Oracle Database Performance Tuning:
The Not SQL Option

Daniel A. Morgan

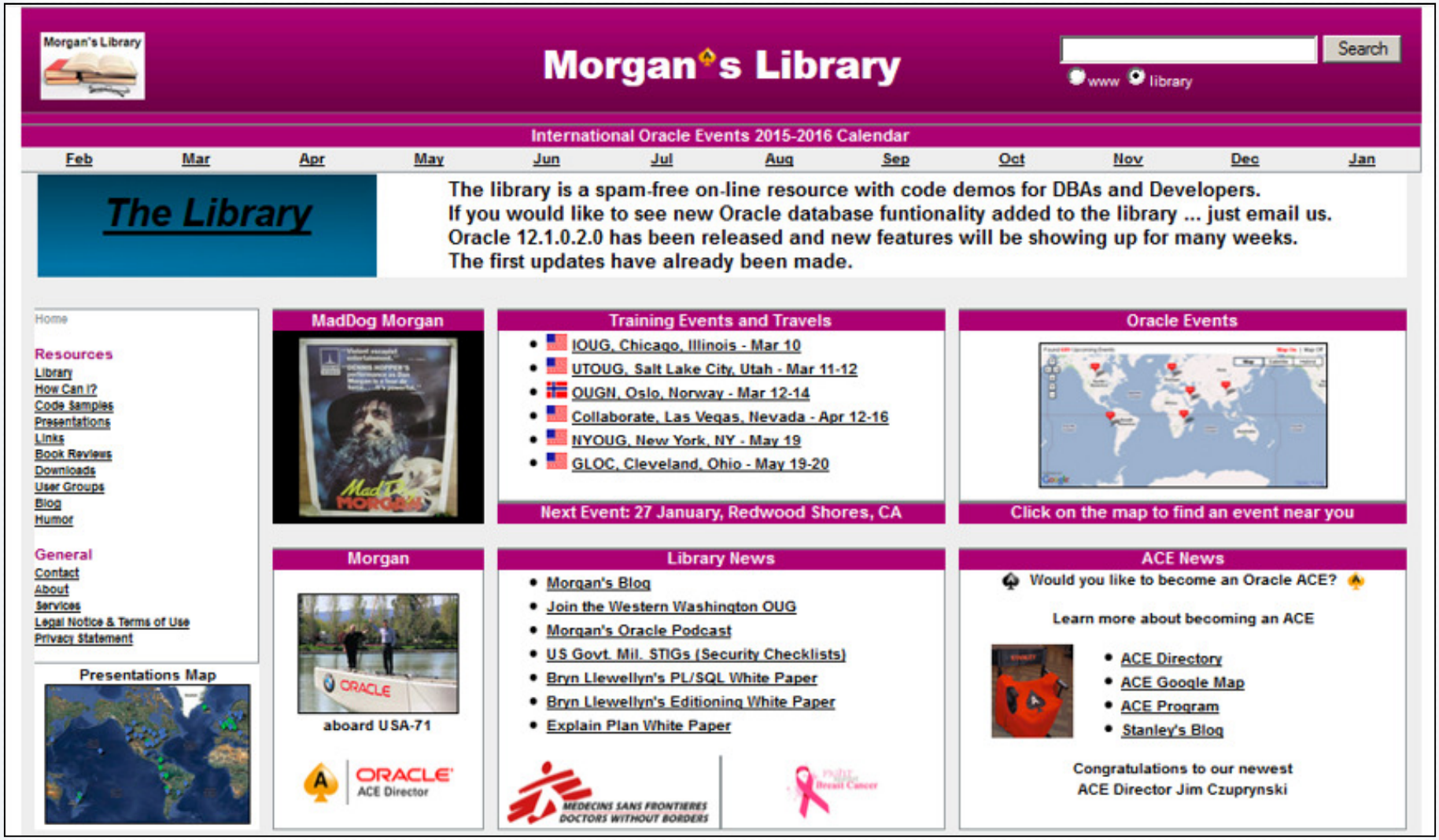
21 August, 2015

Daniel A. Morgan

- Oracle ACE Director
- Educator
  - Author and Primary Instructor: University of Washington Oracle program
  - Oracle Consultant: Harvard University
    - University Guest Lecturers
      - APAC: University of Canterbury (NZ)
      - EMEA: University of Oslo (Norway)
      - Latin America: Universidad Latina de Panama and Technologico de Costa Rica
- The Morgan behind Morgan's Library on the web
  www.morganslibrary.org
- 10g, 11g, and 12c Beta tester
- Co-Founder International GoldenGate Oracle Users Group
- Principal Advisor: Forsythe Meta7

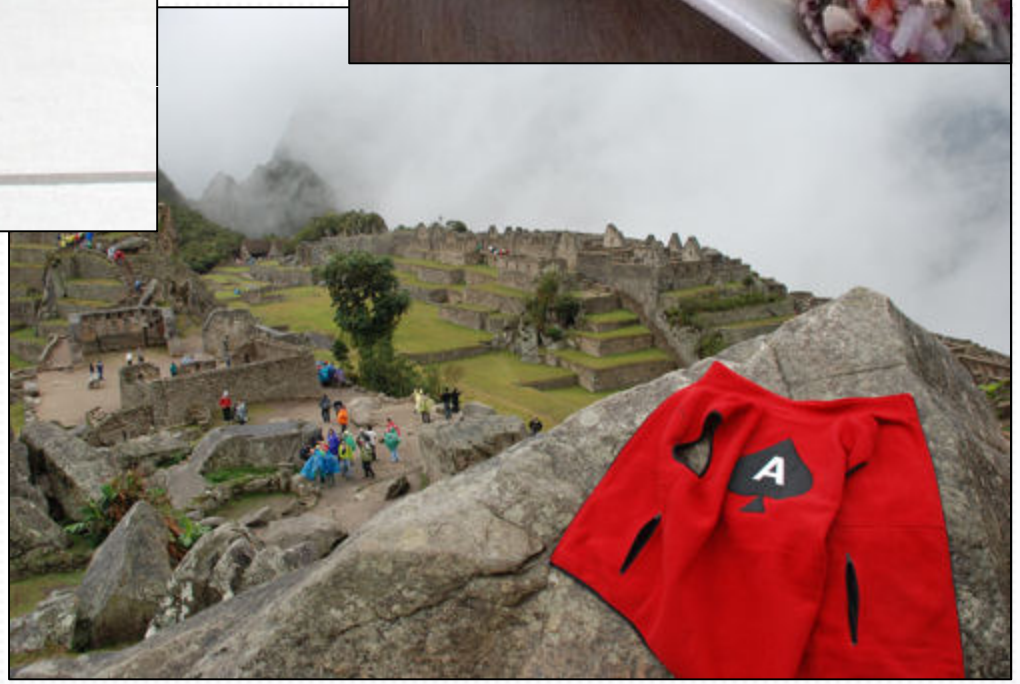
Official Beta Site
ORACLE 12c DATABASE

The Morgan's Library Web Site



Travel Log: OTN Latin America Tour 2009

Travel Log: OTN Latin America Tour 2010

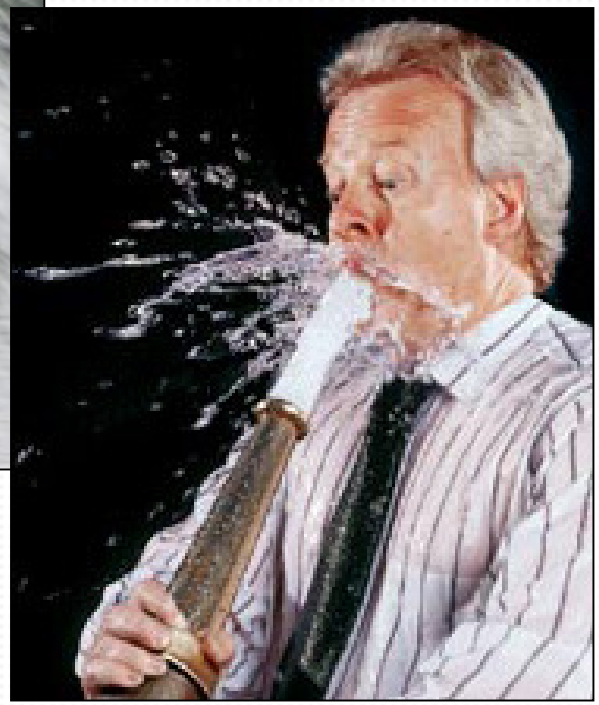


Travel Log: OTN Latin America Tour 2014

Travel Log: OTN Latin America Tour 2014

$ cd $MORGAN_HOME

Content Density Warning



Take Notes ... Ask Questions

The #1 Database Performance Tuning Mythology

Instance Efficiency Percentages (Target 100%)

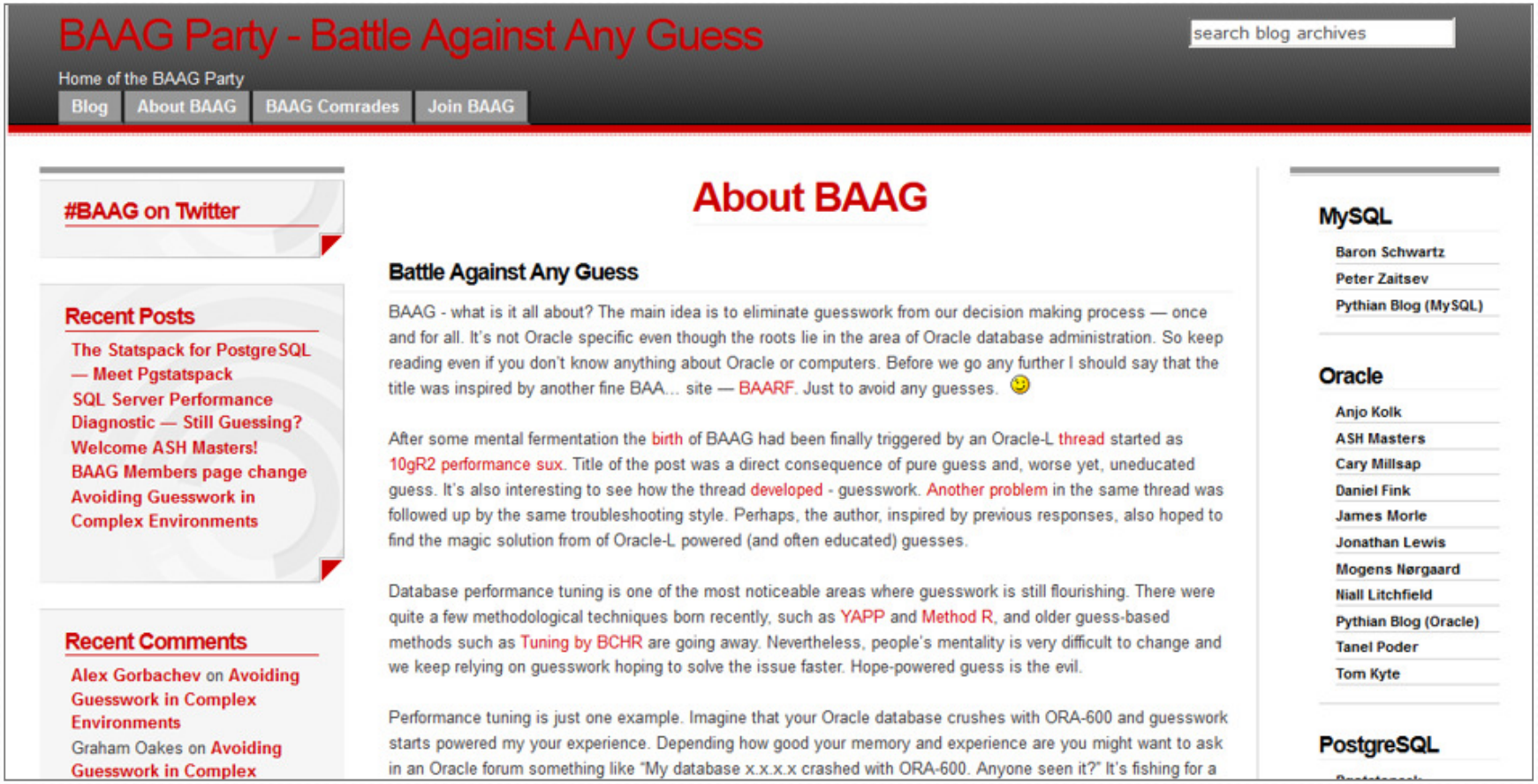
Buffer Nowait %:	99.80	Redo NoWait %:	100.00
Buffer Hit %:	97.34	In-memory Sort %:	99.99
Library Hit %:	99.97	Soft Parse %:	98.79
Execute to Parse %:	99.29	Latch Hit %:	99.57
Parse CPU to Parse Elapsd %:	0.00	% Non-Parse CPU:	96.60

This was from the worst performing Oracle Database I've seen since 2006

The #1 Database Performance Tuning Methodology

Guessing

One Solution



BAAG Comrades

7-Sep-08	Mohammad Illiyaz http://	BAAG Member	Because i also want to BATTLE AGAINST ANY GUESSES
22-Sep-08	Ken Jordan http://	BAAG Member	Seeking THE truth!
27-Sep-08	Martin Berger http://berx.at	BAAG Member	I travel on a sea of wild guesses, with some islands of educated guesses. Knowledge is the stars which help me traveling on these seas. I need more light!
22-Nov-08	Daniel Morgan http://www.morganslibrary.org	BAAG Member	Too often a conclusion is simply the place where someone became tired of thinking. Then the "answer" becomes a guess, a prejudice, or a retreat to mythology. I prefer the application of synapses.
11-Feb-09	Fairlie Rego http://www.el-caro.blogspot.com	BAAG Member	becoz I have had enuff
3-Mar-09	Olivier NOEL http://	BAAG Member	Help me in my daily struggle against Oracle
16-Mar-09	Michael Erwin http://www.oracle.com/consulting/technology/grid-computing.html	BAAG Member	I like sharing what I know as actual fact vs. guessing as well.
18-Mar-09	Randolf Geist http://oracle-randolf.blogspot.com/	BAAG Member	I guess (!) it's time to join.
1-Apr-09	Kevin Fries http://	BAAG Member	I guess it cannot make things worse, I've tried everything else and the Magic 8-Ball broke.
13-Apr-09	Henrik Harsfort http://	BAAG Member	to guess is to not know.
24-Apr-09	Paul Clare http://	BAAG Member	Creative problem solving is evolving into an art form.

BAAG Membership

Christian Antognini
Karl Arao
Mark Bobak
Ronald Bradford
Wolfgang Breitling
Doug Burns
Andrew Clarke
Randolf Geist
Alex Gorbachev
Marco Gralike
Frits Hoogland
John Hurley
Anjo Kolk
David Kurtz
Jonathan Lewis
Robyn Sands
Jared Still
Jeremiah Wilton

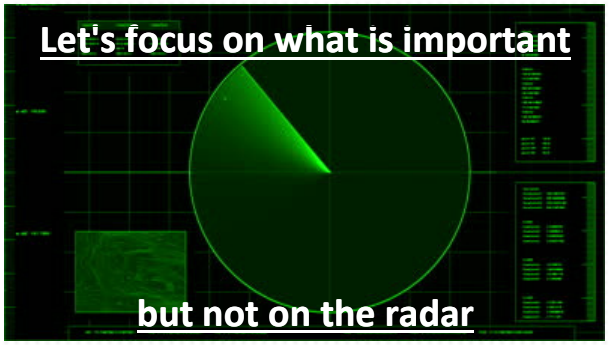
Clearly

- No surprise ... I do not endorse guessing
- Possibly a big surprise ... I do not like normal AWR reports
- I think the solutions to every performance issue is an Exadata
- And I am not attracted to all the tools with pretty GUIs
- So let us take a deep dive into performance tuning and address the root cause of the majority of issues I see in my work
- We will focus on what fixes all issues ... not just one issue
- We all know that 30+ years of doing it the way we have been
  - DBMS_SUPPORT introduced with version 7.2
  - DBMS_TRACE introduced with version 8.1.5
  - DBMS_MONITOR introduced with version 10.1
  - 10053 and 10046 traces and TKPROF

has not eliminated tuning problems

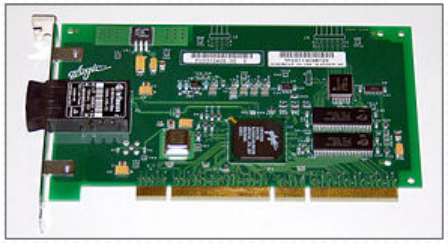
What Affects Performance

- Hardware
  - Servers Resources
    - CPU
    - Memory
    - Bus Bandwidth and Latency
  - Storage Subsystems
  - Networks
- Software
  - Operating System Configuration
  - Virtual Machines
  - Drivers
- Database
  - Memory Allocation
  - Optimizer Configuration
  - SQL Quality
- Application
  - Web Servers
  - Application Servers
  - Middleware Caching
  - Application Code Quality



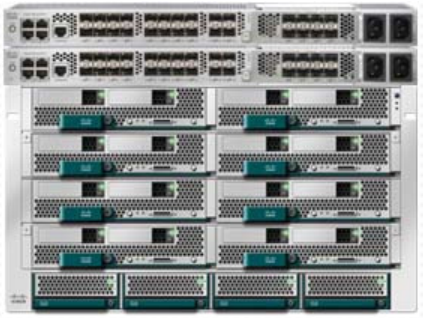
Hardware

- Servers and Operating Systems
  - Blade Servers
  - I/O Cards
  - NUMA Architecture
  - HugePages
  - Swapiness
- Virtual Machines
- Storage
  - Controllers
  - Read-Write Caches
  - LUN Size and Layout
- Networks
  - TCP/IP
  - UDP



Blade Servers (1:2)

- Suitable for web and application servers
- Possibly usable for small databases
- A meltdown waiting to happen with RAC



RAC: Frequently Asked Questions (Doc ID 220970.1)

Cluster interconnect network separation can be satisfied either by using standalone, dedicated switches, which provide the highest degree of network isolation, or Virtual Local Area Networks defined on the Ethernet switch, which provide broadcast domain isolation between IP networks. VLANs are fully supported for Oracle Clusterware interconnect deployments. Partitioning the Ethernet switch with VLANs allows for:
- Sharing the same switch for private and public communication.
- Sharing the same switch for the private communication of more than one cluster.
- Sharing the same switch for private communication and shared storage access.

The following best practices should be followed:

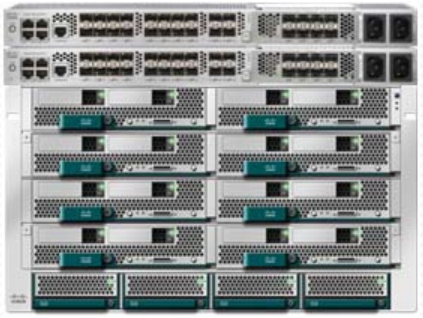
The Cluster Interconnect VLAN must be on a non-routed IP subnet.

All Cluster Interconnect networks must be configured with non-routed IPs. The server-server communication should be single hop through the switch via the interconnect VLAN. There is no VLAN-VLAN communication.

Oracle recommends maintaining a 1:1 mapping of subnet to VLAN.

The most common VLAN deployments maintain a 1:1 mapping of subnet to VLAN. It is strongly recommended to avoid multi-subnet mapping to a single VLAN. Best practice recommends a single access VLAN port configured on the switch for the cluster interconnect VLAN. The server side network interface should have access to a single VLAN.

Blade Servers (2:2)



- Blade servers, of which Cisco UCS is but one example, do not have sufficient independent network cards to avoid the networking becoming a single point of failure
- It is good when the public interface has a "keep alive" enabled but this is a fatal flaw for the cluster interconnect as fail-over will be delayed
- When different types of packets, public, storage, and interconnect are mixed low-level diagnostics are difficult ... if not impossible
- When different types of packets, public, storage, and interconnect are mixed the latency of one is the latency of all
- Traffic from any one blade can impact all blades

Troubleshooting gc block lost and Poor Network Performance in a RAC Environment (Doc ID 563566.1)

6. Interconnect LAN non-dedicated
Description: **Shared public IP traffic and/or shared NAS IP traffic, configured on the interconnect LAN will result in degraded application performance, network congestion and, in extreme cases, global cache block loss.**

Action: **The interconnect/clusterware traffic should be on a dedicated LAN defined by a non-routed subnet**. Interconnect traffic should be isolated to the adjacent switch(es), e.g. interconnect traffic should not extend beyond the access layer switch(es) to which the links are attached. **The interconnect traffic should not be shared with public or NAS traffic**. If Virtual LANs (VLANS) are used, the interconnect should be on a single, dedicated VLAN mapped to a dedicated, non-routed subnet, which is isolated from public or NAS traffic.

I/O (1:3)

- Not all HBA cards are the same
- NIC cards vary widely in capabilities and performance
  - TCP/IP Off-loading
  - Kernel Optimization of the TCP/IP stack

```
--enable TCP kernel auto-tuning
/proc/sys/net/ipv4/tcp_moderate_rcvbuf (1=on)

-- tune TCP max memory: tune to 2xBDP (Bandwidth x Delay Product)
-- For example, with 40 Mbits/sec bandwidth, 25 msec delay,
-- BDP = (40 x 1000 / 8 Kbytes/sec) x (0.025 sec) ~ 128 Kbytes
/proc/sys/net/ipv4/tcp_rmem and tcp_wmem 4096 87380 174760

-- tune the socket buffer sizes by setting to 2xBDP
/proc/sys/net/core/rmem_max and wmem_max

-- ensure that TCP Performance features are enabled
/proc/sys/net/ipv4/tcp_sack
/proc/sys/net/ipv4/tcp_window_scaling
/proc/sys/net/ipv4/tcp_timestamps

-- additionally be sure NIC cards have TCP off-loading capability
```

I/O (2:3)

- Optimize SQL*Net

```
--sqlnet.ora
NAMES.DIRECTORY_PATH= (TNSNAMES, EZCONNECT)
DEFAULT_SDU_SIZE=32767

-- listener.ora
DGLOGSHIPB =
  (DESCRIPTION =
    (SDU = 32767)
    (SEND_BUF_SIZE=9375000)
    (RECV_BUF_SIZE=9375000)
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = 10.0.7.2)(PORT = 1526))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = prodb)
    )
  )
```

more examples: www.morganslibrary.org/reference/data_guard.html

I/O (3:3)

- Optimize for RAC
  - Read the Oracle installation documents with very careful attention to the advice given for kernel parameters
  - If on Linux and you don't know what rmem and wmem are ... read the docs
  - If on Solaris and you don't know what rsize and wsize are ... read the docs

NUMA Memory Allocation

- Non-Uniform Memory Access
  - A memory design used in multiprocessing, where the memory access time depends on the memory location relative to the processor
  - A processor can access its own local memory faster than non-local memory
  - The benefits of NUMA are limited to particular workloads, notably on servers where the data are often associated strongly with certain tasks or users

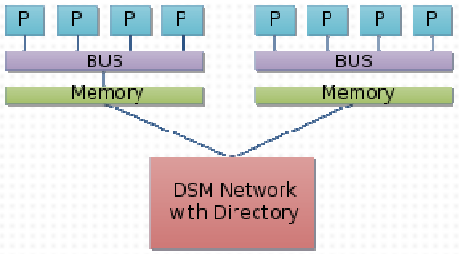


Diagram Source: Wikipedia

Detect NUMA Use

NUMA Not Configured

```
[oracle@alpha1 ~]$ numactl --hardware
available: 1 nodes (0)
node 0 size: 4095 MB
node 9 free: 648MB
node distances:
node   0
  0:  10

[oracle@alpha1 ~]$ numactl --show
policy: default
preferred node: current
physcpubind: 0
nodebind: 0
membind: 0
```

NUMA Configured

```
[dmorgan@lxorap1n5 ~]$ numactl --hardware
available: 2 nodes (0-1)
node 0 size: 48457 MB
node 0 free: 269 MB
node 1 size: 48480 MB
node 1 free: 47 MB
node distances:
node   0   1
  0:  10  20
  1:  20  10

[dmorgan@lxorap1n5 ~]$ numactl --show
policy: default
preferred node: current
physcpubind: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
cpubind: 0 1
nodebind: 0 1
membind: 0 1
```

more examples: www.morganslibrary.org/reference/numa.html

Is Your Database NUMA Aware?

```
SQL> SELECT a.ksppinm PNAME, c.ksppstvl PVAL, a.ksppdesc PDESC
  2  FROM x$ksppi a, x$ksppcv b, x$ksppsv c
  3  WHERE a.indx = b.indx
  4  AND a.indx = c.indx
  5  AND LOWER(a.ksppinm) LIKE '%numa%'
  6* ORDER BY 1;

PNAME                     PVAL          PDESC
------------------------- ------------- ------------------------------------------------------------
_NUMA_instance_mapping    Not specified Set of nodes that this instance should run on
_NUMA_pool_size           Not specified aggregate size in bytes of NUMA pool
_db_block_numa            1             Number of NUMA nodes
_enable_NUMA_interleave   TRUE          Enable NUMA interleave mode
_enable_NUMA_optimization FALSE         Enable NUMA specific optimizations
_enable_NUMA_support      FALSE         Enable NUMA support and optimizations
_numa_buffer_cache_stats  0             Configure NUMA buffer cache stats
_numa_shift_enabled       TRUE          Enable NUMA shift
_numa_shift_value         0             user defined value for numa nodes shift
_numa_trace_level         0             numa trace event
_px_numa_stealing_enabled TRUE          enable/disable PQ granule stealing across NUMA nodes
_px_numa_support_enabled  FALSE         enable/disable PQ NUMA support
_rm_numa_sched_enable     FALSE         Is Resource Manager (RM) related NUMA scheduled policy enabled
_rm_numa_simulation_cpus  0             number of cpus for each pg for numa simulation in resource mgr
_rm_numa_simulation_pgs   0             number of PGs for numa simulation in resource manager
```

more examples: www.morganslibrary.org/reference/numa.html

Enable Database NUMA Support

```
conn / as sysdba

ALTER SYSTEM SET "_enable_NUMA_support" = TRUE
COMMENT= 'NUMA Support Enabled 15-Mar-2015'
CONTAINER=ALL
SCOPE=SPFILE
SID='*';
```

more examples: www.morganslibrary.org/reference/numa.html

HugePages

- Also known as "Large Memory Pages" or just "Large Pages"
- Each page table entry represents a “virtual to physical” translation of a process’s memory
- Can be as large as 64 KB in size per entry
- Can be huge for large memory systems
  - See PageTables in /proc/meminfo
  - As large a 1.5 GB
- The entire SGA must fit inside the HugePages. If it does not fit none of it will use the HugePage memory and you will essentially have walled your database off from using a large portion of the server's memory

Swapiness

- Swapping (aka Paging)
- In a sense the operating system's version of the Oracle Temp tablespace

Value	Strategy
vm.swapiness=0	The kernel will swap only to avoid an out of memory condition
vm.swapiness=60	The default value
vm.swapiness=100	The kernel will swap aggressively

Virtual Machines

- Leave sufficient cpu resources for the bare-metal operating system to perform I/O and manage network traffic
- Disable interrupt coalescing
- Disable chipset power management
- Read the docs
  http://www.vmware.com/pdf/Perf_Best_Practices_vSphere5.0.pdf
- VMs on NUMA machines should be configured to enhance memory allocation
  - This example is from vSphere where 0 and 1 are the processor sockets

```
numa.nodeAffinity=0,1
```

Storage Hardware

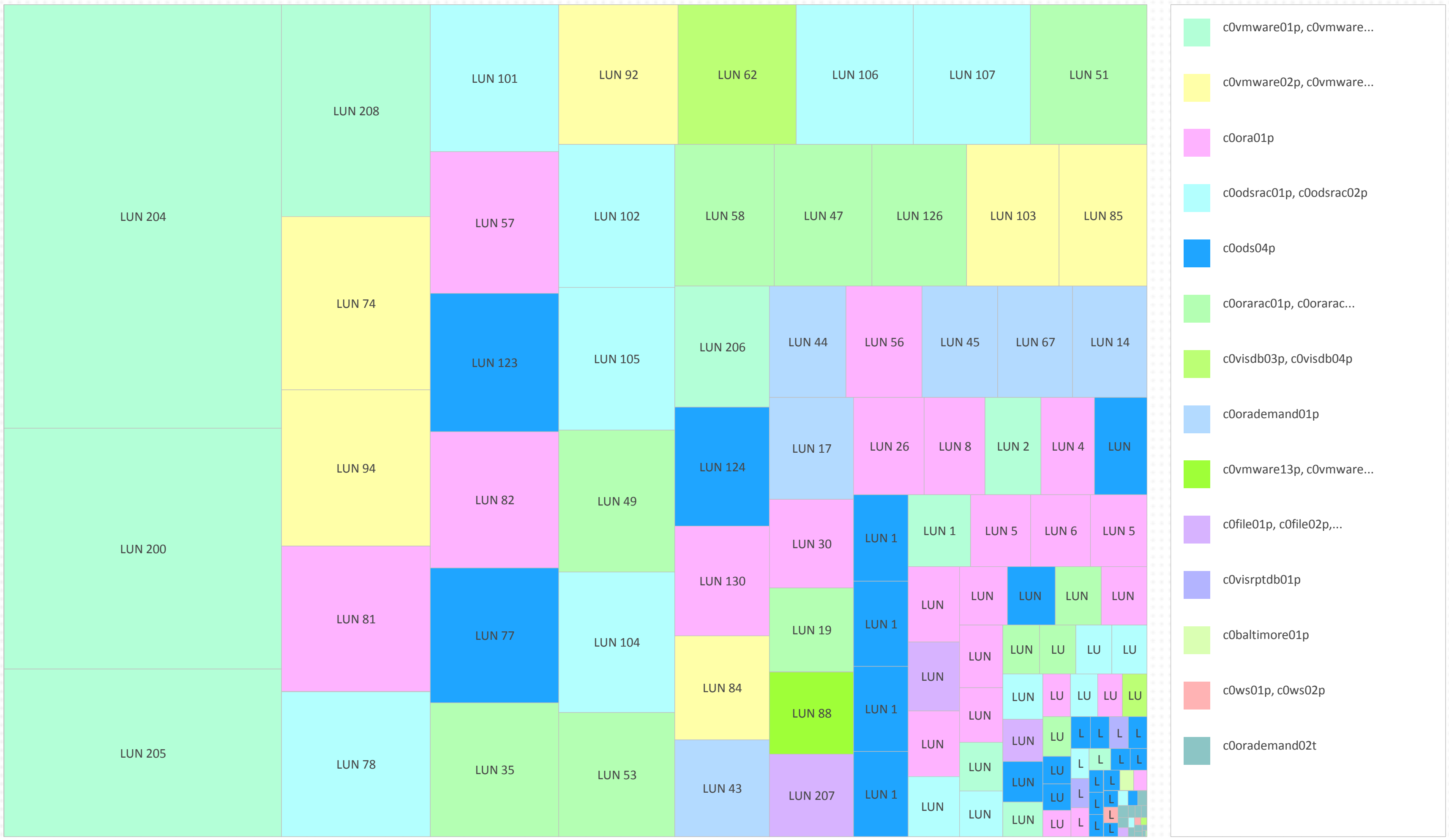
- Storage
  - Spinning Disk
  - Solid State Devices
  - Controllers
  - SAN Switches

Device	Type	IOPS	Interface	Notes
7,200 rpm SATA drives	HDD	~75-100 IOPS[2]	SATA 3 Gbit/s	
10,000 rpm SATA drives	HDD	~125-150 IOPS[2]	SATA 3 Gbit/s	
10,000 rpm SAS drives	HDD	~140 IOPS[2]	SAS	
15,000 rpm SAS drives	HDD	~175-210 IOPS[2]	SAS	

Kaminario K2	Flash/DRAM /Hybrid SSD	Up to 1,200,000 IOPS SPC-1 IOPS with the K2-D (DRAM)[41][42]	FC	
NetApp FAS6240 cluster	Flash/Disk	1,261,145 SPECsfs2008 nfsv3 IOPs using 1,440 15K disks, across 60 shelves, with virtual storage tiering.[43]	NFS, CIFS, FC, FCoE, iSCSI	SPECsfs2008 is the latest version of the Standard Performance Evaluation Corporation benchmark suite measuring file server throughput and response time, providing a standardized method for comparing performance across different vendor platforms. http://www.spec.org/sfs2008.
Fusion-io ioDrive2	SSD	Up to 9,608,000 IOPS[44]	PCIe	Only via demonstration so far.

Storage Heat Map

Storage Layout



Object Optimizations

- Oracle's default tablespace settings waste disk and create unnecessary I/O
  - Often a smallfile tablespace is a bad choice
- Default table creations are almost always inefficient
  - Often heap tables are a bad choice
- Not every index should be a default B*Tree
  - And poor choices lead to poor performance
- Oracle sequence object defaults are almost always inefficient
  - A default cache size of 20 is 99.9% of the time a bad choice
- The vast majority of PL/SQL I see written would have been valid in version 7.3.4
  - I've almost never seen constants declared properly
- Oracle optimizes data dictionary performance by clustering tables ... do you?

I'm Not Afraid To Show You Mine

Page One

Complete List of SQL Text

SQL Id	SQL Text
0cjpvvqy854kh	SELECT 'ERICSSON_CSCORE.EOS' AS TABLENAME, sunday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from (select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then MSC end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then MSC end) ontime_count, count (*) num_of_rows from ERICSSON_CSCORE.EOS where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60)rawt, (select max(datetime)sunday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then MSC end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(s ysdate)+7/24 then MSC end) summarised_ontime_count, sum(entries) num_of_sumrows from ERICSSON_CSCORE.EOS_DY where datetime = trunc(sysdate)-1)sumt union all SELECT 'ERICSSON_CSCORE.MSCSTATS' AS TABLENAME, sunday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from (select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then MSC end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then MSC end) ontime_count, count (*) num_of_rows from ERICSSON_CSCORE.MSCSTATS where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60)rawt, (select max(datetime)sunday, count(dist inct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then MSC end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then MSC end) summarised_ontime_count, sum(entries) num_of_sumrows from ERICSSON_CSCORE.MSCSTATS_DY where datetime = trunc(sysdate)-1)sumt union all SELECT 'ERICSSON_CSCORE.MTRAFTYPE' AS TABLENAME, sunday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from (select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then MSC end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then MSC end) ontime_count , count (*) num_of_rows from ERICSSON_CSCORE.MTRAFTYPE where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60)rawt, (select max(datetime)sunday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then MSC end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then MSC end) summarised_ontime_count, sum(entries) num_of_sumrows from ERICSSON_CSCORE.MTRAFTYPE_DY where datetime = trunc(sysdate)-1)sumt union all SELECT 'ERICSSON_GERAN.CELLGPRS' AS TABLENAME, sunday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from (select count(distinct case when trunc(d atetime_ins, 'mi') > trunc(sysdate)+3/24 then BSC end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then BSC end) ontime_count, count (*) num_of_rows from ERICSSON_GERAN.CELLGPRS where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60)rawt, (select max(datetime)sunday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then BSC end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then BSC end) summarised_ontime_count, sum(entries) num_of_sumrows from ERICSSON_GERAN.CELLGPRS_DY where datetime = trunc(sysdate)-1)sumt union all SELECT 'ERICSSON_GERAN.CELLSTATS' AS TABLENAME, sunday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_ count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from (select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then BSC end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then BSC end) ontime_count, count (*) num_of_rows from ERICSSON_GERAN.CELLSTATS where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60)rawt, (select max(datetime)sunday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then BSC end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then BSC end) summarised_ontime_count, sum(entries) num_of_sumrows from ERICSSON_GERAN.CELLSTATS_DY where datetime = trunc(sysdate)-1)sumt union all SELECT 'ERICSSON_PSCORE.MM' AS TABLENAME, sunday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count,

Page Two

```
(ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count,
2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count curr ent_raw_elements,
summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff,
percent(num_of_sumrows, num_of_rows, 2) accuracy from ( select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then SGSNID end) late_count, count(distinct case when trunc(datetime_ins, 'mi') <
trunc(sysdate)+3/24 then SGSNID end) ontime_count, count (*) num_of_rows from ERICSSON_PSCORE.MM where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60 )rawt, ( select max(datetime)sumday, count(distinct case
when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then SGSNID end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then SGSNID end) summarised_ontime_count, sum(entries)
num_of_sumrows from ERICSSON_PSCORE.MM_DY where datetime = trunc(sysdate)-1 )sumt union all SELECT 'ERICSSON_PSCORE.SGSNSTAT' AS TABLENAME, sumday, CASE WHEN percent(summarised_ontime_count,
(ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0
else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+la te_count, 2)percentage_of_total_by5am,
percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count
current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2)
accuracy from ( select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then SGSNID end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then SGSNID end) ontime_count,
count (*) num_of_rows from ERICSSON_PSCORE.SGSNSTAT where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60 )rawt, ( select max(datetime)sumday, count(distinct case when trunc(datetime_ins, 'mi') >
trunc(sysdate)+7/24 then SGSNID end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then SGSNID end) summarised_ontime_count, sum(entries) num_of_sumrows from
ERICSSON_PSCORE.SGSNSTAT_DY where datetime = trunc(sysdate)-1 )sumt union all SELECT 'ERICSSON_UTRAN.RBS_CARRIER' AS TABLENAME, sumday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count),
2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count- late_count), 2) < 0 then 0 else
100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am,
percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count
current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2)
accuracy from ( select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then NODEB end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then NODEB end) ontime_count,
count (*) num_of_rows from ERICSSON_UTRAN.RBS_CARRIER where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60 )rawt, ( select max(datetime)sumday, count(distinct case when trunc(datetime_ins, 'mi') >
trunc(sysdate)+7/24 then NODEB end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then NODEB end) summarised_ontime_count, sum(entries) num_of_sumrows from
ERICSSON_UTRAN.RBS_CARRIER_DY where datetime = trunc(sysdate)-1 )sumt union all SELECT 'ERICSSON_UTRAN.RBS_EDCHRES' AS TABLENAME, sumday, CASE WHEN percent(summarised_ontime_count, (ontime_count-l
ate_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else
100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am,
percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count
current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2)
accuracy from ( select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then NODEB end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then NODEB end) ontime_count,
count (*) num_of_rows from ERICSSON_UTRAN.RBS_EDCHRES where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60 )rawt, ( select max(datetime)sumday, count(distinct case when trunc(datetime_ins, 'mi') >
trunc(sysdate)+7/24 then NODEB end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then NODEB end) summarised_ontime_count, sum(entries) num_of_sumrows from ERICSSON_UT
RAN.RBS_EDCHRES_DY where datetime = trunc(sysdate)-1 )sumt union all SELECT 'ERICSSON_UTRAN.RBS_HSDSCHRES' AS TABLENAME, sumday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100
THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else
100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am,
percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count
current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2)
accuracy from ( select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then NODEB end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then NODEB end) ontime_count,
count (*) num_of_rows from ERICSSON_UTRAN.RBS_HSDSCHRES where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60 )rawt, ( select max(datetime)sumday, count(distinct case when trunc(datetime_ins, 'mi') >
trunc(sysdate)+7/24 then NODEB e nd) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then NODEB end) summarised_ontime_count, sum(entries) num_of_sumrows from
ERICSSON_UTRAN.RBS_HSDSCHRES_DY where datetime = trunc(sysdate)-1 )sumt union all SELECT 'ERICSSON_UTRAN.RNC_IURLINK' AS TABLENAME, sumday, CASE WHEN percent(summarised_ontime_count, (ontime_count-
late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else
100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am,
```

Page Three

```
percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count
current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2)
accuracy from ( select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then RNC end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then RNC end) ontime_count, count (*)
num_of_rows from ERICSSON_UTRAN.RNC_IURLINK where dateti me between trunc(sysdate)-1 and trunc(sysdate)-1/24/60 )rawt, ( select max(datetime)sunday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24
then RNC end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then RNC end) summarised_ontime_count, sum(entries) num_of_sumrows from ERICSSON_UTRAN.RNC_IURLINK_DY
where datetime = trunc(sysdate)-1 )sumt union all SELECT 'ERICSSON_UTRAN.RNC_RNCFUNC' AS TABLENAME, sunday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE
percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count,
(ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count,
ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count
SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from ( select count(distinct case when trunc(datetime_ins, 'mi') >
trunc(sysdate)+3/24 then RNC end) late_c ount, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then RNC end) ontime_count, count (*) num_of_rows from ERICSSON_UTRAN.RNC_RNCFUNC where datetime
between trunc(sysdate)-1 and trunc(sysdate)-1/24/60 )rawt, ( select max(datetime)sunday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then RNC end) summarised_late_count, count(distinct case when
trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then RNC end) summarised_ontime_count, sum(entries) num_of_sumrows from ERICSSON_UTRAN.RNC_RNCFUNC_DY where datetime = trunc(sysdate)-1 )sumt union all SELECT
'ERICSSON_UTRAN.RNC_UCELL' AS TABLENAME, sunday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END
percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed,
percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count
current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-
late_count availabl e_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from ( select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then RNC end) late_count, count(distinct case when
trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then RNC end) ontime_count, count (*) num_of_rows from ERICSSON_UTRAN.RNC_UCELL where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60 )rawt, ( select
max(datetime)sunday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then RNC end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then RNC end)
summarised_ontime_count, sum(entries) num_of_sumrows from ERICSSON_UTRAN.RNC_UCELL_DY where datetime = trunc(sysdate)-1 )sumt union all SELECT 'NORTEL_CSCORE.BASE_OM' AS TABLENAME, sunday, CASE WHEN
percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count,
(ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count,
2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_
late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows,
num_of_rows, 2) accuracy from ( select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then MSC end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then MSC end)
ontime_count, count (*) num_of_rows from NORTEL_CSCORE.BASE_OM where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60 )rawt, ( select max(datetime)sunday, count(distinct case when trunc(datetime_ins, 'mi') >
trunc(sysdate)+7/24 then MSC end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then MSC end) summarised_ontime_count, sum(entries) num_of_sumrows from
NORTEL_CSCORE.BASE_OM_DY where datetime = trunc(sysdate)-1 )sumt union all SELECT 'NORTEL_CSCORE.C7SP' AS TABLENAME, sunday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN
100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else
100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_c
ount+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements,
summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from ( select
count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then MSC end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then MSC end) ontime_count, count (*) num_of_rows from
NORTEL_CSCORE.C7SP where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60 )rawt, ( select max(datetime)sunday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then MSC end)
summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then MSC end) summarised_ontime_count, sum(entries) num_of_sumrows from NORTEL_CSCORE.C7SP_DY where datetime =
trunc(sysdate)-1 )sumt union all SELECT 'NORTEL_CSCORE.CCS7' AS TABLENAME, sunday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count,
(ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end
percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2)
current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM,
late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from ( select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then MSC end)
```

Page Four

late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then MSC end) ontime_count, count (*) num_of_rows from NORTEL_CSCORE.CCS7 where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60)rawt, (select max(datetime)sunday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then MSC end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then MSC end) summarised_ontime_count, sum(entries) num_of_sumrows from NORTEL_CSCORE.CCS7_DY where datetime = trunc(sysdate)-1)sumt union all SELECT 'NORTEL_CSCORE.LOCATIONAREACODE' AS TABLENAME, sunday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cut off, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from (select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then MSC end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then MSC end) ontime_count, count (*) num_of_rows from NORTEL_CSCORE.LOCATIONAREACODE where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60)rawt, (select max(datetime)sunday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then MSC end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then MSC end) summarised_ontime_count, sum(entries) num_of_sumrows from NORTEL_CSCORE.LOCATIONAREACODE_DY where datetime = trunc(sysdate)-1)sumt union all SELECT 'NORTEL_CSCORE.MSCCP_CP' AS TABLENAME, sumd ay, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from (select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then MSC end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then MSC end) ontime_count, count (*) num_of_rows from NORTEL_CSCORE.MSCCP_CP where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60)rawt, (select max(datetime)sunday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then MSC end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then MSC end) summarised_ontime_c ount, sum(entries) num_of_sumrows from NORTEL_CSCORE.MSCCP_CP_DY where datetime = trunc(sysdate)-1)sumt union all SELECT 'NORTEL_CSCORE.MSCCP' AS TABLENAME, sunday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from (select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then MSC end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then MSC end) ontime_count, count (*) num_of_rows from NORTEL_CSCORE.MSCCP where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60)rawt, (select max(datetime)sunday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(s ysdate)+7/24 then MSC end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then MSC end) summarised_ontime_count, sum(entries) num_of_sumrows from NORTEL_CSCORE.MSCCP_DY where datetime = trunc(sysdate)-1)sumt union all SELECT 'NORTEL_CSCORE.TRKGRP' AS TABLENAME, sunday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from (select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then MSC end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then MSC end) ontime_count, count (*) num_of_rows from NORTEL_CSCORE.TRKGRP where dateti me between trunc(sysdate)-1 and trunc(sysdate)-1/24/60)rawt, (select max(datetime)sunday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then MSC end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then MSC end) summarised_ontime_count, sum(entries) num_of_sumrows from NORTEL_CSCORE.TRKGRP_DY where datetime = trunc(sysdate)-1)sumt union all SELECT 'NORTEL_CSCORE.VLR6' AS TABLENAME, sunday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from (select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then MSC end) late_count, count(d istinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then MSC end) ontime_count, count (*) num_of_rows from NORTEL_CSCORE.VLR6 where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60)rawt, (select max(datetime)sunday, count(distinct case

Page Five

```
when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then MSC end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then MSC end) summarised_ontime_count, sum(entries)
num_of_sumrows from NORTEL_CSCORE.VLR6_DY where datetime = trunc(sysdate)-1 )sumt union all SELECT 'NORTEL_PSCORE.GSCGMM' AS TABLENAME, sumday, CASE WHEN percent(summarised_ontime_count, (ontime_count-
late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else
100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am,
percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count
current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumro ws, num_of_rows, 2)
accuracy from ( select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then SGSN end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then SGSN end) ontime_count, count
(*) num_of_rows from NORTEL_PSCORE.GSCGMM where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60 )rawt, ( select max(datetime)sumday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24
then SGSN end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then SGSN end) summarised_ontime_count, sum(entries) num_of_sumrows from NORTEL_PSCORE.GSCGMM_DY where
datetime = trunc(sysdate)-1 )sumt union all SELECT 'NORTEL_PSCORE.GSCSM_ACT' AS TABLENAME, sumday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE
percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count,
(ontime_count-late_count), 2) end percent_of_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count,
ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count
SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from ( select count(distinct case when trunc(datetime_ins, 'mi') >
trunc(sysdate)+3/24 then SGSN end) late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then SGSN end) ontime_count, count (*) num_of_rows from NORTEL_PSCORE.GSCSM_ACT where datetime
between trunc(sysdate)-1 and trunc(sysdate)-1/24/60 )rawt, ( select max(datetime)sumday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then SGSN end) summarised_late_count, count(distinct case when
trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then SGSN end) summarised_ontime_count, sum(entries) num_of_sumrows from NORTEL_PSCORE.GSCSM_ACT_DY where datetime = trunc(sysdate)-1 )sumt union all SELECT
'NORTEL_PSCORE.GSCSM' AS TABLENAME, sumday, CASE WHEN percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END
percent_of_cutoff, case when 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_of_cutoff_not_summed,
percent(summarised_ontime_count, ontime_count+late_count, 2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count
current_raw_elements, summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-
late_count available_at_cutoff, percent(num_of_sumrows, num_of_rows, 2) accuracy from ( select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then SGSN end) late_count, count(distinct case when
trunc(datetime_ins, 'mi') < trunc(sysdate)+3/24 then SGSN end) ontime_count, count (*) num_of_rows from NORTEL_PSCORE.GSCSM where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60 )rawt, ( select
max(datetime)sumday, count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then SGSN end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then SGSN end)
summarised_ontime_count, sum(entries) num_of_sumrows from NORTEL_PSCORE.GSCSM_DY where datetime = trunc(sysdate)-1 )sumt union all SELECT 'NORTEL_PSCORE.GSDSTATS' AS TABLENAME, sumday, CASE WHEN
percent(summarised_ontime_count, (ontime_count-late_count), 2) > 100 THEN 100 ELSE percent(summarised_ontime_count, (ontime_count-late_count), 2) END percent_of_cutoff, case when 100-percent(summarised_ontime_count,
(ontime_count-late_count), 2) < 0 then 0 else 100-percent(summarised_ontime_count, (ontime_count-late_count), 2) end percent_o f_cutoff_not_summed, percent(summarised_ontime_count, ontime_count+late_count,
2)percentage_of_total_by5am, percent(summarised_ontime_count+summarised_late_count, ontime_count+late_count, 2) current_percentage, ontime_count+late_count current_raw_elements,
summarised_ontime_count+summarised_late_count current_summary_elements, summarised_ontime_count-summarised_late_count SUM_ELEMENTS_AT5AM, late_count late_raw_elements, ontime_count-late_count available_at_cutoff,
percent(num_of_sumrows, num_of_rows, 2) accuracy from ( select count(distinct case when trunc(datetime_ins, 'mi') > trunc(sysdate)+3/24 then SGSN end) late_count, count(distinct case when trunc(datetime_ins, 'mi') <
trunc(sysdate)+3/24 then SGSN end) ontime_count, count (*) num_of_rows from NORTEL_PSCORE.GSDSTATS where datetime between trunc(sysdate)-1 and trunc(sysdate)-1/24/60 )rawt, ( select max(datetime)sumday, count(distinct
case when trunc(datetime_ins, 'mi') > trunc(sysdate)+7/24 then SGSN end) summarised_late_count, count(distinct case when trunc(datetime_ins, 'mi') < trunc(sysdate)+7/24 then SGSN end) summarised_ontime_count, sum(entries)
num_of_sumrows from NORTEL_PSCORE.GSDSTATS_DY where datetime = trunc(sysdate)-1 )sumt
```

Optimizer Plans (1:4)

```
SELECT DISTINCT E1_2.OBJECT_ID
   FROM PMCM.ELEMENT_DETAIL E1_1, PMCM.ELEMENT_DETAIL E1_2, PMCM.MARK_NETW_HIERARCHY H1,
        PMCM.ELEMENT_DETAIL E2_1, PMCM.ELEMENT_DETAIL E2_2, PMCM.MARK_NETW_HIERARCHY H2
   WHERE E1_1.OBJECT_ID = H1.PARENT_ID
     AND E1_2.OBJECT_ID = H1.OBJECT_ID
     AND E2_1.OBJECT_ID = H2.PARENT_ID
     AND E2_2.OBJECT_ID = H2.OBJECT_ID
     AND E1_1.CURRENT_IND = 'Y' AND E2_1.CURRENT_IND = 'Y'
     AND E2_1.CURRENT_IND = 'Y' AND E2_2.CURRENT_IND = 'Y'
     AND H1.CURRENT_IND = 'Y' AND H2.CURRENT_IND = 'Y'
     AND H1.HIERARCHY_TYPE = 'NETWORK' AND H2.HIERARCHY_TYPE = 'NETWORK'
     AND H1.PARENT_TYPE IN ('BSC','RNC')  AND H2.PARENT_TYPE IN ('BSC','RNC')
     AND E2_2.ELEMENT_TYPE = 'CELL' AND E1_2.ELEMENT_TYPE = 'CELL'
     AND H1.PARENT_TYPE IN ('BSC','RNC')
     AND E1_1.ELEMENT_NAME = E2_1.ELEMENT_NAME
     AND E1_1.ELEMENT_ID = E2_1.ELEMENT_ID
     AND E1_2.ELEMENT_NAME = E2_2.ELEMENT_NAME
     AND E1_2.ELEMENT_ID = E2_2.ELEMENT_ID
     AND E1_2.USEID LIKE '*%' AND E2_2.USEID NOT LIKE '*%';
```

Id	Operation	Name	Rows	Bytes	TempSpc	Cost (%CPU)	Time	Pstart	Pstop
0	SELECT STATEMENT		1	78		74M (40)	50:54:42		
1	TEMP TABLE TRANSFORMATION								
2	LOAD AS SELECT								
3	PARTITION RANGE ALL		22M	1111M		38153 (11)	00:01:34	1	29
* 4	TABLE ACCESS FULL	ELEMENT_DETAIL	22M	1111M		38153 (11)	00:01:34		
5	LOAD AS SELECT								
6	PARTITION HASH ALL		337K	9231K		3514 (15)	00:00:09	1	16
* 7	TABLE ACCESS FULL	MARK_NETW_HIERARCHY	337K	9231K		3514 (15)	00:00:09		
8	SORT AGGREGATE		1	78					
* 9	HASH JOIN		927G	65T	534M	74M (40)	50:53:00		
10	VIEW		22M	277M		16808 (12)	00:00:42		
11	TABLE ACCESS FULL	SYS_TEMP_0FDA7485F_6A66C42E	22M	1111M		16808 (12)	00		
* 12	HASH JOIN		21G	1272G	534M	1616K (43)	01:06:04		
13	VIEW		22M	277M		16808 (12)	00:00:42		
14	TABLE ACCESS FULL	SYS_TEMP_0FDA7485F_6A66C42E	22M	1111M		16808 (12)	0		
* 15	HASH JOIN		476M	23G	524M	97327 (22)	00:03:59		
* 16	HASH JOIN		10M	401M	8704K	34520 (10)	00:01:25		
* 17	HASH JOIN		234K	5948K	8256K	783 (10)	00:00:02		
18	VIEW		337K	4286K		142 (14)	00:00:01		
19	TABLE ACCESS FULL	SYS_TEMP_0FDA74860_6A66C42E	337K	3956K		142 (14)	00:00:01		
20	VIEW		337K	4286K		142 (14)	00:00:01		
21	TABLE ACCESS FULL	SYS_TEMP_0FDA74860_6A66C42E	337K	3956K		142 (14)	00:00:01		
22	VIEW		22M	277M		16808 (12)	00:00:42		
23	TABLE ACCESS FULL	SYS_TEMP_0FDA7485F_6A66C42E	22M	1111M		16808 (12)	00:00:42		
24	VIEW		22M	277M		16808 (12)	00:00:42		
25	TABLE ACCESS FULL	SYS_TEMP_0FDA7485F_6A66C42E	22M	1111M		16808 (12)	0		

Optimizer Plans (2:4)

```
-----------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                    | Name                          | Rows  | Bytes |TempSpc| Cost (%CPU)| Time     | Pstart| Pstop |
-----------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |                               |     1 |    78 |       |    14T(100)|999:59:59 |       |       |
|   1 |  TEMP TABLE TRANSFORMATION   |                               |       |       |       |            |          |       |       |
|   2 |   LOAD AS SELECT             |                               |       |       |       |            |          |       |       |
|   3 |    PARTITION RANGE ALL       |                               |    22M|  1111M|       | 38153  (11)| 00:01:34 |     1 |    29 |
|*  4 |     TABLE ACCESS FULL        | ELEMENT_DETAIL                |    22M|  1111M|       | 38153  (11)| 00:01:34 |       |       |
|   5 |   LOAD AS SELECT             |                               |       |       |       |            |          |       |       |
|   6 |    PARTITION HASH ALL        |                               |   337K|  9231K|       |  3514  (15)| 00:00:09 |     1 |    16 |
|*  7 |     TABLE ACCESS FULL        | MARK_NETW_HIERARCHY           |   337K|  9231K|       |  3514  (15)| 00:00:09 |       |       |
|   8 |   SORT AGGREGATE             |                               |     1 |    78 |       |            |          |       |       |
|   9 |    MERGE JOIN                |                               |   471P|   15E|       |    14T(100)|999:59:59 |       |       |
|  10 |     MERGE JOIN               |                               |    10P|   616P|       |   694G (81)|999:59:59 |       |       |
|  11 |      MERGE JOIN              |                               |   231T|    10P|       |   377G (64)|999:59:59 |       |       |
|  12 |       SORT JOIN              |                               |   334T|    11P|    28P|   377G (64)|999:59:59 |       |       |
|  13 |        MERGE JOIN CARTESIAN  |                               |   334T|    11P|       |   140G (14)|999:59:59 |       |       |
|* 14 |         HASH JOIN            |                               |   989M|    23G|   534M| 96010  (38)| 00:03:56 |       |       |
|  15 |          VIEW                |                               |    22M|   277M|       | 16808  (12)| 00:00:42 |       |       |
|  16 |           TABLE ACCESS FULL  | SYS_TEMP_0FDA7485B_6A66C42E   |    22M|  1111M|       | 16808  (12)| 00:00:42 |       |       |
|  17 |          VIEW                |                               |    22M|   277M|       | 16808  (12)| 00:00:42 |       |       |
|  18 |           TABLE ACCESS FULL  | SYS_TEMP_0FDA7485B_6A66C42E   |    22M|  1111M|       | 16808  (12)| 00:00:42 |       |       |
|  19 |         BUFFER SORT          |                               |   337K|  4286K|       |   140G (14)|999:59:59 |       |       |
|  20 |          VIEW                |                               |   337K|  4286K|       |   142  (14)| 00:00:01 |       |       |
|  21 |           TABLE ACCESS FULL  | SYS_TEMP_0FDA7485C_6A66C42E   |   337K|  3956K|       |   142  (14)| 00:00:01 |       |       |
|* 22 |       SORT JOIN              |                               |   337K|  4286K|    12M|   844  (14)| 00:00:03 |       |       |
|  23 |        VIEW                  |                               |   337K|  4286K|       |   142  (14)| 00:00:01 |       |       |
|  24 |         TABLE ACCESS FULL    | SYS_TEMP_0FDA7485C_6A66C42E   |   337K|  3956K|       |   142  (14)| 00:00:01 |       |       |
|* 25 |      SORT JOIN               |                               |    22M|   277M|   855M| 65084  (16)| 00:02:40 |       |       |
|  26 |       VIEW                   |                               |    22M|   277M|       | 16808  (12)| 00:00:42 |       |       |
|  27 |        TABLE ACCESS FULL     | SYS_TEMP_0FDA7485B_6A66C42E   |    22M|  1111M|       | 16808  (12)| 0        |       |       |
|* 28 |     SORT JOIN                |                               |    22M|   277M|   855M| 65084  (16)| 00:02:40 |       |       |
|  29 |      VIEW                    |                               |    22M|   277M|       | 16808  (12)| 00:00:42 |       |       |
|  30 |       TABLE ACCESS FULL      | SYS_TEMP_0FDA7485B_6A66C42E   |    22M|  1111M|       | 16808  (12)| 0        |       |       |
-----------------------------------------------------------------------------------------------------------------------------------------
```

Optimizer Plans (3:4)

```
WITH ed AS (SELECT object_id, element_id, element_name, element_type, useid
            FROM pmcm.element_detail
            WHERE element_type = 'CELL'
            AND current_ind = 'Y'),
     mnh AS (SELECT parent_id, object_id
             FROM pmcm.mark_netw_hierarchy
             WHERE current_ind = 'Y'
             AND hierarchy_type = 'NETWORK'
             AND parent_type IN ('BSC', 'RNC'))
SELECT COUNT(*)
FROM ed e1_1, ed e1_2, ed e2_1, ed e2_2, mnh h1, mnh h2
WHERE e1_1.object_id = h1.parent_id AND   e1_2.object_id = h1.object_id
AND   e2_1.object_id = h2.parent_id AND   e2_2.object_id = h2.object_id
AND   e1_1.element_name = e2_1.element_name
AND   e1_1.element_id = e2_1.element_id
AND   e1_2.element_name = e2_2.element_name
AND   e1_2.element_id = e2_2.element_id
AND   e1_2.useid LIKE '*%'
AND   e2_2.useid NOT LIKE '*%';
```

```
-----------------------------------------------------------------------------------------------------------------------
| Id  | Operation                                | Name                 | Rows  | Bytes |TempSpc| Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                         |                      |     1 |   214 |       |   100K  (6)| 00:04:08 |
|   1 |  HASH UNIQUE                             |                      |     1 |   214 |       |   100K  (6)| 00:04:08 |
|*  2 |   HASH JOIN                              |                      |     1 |   214 |    12M|   100K  (6)| 00:04:08 |
|   3 |    PARTITION HASH ALL                    |                      |   337K|  9231K|       |  3514  (15)| 00:00:09 |
|*  4 |     TABLE ACCESS FULL                    | MARK_NETW_HIERARCHY  |   337K|  9231K|       |  3514  (15)| 00:00:00 |
|*  5 |    HASH JOIN                             |                      |   207K|    36M|    22M| 95860   (6)| 00:03:56 |
|   6 |     PARTITION RANGE ALL                  |                      |   586K|    15M|       | 16233   (2)| 00:00:40 |
|   7 |      TABLE ACCESS BY LOCAL INDEX ROWID   | ELEMENT_DETAIL       |   586K|    15M|       | 16233      | ??:??:?? |
|*  8 |       INDEX SKIP SCAN                    | ED_ET_TECH_CI        |   586K|       |       | 12791   (1)| 00:00:3? |
|*  9 |     HASH JOIN                            |                      |   207K|    31M|    22M| 77982   (7)| 00:03:12 |
|  10 |      PARTITION RANGE ALL                 |                      |   586K|    15M|       | 16233   (2)| 00:00:40 |
|  11 |       TABLE ACCESS BY LOCAL INDEX ROWID  | ELEMENT_DETAIL       |   586K|    15M|       | 16233      | ??:??:?? |
|* 12 |        INDEX SKIP SCAN                   | ED_ET_TECH_CI        |   586K|       |       | 12791   (1)| 00:00:?? |
|* 13 |      HASH JOIN                           |                      |   179K|    22M|    12M| 60372   (8)| 00:02:29 |
|  14 |       PARTITION HASH ALL                 |                      |   337K|  9231K|       |  3514  (15)| 00:00:09 |
|* 15 |        TABLE ACCESS FULL                 | MARK_NETW_HIERARCHY  |   337K|  9231K|       |  3514  (15)| 00:00:?? |
|* 16 |       HASH JOIN                          |                      |   184K|    17M|    10M| 55886   (8)| 00:02:18 |
|  17 |        PARTITION RANGE ALL               |                      |   184K|  9008K|       | 37137   (8)| 00:01:32 |
|* 18 |         TABLE ACCESS FULL                | ELEMENT_DETAIL       |   184K|  9008K|       | 37137   (8)| 00:01:32 |
|  19 |        PARTITION RANGE ALL               |                      |   576K|    28M|       | 17383   (8)| 00:00:43 |
|* 20 |         TABLE ACCESS BY LOCAL INDEX ROWID| ELEMENT_DETAIL       |   576K|    28M|       | 17383   (8)| ??:??:?? |
|* 21 |          INDEX SKIP SCAN                 | ED_ET_TECH_CI        |   583K|       |       | 13939   (9)| 00:00:35 |
-----------------------------------------------------------------------------------------------------------------------
```

Optimizer Plans (4:4)

```
-----------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                                  | Name                     | Rows  | Bytes |TempSpc| Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                           |                          |       |       |       |   264T(100)|          |
|*  1 |  VIEW                                      |                          |  156P|    15E|       |   264T (79)|999:59:59 |
|*  2 |   WINDOW SORT PUSHED RANK                  |                          |  156P|    15E|    15E|   264T (79)|999:59:59 |
|   3 |    MERGE JOIN CARTESIAN                    |                          |  156P|    15E|       |    68T (16)|999:59:59 |
|   4 |     MERGE JOIN CARTESIAN                   |                          |  220G|   205T|       |    96M (16)| 26:57:48 |
|   5 |      MERGE JOIN CARTESIAN                  |                          |  310K|   302M|       |   232  (11)| 00:00:01 |
|   6 |       MERGE JOIN CARTESIAN                 |                          |  779 |   777K|       |    22   (0)| 00:00:01 |
|   7 |        NESTED LOOPS                        |                          |       |       |       |            |          |
|   8 |         NESTED LOOPS                       |                          |    2 |  2044 |       |    20   (0)| 00:00:01 |
|   9 |          NESTED LOOPS OUTER                |                          |    2 |  1990 |       |    18   (0)| 00:00:01 |
|  10 |           NESTED LOOPS                     |                          |    2 |  1868 |       |    17   (0)| 00:00:01 |
|  11 |            NESTED LOOPS                    |                          |    2 |  1712 |       |    15   (0)| 00:00:01 |
|  12 |             NESTED LOOPS                   |                          |    2 |  1564 |       |    13   (0)| 00:00:01 |
|  13 |              MERGE JOIN CARTESIAN          |                          |    2 |  1442 |       |    11   (0)| 00:00:01 |
|  14 |               NESTED LOOPS OUTER           |                          |    1 |   625 |       |     8   (0)| 00:00:01 |
|  15 |                NESTED LOOPS OUTER          |                          |    1 |   613 |       |     7   (0)| 00:00:01 |
|  16 |                 NESTED LOOPS               |                          |    1 |   580 |       |     6   (0)| 00:00:01 |
|  17 |                  NESTED LOOPS OUTER        |                          |    1 |   539 |       |     5   (0)| 00:00:01 |
|  18 |                   NESTED LOOPS OUTER       |                          |    1 |   340 |       |     5   (0)| 00:00:01 |
|  19 |                    TABLE ACCESS BY INDEX ROWID| PA_STUDENT             |    1 |   316 |       |     3   (0)| 00:00:01 |
|* 20 |                     INDEX UNIQUE SCAN      | PK_STUDENT               |    1 |       |       |     2   (0)| 00:00:01 |
|  21 |                    TABLE ACCESS BY INDEX ROWID| PA_STUD_USER           |    1 |    24 |       |     2   (0)| 00:00:01 |
|* 22 |                     INDEX UNIQUE SCAN      | PK_STUD_USER             |    1 |       |       |     1   (0)| 00:00:01 |
|  23 |                   TABLE ACCESS BY INDEX ROWID | PA_ORG                  |    1 |   199 |       |     0   (0)|          |
|* 24 |                    INDEX UNIQUE SCAN       | PK_ORG                   |    1 |       |       |     0   (0)|          |
|  25 |                  TABLE ACCESS BY INDEX ROWID | PA_DOMAIN               |   13 |   533 |       |     1   (0)| 00:00:01 |
|* 26 |                   INDEX UNIQUE SCAN        | PK_DOMAIN                |    1 |       |       |     0   (0)|          |
|  27 |                 TABLE ACCESS BY INDEX ROWID | PA_USRRF_STUD           |  100 |  3300 |       |     1   (0)| 00:00:01 |
|* 28 |                  INDEX UNIQUE SCAN         | PK_USRRF_STUD            |    1 |       |       |     0   (0)|          |
|  29 |                VIEW PUSHED PREDICATE       | PV_STUD_USER             |    1 |    12 |       |     1   (0)| 00:00:01 |
|* 30 |                 FILTER                     |                          |       |       |       |            |          |
|  31 |                  NESTED LOOPS OUTER        |                          |    1 |    22 |       |   264  (11)| 00:00:01 |
|* 32 |                   INDEX UNIQUE SCAN        | PK_STUDENT               |    1 |    10 |       |     2   (0)| 00:00:01 |
|* 33 |                   MAT_VIEW ACCESS FULL     | PV_AP_STUD_USER          |    1 |    12 |       |   262  (11)| 00:00:01 |
|  34 |                BUFFER SORT                 |                          |    2 |   192 |       |    10   (0)| 00:00:01 |
|  35 |                 TABLE ACCESS BY INDEX ROWID | PA_CPNT_COMPLIANCE_DATA |    2 |   192 |       |     3   (0)| 00:00:01 |
|* 36 |                  INDEX RANGE SCAN          | IX_CPNT_CD__EVTHST       |    2 |       |       |     1   (0)| 00:00:01 |
|  37 |               TABLE ACCESS BY INDEX ROWID   | PA_CPNT_TYPE             |    1 |    61 |       |     1   (0)| 00:00:01 |
|* 38 |                INDEX UNIQUE SCAN           | PK_CPNT_TYPE             |    1 |       |       |     0   (0)|          |
|  39 |              TABLE ACCESS BY INDEX ROWID   | PA_RQMT_TYPE             |    1 |    74 |       |     1   (0)| 00:00:01 |
|* 40 |               INDEX UNIQUE SCAN            | PK_RQMT_TYPE             |    1 |       |       |     0   (0)|          |
|  41 |             TABLE ACCESS BY INDEX ROWID    | PA_CMPL_STAT             |    1 |    78 |       |     1   (0)| 00:00:01 |
|* 42 |              INDEX UNIQUE SCAN             | PK_CMPL_STAT             |    1 |       |       |     0   (0)|          |
|  43 |            TABLE ACCESS BY INDEX ROWID     | PA_QUAL                  |    1 |    61 |       |     1   (0)| 00:00:01 |
|* 44 |             INDEX UNIQUE SCAN              | PK_QUAL                  |    1 |       |       |     0   (0)|          |
|* 45 |           INDEX UNIQUE SCAN                | PK_CPNT                  |    1 |       |       |     0   (0)|          |
|  46 |          TABLE ACCESS BY INDEX ROWID       | PA_CPNT                  |    1 |    27 |       |     1   (0)| 00:00:01 |
|  47 |         BUFFER SORT                        |                          |  399 |       |       |    21   (0)| 00:00:01 |
|  48 |          INDEX FAST FULL SCAN              | PK_USRRF_STUD            |  399 |       |       |     1   (0)| 00:00:01 |
|  49 |        BUFFER SORT                         |                          |  399 |       |       |   231  (11)| 00:00:01 |
|  50 |         INDEX FAST FULL SCAN               | PK_USRRF_STUD            |  399 |       |       |     0   (0)|          |
|  51 |       BUFFER SORT                          |                          |  710K|       |       |    96M (16)| 26:57:48 |
|  52 |        INDEX FAST FULL SCAN                | IX_STUD_USER__STUDENT    |  710K|       |       |   309  (16)| 00:00:01 |
|  53 |      BUFFER SORT                           |                          |  710K|       |       |   264T (79)|999:59:59 |
|  54 |       INDEX FAST FULL SCAN                 | IX_STUD_USER__STUDENT    |  710K|       |       |   309  (16)| 00:00:01 |
-----------------------------------------------------------------------------------------------------------------------------------
```

Poorly Written Applications

SQL ordered by Executions

- Total Executions: 29,717,627
- Captured SQL account for 77.4% of Total

Executions	Rows Processed	Rows per Exec	CPU per Exec (s)	Elap per Exec (s)	SQL Id	SQL Module	SQL Text
10,128,178	2,506,529	0.25	0.00	0.00	932srzq1krc33	ASN_07B_DIP(004110016)	SELECT NE_TIMEZONE FROM CMPM.E...
7,576,759	7,579,197	1.00	0.00	0.00	1h698sb62un99	asci_56_RANAPProtocolStats(01611000E)	SELECT DISTINCT NE_TIMEZONE FR...
3,914,621	3,848,268	0.98	0.00	0.00	5tbzddgguu8cc	asci_56_RANAPProtocolStats(01611000E)	SELECT SYS_VERSION FROM CMPM.T...
311,645	311,604	1.00	0.00	0.00	7qtztzv329wq0		select c.name, u.name from co...
301,428	301,325	1.00	0.00	0.00	36s446f9cnwhw		SELECT C.NAME FROM COL$ C WHER...
200,692	200,669	1.00	0.00	0.00	4vs91dcv7u1p6	OMS	insert into sys.aud$(sessioni...
65,044	65,035	1.00	0.00	0.00	fz9xwpt2cvt0k		SELECT par_type, param_clob, ...
64,949	3,945,482	60.75	0.00	0.00	f5ra7dru5fk5n	XML_P7R_RNC_RCS(003110008)	SELECT NAME, PATH, READ, WR...
64,801	64,807	1.00	0.00	0.00	fhzj09a7fnrnb	XML_V7I_IN_LP_DC(00811000V)	SELECT DBTIMEZONE, LENGTH(DBT...
64,632	64,542	1.00	0.00	0.00	15jnrrb6016nd	XML_V7I_IN_LP_DC(00811000V)	SELECT SESSIONTIMEZONE, LENGT...

```
SELECT /*+ RESULT_CACHE */ srvr_id
FROM (
  SELECT srvr_id, SUM(cnt) SUMCNT
  FROM (
    SELECT DISTINCT srvr_id, 1 AS CNT
    FROM servers
    UNION ALL
    SELECT DISTINCT srvr_id, 1
    FROM serv_inst)
  GROUP BY srvr_id)
WHERE sumcnt = 2;
```

more examples: www.morganslibrary.org/reference/pkgs/dbms_result_cache.html

Optimizer Settings

- Default Oracle install favors data warehouse not OLTP

```
NAME                                     TYPE        VALUE
---------------------------------------- ----------- -----------
optimizer_adaptive_features              boolean     TRUE
optimizer_adaptive_reporting_only        boolean     FALSE
optimizer_capture_sql_plan_baselines     boolean     FALSE
optimizer_dynamic_sampling               integer     2
optimizer_features_enable                string      12.1.0.2
optimizer_index_caching                  integer     0
optimizer_index_cost_adj                 integer     100
optimizer_mode                           string      ALL_ROWS
optimizer_secure_view_merging            boolean     TRUE
optimizer_use_invisible_indexes          boolean     FALSE
optimizer_use_pending_statistics         boolean     FALSE
optimizer_use_sql_plan_baselines         boolean     TRUE
```

```
SQL> show parameter optimizer_mode

NAME                                 TYPE        VALUE
------------------------------------ ----------- ---------
optimizer_mode                       string      ALL_ROWS

SQL> ALTER SYSTEM SET OPTIMIZER_MODE='FIRST_ROWS_10';

System altered.

SQL> show parameter optimizer_mode

NAME                      TYPE        VALUE
------------------------- ----------- --------------
optimizer_mode            string      FIRST_ROWS_10
```

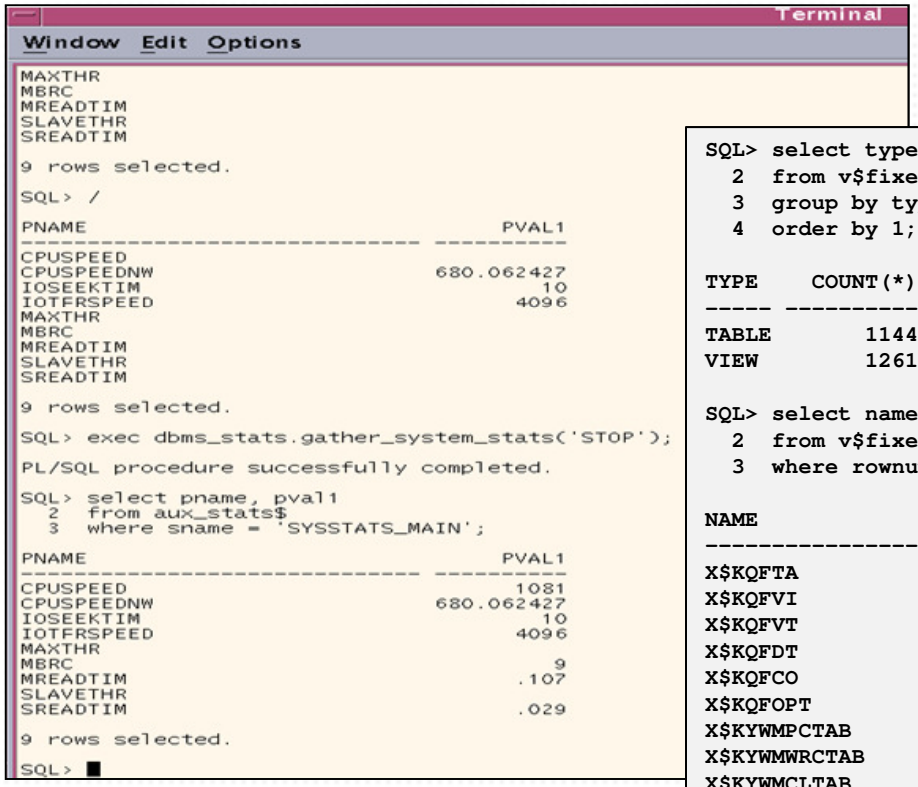
Best for Data Warehouse

Best for OLTP

more examples: www.morganslibrary.org/reference/startup_parms.html

Optimizer Statistics

- Gather System Stats
- Gather Fixed Object Stats
- Gather Dictionary Stats
- Gather Table Stats
- Gather Column Stats
- Gather Index Stats
- Gather Extended Stats
- Stat Generation
  - Copy Stats
  - Set Stats
  - Fixing Stats




```
SQL> select type, count(*)
  2  from v$fixed_table
  3  group by type
  4  order by 1;

TYPE    COUNT(*)
----- ----------
TABLE       1144
VIEW        1261

SQL> select name
  2  from v$fixed_table
  3  where rownum < 11;

NAME
--------------------------
X$KQFTA
X$KQFVI
X$KQFVT
X$KQFDT
X$KQFCO
X$KQFOPT
X$KYWMPCTAB
X$KYWMWRCTAB
X$KYWMCLTAB
X$KYWMNF
```

more examples: www.morganslibrary.org/reference/system_stats.html

Gathering Optimizer Statistics

```
exec dbms_stats.set_table_prefs(USER, 'SERVERS', 'ESTIMATE_PERCENT','90');

exec dbms_stats.set_table_prefs(USER, 'SERVERS', 'DEGREE','8');
```

```
exec dbms_stats.gather_table_stats(USER, 'SERVERS',
'P2010103106',
estimate_percent => dbms_stats.auto_sample_size,
block_sample=>TRUE,
method_opt=>'FOR ALL INDEXED COLUMNS',
degree=>2,
cascade=>TRUE
force=>TRUE);
```

```
SELECT dbms_stats.create_extended_stats(USER, 'SERV_INST', '(srvr_id, si_status)')
FROM dual;
```

more examples: www.morganslibrary.org/reference/pkgs/dbms_stats.html

Manufacturing Optimizer Statistics

```
exec dbms_stats.set_table_stats(USER, 'EMP', numrows=>1000000, numblks=>10000, avgrlen=>74);

exec dbms_stats.set_index_stats(USER, 'ix_emp_deptno', numrows=>1000000, numlblks=>1000,
     numdist=>10000, clstfct=>1);

exec dbms_stats.set_column_stats(USER, 'emp', 'deptno', distcnt=>10000);

exec dbms_stats.set_table_stats(USER, 'dept', numrows=>100, numblks=>100);
```

more examples: www.morganslibrary.org/reference/pkgs/dbms_stats.html

Root Cause Analysis by Sophisticated Guessing

- Check data dictionary for collected stats
- Explain Plans
  - Very few people can read them ... I will prove it next
- AWR Reports
  - Data <> Information
  - If you want value from AWR ... create **AWR Difference** reports with **DBMS_WORKLOAD_REPOSITORY.AWR_DIFF_REPORT_HTML**
- ASH Reports
- SQL Tuning Advisor
- SQL Trace and TKPROF
- Baselines and Evolving Baselines
- DBMS_TRACE, DBMS_MONITOR, TRCSESS, TRACE ANALYZER, SQLTXPLAIN

Reading Explain Plans

- Very few people can read an explain plan
- Here are two plans from identical tables with identical stats
- Which one would you rely on?

The SQL Statement

```
SELECT srvr_id
FROM servers
INTERSECT
SELECT srvr_id
FROM serv_inst;
```

Database 11gR2

```
-------------------------------------------------------------------------------
| Id | Operation              | Name         | Rows | Bytes | Cost (%CPU)|
-------------------------------------------------------------------------------
|  0 | SELECT STATEMENT       |              |  141 | 4560  |     6  (84)|
|  1 |  INTERSECTION          |              |      |       |            |
|  2 |   SORT UNIQUE NOSORT   |              |  141 |  564  |     2  (50)|
|  3 |    INDEX FULL SCAN     | PK_SERVERS   |  141 |  564  |     1   (0)|
|  4 |   SORT UNIQUE          |              |  999 | 3996  |     4  (25)|
|  5 |    INDEX FAST FULL SCAN| IX_SERV_INST |  999 | 3996  |     3   (0)|
-------------------------------------------------------------------------------
```

```
SQL> SELECT table_name, blocks
  2  FROM user_tables
  3* WHERE table_name IN ('SERVERS', 'SERV_INST');

TABLE_NAME                       BLOCKS
------------------------------ ----------
SERVERS                            28
SERV_INST                          28
```

```
-------------------------------------------------------------------------------
| Id  | Operation              | Name         | Rows | Bytes | Cost (%CPU)|
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT       |              |  141 |  4560 |    20  (10)|
|   1 |  INTERSECTION          |              |      |       |            |
|   2 |   SORT UNIQUE          |              |  141 |   564 |    10  (10)|
|   3 |    TABLE ACCESS FULL   | SERVERS      |  141 |   564 |     9   (0)|
|   4 |   SORT UNIQUE          |              |  999 |  3996 |    10  (10)|
|   5 |    TABLE ACCESS FULL   | SERV_INST    |  999 |  3996 |     9   (0)|
-------------------------------------------------------------------------------
```

Database 12gR1

more examples: www.morganslibrary.org/reference/explain_plan.html

This Adds To The Confusion

```
SQL> SELECT blocks
  2  FROM dba_tables
  3  WHERE owner = 'UWCLASS'
  4  AND table_name = 'SERVERS';

    BLOCKS
----------
        28

SQL> SELECT blocks
  2  FROM dba_segments
  3  WHERE owner = 'UWCLASS'
  4  AND segment_name = 'SERVERS';

    BLOCKS
----------
        32
```

This Makes It Even Worse

```
SQL> DECLARE
  2   uf   NUMBER;
  3   ub   NUMBER;
  4   f1   NUMBER;
  5   f1b  NUMBER;
  6   f2   NUMBER;
  7   f2b  NUMBER;
  8   f3   NUMBER;
  9   f3b  NUMBER;
 10   f4   NUMBER;
 11   f4b  NUMBER;
 12   fbl  NUMBER;
 13   fby  NUMBER;
 14  BEGIN
 15    dbms_space.space_usage('UWCLASS','SERVERS', 'TABLE', uf, ub, f1, f1b, f2, f2b, f3, f3b, f4, f4b, fbl, fby);
 16
 17    dbms_output.put_line('unformatted blocks:  ' || TO_CHAR(uf));
 18    dbms_output.put_line('unformatted bytes:   ' || TO_CHAR(ub));
 19    dbms_output.put_line('blocks 0-25% free:   ' || TO_CHAR(f1));
 20    dbms_output.put_line('bytes 0-25% free:    ' || TO_CHAR(f1b));
 21    dbms_output.put_line('blocks 25-50% free:  ' || TO_CHAR(f2));
 22    dbms_output.put_line('bytes 25-50% free:   ' || TO_CHAR(f2b));
 23    dbms_output.put_line('blocks 50-75% free:  ' || TO_CHAR(f3));
 24    dbms_output.put_line('bytes 50-75% free:   ' || TO_CHAR(f3b));
 25    dbms_output.put_line('blocks 75-100% free: ' || TO_CHAR(f4));
 26    dbms_output.put_line('bytes 75-100% free:  ' || TO_CHAR(f4b));
 27    dbms_output.put_line('full blocks:         ' || TO_CHAR(fbl));
 28    dbms_output.put_line('full bytes:          ' || TO_CHAR(fby));
 29  END;
 30  /
unformatted blocks:  16
unformatted bytes:   131072
blocks 0-25% free:   0
bytes 0-25% free:    0
blocks 25-50% free:  1
bytes 25-50% free:   8192
blocks 50-75% free:  0
bytes 50-75% free:   0
blocks 75-100% free: 11
bytes 75-100% free:  90112
full blocks:         0
full bytes:          0
```

only 12 blocks are formatted

more examples: www.morganslibrary.org/reference/pkgs/dbms_space.html

AWR Difference Reports

- SQL scripts located at $ORACLE_HOME/rdbms/admin

File Name	Strategy
awrddrpi.sql	Report on differences between differences between values recorded in two pairs of snapshots. This script requests the user for the dbid and instance number of the instance to report on, for each snapshot pair, before producing the standard report.
awrddrpt.sql	Defaults the dbid and instance number to that of the current instance connected-to, then calls awrddrpi.sql to produce the Compare Periods report.
awrextr.sql	SQL/Plus script to help users extract data from the AWR.
awrgdrpi.sql	RAC Version of Compare Period Report.
awrgdrpt.sql	This script defaults the dbid to that of the current instance connected-to, defaults instance list to all available instances and then calls awrgdrpi.sql to produce the Workload Repository RAC Compare Periods report.
awrgrpt.sql	This script defaults the dbid to that of the current instance connected-to, then calls awrgrpti.sql to produce the Workload Repository RAC report.
awrgrti.sql	SQL*Plus command file to report on RAC-wide differences between values recorded in two snapshots. This script requests the user for the dbid before producing the standard Workload Repository report.

more examples: www.morganslibrary.org/reference/awr_report.html

AWR Difference Reports

- SQL scripts located at $ORACLE_HOME/rdbms/admin

File Name	Strategy
awrinfo.sql	Outputs general AWR information such as the size, data distribution, etc. in AWR and SYSAUX. The intended use of this script is for diagnosing abnormalities in AWR and not for diagnosing issues in the database instance.
awrrpt.sql	Defaults the dbid and instance number to that of the current instance connected-to, then calls awrrpti.sql to produce the report.
awrrpti.sql	SQL*Plus command file to report on differences between values recorded in two snapshots. This script requests the user for the dbid and instance number of the instance to report on, before producing the standard report.
awrsqrpi.sql	SQL*Plus command file to report on differences between values recorded in two snapshots. This script requests the user for the dbid, instance number and the sql id, before producing a report for a particular sql statement in this instance.
awrsqrpt.sql	Defaults the dbid and instance number to that of the current instance connected-to then calls awrsqrpi.sql to produce a Workload report for a particular sql statement.
awrupd12.sql	This script updates AWR data to version 12c. It only modifies AWR data that has been imported using awrload.sql, or data from before changing the database DBID. In other words, it doesn't modify AWR data for the local, active DBID.

more examples: www.morganslibrary.org/reference/awr_report.html

AWR Difference Reports

WORKLOAD REPOSITORY COMPARE PERIOD REPORT

Snapshot Set	DB Name	DB Id	Instance	Inst num	Release	Cluster	Host	Std Block Size
First (1st)	ORAP09	2886124853	orap09	1	11.2.0.3.0	NO	db20p03sh	8192
Second (2nd)	ORAP09	2886124853	orap09	1	11.2.0.3.0	NO	db20p03sh	8192

Snapshot Set	Begin Snap Id	Begin Snap Time	End Snap Id	End Snap Time	Avg Active Users	Elapsed Time (min)	DB time (min)
1st	33759	02-Apr-14 04:00:19 (Wed)	33760	02-Apr-14 04:30:22 (Wed)	0.8	30.1	24.5
2nd	33807	03-Apr-14 04:00:13 (Thu)	33808	03-Apr-14 04:30:16 (Thu)	1.0	30.1	29.8
%Diff					*22.2*	*0.0*	*21.6*

Host Configuration Comparison

	1st	2nd	Diff	%Diff
Number of CPUs:	80	80	0	0.0
Number of CPU Cores:	40	40	0	0.0
Number of CPU Sockets:	4	4	0	0.0
Physical Memory:	1031464.9M	1031464.9M	0M	0.0
Load at Start Snapshot:	19.09	20.68	1.59	8.3
Load at End Snapshot:	11.49	11.18	-.31	-2.7
%User Time:	14.88	14.44	-.44	-3.0
%System Time:	1.08	1.05	-.03	-2.8
%Idle Time:	83.92	84.38	.46	0.5
%IO Wait Time:	.31	.45	.13	45.2

more examples: www.morganslibrary.org/reference/awr_report.html

AWR Difference Reports

Wait Events

- Ordered by absolute value of 'Diff' column of '% of DB time' descending (idle events last)

		% of DB time			# Waits/sec (Elapsed Time)			Total Wait Time (sec)			Avg Wait Time (ms)		
Event	Wait Class	1st	2nd	Diff	1st	2nd	%Diff	1st	2nd	%Diff	1st	2nd	%Diff
db file scattered read	User I/O	0.01	4.46	4.45	2.30	544.72	23,583.48	0.21	79.69	37,847.62	0.05	0.08	60.00
read by other session	User I/O	0.00	0.88	0.87	0.35	66.08	18,780.00	0.03	15.65	52,066.67	0.05	0.13	160.00
log file parallel write	System I/O	3.80	3.41	-0.39	11.54	11.64	0.87	55.83	60.88	9.05	2.68	2.90	8.21
db file sequential read	User I/O	0.66	0.85	0.19	12.60	108.69	762.62	9.72	15.24	56.79	0.43	0.08	-81.40
enq: CR - block range reuse ckpt	Other	0.26	0.17	-0.09	0.40	0.31	-22.50	3.83	2.99	-21.93	5.24	5.35	2.10
log file sync	Commit	3.17	3.11	-0.06	8.42	9.26	9.98	46.65	55.59	19.16	3.07	3.33	8.47
SQL*Net message to client	Network	0.06	0.12	0.06	724.68	1,868.96	157.90	0.85	2.08	144.71	0.00	0.00	0.00
utl_file I/O	User I/O	0.23	0.19	-0.04	85.50	84.68	-0.96	3.40	3.34	-1.76	0.02	0.02	0.00
enq: RO - fast object reuse	Application	0.12	0.08	-0.03	0.41	0.31	-24.39	1.72	1.50	-12.79	2.34	2.68	14.53
db file async I/O submit	System I/O	1.29	1.26	-0.03	5.49	5.96	8.56	18.97	22.51	18.66	1.91	2.09	9.42
control file parallel write	System I/O	0.35	0.32	-0.03	0.75	0.73	-2.67	5.09	5.66	11.20	3.75	4.32	15.20
library cache: mutex X	Concurrency	0.05	0.02	-0.03	1.46	1.10	-24.66	0.77	0.42	-45.45	0.29	0.21	-27.59
latch free	Other	0.05	0.03	-0.02	0.34	0.31	-8.82	0.76	0.60	-21.05	1.23	1.08	-12.20
direct path write	User I/O	0.06	0.05	-0.01	1.67	2.14	28.14	0.90	0.85	-5.56	0.30	0.22	-26.67
direct path write temp	User I/O	0.37	0.39	0.01	1.52	2.30	51.32	5.46	6.88	26.01	1.99	1.66	-16.58
reliable message	Other	0.03	0.02	-0.01	0.81	0.62	-23.46	0.47	0.37	-21.28	0.32	0.33	3.13
log buffer space	Configuration	0.02	0.01	-0.01	0.01	0.01	0.00	0.23	0.11	-52.17	14.15	8.65	-38.87
db file parallel write	System I/O	0.03	0.02	-0.01	0.10	0.08	-20.00	0.40	0.38	-5.00	2.31	2.71	17.32
control file sequential read	System I/O	0.01	0.00	-0.00	5.45	4.86	-10.83	0.10	0.09	-10.00	0.01	0.01	0.00
SQL*Net more data from client	Network	0.01	0.01	-0.00	1.98	1.89	-4.55	0.11	0.10	-9.09	0.03	0.03	0.00
SQL*Net more data to client	Network	0.01	0.01	-0.00	3.84	3.94	2.60	0.12	0.12	0.00	0.02	0.02	0.00
asynch descriptor resize	Other	0.00	0.00	-0.00	2.45	2.43	-0.82	0.05	0.05	0.00	0.01	0.01	0.00
cursor: pin S	Concurrency	0.00	0.00	0.00	0.04	0.03	-25.00	0.00	0.01	100.00	0.01	0.23	2,200.00
latch: redo writing	Configuration	0.00	0.00	-0.00	0.06	0.04	-33.33	0.02	0.01	-50.00	0.17	0.16	-5.88
latch: messages	Other	0.00	0.00	-0.00	0.16	0.13	-18.75	0.04	0.04	0.00	0.14	0.17	21.43
direct path read temp	User I/O	0.00	0.00	-0.00	0.63	0.48	-23.81	0.02	0.01	-50.00	0.02	0.02	0.00
direct path sync	User I/O	0.01	0.00	-0.00	0.02	0.03	50.00	0.08	0.09	12.50	2.19	1.76	-19.63
latch: cache buffers chains	Concurrency	0.00	0.00	0.00	0.14	0.63	350.00	0.02	0.03	50.00	0.06	0.02	-66.67

more examples: www.morganslibrary.org/reference/awr_report.html

AWR Difference Reports

Other Instance Activity Stats

- Ordered by statistic name

Statistic	Value			per Second (DB time)			per Second (Elapsed Time)			per Trans		
	1st	2nd	%Diff	1st	2nd	%Diff	1st	2nd	%Diff	1st	2nd	%Diff
calls to kcmgas	44,985	48,775	8.43	30.62	27.31	-10.81	24.94	27.04	8.42	2.48	2.51	1.21
calls to kcmgcs	327,166	351,412	7.41	222.67	196.76	-11.64	181.40	194.84	7.41	18.03	18.07	0.22
cell physical IO interconnect bytes	2,106,564,608	100,858,843,136	4,687.84	1,433,726.13	56,473,555.08	3,838.94	1,167,996.50	55,921,065.97	4,687.78	116,108.95	5,186,077.91	4,366.56
change write time	518	468	-9.65	0.35	0.26	-25.71	0.29	0.26	-10.34	0.03	0.02	-33.33
cleanout - number of ktugct calls	1,431	1,349	-5.73	0.97	0.76	-21.65	0.79	0.75	-5.06	0.08	0.07	-12.50
cleanouts and rollbacks - consistent read gets	10	29	190.00	0.01	0.02	100.00	0.01	0.02	100.00	0.00	0.00	0.00

more examples: www.morganslibrary.org/reference/awr_report.html

Summary

- Do not guess
  - If your problem is not caused by slow hardware then faster hardware will just create a problem faster
  - Gather information from a date-time range when everything was good and compare with the date-time range when the problem was observed
  - If the problem is slow SQL ... is the SQL statement itself slow or is it the entire environment that should be examined?
  - If the problem is related to CPU ... do you need more/faster CPUs or a better design?
  - If the problem is related to I/O ... do you need a new SAN or better indexing?
  - If the problem is related to network bandwidth ... do you need new NIC cards and to stop network virtualization? Yes you probably do

Conclusion

- Do not tune by guessing
- Do not blame everything on bad SQL statements
- Until you can prove that the root cause is the SQL
- Use the scientific method
  - Construct a hypothesis
  - Test the hypothesis by doing experiments
  - Validate your conclusion
- Only rewrite SQL after you have established conclusively that the root cause is the way a statement has been written and that rewriting the SQL statement will address ALL issues
- You will most likely rewriting very few SQL statements

ERROR at line 1:
ORA-00028: your session has been killed

Thank You

Contact: dmorgan@forsythe.com
+1 206-669-2949
Skype: damorgan11g

ERROR at line 1:
ORA-00028: your session has been killed

Contact: dmorgan@forsythe.com
+1 206-669-2949
Skype: damorgan11g