Reading AWR 11Gr2

# Report Header

This section is self explanatory which provides database name, id, instance if RAC , platform information and snap interval. (database workload time duration in review).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **DB Name** | **DB Id** | **Instance** | **Inst num** | **Startup Time** | **Release** | **RAC** |
| TestRAC | 3626203793 | TestRac2 | 2 | 17-Aug-11 19:08 | 11.1.0.6.0 | YES |

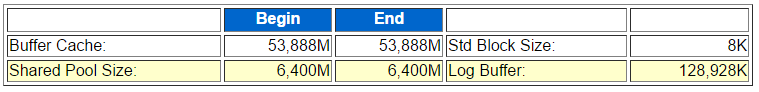
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Host Name** | **Platform** | **CPUs** | **Cores** | **Sockets** | **Memory (GB)** |
| TestRAC | Linux 64-bit for AMD | 8 | 8 | 2 | 31.44 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Snap Id** | **Snap Time** | **Sessions** | **Cursors/Session** |
| Begin Snap: | 28566 | 27-Sep-11 01:00:21 | 130 | 4.8 |
| End Snap: | 28567 | 27-Sep-11 02:00:43 | 135 | 4.5 |
| Elapsed: |  | 60.35 (mins) |  |  |
| DB Time: |  | 15.07 (mins) |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Begin** | **End** |  |  |
| Buffer Cache: | 5,888M | 5,888M | Std Block Size: | 8K |
| Shared Pool Size: | 8,704M | 8,704M | Log Buffer: | 138,328K |

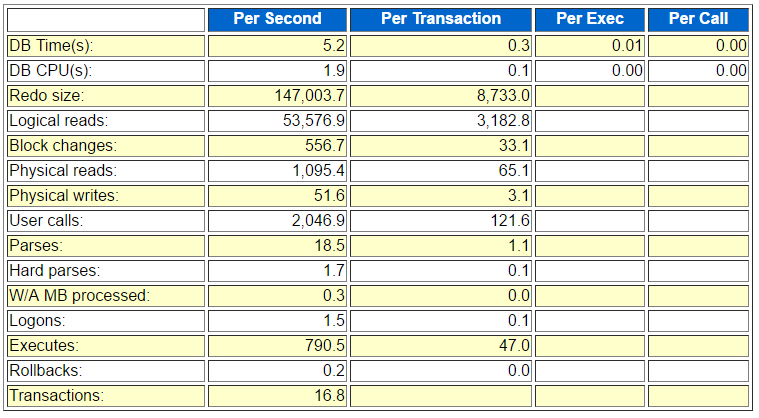
# Report Summary

## Cache Sizes



## Load Profile

This section provides the snapshot of the database workload occurred during the snapshot interval.



**DB time(s):** Its the amount of time oracle has spent performing database user calls. Note it does not include background processes.

**DB CPU(s):** Its the amount of CPU time spent on user calls. Same as DB time it does not include background process. The value is in microseconds

**Redo size:** For example, the table below shows that an average transaction generates about 19,000 of redo data along with around 48,000 redo per second.

**Logical reads:** Consistent Gets+ DB blocks Gets = Logical reads

**Block Changes:** The number of block modified during the sample interval

**Physical reads:** No of block request causing I/O operation

**Physical writes:** Number of physical writes performed

**User calls:** Number of user queries generated

**Parses:** The total of all parses; both hard and soft.

**Hard Parses:** The parses requiring a completely new parse of the SQL statement. These consume both latches and shared pool area.

**Soft Parses:** Soft parses are not listed but derived by subtracting the hard parses from parses. A soft parse reuses a previous hard parse; hence it consumes far fewer resources.

**Sorts:** No of sorts performed

**Logons:** No of logons during the interval

**Executes:** No of SQL Executes

**Transactions:** No of transactions per second

---------------------------------------------------------------------------

Load Profile

The load profile provides an at-a-glance look at some specific operational statistics. You can compare these statistics with a baseline snapshot report to determine if database activity is different. Values for these statistics are presented in two formats. The first is the value per second (for example, how much redo was generated per second) and the second is the value per transaction (for example, 1,024 bytes of redo were generated per transaction).

Statistics presented in the load profile include such things as:

Redo size - An indication of the amount of DML activity the database is experiencing.

Logical and physical reads - A measure of how many IO's (Physical and logical) that the database is performing.

User calls - Indicates how many user calls have occurred during the snapshot period. This value can give you some indication if usage has increased

Parses and hard parses - Provides an indication of the efficiency of SQL re-usage.

Sorts - This number gives you an indication of how much sorting is occurring in the database

Logons - Indicates how many logons occurred during the snapshot period.  
Executes - Indicates how many SQL statements were executed during the snapshot period.  
Transactions - Indicates how many transactions occurred during the snapshot period.

Additionally, the load profile section provides the percentage of blocks that were changed per read, the percentage of recursive calls that occurred, the percentage of transactions that were rolled back and the number of rows sorted per sort operation.

## Instance Efficiency Percentages (Target 100%)

 These statistics include several buffer related ratios including the buffer hit percentage and the library hit percentage. Also, shared pool memory usage statistics are included in this section.

Instance efficiency should be close to 100 %

|  |  |  |  |
| --- | --- | --- | --- |
| Buffer Nowait %: | 99.99 | Redo NoWait %: | 100.00 |
| Buffer Hit %: | 93.06 | In-memory Sort %: | 100.00 |
| Library Hit %: | 98.67 | Soft Parse %: | 98.20 |
| Execute to Parse %: | 3.40 | Latch Hit %: | 99.98 |
| Parse CPU to Parse Elapsd %: | 0.01 | % Non-Parse CPU: | 96.21 |

**Execute to Parse % and Parse CPU to Parse Elapsd %:** If the the value are low like in the above case of 3.40 and 0.01 means that there could be a parsing problem. You may need to look at bind variable issues or shared pool sizing issue.

**Redo NoWait%:** Usually this stats is 99 or greater

**In-memory Sort %:** This can tell you how efficient is you sort\_area\_size, hash\_area\_size or pga\_aggrigate\_target are. If you dont have adequate sizes of sort,hash and pga parameters, then you in-memory sort per cent will go down

**Soft parse %:** with 98.20 % for the soft parse meaning that about 1.72 % (100 -soft parse) is happening for hard parsing. You might want to look at you bind variables issues.

**Latch Hit %:** should be close to 100.

**% Non-Parse CPU:** Most of our statements were already parsed so we weren't doing a lot of re parsing. Re parsing is high on CPU and should be avoided.

------------------------------------------------------------------------------------------------------------

## Shared Pool Statistics

|  |  |  |
| --- | --- | --- |
|  | **Begin** | **End** |
| Memory Usage %: | 73.86 | 75.42 |
| % SQL with executions>1: | 92.61 | 93.44 |
| % Memory for SQL w/exec>1: | 94.33 | 94.98 |

Memory Usage % is the shared pool usage. So here we have use 73.86 per cent of our shared pool and out of that almost 94 percent is being re-used. if Memory Usage % is too large like 90 % it could mean that your shared pool is tool small and if the percent is in 50 for example then this could mean that you shared pool is too large

## Top 5 Timed Foreground Events

This section provides insight into what events the Oracle database is spending most of it's time on (see wait events). Each wait event is listed, along with the number of waits, the time waited (in seconds), the average wait per event (in microseconds) and the associated wait class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Event** | **Waits** | **Time(s)** | **Avg wait (ms)** | **% DB time** | **Wait Class** |
| DB CPU |  | 1,019 |  | 112.73 |  |
| log file sync | 25,642 | 43 | 2 | 4.73 | Commit |
| db file scattered read | 3,064 | 40 | 13 | 4.43 | User I/O |
| library cache pin | 136,267 | 27 | 0 | 2.98 | Concurrency |
| db file sequential read | 7,608 | 24 | 3 | 2.71 | User I/O |

its critical to look into this section. If you turn off the statistic parameter, then the Time(s) wont appear. Wait analysis should be done with respect to Time(s) as there could be million of waits but if that happens for a second or so then who cares. Therefore, time is very important component.

So you have several different types of waits. So you may see the different waits on your AWR report. So lets discuss the most common waits

**df file type waits:**

**db file sequential read**: Is the wait that comes from the physical side of the database. it related to memory starvation and non selective index use. sequential read is an index read followed by table read because it is doing index lookups which tells exactly which block to go to

**db file scattered read:** caused due to full table scans may be because of  insufficient indexes or un-avilablity of updated statistics

**direct Path writes:** You wont see them unless you are doing some appends or data loads

**direct Path reads**: could happen if you are doing a lot of parallel query activity

**db file parallel writes / read:** if you are doing a lot of partition activity then expect to see that wait even. it could be a table or index partition

**db file single write:** if you see this event than probably you have a lot of data files in your database.

**direct path read temp or direct path write temp:** this wait event shows Temp file activity (sort,hashes,temp tables, bitmap). Check pga parameter or sort area or hash area parameters. You might want to increase them

**buffer type waits**so what's going on in your memory

**latch: cache buffer chains:** check hot objects

**free buffer waits:** insufficient buffers, process holding buffers too long or i/o subsystem is over loaded. Also check you db writes may be getting clogged up.

**buffer busy waits:**

see what is causing them further along in report. most of the time its data block related.

**gc buffer busy:**

its in the RAC environment. caused may be because of not enough memory on your nodes,overloaded interconnect. Also look RAC specific section of the report latch:  
**cache buffers lru chain** – Freelist issues, hot blocks latch: cache buffer handles –

Freelist issues, hot blocks

**buffer busy**- See what is causing them further along in report

**no free buffers** – Insufficient buffers, dbwr contention

**Log Type Waits**

**log file parallel write** – Look for log file contention

**log buffer space** – Look at increasing log buffer size

**log file switch (checkpoint incomplete)**– May indicate excessive db files or slow IO subsystem  
**log file switch (archiving needed)** – Indicates archive files are written too slowly

**log file switch completion** – May need more log files per

**log file sync** – Could indicate excessive commits

**GC Events**

**gccr multi block request** – Full table or index scans

**gc current multi block request** – Full table or index scans

**gccr block 2-way** – Blocks are busy in another instance, check for block level contention or hot blocks

**gccr block 3-way** – Blocks are busy in another instance, check for block level contention or hot blocks

**gccr block busy**– Blocks are busy in another instance, check for block level contention or hot blocks

**gccr block congested** – cr block congestion, check for hot blocks or busy interconnect

**gccr block lost**– Indicates interconnect issues and contention

**gc current block 2-way** – Blocks are busy in another instance, check for block level

contention or hot blocks

**gc current block 3-way** – Blocks are busy in another instance, check for block level contention or hot blocks

**gc current block busy** – Block is already involved in GC operation, shows hot blocks or congestion

**gc current block congested**– current block congestion, check for hot blocks or busy interconnect

**gc current block lost** - Indicates interconnect issues and contention

**Undo Events**

**undo segment extension** – If excessive, tune undo

**latch: In memory undo latch** – If excessive could be bug, check for your version, may have to turn off in memory undo

**wait for a undo record**– Usually only during recovery of large transactions, look at turning off parallel undo recovery.

# RAC Statistics

If you are running on a RAC cluster, then the AWRRPT.SQL report will provide various RAC statistics including statistics on the number of RAC instances, as well as global cache and enqueue related performance statistics. Here is an example of the RAC statistics part of the report:

RAC Statistics DB/Inst: A109/a1092 Snaps: 2009-2010

Number of Instances: 2

## Global Cache Load Profile

Global Cache Load Profile Per Second Per Transaction

Global Cache blocks received: 0.11 0.52

Global Cache blocks served: 0.14 0.68

GCS/GES messages received: 0.88 4.23

GCS/GES messages sent: 0.85 4.12

DBWR Fusion writes: 0.01 0.04

Estd Interconnect traffic (KB) 2.31

## Global Cache Efficiency Percentages (Target local+remote 100%)

Buffer access - local cache %: 99.47

Buffer access - remote cache %: 0.53

Buffer access - disk %: 0.00

## Global Cache and Enqueue Services - Workload Characteristics

Avg global enqueue get time (ms): 0.0

Avg global cache cr block receive time (ms): 0.2

Avg global cache current block receive time (ms): 0.3

Avg global cache cr block build time (ms): 0.0

Avg global cache cr block send time (ms): 0.0

Global cache log flushes for cr blocks served %: 1.8

Avg global cache cr block flush time (ms): 4.0

Avg global cache current block pin time (ms): 0.0

Avg global cache current block send time (ms): 0.1

Global cache log flushes for current blocks served %: 0.4

Avg global cache current block flush time (ms): 0.0

## Global Cache and Enqueue Services - Messaging Statistics

Avg message sent queue time (ms): ########

Avg message sent queue time on ksxp (ms): 0.1

Avg message received queue time (ms): 4.6

Avg GCS message process time (ms): 0.0

Avg GES message process time (ms): 0.0

% of direct sent messages: 45.26

% of indirect sent messages: 31.59

% of flow controlled messages: 23.15

## Cluster Interconnect

Begin End

Interface IP Address Pub Source IP Pub Src

ce3 172.16.0.130 N

ce5 172.16.1.2 N

clprivnet0 172.16.4.2 N

# Time Model Statistics

Oracle Database 10g time model related statistics are presented next. The time model allows you to see a summary of where the database is spending it's time. The report will present the various time related statistic (such as DB CPU) and how much total time was spent in the mode of operation represented by that statistic. Here is an example of the time model statistic report where we see that we spent 36.2 seconds on DB CPU time, which was a total of 60.4% of the total DB time. Note that this is a two node RAC system, so the total percentage of overall time available is 200%, not 100%.

Time Model Statistics DB/Inst: A109/a1092 Snaps: 2009-2010

-> Total time in database user-calls (DB Time): 5.5s

-> Statistics including the word "background" measure background process

time, and so do not contribute to the DB time statistic

-> Ordered by % or DB time desc, Statistic name

Statistic Name Time (s) % of DB Time

------------------------------ ------------------ ------------

sql execute elapsed time 4.5 82.8

DB CPU 3.5 64.4

connection management call elapsed time 0.1 1.6

parse time elapsed 0.1 1.3

PL/SQL execution elapsed time 0.0 .9

hard parse elapsed time 0.0 .3

sequence load elapsed time 0.0 .1

repeated bind elapsed time 0.0 .0

DB time 5.5

background elapsed time 33.0

background cpu time 9.7

# Wait class and Wait Events Statistics

Closely associated with the time model section of the report are the wait class and wait event statistics sections. Within Oracle, the duration of a large number of operations (e.g. Writing to disk or to the control file) is metered. These are known as wait events, because each of these operations requires the system to wait for the event to complete. Thus, the execution of some database operation (e.g. a SQL query) will have a number of wait events associated with it. We can try to determine which wait events are causing us problems by looking at the wait classes and the wait event reports generated from AWR.

Wait classes define "buckets" that allow for summation of various wait times. Each wait event is assigned to one of these buckets (for example System I/O or User I/O). These buckets allow one to quickly determine which subsystem is likely suspect in performance problems (e.g. the network, or the cluster). Here is an example of the wait class report section:

Wait Class DB/Inst: A109/a1092 Snaps: 2009-2010

-> s - second

-> cs - centisecond - 100th of a second

-> ms - millisecond - 1000th of a second

-> us - microsecond - 1000000th of a second

-> ordered by wait time desc, waits desc

Avg

%Time Total Wait wait Waits

Wait Class Waits -outs Time (s) (ms) /txn

-------------------- ------- ------ -------- ------ --------

System I/O 8,142 .0 25 3 10.9

Other 439,596 99.6 3 0 589.3

User I/O 112 .0 0 3 0.2

Cluster 443 .0 0 0 0.6

Concurrency 216 .0 0 0 0.3

Commit 16 .0 0 2 0.0

Network 3,526 .0 0 0 4.7

Application 13 .0 0 0 0.0

In this report the system I/O wait class has the largest number of waits (total of 25 seconds) and an average wait of 3 milliseconds.

Wait events are normal occurrences, but if a particular sub-system is having a problem performing (e.g. the disk sub-system) this fact will appear in the form of one or more wait events with an excessive duration. The wait event report then provides some insight into the detailed wait events. Here is an example of the wait event report (we have eliminated some of the bulk of this report, because it can get quite long).

Avg

%Time Total Wait wait Waits

Event Waits -outs Time (s) (ms) /txn

------------ -------------- ------ ----------- ------- --------

control file parallel write 1,220 .0 18 15 1.6

control file sequential read 6,508 .0 6 1 8.7

CGS wait for IPC msg 422,253 100.0 1 0 566.0

change tracking file synchro 60 .0 1 13 0.1

db file parallel write 291 .0 0 1 0.4

db file sequential read 90 .0 0 4 0.1

reliable message 136 .0 0 1 0.2

log file parallel write 106 .0 0 2 0.1

lms flush message acks 1 .0 0 60 0.0

gc current block 2-way 200 .0 0 0 0.3

change tracking file synchro 59 .0 0 1 0.1

In this example our control file parallel write waits (which occurs during writes to the control file) are taking up 18 seconds total, with an average wait of 15 milliseconds per wait. Additionally we can see that we have 1.6 waits per transaction (or 15ms \* 1.6 per transaction = 24ms).

# Operating System Statistics

This part of the report provides some basic insight into OS performance, and OS configuration too. This report may vary depending on the OS platform that your database is running on. Here is an example from a Linux system:

Statistic Total

-------------------------------- --------------------

BUSY\_TIME 128,749

IDLE\_TIME 1,314,287

IOWAIT\_TIME 18,394

NICE\_TIME 54

SYS\_TIME 31,633

USER\_TIME 96,586

LOAD 0

RSRC\_MGR\_CPU\_WAIT\_TIME 0

PHYSICAL\_MEMORY\_BYTES 3,349,528

NUM\_CPUS 4

In this example output, for example, we have 4 CPU's on the box.

# SQL In Need of Tuning

Next in the report we find several different reports that present SQL statements that might be improved by tuning. There are a number of different reports that sort offending SQL statements by the following criteria:

Elapsed time, CPU time, Buffer gets, Physical reads, Executions,

Parse calls, Sharable memory, Version count, Cluster wait time

While these reports might not help tune specific application problems, they can help you find more systemic SQL problems that you might not find when tuning a specific application module. Here is an example of the Buffer gets report:

Gets CPU Elapsed

Buffer Gets Executions per Exec %Total Time (s) Time (s) SQL Id

----------- ---------- -------- ----- -------- ------- -----------

2,163 7 309.0 3.0 0.03 0.04 c7sn076yz7030

In this report we find a SQL statement that seems to be churning through 309 buffers per execution. While the execution times are not terrible we might want to look closer into the SQL statement and try to see if we could tune it (in fact this is Oracle issued SQL that we would not tune anyway).

# Instance Activity Stats

This section provides us with a number of various statistics (such as, how many DBWR Checkpoints occurred, or how many consistent gets occurred during the snapshot). Here is a partial example of the report:

Statistic Total per Second per Trans

---------------- --------- -------------- -------------

consistent changes 9 0.0 0.0

consistent gets 70,445 19.5 94.4

consistent gets - examination 8,728 2.4 11.7

consistent gets direct 0 0.0 0.0

consistent gets from cache 70,445 19.5 94.4

cursor authentications 2 0.0 0.0

data blocks consistent reads - u 5 0.0 0.0

db block changes 1,809 0.5 2.4

db block gets 2,197 0.6 3.0

db block gets direct 0 0.0 0.0

db block gets from cache 2,033 0.6 2.7

# Tablespace and Data File IO Stats

The tablespace and data file IO stats report provides information on tablespace IO performance. From this report you can determine if the tablespace datafiles are suffering from sub-standard performance in terms of IO response from the disk sub-system. Here is a partial example of the tablespace report:

Tablespace

Av Av Av Av Buffer Av Buf

Reads Reads/s Rd(ms) Blks/Rd Writes Writes/s Waits Wt(ms)

-------------- ------- ------ ------- ------------ -------- ---------- --

SYSAUX 1 0 0.0 1.0 159 0 13 0.8

UNDOTBS2 1 0 10.0 1.0 98 0 0 0.0

SYSTEM 1 0 10.0 1.0 46 0 0 0.0

AUD 1 0 0.0 1.0 1 0 0 0.0

UNDOTBS1 1 0 10.0 1.0 1 0 0 0.0

USERS 1 0 10.0 1.0 1 0 0 0.0

USER\_DEF 1 0 10.0 1.0 1 0 0 0.0

If the tablespace IO report seems to indicate a tablespace has IO problems, we can then use the file IO stat report allows us to drill into the datafiles of the tablespace in question and determine what the problem might be. Here is an example of the File IO stat report:

Tablespace Filename

------------------------ ----------------------------------------------------

Av Av Av Av Buffer Av Buf

Reads Reads/s Rd(ms) Blks/Rd Writes Writes/s Waits Wt(ms)

-------------- ------- ------ ------- ------------ -------- ---------- ------

AUD +ASM01/a109/datafile/aud.296.604081931

1 0 0.0 1.0 1 0 0 0.0

CDOL2\_INDEX +ASM01/a109/datafile/cdol2\_index\_001.dbf

1 0 10.0 1.0 1 0 0 0.0

CDOL\_DATA +ASM01/a109/datafile/cdol\_data\_001.dbf

1 0 10.0 1.0 1 0 0 0.0

DBA\_DEF +ASM01/a109/datafile/dba\_def.294.604081931

1 0 10.0 1.0 1 0 0 0.0

SYSAUX +ASM01/a109/datafile/sysaux.299.604081927

1 0 0.0 1.0 159 0 13 0.8

SYSTEM +ASM01/a109/datafile/system.301.604081919

1 0 10.0 1.0 46 0 0 0.0

UNDOTBS1 +ASM01/a109/datafile/undotbs1.300.604081925

1 0 10.0 1.0 1 0 0 0.0

UNDOTBS2 +ASM01/a109/datafile/undotbs2.292.604081931

1 0 10.0 1.0 98 0 0 0.0

USERS +ASM01/a109/datafile/users.303.604081933

1 0 10.0 1.0 1 0 0 0.0

USER\_DEF +ASM01/a109/datafile/user\_def.291.604081933

1 0 10.0 1.0 1 0 0 0.0

-------------------------------------------------------------

Buffer Pool Statistics

The buffer pool statistics report follows. It provides a summary of the buffer pool configuration and usage statistics as seen in this example:

Free Writ Buffer

Number of Pool Buffer Physical Physical Buff Comp Busy

P Buffers Hit% Gets Reads Writes Wait Wait Waits

--- ---------- ---- -------------- ------------ ----------- ---- ---- ----------

D 64,548 100 72,465 0 355 0 0 13

-------------------------------------------------------------

In this case, we have a database where all the buffer pool requests came out of the buffer pool and no physical reads were required. We also see a few (probably very insignificant in our case) buffer busy waits.

Instance Recovery Stats

The instance recovery stats report provides information related to instance recovery. By analyzing this report, you can determine roughly how long your database would have required to perform crash recovery during the reporting period. Here is an example of this report:

-> B: Begin snapshot, E: End snapshot

Targt Estd Log File Log Ckpt Log Ckpt

MTTR MTTR Recovery Actual Target Size Timeout Interval

(s) (s) Estd IOs Redo Blks Redo Blks Redo Blks Redo Blks Redo Blks

- ----- ----- ---------- --------- --------- ---------- --------- ------------

B 0 19 196 575 183 92160 183 N/A

E 0 19 186 258 96 92160 96 N/A

-------------------------------------------------------------

# Buffer Pool Advisory

The buffer pool advisory report answers the question, how big should you make your database buffer cache. It provides an extrapolation of the benefit or detriment that would result if you added or removed memory from the database buffer cache. These estimates are based on the current size of the buffer cache and the number of logical and physical IO's encountered during the reporting point. This report can be very helpful in "rightsizing" your buffer cache. Here is an example of the output of this report:

Est

Phys

Size for Size Buffers for Read Estimated

P Est (M) Factor Estimate Factor Physical Reads

--- -------- ------ ---------------- ------ ------------------

D 48 .1 5,868 4.9 803,496

D 96 .2 11,736 4.0 669,078

D 144 .3 17,604 3.3 550,831

D 192 .4 23,472 2.8 462,645

D 240 .5 29,340 2.3 379,106

D 288 .5 35,208 1.8 305,342

D 336 .6 41,076 1.4 238,729

D 384 .7 46,944 1.2 200,012

D 432 .8 52,812 1.1 183,694

D 480 .9 58,680 1.0 172,961

D 528 1.0 64,548 1.0 165,649

D 576 1.1 70,416 1.0 161,771

D 624 1.2 76,284 1.0 159,728

D 672 1.3 82,152 1.0 158,502

D 720 1.4 88,020 1.0 157,723

D 768 1.5 93,888 0.9 157,124

D 816 1.5 99,756 0.9 156,874

D 864 1.6 105,624 0.9 156,525

D 912 1.7 111,492 0.9 156,393

D 960 1.8 117,360 0.9 155,388

-------------------------------------------------------------

In this example we currently have 528GB allocated to the SGA (represented by the size factor column with a value of 1.0. It appears that if we were to reduce the memory allocated to the SGA to half of the size of the current SGA (freeing the memory to the OS for other processes) we would incur an increase of about 1.8 times the number of physical IO's in the process.

PGA Reports

The PGA reports provide some insight into the health of the PGA. The PGA Aggr Target Stats report provides information on the configuration of the PGA Aggregate Target parameter during the reporting period.

The PGA Aggregate Target Histogram report provides information on the size of various operations (e.g. sorts). It will indicate if PGA sort operations occurred completely in memory, or if some of those operations were written out to disk.

Finally the PGA Memory Advisor, much like the buffer pool advisory report, provides some insight into how to properly size your PGA via the PGA\_AGGREGATE\_TARGET database parameter. The PGA Memory Advisor report is shown here:

Estd Extra Estd PGA Estd PGA

PGA Target Size W/A MB W/A MB Read/ Cache Overalloc

Est (MB) Factr Processed Written to Disk Hit % Count

---------- ------- ---------------- ---------------- -------- ----------

44 0.1 289,899.2 7,844.9 97.0 1,124

88 0.3 289,899.2 7,576.9 97.0 1,073

176 0.5 289,899.2 3.3 100.0 0

263 0.8 289,899.2 3.3 100.0 0

351 1.0 289,899.2 3.3 100.0 0

421 1.2 289,899.2 0.0 100.0 0

491 1.4 289,899.2 0.0 100.0 0

562 1.6 289,899.2 0.0 100.0 0

632 1.8 289,899.2 0.0 100.0 0

702 2.0 289,899.2 0.0 100.0 0

1,053 3.0 289,899.2 0.0 100.0 0

1,404 4.0 289,899.2 0.0 100.0 0

2,106 6.0 289,899.2 0.0 100.0 0

2,808 8.0 289,899.2 0.0 100.0 0

-------------------------------------------------------------

Shared Pool Advisory

The shared pool advisory report provides assistance in right sizing the Oracle shared pool. Much like the PGA Memory Advisor or the Buffer Pool advisory report, it provides some insight into what would happen should you add or remove memory from the shared pool. This can help you reclaim much needed memory if you have over allocated the shared pool, and can significantly improve performance if you have not allocated enough memory to the shared pool. Here is an example of the shared pool advisory report:

Est LC Est LC Est LC Est LC

Shared SP Est LC Time Time Load Load Est LC

Pool Size Size Est LC Saved Saved Time Time Mem

Size(M) Factr (M) Mem Obj (s) Factr (s) Factr Obj Hits

---------- ----- -------- ------------ ------- ------ ------- ------ -----------

192 .4 54 3,044 ####### .8 ####### 382.1 22,444,274

240 .5 92 5,495 ####### .9 ####### 223.7 22,502,102

288 .6 139 8,122 ####### .9 53,711 102.5 22,541,782

336 .7 186 12,988 ####### 1.0 17,597 33.6 22,562,084

384 .8 233 17,422 ####### 1.0 7,368 14.1 22,569,402

432 .9 280 23,906 ####### 1.0 3,553 6.8 22,571,902

480 1.0 327 28,605 ####### 1.0 524 1.0 22,573,396

528 1.1 374 35,282 ####### 1.0 1 .0 22,574,164

576 1.2 421 40,835 ####### 1.0 1 .0 22,574,675

624 1.3 468 46,682 ####### 1.0 1 .0 22,575,055

672 1.4 515 52,252 ####### 1.0 1 .0 22,575,256

720 1.5 562 58,181 ####### 1.0 1 .0 22,575,422

768 1.6 609 64,380 ####### 1.0 1 .0 22,575,545

816 1.7 656 69,832 ####### 1.0 1 .0 22,575,620

864 1.8 703 75,168 ####### 1.0 1 .0 22,575,668

912 1.9 750 78,993 ####### 1.0 1 .0 22,575,695

960 2.0 797 82,209 ####### 1.0 1 .0 22,575,719

-------------------------------------------------------------

SGA Target Advisory

The SGA target advisory report is somewhat of a summation of all the advisory reports previously presented in the AWR report. It helps you determine the impact of changing the settings of the SGA target size in terms of overall database performance. The report uses a value called DB Time as a measure of the increase or decrease in performance relative to the memory change made. Also the report will summarize an estimate of physical reads associated with the listed setting for the SGA. Here is an example of the SGA target advisory report:

SGA Target SGA Size Est DB Est Physical

Size (M) Factor Time (s) Reads

---------- ---------- ------------ ----------------

528 0.5 25,595 769,539

792 0.8 20,053 443,095

1,056 1.0 18,443 165,649

1,320 1.3 18,354 150,476

1,584 1.5 18,345 148,819

1,848 1.8 18,345 148,819

2,112 2.0 18,345 148,819

In this example, our SGA Target size is currently set at 1056MB. We can see from this report that if we increased the SGA target size to 2112MB, we would see almost no performance improvement (about a 98 second improvement overall). In this case, we may determine that adding so much memory to the database is not cost effective, and that the memory can be better used elsewhere.

Memory Advisory

Memory advisory reports for the streams pool and the java pool also appear in the report (assuming you are using the streams pool). These reports take on the same general format as the other memory advisor reports.

Buffer Wait Statistics

The buffer wait statistics report helps you drill down on specific buffer wait events, and where the waits are occurring. In the following report we find that the 13 buffer busy waits we saw in the buffer pool statistics report earlier are attributed to data block waits. We might then want to pursue tuning remedies to these waits if the waits are significant enough. Here is an example of the buffer wait statistics report:

Class Waits Total Wait Time (s) Avg Time (ms)

------------------ ----------- ------------------- --------------

data block 13 0 1

# Enqueue Activity

The Enqueue activity report provides information on enqueues (higher level Oracle locking) that occur. As with other reports, if you see high levels of wait times in these reports, you might dig further into the nature of the enqueue and determine the cause of the delays. Here is an example of this report section:

Enqueue Type (Request Reason)

------------------------------------------------------------------------------

Requests Succ Gets Failed Gets Waits Wt Time (s) Av Wt Time(ms)

------------ ------------ ----------- ----------- ------------ --------------

PS-PX Process Reservation

386 358 28 116 0 .43

US-Undo Segment

276 276 0 228 0 .18

TT-Tablespace

90 90 0 42 0 .71

WF-AWR Flush

12 12 0 7 0 1.43

MW-MWIN Schedule

2 2 0 2 0 5.00

TA-Instance Undo

12 12 0 12 0 .00

UL-User-defined

7 7 0 7 0 .00

CF-Controlfile Transaction

5,737 5,737 0 5 0 .00

Undo Segment Summary

The undo segment summary report provides basic information on the performance of undo tablespaces.

Latch Activity

The latch activity report provides information on Oracle's low level locking mechanism called a latch. From this report you can determine if Oracle is suffering from latching problems, and if so, which latches are causing the greates amount of contention on the system. Here is a partial example of the latch activity report (it is quite long):

Pct Avg Wait Pct

Get Get Slps Time NoWait NoWait

Latch Name Requests Miss /Miss (s) Requests Miss

------------------------ -------------- ------ ------ ------ ------------ ------

ASM allocation 122 0.0 N/A 0 0 N/A

ASM map headers 60 0.0 N/A 0 0 N/A

ASM map load waiting lis 11 0.0 N/A 0 0 N/A

ASM map operation freeli 30 0.0 N/A 0 0 N/A

ASM map operation hash t 45,056 0.0 N/A 0 0 N/A

ASM network background l 1,653 0.0 N/A 0 0 N/A

AWR Alerted Metric Eleme 14,330 0.0 N/A 0 0 N/A

Consistent RBA 107 0.0 N/A 0 0 N/A

FAL request queue 75 0.0 N/A 0 0 N/A

FAL subheap alocation 75 0.0 N/A 0 0 N/A

FIB s.o chain latch 14 0.0 N/A 0 0 N/A

FOB s.o list latch 93 0.0 N/A 0 0 N/A

JS broadcast add buf lat 826 0.0 N/A 0 0 N/A

JS broadcast drop buf la 826 0.0 N/A 0 0 N/A

In this example our database does not seem to be experiencing any major latch problems, as the wait times on the latches are 0, and our get miss pct (Pct Get Miss) is 0 also.

There is also a latch sleep breakdown report which provides some additional detail if a latch is being constantly moved into the sleep cycle, which can cause additional performance issues.

The latch miss sources report provides a list of latches that encountered sleep conditions. This report can be of further assistance when trying to analyze which latches are causing problems with your database.

Segments by Logical Reads and Segments by Physical Reads

The segments by logical reads and segments by physical reads reports provide information on the database segments (tables, indexes) that are receiving the largest number of logical or physical reads. These reports can help you find objects that are "hot" objects in the database. You may want to review the objects and determine why they are hot, and if there are any tuning opportunities available on those objects (e.g. partitioning), or on SQL accessing those objects.

For example, if an object is showing up on the physical reads report, it may be that an index is needed on that object. Here is an example of the segments by logical reads report:

Segments by Logical Reads DB/Inst: A109/a1092 Snaps: 2009-2010

-> Total Logical Reads: 72,642

-> Captured Segments account for 96.1% of Total

Tablespace Subobject Obj. Logical

Owner Name Object Name Name Type Reads %Total

---------- ---------- -------------------- ---------- ----- ------------ -------

SYS SYSAUX SYS\_IOT\_TOP\_8813 INDEX 52,192 71.85

SYS SYSTEM SMON\_SCN\_TIME TABLE 4,704 6.48

SYS SYSTEM I\_JOB\_NEXT INDEX 2,432 3.35

SYS SYSTEM OBJ$ TABLE 1,344 1.85

SYS SYSTEM TAB$ TABLE 1,008 1.39

-------------------------------------------------------------

Additional Reports

Several segment related reports appear providing information on:

Segments with ITL waits

Segments with Row lock waits

Segments with buffer busy waits

Segments with global cache buffer waits

Segments with CR Blocks received

Segments with current blocks received

These reports help provide more detailed information on specific segments that might be experiencing performance problems.

The dictionary cache and library cache statistics reports provide performance information on the various areas in the data dictionary cache and the library cache.

The process memory summary, SGA memory summary, and the SGA breakdown difference reports provide summary information on how memory allocated to the database is allocated amongst the various components. Other memory summary reports may occur if you have certain optional components installed (such as streams).

The database parameter summary report provides a summary of the setting of all the database parameters during the snapshot report. If the database parameters changed during the period of the report, then the old and new parameters will appear on the report.