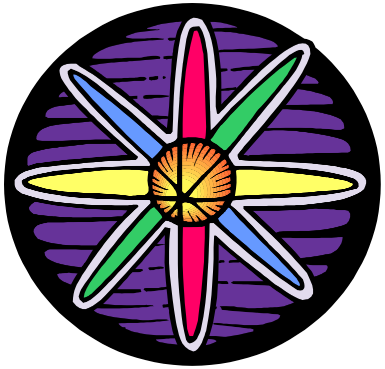
Using AWR For Memory Analysis

Mike Ault, Oracle Guru

May, 2011

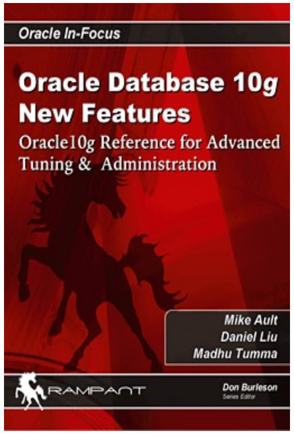
Michael R. Ault, Oracle Guru

- Nuclear Navy 6 years
- Nuclear Chemist/Programmer 10 years
- Kennedy Western University Graduate
- Bachelors Degree Computer Science
- Certified in all Oracle Versions Since 6
- Oracle DBA, author, since 1990



ORACLE | CERTIFIED PROFESSIONAL

Books by Michael R. Ault


Oracle In-Focus

Oracle Database 10g New Features

Oracle10g Reference for Advanced Tuning & Administration

Mike Ault
Daniel Liu
Madhu Tumma

RAMPANT

Don Burleson
Series Editor

Oracle In-Focus

Mike Ault's Oracle Internals Monitoring & Tuning Scripts

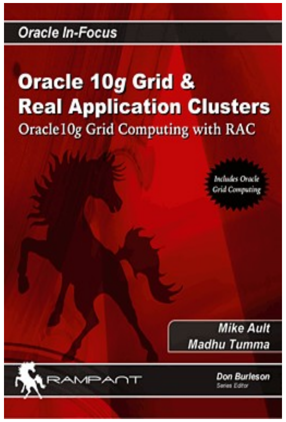
Advanced internals & OCP certification insights for the master DBA

Includes Oracle10g

Mike Ault

RAMPANT

Don Burleson
Series Editor


Oracle In-Focus

Oracle 10g Grid & Real Application Clusters

Oracle10g Grid Computing with RAC

Includes Oracle Grid Computing

Mike Ault
Madhu Tumma

RAMPANT

Don Burleson
Series Editor

Oracle In-Focus

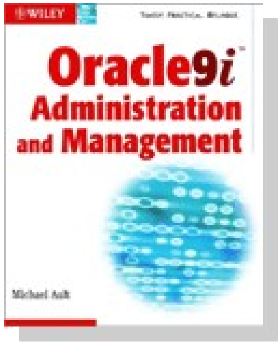
Oracle disk I/O Tuning

Disk I/O Performance & Optimization for Oracle 10g Databases

Mike Ault

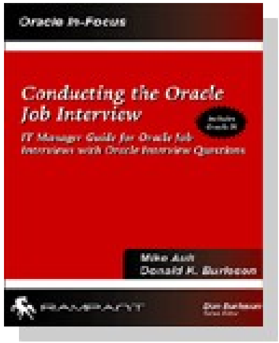
RAMPANT

Don Burleson
Series Editor


WILEY

Oracle9i Administration and Management

Michael Ault


Oracle In-Focus

Conducting the Oracle Job Interview

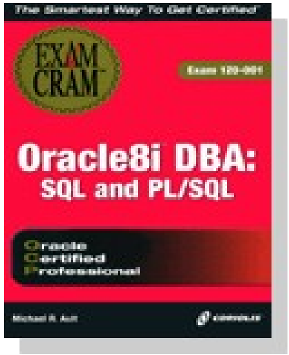
IT Manager Guide for Oracle Job Interviews with Oracle Interview Questions

Includes Oracle 9i

Mike Ault
Donald K. Burleson

RAMPANT

Don Burleson
Series Editor


The Smartest Way To Get Certified

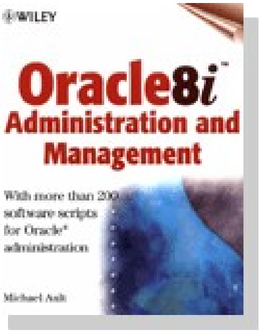
EXAM CRAM

Exam 1Z0-001

Oracle8i DBA: SQL and PL/SQL

Oracle
Certified
Professional

Michael R. Ault


WILEY

Oracle8i Administration and Management

With more than 200 software scripts for Oracle administration

Michael Ault

StatspackAnalyzer.com

Free Statspack/AWR Analysis

Sponsored by Texas Memory Systems

-Looks for IO bottlenecks and other configuration issues.

-Straightforward tuning advice

Preparation for Analysis

- Know your systems normal performance fingerprint
- Be familiar with Concepts and Tuning Guides
- Have “normal” AWR/Statspacks for comparison

Oracle and memory

- DB cache – other than direct read/write, all data, index and undo go through here
- Shared Pool – SQL area, PL/SQL library, dictionary caches and lots more
- Streams pool – Only if streams are used
- Java Pool – usually small
- PGA – Each process gets a PGA, stores cursor and other process related information
- Log buffers – Circular buffers for redo information

DB Cache

- Default – should be largest area
- Recycle – for frequently scanned large objects
- Keep – For frequently accessed small objects
- 2-32K areas – Was originally for TTS, now used to tune items (usually in RAC)

Shared Pool

- Shared SQL, PL/SQL, dictionary cache plus
- 34 other areas in 9i
- 551 in 10gR2 (non-RAC) 670 (with RAC)
- 878 in 11gR2 (non-RAC)
  - Report only shows those that change
  - There have been bugs with space leaks in shared pool sub-pools

Top-Down Approach

- Report starts with settings overview
- Next provides Top-5 waits
- Use the Waits to guide further investigation

AWR Report Header

```
WORKLOAD REPOSITORY report for
DB Name         DB Id    Instance     Inst Num Startup Time    Release     RAC
------------ ----------- ------------ -------- --------------- ----------- ---
AULTDB        4030696936 aultdb1             1 04-Aug-08 10:16 11.1.0.6.0  YES
Host Name        Platform                         CPUs Cores Sockets Memory(GB)
---------------- -------------------------------- ---- ----- ------- ----------
aultlinux3       Linux IA (32-bit)                   2     1       1       2.97
              Snap Id      Snap Time      Sessions Curs/Sess
            --------- ------------------- -------- ---------
Begin Snap:        91 04-Aug-08 12:00:15        41       1.2
  End Snap:        92 04-Aug-08 13:00:28        47       1.1
   Elapsed:               60.22 (mins)
   DB Time:              139.52 (mins)
Cache Sizes                       Begin        End
~~~~~~~~~~~                  ---------- ----------
               Buffer Cache:     1,312M     1,312M  Std Block Size:         8K
           Shared Pool Size:       224M       224M      Log Buffer:    10,604K
```

Signs of Memory Issues

- High sequential reads
- Excessive library latches
- Large number of sorts/hashes/GTT/bitmap ops to disk
- Large amount of IO to the temporary tablespace
- Buffer busy waits with free buffer waits
- Indications in Cache, shared, streams or java pool advisors
- Excessive reparsing and reloads of SQL and PL/SQL
- High percentage of use in the shared pool
- High CPU cycles

Load Profile Section

```
Load Profile              Per Second    Per Transaction   Per Exec   Per Call
~~~~~~~~~~~~         ---------------    --------------- ---------- ----------
      DB Time(s):                2.3                7.1       0.63       1.05
       DB CPU(s):                0.3                0.9       0.07       0.13
       Redo size:              800.5            2,461.8
   Logical reads:            6,307.6           19,396.7
   Block changes:                3.6               10.9
  Physical reads:            2,704.9            8,317.8
 Physical writes:               86.9              267.3
      User calls:                2.2                6.8
          Parses:                2.0                6.1
     Hard parses:                0.0                0.1
W/A MB processed:          932,965.4        2,868,990.9
          Logons:                0.1                0.2
        Executes:                3.7               11.3
       Rollbacks:                0.1                0.3
    Transactions:                0.3
```

What Are Your Efficiencies

- Should be close to 100%
- Parse issues usually are a result of:
  - Bad bind variable usage
  - Insufficient memory
  - Will also be co-indicated by low percentage of memory for multiple SQL execution

Load Profile Section

```
Instance Efficiency Percentages (Target 100%)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
            Buffer Nowait %:  100.00       Redo NoWait %:    99.97
            Buffer  Hit   %:   96.09    In-memory Sort %:  100.00
            Library Hit   %:   98.17        Soft Parse %:    97.88
         Execute to Parse %:   45.80         Latch Hit %:    99.95
Parse CPU to Parse Elapsd %:    0.00     % Non-Parse CPU:    99.77

 Shared Pool Statistics        Begin    End
                              ------  ------
             Memory Usage %:   81.53   85.39
    % SQL with executions>1:   79.29   79.48
  % Memory for SQL w/exec>1:   76.73   78.19
```

Top 5 Waits Section

- Critical to look closely at this section
- Use highest wait times to guide investigation
  - DB FILE type waits – physical IO
  - BUFFER type waits – Logical IO
  - LOG type waits – Redo related
  - PX – Parallel Query
  - GC – Global Cache (RAC related)
  - Undo – Undo or rollback segment related

Top 5 Waits Section With possible cache starvation

```
Top 5 Timed Foreground Events
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                                                            Avg
                                                           wait   % DB
Event                                 Waits     Time(s)    (ms)   time Wait Class
------------------------------ ------------ ----------- ------ ------ ----------
db file sequential read             465,020       3,969      9   47.4 User I/O
DB CPU                                              995          11.9
db file parallel read                 2,251         322    143    3.8 User I/O
db file scattered read               15,268         153     10    1.8 User I/O
gc current block 2-way              108,739         116      1    1.4 Cluster
```

Top 5 Waits Section With Shared Pool Issues

```
Top 5 Timed Events                                         Avg %Total
~~~~~~~~~~~~~~~~~~                                        wait   Call
Event                                 Waits    Time (s)   (ms)   Time Wait Class
------------------------------ ------------ ----------- ------ ------ ----------
CPU time                                        435,461            41.1
PX Deq Credit: send blkd        124,829,330     138,223      1   13.0      Other
library cache pin                    20,347      57,692   2835    5.4 Concurrenc
library cache lock                   19,226      56,078   2917    5.3 Concurrenc
db file sequential read          16,798,329      42,215      3    4.0   User I/O
          -------------------------------------------------------------
Top 5 Timed Events                                         Avg %Total
~~~~~~~~~~~~~~~~~~                                        wait   Call
Event                                 Waits    Time (s)   (ms)   Time Wait Class
------------------------------ ------------ ----------- ------ ------ ----------
CPU time                                         24,956            29.3
latch: library cache              1,757,331       9,886      6   11.6 Concurrenc
db file sequential read             759,605       6,146      8    7.2   User I/O
cursor: pin S                     2,103,389       4,988      2    5.9      Other
log file sync                       250,039       2,387     10    2.8     Commit
          -------------------------------------------------------------
```

Buffer Type Waits

- latch: cache buffers chains – Hot blocks, check for hot objects, high IO rates
- free buffer waits – Insufficient buffers, processes holding buffers too long, IO subsystem over loaded
- buffer busy waits – See what is causing them further along in report
- gc buffer busy – Overloaded interconnect, find problem objects and tune
- log buffer space – High load, too small a log buffer, increase log buffer size
- latch: cache buffers lru chain – Freelist issues, hot blocks, new buffers, buffers being writen
- 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
- Free buffer waits – insufficient buffers

Fixing Cache Waits

- Reduce logical IO rates (buffer caches latch)
- Increase the cache size (lru chain latch)
- Increase the cache size (free buffer waits)
- Increase _db_block_lru_latches
- Increase _db_block_hash_buckets
- Reduce hot blocks

Shared Pool Waits

- library cache pin – Loading or compiling same SQL
- library cache lock – Loading or compiling same SQL
- *latch: library cache – Usually a result of excessive parsing
- latch: shared pool latch –Parsing issues
- *latch: library cache lock – Usually a result of excessive parsing
- *latch: library cache – Usually a result of excessive parsing
- row cache lock – shared pool too small
- Library cache load lock – Wait for a reload by another session. Excessive hard/soft parsing.

* Gone in 11g to mutex

Shared Pool Mutexes

- Cursor:mutex X – resource is busy, requestor needs exclusive access
- Cursor:mutex S – resource is held in X mode by another session
- Cursor:pin X – resource is held n S or X by another session
- Cursor:pin S – re-execute of same cursor
- Cursor:pin S wait on X – resource is held in X mode by another session
- Library cache: mutex X – Bind variable issues
- Library cache: mutex S – Bind variable issues

- Less costly than latches

What to Do?

- Share cursors (avoid hard parsing)
  - BIND VARIABLES!!!!!
  - Cursor_sharing
- Avoid soft parsing
  - Cursor_space_for_time
  - Session_cached_cursors
- Avoid invalidations and reloads
  - Make sure shared pool is large enough

What Next?

- Determine wait events of concern
- Drill down to specific sections of report for deeper analysis
- Use custom scripts, ADDM and Ash to investigate issues

Classes

```
Wait Class                                DB/Inst: Snaps: 84084-84108
-> 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
-------------------- ---------------- ------ ---------------- ------- ---------
Other                     153,619,985   16.5          192,921       1     102.3
Concurrency                 2,536,362   26.9          128,816      51       1.7
User I/O                   30,594,385     .0          124,207       4      20.4
System I/O                  5,104,873     .0           17,633       3       3.4
Application                    65,645    5.0            6,508      99       0.0
Commit                        267,317     .0            4,234      16       0.2
Configuration                 553,825   69.5              858       2       0.4
Network                    13,513,847     .0              274       0       9.0
Administrative                     30   70.0                0      10       0.0
          -------------------------------------------------------------
```

Operating System Statistics

```
Operating System Statistics         DB/Inst: Snaps: 84084-84108

Statistic                                       Total
-------------------------------- --------------------
BUSY_TIME                                  45,601,415
IDLE_TIME                                   6,316,939
IOWAIT_TIME                                   567,343
NICE_TIME                                     810,986
SYS_TIME                                    3,169,946
USER_TIME                                  41,265,848
LOAD                                               50
RSRC_MGR_CPU_WAIT_TIME                              0
PHYSICAL_MEMORY_BYTES                 270,208,987,136
NUM_CPUS                                           24
NUM_CPU_SOCKETS                                     4

-------------------------------------------------------------
```

SQL Areas

SQL ordered by CPU Time – Sorting, bad paths
SQL ordered by Gets – Excessive logical IO
SQL ordered by Reads – Cache starvation
SQL ordered by Parse Calls – Cursor sharing, cursor caching
SQL ordered by Version Count – Versioning is usually due to a bug, check with support

- **Tune SQL that appears in more than one of these areas**
- **Tune SQL at the top of these sections**

System Statistics

- Many-many statistics
- Many are not useful to the DBA
- Many are useless for memory area tuning
- Many give ideas of how memory is used, but not how to tune it
- Usually these will confirm what you have already found

System Statistics

```
Instance Activity Stats                  DB/Inst: Snaps: 84084-84108

Statistic                                     Total     per Second     per Trans
-------------------------------- ------------------ -------------- -------------
dirty buffers inspected                     686,267           31.8           0.5
execute count                            78,907,090        3,656.2          52.6
free buffer inspected                   161,591,258        7,487.4         107.6
free buffer requested                   176,367,274        8,172.1         117.5
hot buffers moved to head of LRU         15,346,759          711.1          10.2
immediate (CR) block cleanout ap          2,267,512          105.1           1.5
immediate (CURRENT) block cleano          5,139,016          238.1           3.4
no buffer to keep pinned count              136,849            6.3           0.1
no work - consistent read gets        4,459,140,613      206,616.1       2,969.8
opened cursors cumulative                26,795,933        1,241.6          17.9
parse count (failures)                          160            0.0           0.0
parse count (hard)                          398,147           18.5           0.3
parse count (total)                      20,200,501          936.0          13.5
parse time cpu                            3,883,178          179.9           2.6
parse time elapsed                        7,474,786          346.4           5.0
physical read total IO requests          37,303,810        1,728.5          24.8
physical reads direct temporary          58,378,313        2,705.0          38.9
physical write total IO requests         13,738,098          636.6           9.2
physical writes direct temporary         58,440,795        2,707.9          38.9
```

System Statistics

```
pinned buffers inspected                    120,843            5.6           0.1
recursive calls                         749,184,714       34,713.8         499.0
recursive cpu usage                      39,323,240        1,822.1          26.2
redo log space requests                         190            0.0           0.0
redo log space wait time                        333            0.0           0.0
redo synch time                             433,625           20.1           0.3
redo synch writes                           236,148           10.9           0.2
redo write time                             567,670           26.3           0.4
redo writer latching time                    56,827            2.6           0.0
redo writes                               1,127,300           52.2           0.8
rollback changes - undo records           1,395,329           64.7           0.9
rollbacks only - consistent read            346,504           16.1           0.2
session cursor cache hits                21,520,355          997.2          14.3
session logical reads                10,474,545,504      485,342.3       6,976.0
sorts (disk)                                  3,529            0.2           0.0
sorts (memory)                            9,012,270          417.6           6.0
sorts (rows)                        110,063,794,220    5,099,850.2      73,302.3
sql area evicted                            327,084           15.2           0.2
sql area purged                              29,720            1.4           0.0
table scans (long tables)                 1,149,945           53.3           0.8
table scans (short tables)                7,528,140          348.8           5.0
transaction rollbacks                       252,407           11.7           0.2
```

System Statistics

```
user I/O wait time                       12,422,069          575.6            8.3
user calls                                8,038,839          372.5            5.4
user commits                              1,439,821           66.7            1.0
user rollbacks                               61,684            2.9            0.0
workarea executions - multipass                   0            0.0            0.0
workarea executions - onepass                 5,293            0.3            0.0
workarea executions - optimal             7,113,060          329.6            4.7
```

Instance Activity Statistics

```
Instance Activity Stats - Absolute
-> Statistics with absolute values (should not be diffed)

Statistic                            Begin Value       End Value
-------------------------------- --------------- ---------------
session cursor cache count            28,024,069      28,789,659
opened cursors current                     2,921           6,982
workarea memory allocated                289,532       2,531,741
logons current                               144             287
          --------------------------------------------------
```

Tablespace/File IO Reports

- Helps confirm IO issues
- Also helps with temp area issue determination

Tablespace IO

```
Tablespace IO Stats
-> ordered by IOs (Reads + Writes) desc

Tablespace
------------------------------
                 Av      Av     Av                       Av     Buffer Av Buf
         Reads Reads/s Rd(ms) Blks/Rd       Writes Writes/s      Waits Wt(ms)
-------------- ------- ------ ------- ------------ -------- ---------- ------
TEMP
    11,484,000     532   16.3     4.1    3,478,365      161     12,266    2.0
REPMAN_TEMP
     1,703,767      79   27.2     8.2    1,457,241       68          0    0.0
UNDOTBS3
        30,012       1    8.0     1.0    1,512,571       70    142,889    1.1
RSNET_DTSA
     1,496,441      69    1.2     2.0        2,454        0    130,753    1.3
LOREAL_D_CVS_DAILY_ITSA
       846,665      39    0.9     1.0          338        0          0    0.0
```

Buffer Pool Statistics

```
Buffer Pool Statistics                    DB/Inst: CC1/cc1  Snaps: 84084-84108
-> Standard block size Pools  D: default,  K: keep,  R: recycle
-> Default Pools for other block sizes: 2k, 4k, 8k, 16k, 32k

                                                            Free Writ     Buffer
     Number of Pool         Buffer     Physical    Physical Buff Comp       Busy
P      Buffers Hit%           Gets        Reads      Writes Wait Wait      Waits
--- ---------- ---- -------------- ------------ ----------- ---- ---- ----------
D    3,361,107   96  3,643,978,600  163,055,679  14,338,623    0    0    711,281
K      321,600  100  2,527,600,634        7,379      28,755    0    0        123
          -------------------------------------------------------------
```

- Note that there are Buffer Busy Waits, but no Free Buffer Waits
- These are due to hot block contention
- Increasing memory probably won’t help with this
- However…also look at db file sequential read waits and the cache advisory section

Buffer Pool Advisory Section

```
Buffer Pool Advisory
-> Only rows with estimated physical reads >0 are displayed
-> ordered by Block Size, Buffers For Estimate

                                        Est
                                       Phys
    Size for   Size      Buffers for   Read          Estimated
P    Est (M) Factor         Estimate Factor     Physical Reads
--- -------- ------ ---------------- ------ ------------------
D      5,344     .1          335,670    1.9     15,767,325,073
D     10,688     .2          671,340    1.4     11,371,357,960
…
D    106,880    2.0        6,713,400    1.0      7,964,367,701
K        512     .1           32,160  102.8      3,507,100,178
K      1,024     .2           64,320    7.8        264,615,629
K      1,536     .3           96,480    1.4         49,384,590
…
K     10,240    2.0          643,200    1.0         32,639,643
          ----------------------------------------------
```

Buffer Pool Advisory Section

- As you can see, this repor shows even doubling the default or keep would be no benefit
- Let’s look at one that would benefit from increased buffer pool

Buffer Pool Advisor Section

```
Buffer Pool Statistics           DB/Inst:   Snaps: 26064-26097
-> Standard block size Pools  D: default,  K: keep,  R: recycle
-> Default Pools for other block sizes: 2k, 4k, 8k, 16k, 32k

                                                             Free Writ     Buffer
     Number of Pool         Buffer     Physical    Physical Buff Comp       Busy
P      Buffers Hit%           Gets        Reads      Writes Wait Wait      Waits
--- ---------- ---- --------------- ------------ ----------- ---- ---- ----------
D      818,201   99     130,795,544    1,578,580     276,075    0    0      3,418
          -------------------------------------------------------------
```

- Before we go there…look here
- Notice no free buffer waits

Buffer Pool Advisor Section

```
Buffer Pool Advisory                     Snap: 26097
-> Only rows with estimated physical reads >0 are displayed
-> ordered by Block Size, Buffers For Estimate

                                        Est
                                       Phys
    Size for   Size      Buffers for   Read          Estimated
P    Est (M) Factor         Estimate Factor     Physical Reads
--- -------- ------ ---------------- ------ ------------------
D        656     .1           81,139    2.0        125,592,784
D      1,312     .2          162,278    1.8        113,080,052
…
D     10,496    1.6        1,298,224    0.7         42,942,366
D     11,152    1.7        1,379,363    0.7         41,649,501
D     11,808    1.8        1,460,502    0.6         40,403,058
-------------------------------------------------------------
```

- Notice that at 1.8 times the current size physical reads down by 40%
- Db file sequential reads was 13% of waits

PGA Analysis

- Several of the next sections deal with PGA
- PGA_AGGREGATE_TARGET sets the PGA area
- 5% of PGA_AGGREGATE_TARGET can be allocated to each session up to a maximum of "_PGA_MAX_SIZE" which is usually 200 or 500 megabytes
- Manually setting SORT_AREA_SIZE or HASH_AREA_SIZE overrides at the session level
- Some processes such as RMAN and shared servers don't use PGA_AGGREGATE_TARGET but use the old manual settings, indicated by 4-8 or 8-16m sorts even with adequate PGA_AGGREGATE_TARGET settings

PGA Analysis

```
PGA Aggr Summary                         DB/Inst: Snaps: 84084-84108
-> PGA cache hit % - percentage of W/A (WorkArea) data processed only in-
memory

PGA Cache Hit %   W/A MB Processed  Extra W/A MB Read/Written
--------------- ------------------ --------------------------
           82.1          4,495,435                    979,073
          -------------------------------------------------------------
```

PGA Analysis

```
PGA Aggr Target Stats                     DB/Inst: Snaps: 84084-84108
-> B: Begin snap   E: End snap (rows dentified with B or E contain data
   which is absolute i.e. not diffed over the interval)
-> Auto PGA Target - actual workarea memory target
-> W/A PGA Used    - amount of memory used for all Workareas (manual + auto)
-> %PGA W/A Mem    - percentage of PGA memory allocated to workareas
-> %Auto W/A Mem   - percentage of workarea memory controlled by Auto Mem Mgmt
-> %Man W/A Mem    - percentage of workarea memory under manual control

                                                %PGA  %Auto   %Man
    PGA Aggr   Auto PGA   PGA Mem    W/A PGA     W/A    W/A    W/A Global Mem
   Target(M)  Target(M)  Alloc(M)    Used(M)     Mem    Mem    Mem   Bound(K)
- ---------- ---------- ---------- ---------- ------ ------ ------ ----------
B      5,120      4,320    1,680.5      193.5   11.5   99.7     .3    524,280
E      5,120      4,202    4,400.5    2,219.2   50.4   99.9     .1    524,280
          --------------------------------------------------------------
```

PGA Analysis

```
PGA Aggr Target Histogram                  Snaps: 84084-84108
Low     High
Optimal Optimal    Total Execs  Optimal Execs 1-Pass Execs M-Pass Execs
------- ------- -------------- -------------- ------------ ------------
     2K      4K      6,308,435      6,308,435            0            0
    64K    128K         32,500         32,500            0            0
   128K    256K         36,535         36,535            0            0
   256K    512K         47,477         47,477            0            0
   512K   1024K        353,344        353,344            0            0
     1M      2M        201,558        201,558            0            0
     2M      4M         22,468         22,468            0            0
     4M      8M         21,796         21,725           71            0
     8M     16M         28,892         28,714          178            0
    16M     32M         30,478         30,346          132            0
    32M     64M         19,898         18,690        1,208            0
    64M    128M          9,080          7,284        1,796            0
   128M    256M          1,682            732          950            0
   256M    512M            734            179          555            0
   512M   1024M            329             58          271            0
     1G      2G            131             14          117            0
     2G      4G             17              4           13            0
          -------------------------------------------------------------
```

PGA Analysis

- SS*100/2.5*NOS
  - SS- sort size
  - NOS – number of sorts
- 128M*100/2.5=5120
- Hash gets full 5%
- Sort gets ½ of hash
- ½ of 5 is 2.5
- 10|10 rule:
  - 10% of sessions are active
  - 10% of active sessions doing sorts
- 10% of 208 sessions is 20.8; 10% of 20.8 is 2
- 2*5120 is 10420 gb
- Let’s see what the advisor says…

PGA Analysis

```
PGA Memory Advisory                         Snap: 84108
-> When using Auto Memory Mgmt, minimally choose a pga_aggregate_target
value where Estd PGA Overalloc Count is 0
                                       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
---------- ------- ---------------- ---------------- -------- ----------
       640     0.1    167,211,258.2    133,135,735.1     56.0     18,399
     1,280     0.3    167,211,258.2     65,323,839.1     72.0      1,933
     2,560     0.5    167,211,258.2     34,248,434.3     83.0      1,842
     3,840     0.8    167,211,258.2     28,768,442.5     85.0      1,803
     5,120     1.0    167,211,258.2     19,597,963.1     90.0        860
     6,144     1.2    167,211,258.2     14,425,059.9     92.0        845
     7,168     1.4    167,211,258.2     13,624,988.5     92.0        839
     8,192     1.6    167,211,258.2     12,882,080.7     93.0          0
     9,216     1.8    167,211,258.2     12,146,846.2     93.0          0
    10,240     2.0    167,211,258.2     11,689,119.6     93.0          0
    15,360     3.0    167,211,258.2     10,710,132.3     94.0          0
    20,480     4.0    167,211,258.2     10,563,396.2     94.0          0
    30,720     6.0    167,211,258.2     10,539,291.8     94.0          0
    40,960     8.0    167,211,258.2     10,539,291.8     94.0          0
          -----------------------------------------------------------
```

PGA Analysis

- Advisor says 8 gb
- Calculation says 10 gb
- Take whichever you feel will give best results
- Won't be used unless it needs to be

Shared Pool Advisor

- Use reloads and used percentages to guide you
- The advisor rarely has any meaningful information
- Even when reloads are huge and other factors show the pool should be increased, or, decreased it has told me things where fine

Shared Pool Advisor

```
Shared Pool Advisory                            DB/Inst: Snap: 84108
-> SP: Shared Pool     Est LC: Estimated Library Cache   Factr: Factor
-> Note there is often a 1:Many correlation between a single logical object
   in the Library Cache, and the physical number of memory objects associated
   with it.  Therefore comparing the number of Lib Cache objects (e.g. in
   v$librarycache), with the number of Lib Cache Memory Objects is invalid.

                                        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
---------- ----- -------- ------------ ------- ------ ------- ------ -----------
     2,160    .4      619       76,837 #######    1.0 #######    2.4  88,538,740
     2,736    .5    1,188       95,749 #######    1.0 #######    2.1  88,936,333
     3,312    .6    1,761      110,785 #######    1.0 #######    1.8  89,297,339
     3,888    .7    2,333      125,755 #######    1.0 #######    1.6  89,610,155
     …
     8,496   1.5    6,916      238,592 #######    1.0 #######     .5  91,076,008
     9,072   1.6    7,489      252,248 #######    1.0 #######     .4  91,187,541
     9,648   1.7    8,061      264,748 #######    1.0 #######     .3  91,291,114
    10,224   1.8    8,632      274,837 #######    1.0 #######     .3  91,388,340
    10,800   1.9    9,201      284,723 #######    1.0 #######     .2  91,480,577
    11,376   2.0    9,774      293,073 #######    1.0 #######     .2  91,569,191
          -------------------------------------------------------------
```

SGA Target Advisor

- I believe you should set the base parameters and then allow SGA_TARGET to set itself
- Other calculation schemes involve adding up the proposed sizes needed and then setting the value
- Same logic applies to MEMORY_TARGET only include PGA_AGGREGATE_TARGET in the base
- Then set SGA_MAX_SIZE or MEMORAY_MAX_SIZE 10-20% higher than their associated TARGET values.

SGA Target

```
SGA Target Advisory                        Snap: 26097

SGA Target   SGA Size       Est DB     Est Physical
  Size (M)     Factor     Time (s)            Reads
---------- ---------- ------------ ----------------
     2,048        0.3    8,957,297      125,595,242
     4,096        0.5      745,014       99,011,933
     6,144        0.8      683,241       84,201,232
     8,192        1.0      603,250       62,255,994
    10,240        1.3      573,691       48,902,083
    12,288        1.5      573,630       42,944,185
    14,336        1.8      573,630       37,745,809
    16,384        2.0      573,630       37,745,809
          ----------------------------------------------
```

- At 2x shows a 60% reduction in PR, same as doubling the cache for this instance

Streams Pool Advisor

- Only valid if you use streams
- If it shows you have spills to disk make pool larger

Streams Pool Advisor

```
Streams Pool Advisory                            DB/Inst: Snap: 84108

  Size for      Size   Est Spill   Est Spill Est Unspill Est Unspill
  Est (MB)    Factor       Count    Time (s)       Count    Time (s)
---------- --------- ----------- ----------- ----------- -----------
        16       0.5           0           0           0           0
        32       1.0           0           0           0           0
        48       1.5           0           0           0           0
        64       2.0           0           0           0           0
        80       2.5           0           0           0           0
…
       240       7.5           0           0           0           0
       256       8.0           0           0           0           0
       272       8.5           0           0           0           0
       288       9.0           0           0           0           0
       304       9.5           0           0           0           0
       320      10.0           0           0           0           0
          -------------------------------------------------------------
```

Java Pool Advisor

- Another area usually not used
- If you use it you will get errors if it is too small
- Under AMM will grow but usually won’t shrink

Java Pool Advisor

```
Java Pool Advisory                         DB/Inst: Snap: 37
                                      Est LC Est LC  Est LC Est LC
      Java    JP   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
---------- ----- -------- ---------- ------- ------ ------- ------ -----------
        64    .5       10        168      10    1.0  11,974    1.0         389
       128   1.0       12        201      10    1.0  11,974    1.0         465
       192   1.5       12        201      10    1.0  11,974    1.0         465
       256   2.0       12        201      10    1.0  11,974    1.0         465
          -------------------------------------------------------------
```

Buffer Wait Analysis

- Look here to see what is causing buffer waits
- Data blocks usually predominate
- High buffer waits for data blocks plus high db file sequential reads will usually indicate memory starvation even without free buffer waits

Buffer Wait Analysis

```
Buffer Wait Statistics                    DB/Inst:
Snaps: 84084-84108
-> ordered by wait time desc, waits desc

Class                    Waits Total Wait Time (s)  Avg Time (ms)
------------------ ----------- ------------------- --------------
data block             539,313               1,770              3
1st level bmb            5,873                  88             15
undo block              49,801                  71              1
undo header             35,848                  66              2
file header block       57,833                  39              1
segment header          12,980                  28              2
3rd level bmb            4,900                  22              5
2nd level bmb            5,206                  20              4
extent map                  73                   0              5
          ----------------------------------------------------
```

Enqueues and Latches

- Enqueues are usually for physical objects or transactions and rarely memory related
- Latches can be used to verify findings

Latches

```
Latch Activity                           DB/Inst: Snaps: 84084-84108
-> "Get Requests", "Pct Get Miss" and "Avg Slps/Miss" are statistics for
   willing-to-wait latch get requests
-> "NoWait Requests", "Pct NoWait Miss" are for no-wait latch get requests
-> "Pct Misses" for both should be very close to 0.0
                                            Pct    Avg   Wait                 Pct
                                    Get     Get   Slps   Time       NoWait NoWait
Latch Name                     Requests    Miss  /Miss    (s)     Requests   Miss
------------------------ -------------- ------ ------ ------ ------------ ------
SGA IO buffer pool latch        154,979    0.0    0.7      0      261,675    0.3
SQL memory manager latch         21,353   12.7    1.2     32        7,066    0.1
active service list           2,524,503    1.7    0.2     10        7,615    0.1
cache buffers lru chain      66,153,522    1.0    0.6   1172  469,078,119    1.8
cache table scan latch        3,749,595    0.3    0.3      9    3,986,131    0.3
library cache               272,976,941    0.1    0.4   2242    1,582,568  216.3
library cache lock          179,441,868    0.1    0.1     34       58,383    0.1
object queue header oper    883,815,615    0.1    0.1    365       45,167    0.2
process queue reference   8,224,189,548    0.0    0.0     15  613,910,934    4.3
redo allocation               8,482,485    8.1    0.1    271  435,767,620    1.4
row cache objects           871,266,120    0.4    0.2   1765      580,035    1.6
simulator lru latch               7,890    1.1    1.1      1 ############   17.1
undo global data             68,788,574    0.0    0.1      8       42,419    0.1

          ------------------------------------------------------------
```

Latches

- When a latch can’t be obtained a process will “sleep”
- The number of CPU cycles is set by the “_SPIN_COUNT” setting
- Defaults to 2000, was set in the era of slow CPUs
- Experts agree setting as high as 8000-10000 is acceptable
- Reduces sleeps and improved performance

Latch Sleeps

```
Latch Sleep Breakdown                    DB/Inst: Snaps: 84084-84108
-> ordered by misses desc

Latch Name
----------------------------------------
  Get Requests      Misses      Sleeps  Spin Gets   Sleep1   Sleep2   Sleep3
-------------- ----------- ----------- ---------- -------- -------- --------
cache buffers chains
 9,561,948,628 ###########     402,582 ##########        0        0        0
session allocation
   570,516,196  14,728,230     461,868 ##########        0        0        0
messages
   105,586,860   3,521,238      16,783  3,506,276        0        0        0
row cache objects
   871,266,120   3,508,138     655,697  2,914,340        0        0        0
process queue reference
 8,224,189,548   2,645,262      19,802  2,629,967        0        0        0
active checkpoint queue latch
    50,214,009   1,082,455      13,133  1,070,658        0        0        0
redo writing
    52,201,162   1,026,204       6,893  1,020,132        0        0        0
object queue header operation
   883,815,615     936,882     137,869    812,882        0        0        0
```

Latch Miss Sources

```
Latch Miss Sources                       DB/Inst: Snaps: 84084-84108
-> only latches with sleeps are shown
-> ordered by name, sleeps desc

                                                     NoWait              Waiter
Latch Name               Where                       Misses     Sleeps   Sleeps
------------------------ -------------------------- ------- ---------- --------
In memory undo latch     ktiFlush: child                  0     12,727    3,775
...
cache buffers chains     kcbgtcr: kslbegin excl           0    821,219  689,229
...
cache buffers lru chain  kcbbxsv: move to being wri       0     19,728   87,892
...
library cache            kglpndl: child: after proc       0     28,574   10,336
...
library cache lock       kgllkdl: child: cleanup          0      7,106    6,749
...
object queue header oper kcbw_unlink_q                    0    139,971   27,436
...
parameter table allocati ksp_param_table_free             0     60,653   60,605
row cache objects        kqrpre: find obj                 0    381,022  379,948
...
session allocation       ksuprc                           0    191,233  175,384
...
shared pool              kghalo                           0    194,517  123,739
...
simulator hash latch     kcbsacc: lookup dba              0     16,413   26,348
```

Dictionary Cache

- Misses can be costly
- Only correction is larger shared pool
- Used to be individually controlled by DC parameters (v6)
- Now controlled automatically

Dictionary Cache

```
Dictionary Cache Stats                   DB/Inst: Snaps: 84084-84108
-> "Pct Misses"  should be very low (< 2% in most cases)
-> "Final Usage" is the number of cache entries being used
                                   Get    Pct    Scan   Pct      Mod      Final
Cache                         Requests   Miss    Reqs  Miss     Reqs      Usage
------------------------- ------------ ------ ------- ----- -------- ----------
dc_awr_control                     442    9.3       0   N/A       48          1
dc_constraints                  60,034   41.5       0   N/A   60,034         49
dc_files                           559   97.3       0   N/A        0          0
dc_global_oids                  20,580    3.0       0   N/A        0         25
dc_histogram_data            7,832,289    2.0       0   N/A   19,621      7,472
dc_histogram_defs            5,795,619   15.5       0   N/A  322,942      6,953
dc_object_grants                99,052   11.6       0   N/A        0        370
dc_object_ids              105,285,794    2.3       0   N/A   38,220     66,669
dc_objects                  53,343,945    4.8       0   N/A  161,568      8,250
dc_profiles                     60,308    0.1       0   N/A        0          5
dc_segments                 19,064,933   58.4       0   N/A  256,828     16,244
dc_sequences                   137,407    0.9       0   N/A  137,407         35
dc_table_scns                  180,866    4.2       0   N/A       88         14
dc_tablespace_quotas           101,823    2.5       0   N/A  101,679         43
dc_tablespaces              11,521,250    1.3       0   N/A       18     10,945
dc_usernames                   750,849    0.4       0   N/A        0        104
global database name               118    6.8       0   N/A        0          1
kqlsubheap_object                    1  100.0       0   N/A        0          0
outstanding_alerts             394,019   41.7       0   N/A        0     10,943
           ------------------------------------------------------------
```

Library Cache Activity

- Usually only see issues in SQL and PL/SQL areas
- Invalidations high means DDL activity
- Reloads high means SQL or PL/SQL issues

Library Cache Activity

```
Library Cache Activity                    DB/Inst: Snaps: 84084-84108
-> "Pct Misses"  should be very low

                         Get    Pct            Pin    Pct               Invali-
Namespace           Requests   Miss       Requests   Miss     Reloads  dations
--------------- ------------ ------ -------------- ------ ---------- --------
BODY                 106,197    0.3      2,372,306    0.1      1,288        0
CLUSTER               11,232    0.4         19,082    0.7         81        0
INDEX              1,728,015    1.9     28,528,015    0.2     17,973        0
PIPE                       9    0.0              9    0.0          0        0
SQL AREA           3,482,631   14.7    108,055,227    2.4  1,276,544 ########
TABLE/PROCEDURE      290,691    6.1     32,986,432    0.5     76,695        0
TRIGGER              621,821    0.1      2,312,915    0.1      2,423        0
          -------------------------------------------------------------
```

Process Statistics

```
Process Memory Summary                   DB/Inst: Snaps: 84084-84108
-> B: Begin snap   E: End snap
-> All rows below contain absolute values (i.e. not diffed over the interval)
-> Max Alloc is Maximum PGA Allocation size at snapshot time
-> Hist Max Alloc is the Historical Max Allocation for still-connected processes
-> ordered by Begin/End snapshot, Alloc (MB) desc

                                                            Hist
                                    Avg  Std Dev     Max     Max
               Alloc      Used    Alloc    Alloc   Alloc   Alloc    Num    Num
  Category      (MB)      (MB)     (MB)     (MB)    (MB)    (MB)   Proc  Alloc
- -------- --------- --------- -------- -------- ------- ------- ------ ------
B Freeable     981.6        .0      6.2     23.0     141     N/A    158    158
  Other        647.4       N/A      3.3      5.8      30     102    195    191
  PL/SQL        38.8       7.0       .2       .6       3       3    193    154
  SQL           15.7      10.9       .1       .2       2   1,289    174    128
E SQL        1,953.6   1,939.0      6.7     36.8     488   1,289    291    264
  Other      1,772.1       N/A      5.6     16.7     177     498    315    311
  Freeable     640.8        .0      4.1     14.2      92     N/A    156    156
  PL/SQL        77.0      13.8       .2       .5       3      13    313    313
          -------------------------------------------------------------
```

SGA Memory Summary

```
SGA Memory Summary                   DB/Inst: Snaps: 84084-84108

                                                End Size (Bytes)
SGA regions                Begin Size (Bytes)      (if different)
------------------------- ------------------- -------------------
Database Buffers               61,035,511,808      61,538,828,288
Fixed Size                          2,212,832
Redo Buffers                       14,561,280
Variable Size                   7,667,190,816       7,163,874,336
                          -------------------
sum                            68,719,476,736
          -------------------------------------------------------
```

SGA Breakdown

```
SGA breakdown difference                  DB/Inst: Snaps: 84084-84108
-> ordered by Pool, Name
-> N/A value for Begin MB or End MB indicates the size of that Pool/Name was
   insignificant, or zero in that snapshot

Pool   Name                                 Begin MB         End MB  % Diff
------ ------------------------------ -------------- -------------- -------
java   free memory                             160.0          160.0    0.00
large  free memory                           1,022.6        1,020.6   -0.19
shared CCursor                                 483.3          425.2  -12.04
shared Checkpoint queue                        175.8          175.8    0.00
shared KGH: NO ACCESS                        2,445.0        3,079.3   25.94
shared KQR L PO                                355.6            N/A -100.00
shared KQR M PO                                430.4            N/A -100.00
shared PCursor                                 221.9          190.5  -14.16
shared db_block_hash_buckets                   180.0          180.0    0.00
shared free memory                           2,191.3        2,890.1   31.89
shared kzctxgjsi ksuseclid memor             1,644.5        1,676.7    1.96
shared library cache                           334.2          308.2   -7.79
shared obj stat memo                           466.8          474.8    1.71
shared partitioning d                          131.8            N/A -100.00
shared sql area                                483.6          236.8  -51.04
stream free memory                              31.9           31.9    0.00
       buffer_cache                         58,208.0       58,688.0    0.82
       fixed_sga                                 2.1            2.1    0.00
       log_buffer                               13.9           13.9    0.00
          ------------------------------------------------------------
```

Resize Operations

- Using AMM resize operations are automatic
- If there is room to grow, shouldn’t have to borrow (steal) from other areas
- If SGA_TARGET=SGA_MAX_SIZE or MEMORY_TARGET=MEMORY_MAX_SIZE then no room to grow

Resize Operations

```
Memory Dynamic Components                  DB/Inst: Snaps: 22930-22958
-> Min/Max sizes since instance startup
-> Oper Types/Modes: INItializing,GROw,SHRink,STAtic/IMMediate,DEFerred
-> ordered by Component

                  Begin Snap     Current         Min         Max     Oper Last Op
Component          Size (Mb)   Size (Mb)   Size (Mb)   Size (Mb)    Count Typ/Mod
--------------- ----------- ----------- ----------- ----------- ------ -------
ASM Buffer Cach         .00         .00         .00         .00      0 STA/
DEFAULT 16K buf         .00         .00         .00         .00      0 STA/
DEFAULT 2K buff         .00         .00         .00         .00      0 STA/
DEFAULT 32K buf         .00         .00         .00         .00      0 STA/
DEFAULT 4K buff         .00         .00         .00         .00      0 STA/
DEFAULT 8K buff         .00         .00         .00         .00      0 STA/
DEFAULT buffer    20,480.00   20,480.00   19,712.00   20,736.00      0 GRO/DEF
KEEP buffer cac         .00         .00         .00         .00      0 STA/
PGA Target        14,848.00   14,848.00   14,848.00   14,848.00      0 STA/
RECYCLE buffer          .00         .00         .00         .00      0 STA/
SGA Target        28,160.00   28,160.00   28,160.00   28,160.00      0 STA/
Shared IO Pool          .00         .00         .00         .00      0 STA/
java pool          1,024.00    1,024.00    1,024.00    1,024.00      0 SHR/DEF
large pool           256.00      256.00      256.00      256.00      0 STA/
shared pool        4,096.00    4,096.00    3,840.00    4,864.00      0 SHR/DEF
streams pool       2,048.00    2,048.00    2,048.00    2,048.00      0 STA/
```

V$SGA_RESIZE_OPS -10g V$MEMORY_RESIZE_OPS – 11g

- A dynamic performance view
- Use to get details of resize operations
- Every memory change since startup

V$SGA_RESIZE_OPS

```
Date: 02/18/08                                                                                   Page:   1
Time: 01:04 PM                                           Component Resize Operations                 SYSTEM
                                                              ault10g1 database

COMPONENT            Oper   OPER_MODE INITIAL_SIZE TARGET_SIZE FINAL_SIZE STATUS    START_TIME   END_TIME
-------------------- ------ --------- ------------ ----------- ---------- --------- ------------ ------------
shared pool          SHRINK DEFERRED     318767104   301989888  301989888 COMPLETE  0217 15:23   0217 15:23
DEFAULT buffer cache GROW   DEFERRED    1040187392  1056964608 1056964608 COMPLETE  0217 15:23   0217 15:23
shared pool          SHRINK DEFERRED     301989888   285212672  285212672 COMPLETE  0217 15:24   0217 15:24
DEFAULT buffer cache GROW   DEFERRED    1073741824  1090519040 1090519040 COMPLETE  0217 15:24   0217 15:24
shared pool          SHRINK DEFERRED     285212672   268435456  268435456 COMPLETE  0217 15:24   0217 15:24
DEFAULT buffer cache GROW   DEFERRED    1056964608  1073741824 1073741824 COMPLETE  0217 15:24   0217 15:24
shared pool          SHRINK DEFERRED     268435456   251658240  251658240 COMPLETE  0217 15:25   0217 15:25
DEFAULT buffer cache GROW   DEFERRED    1090519040  1107296256 1107296256 COMPLETE  0217 15:25   0217 15:25
shared pool          SHRINK DEFERRED     251658240   234881024  234881024 COMPLETE  0217 15:28   0217 15:28
DEFAULT buffer cache GROW   DEFERRED    1107296256  1124073472 1124073472 COMPLETE  0217 15:28   0217 15:28
shared pool          SHRINK DEFERRED     234881024   218103808  218103808 COMPLETE  0217 15:28   0217 15:28
```

Parameters

```
init.ora Parameters                 DB/Inst:   Snaps: 84084-84108
                                                          End value
Parameter Name                Begin value                  (if different)
----------------------------- --------------------------- ---------
_spin_count                   2000
cursor_sharing                EXACT
db_block_size                 16384
db_cache_size                 46170898432
db_keep_cache_size            5368709120
db_name                       cc1
db_writer_processes           4
java_pool_size                167772160
large_pool_size               1073741824
olap_page_pool_size           33554432
open_cursors                  900
pga_aggregate_target          5368709120
processes                     700
sessions                      1000
sga_max_size                  68719476736
sga_target                    68719476736
shared_pool_reserved_size     262144000
shared_pool_size              4294967296
          -------------------------------------------------------------
```

Quick Word on RAC

- The TCP buffers are the biggest memory issue with RAC performance, size them big
- The global dictionary takes memory out of the shared pool, the more memory in the db caches, the more is needed in the shared pools to track it.

Questions/Comments?

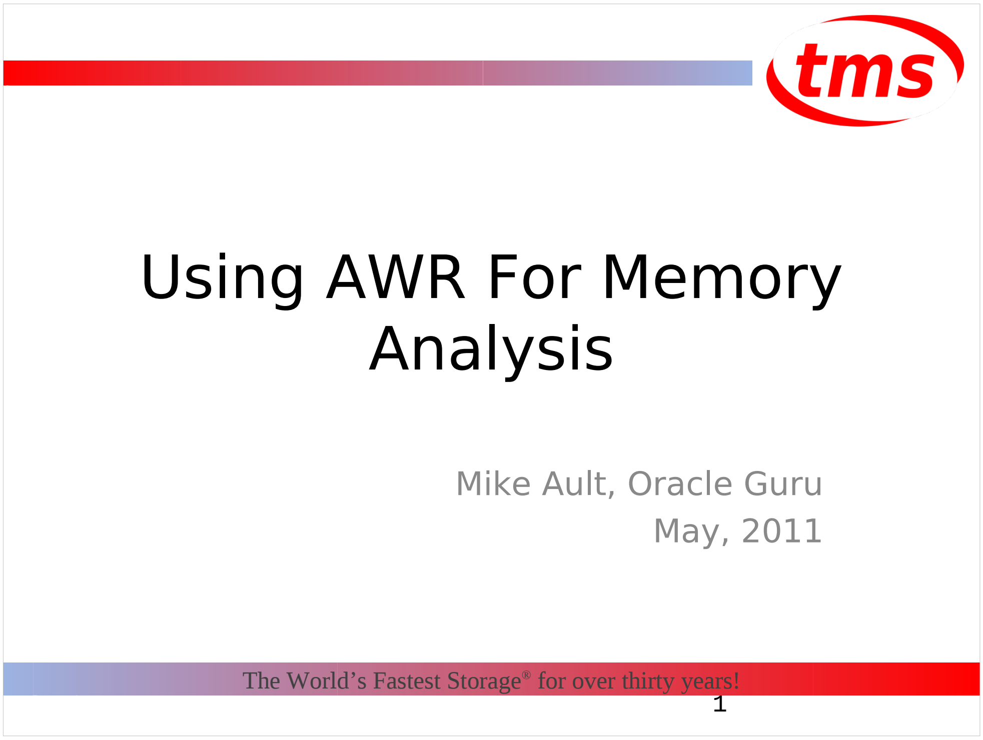
Thank You!

Mike Ault

Oracle@RamSan.com

www.ramsan.com

www.statspackanalyzer.com

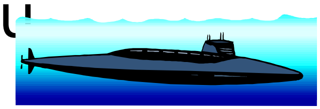


Using AWR For Memory Analysis

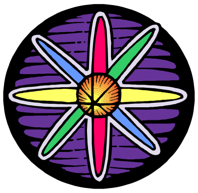
Mike Ault, Oracle Guru
May, 2011

The World’s Fastest Storage® for over thirty years!

Michael R. Ault, Oracle Guru



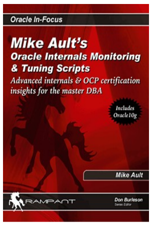
- Nuclear Navy 6 years
- Nuclear Chemist/Programmer 10 years
- Kennedy Western University Graduate
- Bachelors Degree Computer Science
- Certified in all Oracle Versions Since 6
- Oracle DBA, author, since 1990

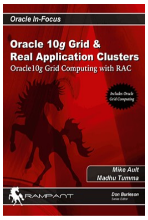


ORACLE CERTIFIED PROFESSIONAL

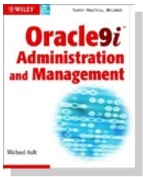
Books by Michael R. Ault

Oracle In-Focus
Oracle Database 10g New Features
Oracle10g Reference for Advanced Tuning & Administration
Mike Ault
Daniel Liu
Madhu Tumma
RAMPANT
Don Burleson

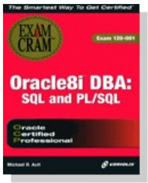

Oracle In-Focus
Mike Ault's Oracle Internals Monitoring & Tuning Scripts
Advanced internals & OCP certification insights for the master DBA
Includes Oracle10g
Mike Ault
RAMPANT
Don Burleson

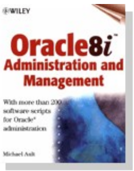

Oracle In-Focus
Oracle 10g Grid & Real Application Clusters
Oracle10g Grid Computing with RAC
Includes Oracle Grid Computing
Mike Ault
Madhu Tumma
RAMPANT
Don Burleson

Oracle In-Focus
Oracle disk I/O Tuning
Disk I/O Performance & Optimization for Oracle 10g Databases
Mike Ault
RAMPANT
Don Burleson


WILEY
Oracle9i Administration and Management
Michael Ault

Oracle In-Focus
Conducting the Oracle Job Interview
IT Manager Guide for Oracle Job Interviews with Oracle Interview Questions
Mike Ault
Donald K. Burleson
RAMPANT
Don Burleson


The Smartest Way To Get Certified
EXAM CRAM
Exam 1Z0-001
Oracle8i DBA: SQL and PL/SQL
Oracle Certified Professional
Michael R. Ault
CORIOLIS


WILEY
Oracle8i Administration and Management
With more than 200 software scripts for Oracle administration
Michael Ault

tms StatspackAnalyzer.com

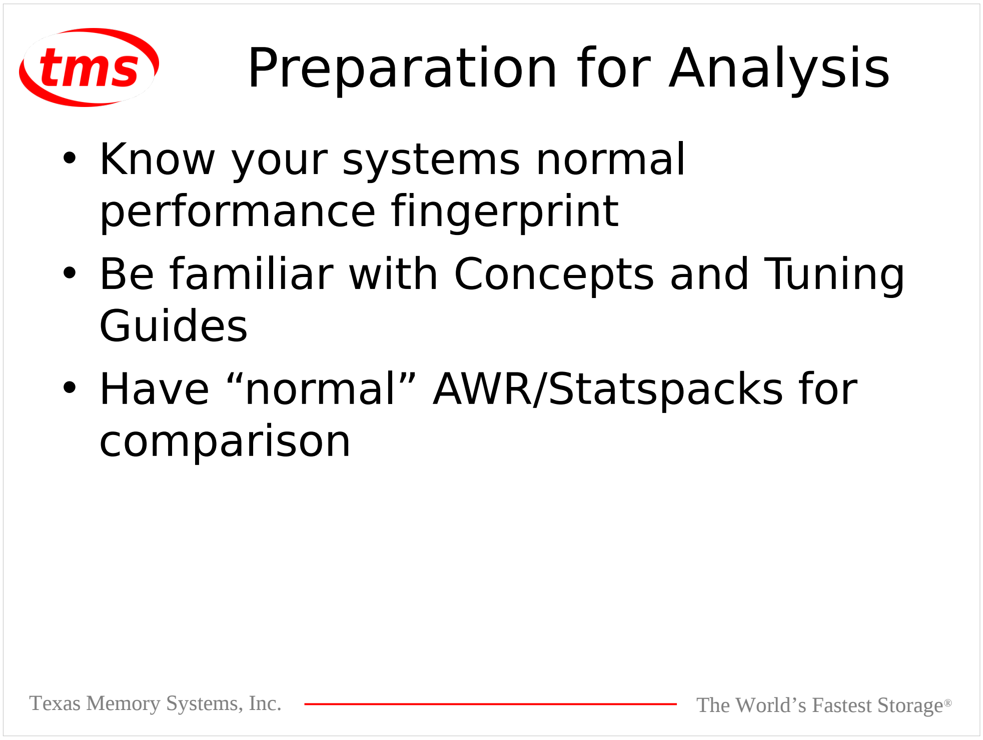
Free Statspack/AWR Analysis

Sponsored by Texas Memory Systems

-Looks for IO bottlenecks and other configuration issues.

-Straightforward tuning advice

Statspack Analyzer



Preparation for Analysis

- Know your systems normal performance fingerprint
- Be familiar with Concepts and Tuning Guides
- Have “normal” AWR/Statspacks for comparison

Oracle and memory

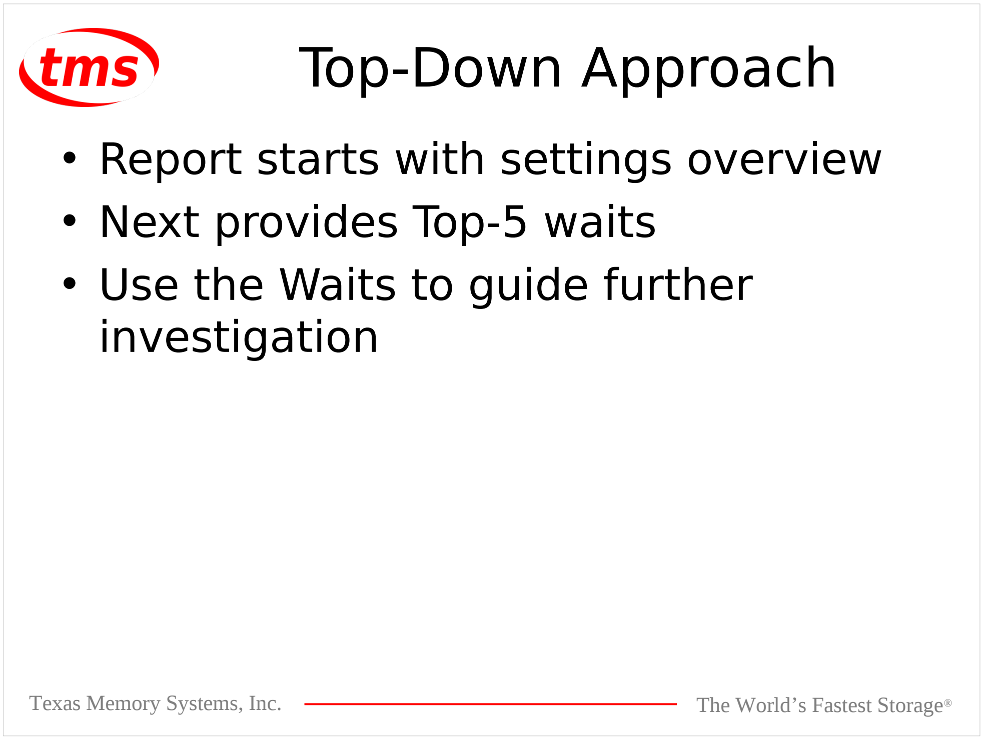
- DB cache – other than direct read/write, all data, index and undo go through here
- Shared Pool – SQL area, PL/SQL library, dictionary caches and lots more
- Streams pool – Only if streams are used
- Java Pool – usually small
- PGA – Each process gets a PGA, stores cursor and other process related information
- Log buffers – Circular buffers for redo information

DB Cache

- Default – should be largest area
- Recycle – for frequently scanned large objects
- Keep – For frequently accessed small objects
- 2-32K areas – Was originally for TTS, now used to tune items (usually in RAC)

Shared Pool

- Shared SQL, PL/SQL, dictionary cache plus
- 34 other areas in 9i
- 551 in 10gR2 (non-RAC) 670 (with RAC)
- 878 in 11gR2 (non-RAC)
  - Report only shows those that change
  - There have been bugs with space leaks in shared pool sub-pools

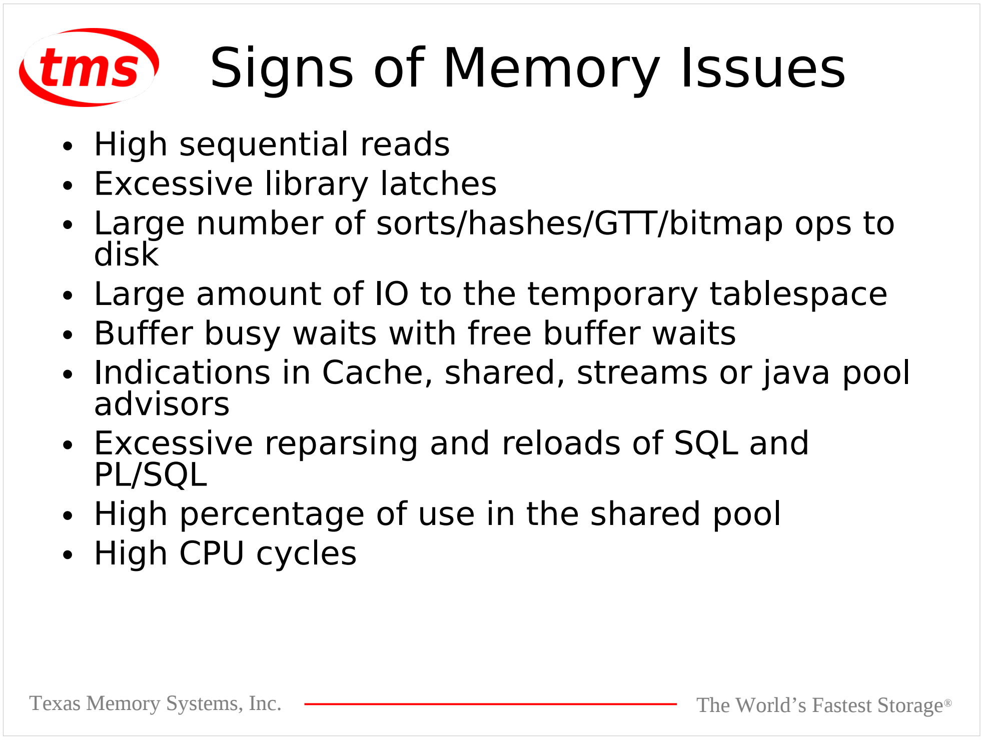


Top-Down Approach

- Report starts with settings overview
- Next provides Top-5 waits
- Use the Waits to guide further investigation

AWR Report Header

```
WORKLOAD REPOSITORY report for
DB Name         DB Id    Instance     Inst Num Startup Time    Release     RAC
------------ ----------- ------------ -------- --------------- ----------- ---
AULTDB        4030696936 aultdb1             1 04-Aug-08 10:16 11.1.0.6.0  YES
Host Name        Platform                         CPUs Cores Sockets Memory(GB)
---------------- -------------------------------- ---- ----- ------- ----------
aultlinux3       Linux IA (32-bit)                   2     1       1       2.97
              Snap Id      Snap Time      Sessions Curs/Sess
            --------- ------------------- -------- ---------
Begin Snap:         91 04-Aug-08 12:00:15        41       1.2
  End Snap:         92 04-Aug-08 13:00:28        47       1.1
   Elapsed:               60.22 (mins)
   DB Time:              139.52 (mins)
Cache Sizes                       Begin        End
~~~~~~~~~~~                  ---------- ----------
               Buffer Cache:     1,312M     1,312M  Std Block Size:         8K
           Shared Pool Size:       224M       224M      Log Buffer:     10,604K
```

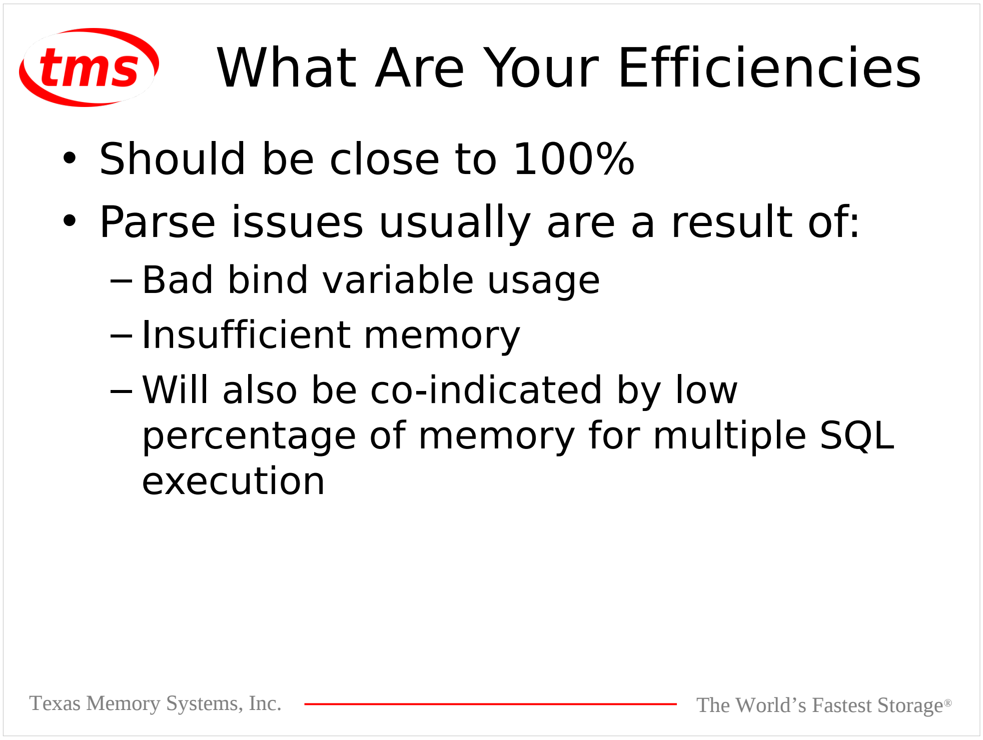


Signs of Memory Issues

- High sequential reads
- Excessive library latches
- Large number of sorts/hashes/GTT/bitmap ops to disk
- Large amount of IO to the temporary tablespace
- Buffer busy waits with free buffer waits
- Indications in Cache, shared, streams or java pool advisors
- Excessive reparsing and reloads of SQL and PL/SQL
- High percentage of use in the shared pool
- High CPU cycles

Load Profile Section

```
Load Profile              Per Second    Per Transaction   Per Exec   Per Call
~~~~~~~~~~~~         ---------------    --------------- ---------- ----------
      DB Time(s):                2.3                7.1       0.63       1.05
       DB CPU(s):                0.3                0.9       0.07       0.13
       Redo size:              800.5            2,461.8
   Logical reads:            6,307.6           19,396.7
   Block changes:                3.6               10.9
  Physical reads:            2,704.9            8,317.8
 Physical writes:               86.9              267.3
      User calls:                2.2                6.8
          Parses:                2.0                6.1
     Hard parses:                0.0                0.1
W/A MB processed:          932,965.4        2,868,990.9
          Logons:                0.1                0.2
        Executes:                3.7               11.3
       Rollbacks:                0.1                0.3
    Transactions:                0.3
```



What Are Your Efficiencies

- Should be close to 100%
- Parse issues usually are a result of:
  - Bad bind variable usage
  - Insufficient memory
  - Will also be co-indicated by low percentage of memory for multiple SQL execution

Load Profile Section

```
Instance Efficiency Percentages (Target 100%)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
            Buffer Nowait %:  100.00       Redo NoWait %:   99.97
            Buffer  Hit   %:   96.09    In-memory Sort %:  100.00
            Library Hit   %:   98.17        Soft Parse %:   97.88
         Execute to Parse %:   45.80         Latch Hit %:   99.95
Parse CPU to Parse Elapsd %:    0.00     % Non-Parse CPU:   99.77

 Shared Pool Statistics        Begin    End
                              ------   ------
             Memory Usage %:   81.53    85.39
    % SQL with executions>1:   79.29    79.48
  % Memory for SQL w/exec>1:   76.73    78.19
```

Top 5 Waits Section

- Critical to look closely at this section
- Use highest wait times to guide investigation
  - DB FILE type waits – physical IO
  - BUFFER type waits – Logical IO
  - LOG type waits – Redo related
  - PX – Parallel Query
  - GC – Global Cache (RAC related)
  - Undo – Undo or rollback segment related

Top 5 Waits Section With possible cache starvation

```
Top 5 Timed Foreground Events
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                                                           Avg
                                                          wait   % DB
Event                                 Waits     Time(s)   (ms)   time Wait Class
------------------------------ ------------ ----------- ------ ------ ----------
db file sequential read             465,020       3,969      9   47.4 User I/O
DB CPU                                              995          11.9
db file parallel read                 2,251         322    143    3.8 User I/O
db file scattered read               15,268         153     10    1.8 User I/O
gc current block 2-way              108,739         116      1    1.4 Cluster
```

Top 5 Waits Section With Shared Pool Issues

```
Top 5 Timed Events                                         Avg %Total
~~~~~~~~~~~~~~~~~~                                        wait   Call
Event                                 Waits    Time (s)   (ms)   Time Wait Class
------------------------------ ------------ ----------- ------ ------ ----------
CPU time                                        435,461          41.1
PX Deq Credit: send blkd        124,829,330     138,223      1   13.0      Other
library cache pin                    20,347      57,692   2835    5.4 Concurrenc
library cache lock                   19,226      56,078   2917    5.3 Concurrenc
db file sequential read          16,798,329      42,215      3    4.0   User I/O
          -------------------------------------------------------------
Top 5 Timed Events                                         Avg %Total
~~~~~~~~~~~~~~~~~~                                        wait   Call
Event                                 Waits    Time (s)   (ms)   Time Wait Class
------------------------------ ------------ ----------- ------ ------ ----------
CPU time                                         24,956          29.3
latch: library cache              1,757,331       9,886      6   11.6 Concurrenc
db file sequential read             759,605       6,146      8    7.2   User I/O
cursor: pin S                     2,103,389       4,988      2    5.9      Other
log file sync                       250,039       2,387     10    2.8     Commit
          -------------------------------------------------------------
```

Buffer Type Waits

- latch: cache buffers chains – Hot blocks, check for hot objects, high IO rates
- free buffer waits – Insufficient buffers, processes holding buffers too long, IO subsystem over loaded
- buffer busy waits – See what is causing them further along in report
- gc buffer busy – Overloaded interconnect, find problem objects and tune
- log buffer space – High load, too small a log buffer, increase log buffer size
- latch: cache buffers lru chain – Freelist issues, hot blocks, new buffers, buffers being writen
- 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
- Free buffer waits – insufficient buffers

Fixing Cache Waits

- Reduce logical IO rates (buffer caches latch)
- Increase the cache size (lru chain latch)
- Increase the cache size (free buffer waits)
- Increase _db_block_lru_latches
- Increase _db_block_hash_buckets
- Reduce hot blocks

Shared Pool Waits

- library cache pin – Loading or compiling same SQL
- library cache lock – Loading or compiling same SQL
- *latch: library cache – Usually a result of excessive parsing
- latch: shared pool latch –Parsing issues
- *latch: library cache lock – Usually a result of excessive parsing
- *latch: library cache – Usually a result of excessive parsing
- row cache lock – shared pool too small
- Library cache load lock – Wait for a reload by another session. Excessive hard/soft parsing.

* Gone in 11g to mutex

Shared Pool Mutexes

- Cursor:mutex X – resource is busy, requestor needs exclusive access
- Cursor:mutex S – resource is held in X mode by another session
- Cursor:pin X – resource is held n S or X by another session
- Cursor:pin S – re-execute of same cursor
- Cursor:pin S wait on X – resource is held in X mode by another session
- Library cache: mutex X – Bind variable issues
- Library cache: mutex S – Bind variable issues

- Less costly than latches

What to Do?

- Share cursors (avoid hard parsing)
  - BIND VARIABLES!!!!!
  - Cursor_sharing
- Avoid soft parsing
  - Cursor_space_for_time
  - Session_cached_cursors
- Avoid invalidations and reloads
  - Make sure shared pool is large enough

What Next?

- Determine wait events of concern
- Drill down to specific sections of report for deeper analysis
- Use custom scripts, ADDM and Ash to investigate issues

Classes

```
Wait Class                                DB/Inst: Snaps: 84084-84108
-> 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
-------------------- ---------------- ------ ---------------- ------- ---------
Other                     153,619,985   16.5          192,921       1     102.3
Concurrency                 2,536,362   26.9          128,816      51       1.7
User I/O                   30,594,385     .0          124,207       4      20.4
System I/O                  5,104,873     .0           17,633       3       3.4
Application                    65,645    5.0            6,508      99       0.0
Commit                        267,317     .0            4,234      16       0.2
Configuration                 553,825   69.5              858       2       0.4
Network                    13,513,847     .0              274       0       9.0
Administrative                     30   70.0                0      10       0.0
          -------------------------------------------------------------
```

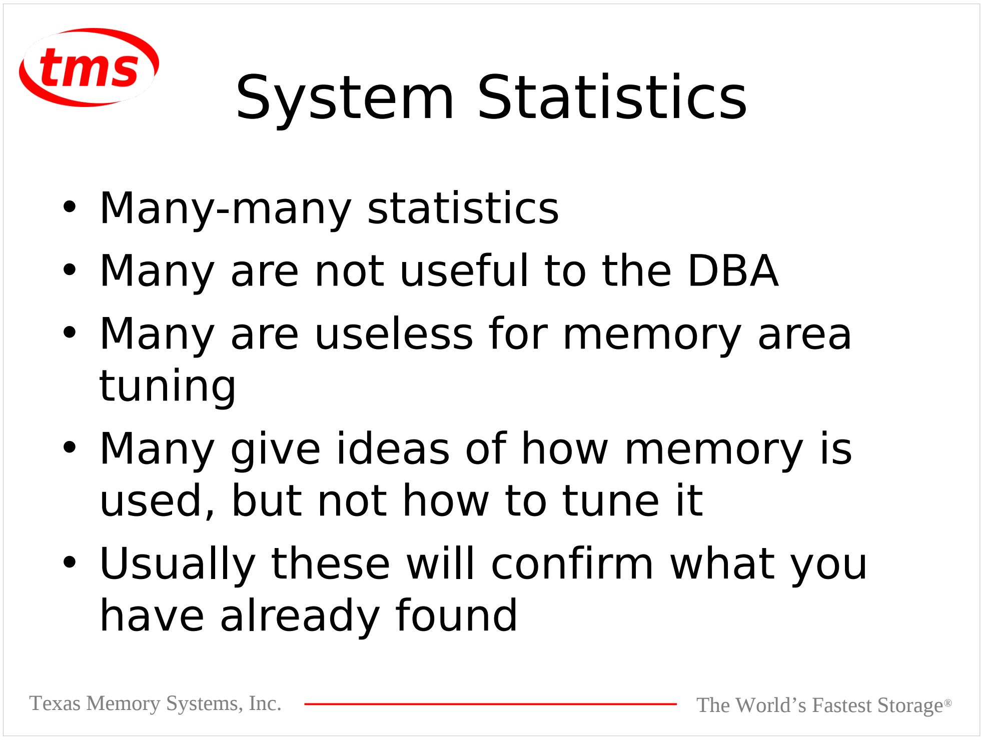
Operating System Statistics

```
Operating System Statistics         DB/Inst: Snaps: 84084-84108

Statistic                                            Total
-------------------------------- --------------------
BUSY_TIME                                       45,601,415
IDLE_TIME                                        6,316,939
IOWAIT_TIME                                        567,343
NICE_TIME                                          810,986
SYS_TIME                                         3,169,946
USER_TIME                                       41,265,848
LOAD                                                    50
RSRC_MGR_CPU_WAIT_TIME                                   0
PHYSICAL_MEMORY_BYTES                      270,208,987,136
NUM_CPUS                                                24
NUM_CPU_SOCKETS                                          4

-------------------------------------------------------------
```

SQL Areas

SQL ordered by CPU Time – Sorting, bad paths
SQL ordered by Gets – Excessive logical IO
SQL ordered by Reads – Cache starvation
SQL ordered by Parse Calls – Cursor sharing, cursor caching
SQL ordered by Version Count – Versioning is usually due to a bug, check with support

- **Tune SQL that appears in more than one of these areas**
- **Tune SQL at the top of these sections**



System Statistics

- Many-many statistics
- Many are not useful to the DBA
- Many are useless for memory area tuning
- Many give ideas of how memory is used, but not how to tune it
- Usually these will confirm what you have already found

System Statistics

```
Instance Activity Stats                   DB/Inst: Snaps: 84084-84108

Statistic                                     Total     per Second     per Trans
-------------------------------- ------------------ -------------- -------------
dirty buffers inspected                     686,267           31.8           0.5
execute count                            78,907,090        3,656.2          52.6
free buffer inspected                   161,591,258        7,487.4         107.6
free buffer requested                   176,367,274        8,172.1         117.5
hot buffers moved to head of LRU         15,346,759          711.1          10.2
immediate (CR) block cleanout ap          2,267,512          105.1           1.5
immediate (CURRENT) block cleano          5,139,016          238.1           3.4
no buffer to keep pinned count              136,849            6.3           0.1
no work - consistent read gets        4,459,140,613      206,616.1       2,969.8
opened cursors cumulative                26,795,933        1,241.6          17.9
parse count (failures)                          160            0.0           0.0
parse count (hard)                          398,147           18.5           0.3
parse count (total)                      20,200,501          936.0          13.5
parse time cpu                            3,883,178          179.9           2.6
parse time elapsed                        7,474,786          346.4           5.0
physical read total IO requests          37,303,810        1,728.5          24.8
physical reads direct temporary          58,378,313        2,705.0          38.9
physical write total IO requests         13,738,098          636.6           9.2
physical writes direct temporary         58,440,795        2,707.9          38.9
```

System Statistics

```
pinned buffers inspected                   120,843          5.6          0.1
recursive calls                        749,184,714     34,713.8        499.0
recursive cpu usage                     39,323,240      1,822.1         26.2
redo log space requests                        190          0.0          0.0
redo log space wait time                       333          0.0          0.0
redo synch time                            433,625         20.1          0.3
redo synch writes                          236,148         10.9          0.2
redo write time                            567,670         26.3          0.4
redo writer latching time                   56,827          2.6          0.0
redo writes                              1,127,300         52.2          0.8
rollback changes - undo records          1,395,329         64.7          0.9
rollbacks only - consistent read           346,504         16.1          0.2
session cursor cache hits               21,520,355        997.2         14.3
session logical reads               10,474,545,504    485,342.3      6,976.0
sorts (disk)                                 3,529          0.2          0.0
sorts (memory)                           9,012,270        417.6          6.0
sorts (rows)                       110,063,794,220  5,099,850.2     73,302.3
sql area evicted                           327,084         15.2          0.2
sql area purged                             29,720          1.4          0.0
table scans (long tables)                1,149,945         53.3          0.8
table scans (short tables)               7,528,140        348.8          5.0
transaction rollbacks                      252,407         11.7          0.2
```

System Statistics

```
user I/O wait time                          12,422,069          575.6            8.3
user calls                                   8,038,839          372.5            5.4
user commits                                 1,439,821           66.7            1.0
user rollbacks                                  61,684            2.9            0.0
workarea executions - multipass                      0            0.0            0.0
workarea executions - onepass                    5,293            0.3            0.0
workarea executions - optimal                7,113,060          329.6            4.7
```

Instance Activity Statistics

```
Instance Activity Stats - Absolute
-> Statistics with absolute values (should not be diffed)

Statistic                            Begin Value       End Value
-------------------------------- --------------- ---------------
session cursor cache count            28,024,069      28,789,659
opened cursors current                     2,921           6,982
workarea memory allocated                289,532       2,531,741
logons current                               144             287
          --------------------------------------------------
```

Tablespace/File IO Reports

- Helps confirm IO issues
- Also helps with temp area issue determination

Tablespace IO

```
Tablespace IO Stats
-> ordered by IOs (Reads + Writes) desc

Tablespace
------------------------------
                 Av       Av     Av                       Av     Buffer Av Buf
         Reads Reads/s Rd(ms) Blks/Rd       Writes Writes/s      Waits Wt(ms)
-------------- ------- ------ ------- ------------ -------- ---------- ------
TEMP
    11,484,000     532   16.3     4.1    3,478,365      161     12,266    2.0
REPMAN_TEMP
     1,703,767      79   27.2     8.2    1,457,241       68          0    0.0
UNDOTBS3
        30,012       1    8.0     1.0    1,512,571       70    142,889    1.1
RSNET_DTSA
     1,496,441      69    1.2     2.0        2,454        0    130,753    1.3
LOREAL_D_CVS_DAILY_ITSA
       846,665      39    0.9     1.0          338        0          0    0.0
```

Buffer Pool Statistics

```
Buffer Pool Statistics                   DB/Inst: CC1/cc1  Snaps: 84084-84108
-> Standard block size Pools  D: default,  K: keep,  R: recycle
-> Default Pools for other block sizes: 2k, 4k, 8k, 16k, 32k

                                                            Free Writ     Buffer
     Number of Pool         Buffer     Physical    Physical Buff Comp       Busy
P      Buffers Hit%           Gets        Reads      Writes Wait Wait      Waits
--- ---------- ---- -------------- ------------ ----------- ---- ---- ----------
D    3,361,107   96  3,643,978,600  163,055,679  14,338,623    0    0    711,281
K      321,600  100  2,527,600,634        7,379      28,755    0    0        123
          -------------------------------------------------------------
```

- Note that there are Buffer Busy Waits, but no Free Buffer Waits
- These are due to hot block contention
- Increasing memory probably won’t help with this
- However…also look at db file sequential read waits and the cache advisory section

Buffer Pool Advisory Section

```
Buffer Pool Advisory
-> Only rows with estimated physical reads >0 are displayed
-> ordered by Block Size, Buffers For Estimate

                                                Est
                                               Phys
    Size for   Size        Buffers for         Read          Estimated
P    Est (M) Factor           Estimate       Factor    Physical Reads
--- -------- ------ ---------------- ------ ------------------
D      5,344     .1          335,670    1.9     15,767,325,073
D     10,688     .2          671,340    1.4     11,371,357,960
…
D    106,880    2.0        6,713,400    1.0      7,964,367,701
K        512     .1           32,160  102.8      3,507,100,178
K      1,024     .2           64,320    7.8        264,615,629
K      1,536     .3           96,480    1.4         49,384,590
…
K     10,240    2.0          643,200    1.0         32,639,643
          ---------------------------------------------
```

Buffer Pool Advisory Section

- As you can see, this repor shows even doubling the default or keep would be no benefit
- Let’s look at one that would benefit from increased buffer pool

Buffer Pool Advisor Section

```
Buffer Pool Statistics           DB/Inst:   Snaps: 26064-26097
-> Standard block size Pools  D: default,  K: keep,  R: recycle
-> Default Pools for other block sizes: 2k, 4k, 8k, 16k, 32k

                                                            Free Writ     Buffer
     Number of Pool         Buffer     Physical    Physical Buff Comp       Busy
P      Buffers Hit%           Gets        Reads      Writes Wait Wait      Waits
--- ---------- ---- -------------- ------------ ----------- ---- ---- ----------
D      818,201   99    130,795,544    1,578,580     276,075    0    0      3,418
          -------------------------------------------------------------
```

- Before we go there…look here
- Notice no free buffer waits

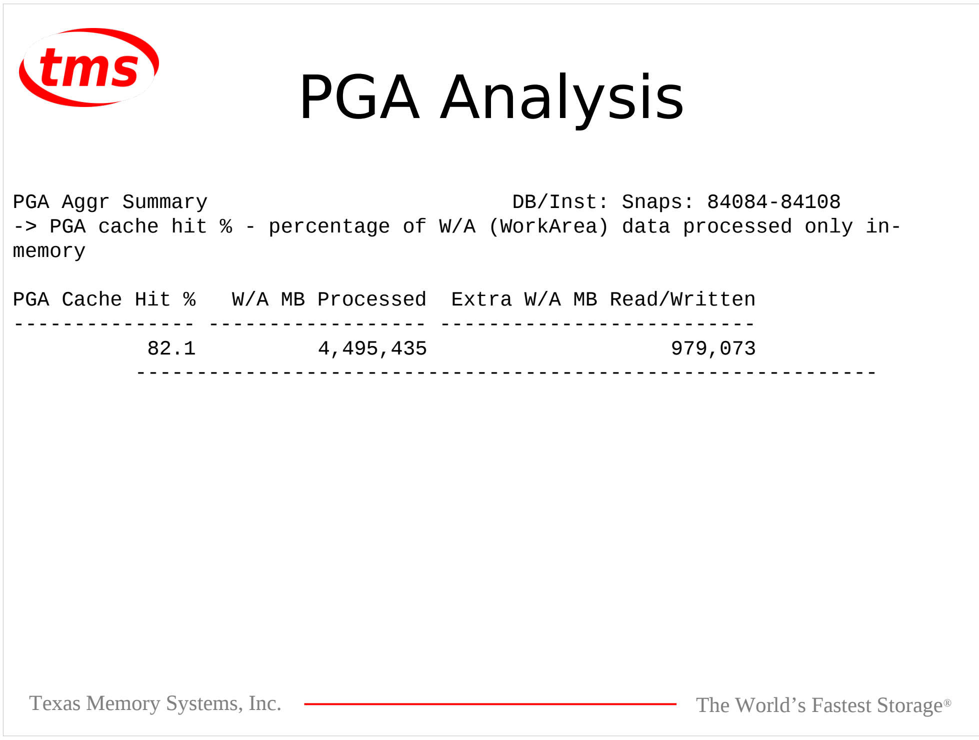
Buffer Pool Advisor Section

```
Buffer Pool Advisory                         Snap: 26097
-> Only rows with estimated physical reads >0 are displayed
-> ordered by Block Size, Buffers For Estimate

                                            Est
                                           Phys
    Size for   Size      Buffers for   Read          Estimated
P    Est (M) Factor         Estimate Factor     Physical Reads
--- -------- ------ ---------------- ------ ------------------
D        656     .1           81,139    2.0        125,592,784
D      1,312     .2          162,278    1.8        113,080,052
…
D     10,496    1.6        1,298,224    0.7         42,942,366
D     11,152    1.7        1,379,363    0.7         41,649,501
D     11,808    1.8        1,460,502    0.6         40,403,058
-------------------------------------------------------------
```

- Notice that at 1.8 times the current size physical reads down by 40%
- Db file sequential reads was 13% of waits

PGA Analysis

- Several of the next sections deal with PGA
- PGA_AGGREGATE_TARGET sets the PGA area
- 5% of PGA_AGGREGATE_TARGET can be allocated to each session up to a maximum of "_PGA_MAX_SIZE" which is usually 200 or 500 megabytes
- Manually setting SORT_AREA_SIZE or HASH_AREA_SIZE overrides at the session level
- Some processes such as RMAN and shared servers don't use PGA_AGGREGATE_TARGET but use the old manual settings, indicated by 4-8 or 8-16m sorts even with adequate PGA_AGGREGATE_TARGET settings



PGA Analysis

```
PGA Aggr Summary                         DB/Inst: Snaps: 84084-84108
-> PGA cache hit % - percentage of W/A (WorkArea) data processed only in-
memory

PGA Cache Hit %   W/A MB Processed  Extra W/A MB Read/Written
--------------- ------------------ --------------------------
           82.1          4,495,435                    979,073
          -------------------------------------------------------------
```

PGA Analysis

```
PGA Aggr Target Stats                     DB/Inst: Snaps: 84084-84108
-> B: Begin snap   E: End snap (rows dentified with B or E contain data
   which is absolute i.e. not diffed over the interval)
-> Auto PGA Target - actual workarea memory target
-> W/A PGA Used    - amount of memory used for all Workareas (manual + auto)
-> %PGA W/A Mem    - percentage of PGA memory allocated to workareas
-> %Auto W/A Mem   - percentage of workarea memory controlled by Auto Mem Mgmt
-> %Man W/A Mem    - percentage of workarea memory under manual control

                                                %PGA  %Auto   %Man
    PGA Aggr   Auto PGA   PGA Mem    W/A PGA     W/A    W/A    W/A Global Mem
   Target(M)  Target(M)  Alloc(M)    Used(M)     Mem    Mem    Mem   Bound(K)
- ---------- ---------- ---------- ---------- ------ ------ ------ ----------
B      5,120      4,320    1,680.5      193.5   11.5   99.7     .3    524,280
E      5,120      4,202    4,400.5    2,219.2   50.4   99.9     .1    524,280
          -------------------------------------------------------------
```

PGA Analysis

```
PGA Aggr Target Histogram                    Snaps: 84084-84108
Low      High
Optimal Optimal    Total Execs  Optimal Execs 1-Pass Execs M-Pass Execs
------- ------- -------------- -------------- ------------ ------------
     2K      4K      6,308,435      6,308,435            0            0
    64K    128K         32,500         32,500            0            0
   128K    256K         36,535         36,535            0            0
   256K    512K         47,477         47,477            0            0
   512K   1024K        353,344        353,344            0            0
     1M      2M        201,558        201,558            0            0
     2M      4M         22,468         22,468            0            0
     4M      8M         21,796         21,725           71            0
     8M     16M         28,892         28,714          178            0
    16M     32M         30,478         30,346          132            0
    32M     64M         19,898         18,690        1,208            0
    64M    128M          9,080          7,284        1,796            0
   128M    256M          1,682            732          950            0
   256M    512M            734            179          555            0
   512M   1024M            329             58          271            0
     1G      2G            131             14          117            0
     2G      4G             17              4           13            0
          -------------------------------------------------------------
```

PGA Analysis

- SS*100/2.5*NOS
  - SS- sort size
  - NOS – number of sorts
- 128M*100/2.5=5120
- Hash gets full 5%
- Sort gets ½ of hash
- ½ of 5 is 2.5
- 10|10 rule:
  - 10% of sessions are active
  - 10% of active sessions doing sorts
- 10% of 208 sessions is 20.8; 10% of 20.8 is 2
- 2*5120 is 10420 gb
- Let’s see what the advisor says…

PGA Analysis

```
PGA Memory Advisory                                        Snap: 84108
-> When using Auto Memory Mgmt, minimally choose a pga_aggregate_target
value where Estd PGA Overalloc Count is 0
                                             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
---------- ------- ---------------- ---------------- -------- ----------
       640     0.1    167,211,258.2    133,135,735.1     56.0     18,399
     1,280     0.3    167,211,258.2     65,323,839.1     72.0      1,933
     2,560     0.5    167,211,258.2     34,248,434.3     83.0      1,842
     3,840     0.8    167,211,258.2     28,768,442.5     85.0      1,803
     5,120     1.0    167,211,258.2     19,597,963.1     90.0        860
     6,144     1.2    167,211,258.2     14,425,059.9     92.0        845
     7,168     1.4    167,211,258.2     13,624,988.5     92.0        839
     8,192     1.6    167,211,258.2     12,882,080.7     93.0          0
     9,216     1.8    167,211,258.2     12,146,846.2     93.0          0
    10,240     2.0    167,211,258.2     11,689,119.6     93.0          0
    15,360     3.0    167,211,258.2     10,710,132.3     94.0          0
    20,480     4.0    167,211,258.2     10,563,396.2     94.0          0
    30,720     6.0    167,211,258.2     10,539,291.8     94.0          0
    40,960     8.0    167,211,258.2     10,539,291.8     94.0          0
          -----------------------------------------------------------
```

PGA Analysis

- Advisor says 8 gb
- Calculation says 10 gb
- Take whichever you feel will give best results
- Won’t be used unless it needs to be

Shared Pool Advisor

- Use reloads and used percentages to guide you
- The advisor rarely has any meaningful information
- Even when reloads are huge and other factors show the pool should be increased, or, decreased it has told me things where fine

Shared Pool Advisor

```
Shared Pool Advisory                            DB/Inst: Snap: 84108
-> SP: Shared Pool     Est LC: Estimated Library Cache   Factr: Factor
-> Note there is often a 1:Many correlation between a single logical object
   in the Library Cache, and the physical number of memory objects associated
   with it.  Therefore comparing the number of Lib Cache objects (e.g. in
   v$librarycache), with the number of Lib Cache Memory Objects is invalid.

                                        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
---------- ----- -------- ------------ ------- ------ ------- ------ -----------
     2,160    .4      619       76,837 #######    1.0 #######    2.4  88,538,740
     2,736    .5    1,188       95,749 #######    1.0 #######    2.1  88,936,333
     3,312    .6    1,761      110,785 #######    1.0 #######    1.8  89,297,339
     3,888    .7    2,333      125,755 #######    1.0 #######    1.6  89,610,155
     …
     8,496   1.5    6,916      238,592 #######    1.0 #######     .5  91,076,008
     9,072   1.6    7,489      252,248 #######    1.0 #######     .4  91,187,541
     9,648   1.7    8,061      264,748 #######    1.0 #######     .3  91,291,114
    10,224   1.8    8,632      274,837 #######    1.0 #######     .3  91,388,340
    10,800   1.9    9,201      284,723 #######    1.0 #######     .2  91,480,577
    11,376   2.0    9,774      293,073 #######    1.0 #######     .2  91,569,191
          -------------------------------------------------------------
```

SGA Target Advisor

- I believe you should set the base parameters and then allow SGA_TARGET to set itself
- Other calculation schemes involve adding up the proposed sizes needed and then setting the value
- Same logic applies to MEMORY_TARGET only include PGA_AGGREGATE_TARGET in the base
- Then set SGA_MAX_SIZE or MEMORAY_MAX_SIZE 10-20% higher than their associated TARGET values.

SGA Target

```
SGA Target Advisory                               Snap: 26097

SGA Target   SGA Size       Est DB     Est Physical
  Size (M)     Factor     Time (s)            Reads
---------- ---------- ------------ ----------------
     2,048        0.3    8,957,297      125,595,242
     4,096        0.5      745,014       99,011,933
     6,144        0.8      683,241       84,201,232
     8,192        1.0      603,250       62,255,994
    10,240        1.3      573,691       48,902,083
    12,288        1.5      573,630       42,944,185
    14,336        1.8      573,630       37,745,809
    16,384        2.0      573,630       37,745,809
          ---------------------------------------------
```

- At 2x shows a 60% reduction in PR, same as doubling the cache for this instance

Streams Pool Advisor

- Only valid if you use streams
- If it shows you have spills to disk make pool larger

Streams Pool Advisor

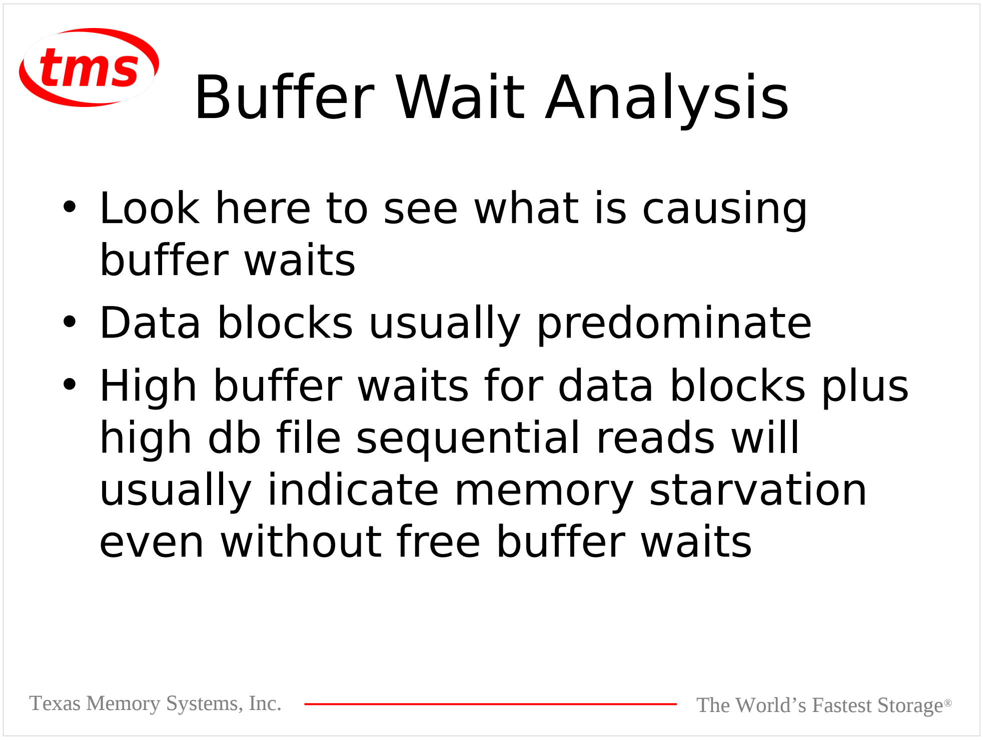
```
Streams Pool Advisory                              DB/Inst: Snap: 84108

  Size for       Size   Est Spill   Est Spill Est Unspill Est Unspill
  Est (MB)     Factor       Count    Time (s)       Count    Time (s)
---------- --------- ----------- ----------- ----------- -----------
        16       0.5           0           0           0           0
        32       1.0           0           0           0           0
        48       1.5           0           0           0           0
        64       2.0           0           0           0           0
        80       2.5           0           0           0           0
…
       240       7.5           0           0           0           0
       256       8.0           0           0           0           0
       272       8.5           0           0           0           0
       288       9.0           0           0           0           0
       304       9.5           0           0           0           0
       320      10.0           0           0           0           0
          -------------------------------------------------------------
```

Java Pool Advisor

- Another area usually not used
- If you use it you will get errors if it is too small
- Under AMM will grow but usually won’t shrink

Java Pool Advisor

```
Java Pool Advisory                       DB/Inst: Snap: 37
                                      Est LC Est LC  Est LC Est LC
      Java    JP   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
---------- ----- -------- ---------- ------- ------ ------- ------ -----------
        64    .5       10        168      10    1.0  11,974    1.0         389
       128   1.0       12        201      10    1.0  11,974    1.0         465
       192   1.5       12        201      10    1.0  11,974    1.0         465
       256   2.0       12        201      10    1.0  11,974    1.0         465
          -------------------------------------------------------------
```

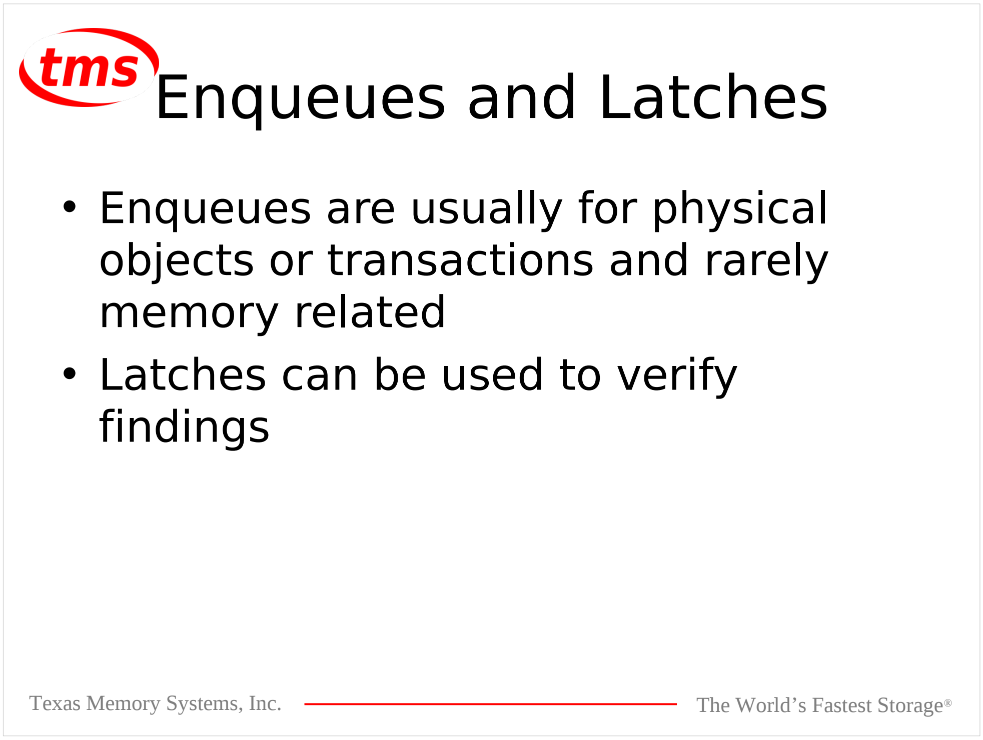


Buffer Wait Analysis

- Look here to see what is causing buffer waits
- Data blocks usually predominate
- High buffer waits for data blocks plus high db file sequential reads will usually indicate memory starvation even without free buffer waits

Buffer Wait Analysis

```
Buffer Wait Statistics                    DB/Inst:
Snaps: 84084-84108
-> ordered by wait time desc, waits desc

Class                        Waits Total Wait Time (s)  Avg Time (ms)
------------------ ----------- ------------------- --------------
data block             539,313               1,770              3
1st level bmb            5,873                  88             15
undo block              49,801                  71              1
undo header             35,848                  66              2
file header block       57,833                  39              1
segment header          12,980                  28              2
3rd level bmb            4,900                  22              5
2nd level bmb            5,206                  20              4
extent map                  73                   0              5
          ----------------------------------------------------
```



Enqueues and Latches

- Enqueues are usually for physical objects or transactions and rarely memory related
- Latches can be used to verify findings

Latches

```
Latch Activity                          DB/Inst: Snaps: 84084-84108
-> "Get Requests", "Pct Get Miss" and "Avg Slps/Miss" are statistics for
   willing-to-wait latch get requests
-> "NoWait Requests", "Pct NoWait Miss" are for no-wait latch get requests
-> "Pct Misses" for both should be very close to 0.0
                                              Pct    Avg   Wait                 Pct
                                   Get        Get   Slps   Time       NoWait NoWait
Latch Name                    Requests       Miss  /Miss    (s)     Requests   Miss
------------------------ -------------- ------ ------ ------ ------------ ------
SGA IO buffer pool latch        154,979    0.0    0.7      0      261,675    0.3
SQL memory manager latch         21,353   12.7    1.2     32        7,066    0.1
active service list           2,524,503    1.7    0.2     10        7,615    0.1
cache buffers lru chain      66,153,522    1.0    0.6   1172  469,078,119    1.8
cache table scan latch        3,749,595    0.3    0.3      9    3,986,131    0.3
library cache               272,976,941    0.1    0.4   2242    1,582,568  216.3
library cache lock          179,441,868    0.1    0.1     34       58,383    0.1
object queue header oper    883,815,615    0.1    0.1    365       45,167    0.2
process queue reference   8,224,189,548    0.0    0.0     15  613,910,934    4.3
redo allocation               8,482,485    8.1    0.1    271  435,767,620    1.4
row cache objects           871,266,120    0.4    0.2   1765      580,035    1.6
simulator lru latch               7,890    1.1    1.1      1 ############   17.1
undo global data             68,788,574    0.0    0.1      8       42,419    0.1

          -------------------------------------------------------------
```

Latches

- When a latch can’t be obtained a process will “sleep”
- The number of CPU cycles is set by the “_SPIN_COUNT” setting
- Defaults to 2000, was set in the era of slow CPUs
- Experts agree setting as high as 8000-10000 is acceptable
- Reduces sleeps and improved performance

Latch Sleeps

```
Latch Sleep Breakdown                    DB/Inst: Snaps: 84084-84108
-> ordered by misses desc

Latch Name
----------------------------------------
  Get Requests      Misses      Sleeps  Spin Gets   Sleep1   Sleep2   Sleep3
-------------- ----------- ----------- ---------- -------- -------- --------
cache buffers chains
 9,561,948,628 ###########     402,582 ##########        0        0        0
session allocation
   570,516,196  14,728,230     461,868 ##########        0        0        0
messages
   105,586,860   3,521,238      16,783  3,506,276        0        0        0
row cache objects
   871,266,120   3,508,138     655,697  2,914,340        0        0        0
process queue reference
 8,224,189,548   2,645,262      19,802  2,629,967        0        0        0
active checkpoint queue latch
    50,214,009   1,082,455      13,133  1,070,658        0        0        0
redo writing
    52,201,162   1,026,204       6,893  1,020,132        0        0        0
object queue header operation
   883,815,615     936,882     137,869    812,882        0        0        0
```

Latch Miss Sources

```
Latch Miss Sources                        DB/Inst: Snaps: 84084-84108
-> only latches with sleeps are shown
-> ordered by name, sleeps desc

                                                     NoWait              Waiter
Latch Name               Where                       Misses     Sleeps   Sleeps
------------------------ -------------------------- ------- ---------- --------
In memory undo latch     ktiFlush: child                  0     12,727    3,775
...
cache buffers chains     kcbgtcr: kslbegin excl           0    821,219  689,229
...
cache buffers lru chain  kcbbxsv: move to being wri       0     19,728   87,892
...
library cache            kglpndl: child: after proc       0     28,574   10,336
...
library cache lock       kgllkdl: child: cleanup          0      7,106    6,749
...
object queue header oper kcbw_unlink_q                    0    139,971   27,436
...
parameter table allocati ksp_param_table_free             0     60,653   60,605
row cache objects        kqrpre: find obj                 0    381,022  379,948
...
session allocation       ksuprc                           0    191,233  175,384
...
shared pool              kghalo                           0    194,517  123,739
...
simulator hash latch     kcbsacc: lookup dba              0     16,413   26,348
```

Dictionary Cache

- Misses can be costly
- Only correction is larger shared pool
- Used to be individually controlled by DC parameters (v6)
- Now controlled automatically

Dictionary Cache

```
Dictionary Cache Stats                    DB/Inst: Snaps: 84084-84108
-> "Pct Misses"  should be very low (< 2% in most cases)
-> "Final Usage" is the number of cache entries being used
                                   Get    Pct    Scan   Pct      Mod      Final
Cache                         Requests   Miss    Reqs  Miss     Reqs      Usage
------------------------- ------------ ------ ------- ----- -------- ----------
dc_awr_control                     442    9.3       0   N/A       48          1
dc_constraints                  60,034   41.5       0   N/A   60,034         49
dc_files                           559   97.3       0   N/A        0          0
dc_global_oids                  20,580    3.0       0   N/A        0         25
dc_histogram_data            7,832,289    2.0       0   N/A   19,621      7,472
dc_histogram_defs            5,795,619   15.5       0   N/A  322,942      6,953
dc_object_grants                99,052   11.6       0   N/A        0        370
dc_object_ids              105,285,794    2.3       0   N/A   38,220     66,669
dc_objects                  53,343,945    4.8       0   N/A  161,568      8,250
dc_profiles                     60,308    0.1       0   N/A        0          5
dc_segments                 19,064,933   58.4       0   N/A  256,828     16,244
dc_sequences                   137,407    0.9       0   N/A  137,407         35
dc_table_scns                  180,866    4.2       0   N/A       88         14
dc_tablespace_quotas           101,823    2.5       0   N/A  101,679         43
dc_tablespaces              11,521,250    1.3       0   N/A       18     10,945
dc_usernames                   750,849    0.4       0   N/A        0        104
global database name               118    6.8       0   N/A        0          1
kqlsubheap_object                    1  100.0       0   N/A        0          0
outstanding_alerts             394,019   41.7       0   N/A        0     10,943
          -------------------------------------------------------------
```

Library Cache Activity

- Usually only see issues in SQL and PL/SQL areas
- Invalidations high means DDL activity
- Reloads high means SQL or PL/SQL issues

Library Cache Activity

```
Library Cache Activity                    DB/Inst: Snaps: 84084-84108
-> "Pct Misses"  should be very low

                         Get    Pct            Pin    Pct             Invali-
Namespace           Requests   Miss       Requests   Miss    Reloads  dations
--------------- ------------ ------ -------------- ------ ---------- --------
BODY                 106,197    0.3      2,372,306    0.1      1,288        0
CLUSTER               11,232    0.4         19,082    0.7         81        0
INDEX              1,728,015    1.9     28,528,015    0.2     17,973        0
PIPE                       9    0.0              9    0.0          0        0
SQL AREA           3,482,631   14.7    108,055,227    2.4  1,276,544 ########
TABLE/PROCEDURE      290,691    6.1     32,986,432    0.5     76,695        0
TRIGGER              621,821    0.1      2,312,915    0.1      2,423        0
          -------------------------------------------------------------
```

Process Statistics

```
Process Memory Summary                   DB/Inst: Snaps: 84084-84108
-> B: Begin snap   E: End snap
-> All rows below contain absolute values (i.e. not diffed over the interval)
-> Max Alloc is Maximum PGA Allocation size at snapshot time
-> Hist Max Alloc is the Historical Max Allocation for still-connected processes
-> ordered by Begin/End snapshot, Alloc (MB) desc

                                                            Hist
                                    Avg  Std Dev     Max     Max
               Alloc      Used    Alloc    Alloc   Alloc   Alloc    Num    Num
  Category      (MB)      (MB)     (MB)     (MB)    (MB)    (MB)   Proc  Alloc
- -------- --------- --------- -------- -------- ------- ------- ------ ------
B Freeable     981.6        .0      6.2     23.0     141     N/A    158    158
  Other        647.4       N/A      3.3      5.8      30     102    195    191
  PL/SQL        38.8       7.0       .2       .6       3       3    193    154
  SQL           15.7      10.9       .1       .2       2   1,289    174    128
E SQL        1,953.6   1,939.0      6.7     36.8     488   1,289    291    264
  Other      1,772.1       N/A      5.6     16.7     177     498    315    311
  Freeable     640.8        .0      4.1     14.2      92     N/A    156    156
  PL/SQL        77.0      13.8       .2       .5       3      13    313    313
          -------------------------------------------------------------
```

SGA Memory Summary

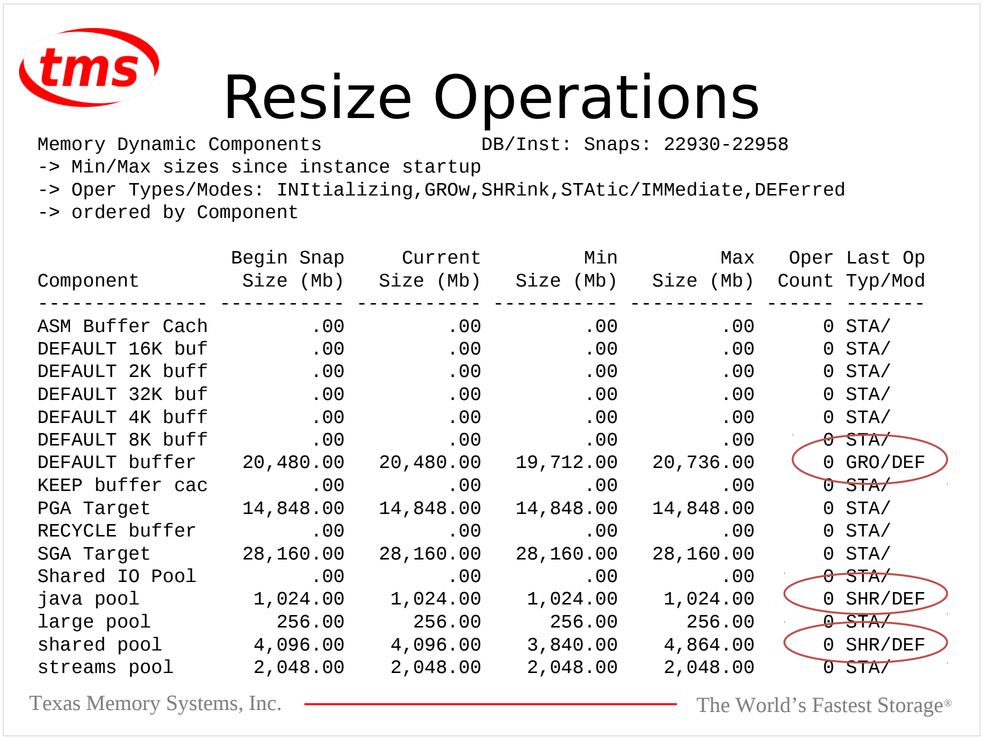
```
SGA Memory Summary                    DB/Inst: Snaps: 84084-84108

                                                   End Size (Bytes)
SGA regions                     Begin Size (Bytes)     (if different)
------------------------- ------------------- -------------------
Database Buffers              61,035,511,808      61,538,828,288
Fixed Size                         2,212,832
Redo Buffers                      14,561,280
Variable Size                  7,667,190,816       7,163,874,336
                          -------------------
sum                           68,719,476,736
          -------------------------------------------------------
```

SGA Breakdown

```
SGA breakdown difference                  DB/Inst: Snaps: 84084-84108
-> ordered by Pool, Name
-> N/A value for Begin MB or End MB indicates the size of that Pool/Name was
   insignificant, or zero in that snapshot

Pool   Name                                 Begin MB         End MB  % Diff
------ ------------------------------ -------------- -------------- -------
java   free memory                             160.0          160.0    0.00
large  free memory                           1,022.6        1,020.6   -0.19
shared CCursor                                 483.3          425.2  -12.04
shared Checkpoint queue                        175.8          175.8    0.00
shared KGH: NO ACCESS                        2,445.0        3,079.3   25.94
shared KQR L PO                                355.6            N/A -100.00
shared KQR M PO                                430.4            N/A -100.00
shared PCursor                                 221.9          190.5  -14.16
shared db_block_hash_buckets                   180.0          180.0    0.00
shared free memory                           2,191.3        2,890.1   31.89
shared kzctxgjsi ksuseclid memor             1,644.5        1,676.7    1.96
shared library cache                           334.2          308.2   -7.79
shared obj stat memo                           466.8          474.8    1.71
shared partitioning d                          131.8            N/A -100.00
shared sql area                                483.6          236.8  -51.04
stream free memory                              31.9           31.9    0.00
       buffer_cache                         58,208.0       58,688.0    0.82
       fixed_sga                                 2.1            2.1    0.00
       log_buffer                               13.9           13.9    0.00
          -------------------------------------------------------------
```

Resize Operations

- Using AMM resize operations are automatic
- If there is room to grow, shouldn’t have to borrow (steal) from other areas
- If SGA_TARGET=SGA_MAX_SIZE or MEMORY_TARGET=MEMORY_MAX_SIZE then no room to grow



Resize Operations

```
Memory Dynamic Components                DB/Inst: Snaps: 22930-22958
-> Min/Max sizes since instance startup
-> Oper Types/Modes: INItializing,GROw,SHRink,STAtic/IMMediate,DEFerred
-> ordered by Component

                 Begin Snap     Current         Min         Max   Oper Last Op
Component        Size (Mb)   Size (Mb)   Size (Mb)   Size (Mb)   Count Typ/Mod
--------------- ----------- ----------- ----------- ----------- ------ -------
ASM Buffer Cach         .00         .00         .00         .00      0 STA/
DEFAULT 16K buf         .00         .00         .00         .00      0 STA/
DEFAULT 2K buff         .00         .00         .00         .00      0 STA/
DEFAULT 32K buf         .00         .00         .00         .00      0 STA/
DEFAULT 4K buff         .00         .00         .00         .00      0 STA/
DEFAULT 8K buff         .00         .00         .00         .00      0 STA/
DEFAULT buffer    20,480.00   20,480.00   19,712.00   20,736.00      0 GRO/DEF
KEEP buffer cac         .00         .00         .00         .00      0 STA/
PGA Target        14,848.00   14,848.00   14,848.00   14,848.00      0 STA/
RECYCLE buffer          .00         .00         .00         .00      0 STA/
SGA Target        28,160.00   28,160.00   28,160.00   28,160.00      0 STA/
Shared IO Pool          .00         .00         .00         .00      0 STA/
java pool          1,024.00    1,024.00    1,024.00    1,024.00      0 SHR/DEF
large pool           256.00      256.00      256.00      256.00      0 STA/
shared pool        4,096.00    4,096.00    3,840.00    4,864.00      0 SHR/DEF
streams pool       2,048.00    2,048.00    2,048.00    2,048.00      0 STA/
```

V$SGA_RESIZE_OPS -10g V$MEMORY_RESIZE_OPS – 11g

- A dynamic performance view
- Use to get details of resize operations
- Every memory change since startup

V$SGA_RESIZE_OPS

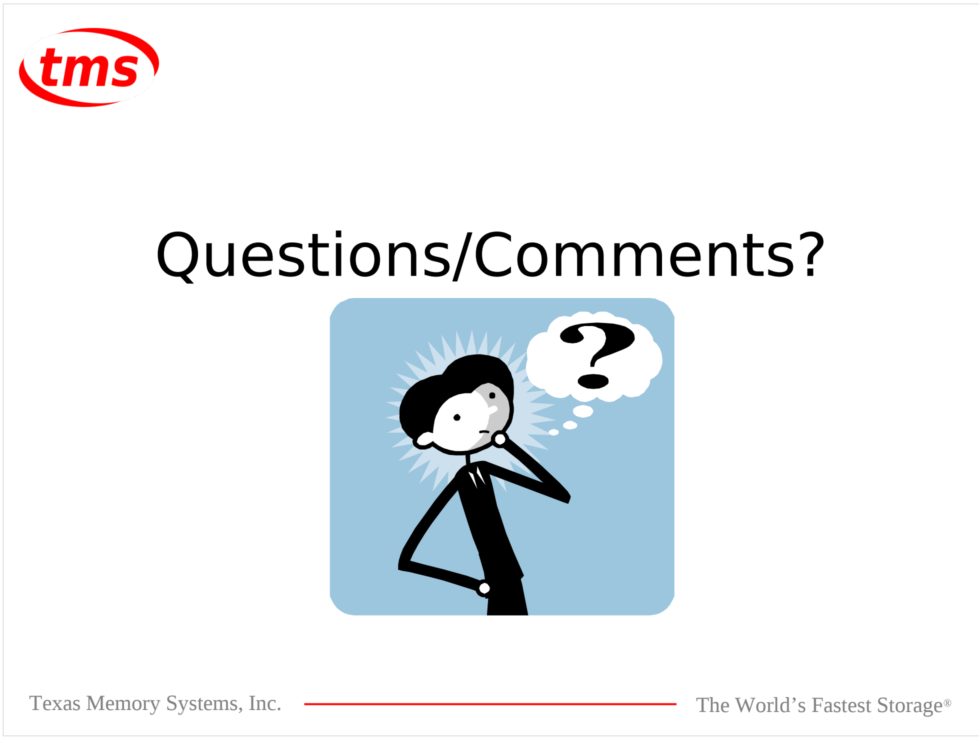
```
Date: 02/18/08                                                                                           Page:   1
Time: 01:04 PM                                   Component Resize Operations                                  SYSTEM
                                                     ault10g1 database

COMPONENT            Oper   OPER_MODE INITIAL_SIZE TARGET_SIZE FINAL_SIZE STATUS    START_TIME   END_TIME
-------------------- ------ --------- ------------ ----------- ---------- --------- ------------ ------------
shared pool          SHRINK DEFERRED     318767104   301989888  301989888 COMPLETE  0217 15:23   0217 15:23
DEFAULT buffer cache GROW   DEFERRED    1040187392  1056964608 1056964608 COMPLETE  0217 15:23   0217 15:23
shared pool          SHRINK DEFERRED     301989888   285212672  285212672 COMPLETE  0217 15:24   0217 15:24
DEFAULT buffer cache GROW   DEFERRED    1073741824  1090519040 1090519040 COMPLETE  0217 15:24   0217 15:24
shared pool          SHRINK DEFERRED     285212672   268435456  268435456 COMPLETE  0217 15:24   0217 15:24
DEFAULT buffer cache GROW   DEFERRED    1056964608  1073741824 1073741824 COMPLETE  0217 15:24   0217 15:24
shared pool          SHRINK DEFERRED     268435456   251658240  251658240 COMPLETE  0217 15:25   0217 15:25
DEFAULT buffer cache GROW   DEFERRED    1090519040  1107296256 1107296256 COMPLETE  0217 15:25   0217 15:25
shared pool          SHRINK DEFERRED     251658240   234881024  234881024 COMPLETE  0217 15:28   0217 15:28
DEFAULT buffer cache GROW   DEFERRED    1107296256  1124073472 1124073472 COMPLETE  0217 15:28   0217 15:28
shared pool          SHRINK DEFERRED     234881024   218103808  218103808 COMPLETE  0217 15:28   0217 15:28
```

Parameters

```
init.ora Parameters                 DB/Inst:   Snaps: 84084-84108
                                                          End value
Parameter Name                Begin value                  (if different)
----------------------------- --------------------------- ---------
_spin_count                   2000
cursor_sharing                EXACT
db_block_size                 16384
db_cache_size                 46170898432
db_keep_cache_size            5368709120
db_name                       cc1
db_writer_processes           4
java_pool_size                167772160
large_pool_size               1073741824
olap_page_pool_size           33554432
open_cursors                  900
pga_aggregate_target          5368709120
processes                     700
sessions                      1000
sga_max_size                  68719476736
sga_target                    68719476736
shared_pool_reserved_size     262144000
shared_pool_size              4294967296
          -------------------------------------------------------------
```

Quick Word on RAC

- The TCP buffers are the biggest memory issue with RAC performance, size them big
- The global dictionary takes memory out of the shared pool, the more memory in the db caches, the more is needed in the shared pools to track it.



Questions/Comments?

Texas Memory Systems, Inc. The World's Fastest Storage®

Thank You!

Mike Ault

Oracle@RamSan.com

www.ramsan.com

www.statspackanalyzer.com