22:02
0:08 oraclePROD (LOCAL=NO)
```

immediately with ALTER SYSTEM KILL SESSION ... IMMEDIATE, you can end it by killing the session's process at the OS level. For instance, to kill a session with OS PID 4924:

```
[oracle@oraserver ~]$ kill -9
4924
```

If your Oracle server is running on Windows, the "OS PID" actually refers to the operating system thread ID. Oracle runs as a single process on Windows; all sessions run as threads under this process. You will need to use the Oracle-provided utility ORAKILL.EXE, found in %ORACLE_HOME%\BIN, to kill a specific thread.

```
C:\> ORAKILL.EXE PROD 4924
```

ORAKILL.EXE takes the Oracle SID and the thread ID as arguments.

The second reason that the OS PID comes in handy is that you can use this PID in operating-system monitoring tools. For example, on Windows, you can use the Microsoft utility qslice.exe (available for download on the Microsoft site at www.microsoft.com/downloads/details.aspx?familyid=6247BB76-13C5-4E0E-B800-53DC1B84A94C&displaylang=en). qslice displays the % cpu usage per process, and per thread.

To monitor how much CPU thread 4924 is using, open qslice.exe and find the oracle.exe process. Click on the oracle.exe process to see a graphical breakdown of % CPU usage for that process by thread. The thread ID's shown here are in hexadecimal, so you must convert 4924 to hexadecimal (133c), then monitor the CPU usage for thread 133c.

The second reason that the OS PID comes in handy is that you can use this PID in operating-system monitoring tools.

Drilling Down on Waits

OAM's session Database Session Information page (Figure 6) includes a section on session wait events.

Note: If you do not see non-zero values in this section, your instance most likely has the initialization parameter TIMED_STATISTICS set to FALSE. Oracle recommends setting this parameter to TRUE, as it's invaluable for instance tuning and, under normal circumstances, has very little overhead.

However, it is also important to note that this session wait information is aggregate information – the average, total, and max values are the average/total/max since logon.

Session Wait Information

Previous 1-5 of 7 Next 2

Event	Wait Time	Timeouts	Avg. Wait	Total Wait	Max. Wait
db file sequential read	1733.0	0	1.0	2104	9.0
db file scattered read	214.0	0	1.0	373	11.0
log file sync	18.0	0	0.0	129	7.0
SQL*Net more data to client	1.0	0	0.0	1663	0.0
SQL*Net message to client	1.0	0	0.0	8996	0.0

Figure 6: Session Waits Information in OAM

If you're wondering what a given session is currently waiting on, this information can be next to useless. Fortunately, it's easy to get this information from the database.

Use the SID found under "Instance Attributes" in this query:

```
select *
from v$session_wait
where sid=<SID> ;
```

If a given session seems to be hanging, a quick look at the results of this query will let you know what the session is waiting on. Check the value of the "Event" column to see what the session is waiting on; the ID1 and ID2 columns hold more details you can often use to drill down on that wait event, and the WAIT_TIME and SECONDS_IN_WAIT indicate how long the session has been waiting on this event. There is a vast body of Oracle literature available on addressing specific wait events. I'll only address one of the more common wait events here: lock waits.

Wait Event: Locks

If you see a lock wait, usually called an "enq" or "enqueue" wait, in v$session_wait, you can find out who holds the blocking lock, and whether your stuck session is blocking another. Again, use the SID found under "Instance Attributes" in this query to produce the output shown in Figure 7.

And, if the lock type is TM, you can use the blocking SID (1030 in the example) to find out which object that SID is locking.

```
LOCK_INFO
----------------------------------------
SCOTT@mercury ( SID=1030 ) is blocking SCOTT@venus ( SID=982 )
```

```
select sessA.username || '@' || sessA.machine
  || ' ( SID=' || sessA.sid || ' ) is blocking '
  || sessB.username || '@' || sessB.machine
  || ' ( SID=' || sessB.sid || ' ) ' AS lock_info
from v$lock lckA, v$session sessA, v$lock lckB, v$session sessB
where sessA.sid=lckA.sid and sessB.sid=lckB.sid
and ( sessB.sid=<SID> or sessA.sid=<SID> )
and lckA.BLOCK=1 and lckB.request > 0
and lckA.id1 = lckB.id1
and lckB.id2 = lckB.id2 ;
```

Figure 7: Sample Output

USERNAME	OSUSER	SID	TY	I D1	ID2	LM	REQ	BLOCK	OBJ_NAME	OWNER
SCOTT	oracle	982	TM	187235	0	3	0	2	THE_TAB	SCOTT
SCOTT	oracle	1030	TM	187235	0	3	0	2	THE_TAB	SCOTT

Figure 8: Sample Output

```
select se.USERNAME, se.osuser, l.sid, l.type, l.id1, l.id2, lmode, request, block, do.OBJECT_NAME, do.owner

from gv$lock l, dba_objects do, gv$session se

where l.sid=se.sid and l.id1=do.object_id(+)
and l.sid in (<SID>, <BLOCKING SID>)

order by block desc, l.sid ;
```

Now that you have identified the blocking session and object, you can also look the blocking session's SID up in the Oracle Application Monitor. You will need to get the session's AUDSID from v$session. For example:

Return to the Session Monitor (Site Map > Current Activity > Database Sessions) and click on the AUDSID found above to show the Session Monitor screen for the blocking session.

Conclusion

We've seen how to drill down on sessions in Oracle Applications Manager, and how to extend OAM's information using

```
select audsid
from v$session
where sid=1030;
```

Natalka Roshak – Natalka is a Senior Oracle Database Administrator and an Oracle Certified Professional in Database Administration. She provides expert database consulting solutions across North America from her base in Southern Ontario. More of her scripts and tips can be found in her online DBA toolkit at www./toolkit.rdbms-insight.com. Natalka may be contacted at Natalka.Roshak@ERPtips.com.

Acknowledgements

The Oracle Applications Manager screenshots in this article are thanks to Jeff Slavitz, Oracle Applications Consultant, of www.OracleAppsPro.com.

This article was originally published by Klee Associates, Inc., publishers of JDEtips and SAPtips. For training, consulting, and articles on JD Edwards or SAP, please visit our websites: www.JDEtips.com and www.SAPtips.com.