AWR & ADDM are like YIN & YANG

- 7 Business Cases Explained

Yuri van Buren
Senior Oracle DBA Specialist
End-2-End Performance Engineer
CGI Nederland B.V.
Rotterdam – The Netherlands

Keywords: Performance Management, Trouble Shooting, Capacity Management

Introduction

In this presentation Yuri will demonstrate the power of combining the Automatic Workload Repository (AWR) and Automatic Database Diagnostics Monitor (ADDM) reports.

Using 7 real life business cases, it will become clear that old school gut feeling from an AWR report can be made very firm by combining AWR data with the information you can get from an ADDM report. The information of these two reports is so complementary that they fit like Yin and Yang.

From this presentation onwards, you will always use the power of those two reports together. Together these reports will give you "*the whole picture*" of performance issues on your Oracle database system.

After a basic introduction of AWR and ADDM. You will learn my method to read an AWR report.
During my presentation in I will explain 7 different business cases.

Basics of generating the AWR and ADDM reports

Scripts to create these reports are available on the Oracle RDBMS Server from 10g onwards.
They are located in the $ORACLE_HOME/rdbms/admin directory.

Script to generate the AWR report - awrrpt.sql (Main report, like Statspack in text or HTML)

Other great AWR scripts:

- awrsqrpt.sql (Workload report for a particular sql_id)
- awrddrpt.sql (Workload Repository Compare Periods report)
- awrextr.sql (To extract AWR data with datapump)
- awrload.sql (To load data from an awrextr dump).

Script for generating the ADDM report - addmrpt.sql (Only text output possible).

AWR is necessary for the advisors to give recommendations.
AWR is installed in the SYSAUX tablespace and cannot be moved!
AWR data can be exported to a data warehouse.

Setting the interval and retention times for the Automatic Workload Repository

AWR is enabled by default every hour and kept for 7 days (10g) or 8 days (11g) only!

Query the current values with:

```
 select snap_interval, retention from dba_hist_wr_control;

SNAP_INTERVAL                  RETENTION
------------------------------ ------------------------------
+00000 01:00:00.0              +00007 00:00:00.0
```

Here is an example of setting the interval to 20 minutes and the retention time to 14 days.

```
begin
    dbms_workload_repository.modify_snapshot_settings (
       interval => 0, retention => 14*24*60
    );
end;
/
```

Setting Interval to zero minutes, disables AWR data collection!!

A manual AWR snapshot can be created with:

```
execute dbms_workload_repository.create_snapshot;
```

This command is necessary when you are not allowed to change the interval but want to get measurements with different intervals.

Method to read an AWR report

When I read an AWR Report I follow these steps.

1) Check that you look at the correct database instance on the correct machine and check that the time interval is the one you want to analyze. Calculate the average active sessions (AAS=DB Time / Elapsed Time). When AAS >> Nr of CPU's on the server you are in trouble.

2) Go to the top Timed Events section and see which Events are on top. Is it **CPU**, **physical IO** (db file sequential read + db file scattered read, direct path read) or physical IO caused by the Logwriter visible as log file sync or log file parallel write, **or is it contention** e.g. Latch issues (buffer busy waits, library cache, shared pool etc). or Locking (Enqueue) issues.

   a) **High CPU usage** is most often caused by too many Logical IOs per second (LIOs/sec). Check the Logical Reads Per Second in the "Load Profile" section of the AWR report. Check the amount of CPU used per second. This can be found in the "Instance Activity Stats" section when you look at the value of "CPU used by this session" per second. The per second metric shows the amount of 1/100st of CPU seconds. So a value of 140 means you had 1.4 CPU's working for the Oracle instance during your snapshot interval. Remember when you get close to the actual number of CPU's in the system you are reaching the non-linear zone. On a response time curve you are above the knee of the curve. From the CPU used by this session metric and the Logical Reads per second metric you can calculate and "spec" your instance its "Effective CPU Speed", expressed in µs per LIO. The reciprocal value of this gives you the amount of LIOs/sec that the Oracle instance can run on one CPU.

      E.g. Logical reads per second: 14180 and CPU used by this session per second 20.7

      0.207/14180 = 14,5us CPU per LIO.
      And reciprocal it is 14180/0.207 = 68502 LIOs/sec per CPU.

      Go to the top LIO per segment section. Check how many objects cause in total 80% of the load. Sometimes only 1 to 5 segments are causing almost all of the LIOs. Go to the "SQL ordered by (buffer) Gets section". A bufferget is a synonym for a Logical IO. See which queries take the most time.

b) When a lot of time is spent on **physical IO** on "db file ... " related events. It is good to calculate the average physical IO times for the scattered and sequential read. For the whole instance I can calculate the average PIO call time:

   Average PIO call (ms) = (time waited for scattered and sequential reads) / physical reads count
   This gives an indication if there are slow disks (E.g. PIO Call times well above 6 or 10ms).

   Go to the top PIO per segment section. Check how many objects cause in total 80% of the load. Sometimes only 1 to 5 segments are causing almost all of the PIOs. Go to the "SQL ordered by Reads section". A "Read" is a synonym for a Physical IO. See which queries take the most time. If you have a very high PIO rate, check the sizing of the db_cache_size. A good estimated of a bigger db_cache size and its effect on reducing the number of physical reads can be seen at the Buffer Pool Advisory. When you did find Full Table scans in the SQL section, you should first tune these before you start bumping up the db_cache_size.

   For the pga_aggregate_target you can check the PGA Memory Advisory section, set the pga_aggregate_target to the size that is mentioned in the first line where Estd PGA Overalloc Count is 0. Size factor 1.0 shows what the current pga_aggregate_target value is.

   In the SGA breakdown difference section you can check the free memory settings of your java large pool and shared pool. Check if these pools are oversized (or even used .. for the java and

large pool!). You might be able to shift memory from these pools to the db_cache_size or pga_aggregate_target.

When a lot of physical IO is spent on **"log file sync" and "log file parallel write"** , you need to investigate the commit times. Are there many concurrent users and is there a high commit rate or is time lost due to the slow writing into the redologfiles?

c) When there is **contention** look in the respective sections of "Latch Activity" or "Enqueue Activity" to determine the type of problem. Try to understand why Oracle needs so many latches and/or enqueues at the same time. If it is the dictionary of shared pool latch you might have a shared_pool_size that is too small. If it is the "cache buffer chain latch", you might see a symptom of a system that is doing way too much LIOs/sec. It might be that you are scanning a lot of unselective indexes for instance. Fixing the excessive LIO problem often fixes the latch issue!

A glimpse of the 7 business cases:

1) High CPU time ... caused by Full Table Scans?!
2) High SQL*Net more data from client ... Sql Id not in AWR, but in ADDM!
3) High direct path reads ... Which Lob is Hot?
4) High CPU time ... Is it SQL or PL/SQL?
5) High "enq: TX - row lock contention" ... what is causing it?
6) High Buffer Busy Waits ... We are off the charts.
7) High Log File Sync and Log File Parallel Write times.

I will show that the assumptions from reading an AWR report can be made very firm (up to 100%) by combining the AWR data with data from ADDM.

Summary

From now on you will always generate both AWR and ADDM reports from the same time and use the power of these two tools together to provide you with more proof (up to 100%) of performance issues that occur.

Contact address:

Name	**Yuri van Buren**
	Principal IT Consultant | Senior Oracle DBA Specialist
Company	**CGI Nederland B.V.**
Address	George Hintzenweg 89
	3068AX Rotterdam
	The Netherlands
Phone:	+31 (0)88 5640000
Email	yuri.van.buren@cgi.com
Internet:	www.cgi.com