

Developing and Deploying Insanely Large Databases with Oracle 11gR2

Dan Morgan
Morgan's Library

Daniel A. Morgan



Oracle ACE Director



Consultant to Harvard University



University of Washington Oracle Instructor, ret.



The Morgan of Morgan's Library on the web



Board Member: Western Washington OUG

■ Upcoming Presentations

- Feb 23: Vancouver B.C.
- Mar 19: University of Oslo
- Mar 21-24: Oracle User Group Norway
- Apr 24: Benelux Connect (Netherlands)
- May 30-31: Harmony Finland
- Jun 1: Harmony Latvia



Official Beta Site
ORACLE
DATABASE **11g**

ORACLE
RAC SIG

International
zSeries
Oracle SIG

Daniel A. Morgan | damorgan11g@gmail.com | www.morganslibrary.org

Developing and Deploying Insanely Large Databases with Oracle 11gR2

Presented: RMOUG 16 Feb 2012

cd \$MORGAN_HOME



Morgan's Library: www.morganslibrary.org





Morgan's Library

www library

Search

Morgan's 2012 - 2013 Calendar

Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec Jan

[[don't censor the web](#)] [[don't be silent](#)]

Home

Community

Events
Training
Evening Workshops

Resources

Library
How Can I?
Code Samples
Presentations
Links
Book Reviews
Downloads
User Groups
Humor

General

Contact
About
Services
Legal Notice & Terms of Use
Privacy Statement

Presentations Map



Oracle Core



Training Events

- [Harvard University](#) - Feb 6 - 10, 2012
- [RMOUG Training Days](#) - Feb 14 - 16, 2012
- [VanOUG Canada](#) - Feb 23, 2012
- [OUGN: Oslo, Norway](#) - Mar 21 - 24, 2012
- [BeNeLux Connect: Maastricht](#) - Apr 24, 2012
- [OUG Harmony Finland](#) - May 30 - 31, 2012
- [OUG Harmony Latvia](#) - Jun 01, 2012

Oracle Events



Next Event: RMOUG Denver, CO Feb 14-16

Morgan



aboard USA-71



ORACLE
ACE Director

Library News

- [Morgan's Notepad vi \(Blog\)](#)
- [Join the Western Washington OUG](#)
- [Morgan's Oracle Podcast](#)
- [US Government STIGs \(Security Checklists\)](#)
- [Bryn Llewellyn's PL/SQL White Paper](#)
- [Bryn Llewellyn's Editioning White Paper](#)
- [Explain Plan White Paper](#)
- [Troubleshooting Performance](#)

ACE News

 Would you like to become an Oracle ACE? 

Learn more about becoming an ACE



- [ACE Directory](#)
- [ACE Google Map](#)
- [ACE Program](#)
- [Stanley's Blog](#)

Congratulations to our newest ACEs

If it is not necessary ... don't do it!

Syllabus



- What is an ILDB
- Partitioning Options
- Cost Based Optimizer
- Resource Utilization
- Parallel Access
- Metadata Staging
- Developer Practices
- Stats Collection
- Storage

Discussion

What is a ILDB

Define Large



- The size of the storage footprint?
- The size of the queries?
- The size of the DML statements?
- The number of transactions per second?
- The number of simultaneously connected active users?

WORKLOAD REPOSITORY report for

DB Name	DB Id	Instance	Inst num	Startup Time	Release	RAC
OPM01P	782247420	opm01p6	6	18-Aug-10 21:08	11.1.0.7.0	YES

Host Name	Platform	CPUs	Cores	Sockets	Memory (GB)
usp9004b	Solaris[tm] OE (64-bit)	128	64	16	503.16

	Snap Id	Snap Time	Sessions	Cursors/Session
Begin Snap:	7037	15-Sep-10 13:00:18	406	7.5
End Snap:	7038	15-Sep-10 14:01:28	318	8.5
Elapsed:		61.17 (mins)		
DB Time:		6,076.88 (mins)		

Are my customer's happy?



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 Elapsed %:	0.00	% Non-Parse CPU:	96.60

No!



Foreground Wait Events

- s - second, ms - millisecond - 1000th of a second
- Only events with Total Wait Time (s) >= .001 are shown
- ordered by wait time desc, waits desc (idle events last)
- %Timeouts: value of 0 indicates value was < .5%. Value of null is truly 0

Event	Waits	%Time -outs	Total Wait Time (s)	Avg wait (ms)	Waits /txn	% DB time
cursor: pin S wait on X	559,981	100	11,250	20	15.41	23.77
db file sequential read	1,824,756	0	4,231	2	50.23	8.94
unspecified wait event	462,648	0	1,996	4	12.73	4.22
gc buffer busy acquire	322,335	0	1,342	4	8.87	2.84
external table misc IO	57,038	0	1,131	20	1.57	2.39
db file scattered read	69,467	0	989	14	1.91	2.09
PX Deq: reap credit	68,532,223	100	630	0	1,886.33	1.33
IPC send completion sync	3,694,966	98	389	0	101.70	0.82
PX Deq: Slave Session Stats	132,529	12	350	3	3.65	0.74
external table read	96,704	0	339	4	2.66	0.72
PX Nsq: PQ load info query	1,668	97	328	197	0.05	0.69
read by other session	91,147	0	300	3	2.51	0.63
latch free	4,480	0	297	66	0.12	0.63
gc cr grant 2-way	417,285	0	272	1	11.49	0.57
DFS lock handle	13,528	11	261	19	0.37	0.55
gc cr multi block request	547,105	0	217	0	15.06	0.46
enq: PS - contention	186,170	56	201	1	5.12	0.42
kksfbc child completion	2,097	100	138	66	0.06	0.29
...	...	...	...	...	...	...

No!



Dictionary Cache Stats

- "Pct Misses" should be very low ($\leq 2\%$ in most cases)
- "Final Usage" is the number of cache entries being used

Cache	Get Requests	Pct Miss	Scan Reqs	Pct Miss	Mod Reqs	Final Usage
dc_avwr_control	94	1.06	0		1	1
dc_constraints	75	81.33	0		75	268
dc_database_links	12,557	0.00	0		0	3
dc_files	18,320	0.00	0		0	482
dc_global_oids	414,904	0.00	0		0	355
dc_histogram_data	303,148	0.46	0		307	146,041
dc_histogram_defs	4,462,445	0.10	0		714	496,635
dc_object_grants	23,089	0.00	0		0	1,094
dc_objects	2,748,959	0.45	0		2,384	281,443
dc_partition_scns	3	0.00	0		0	5
dc_profiles	12,918	0.00	0		0	4
dc_rollback_segments	6,824,069	0.00	0		0	1,662
dc_segments	242,656	4.68	0		9,231	877,062
dc_sequences	396	3.28	0		396	33
dc_table_scns	43	37.21	0		0	8
dc_tablespace_quotas	8,495	21.27	0		2,365	66
dc tablespaces	11,893,072	0.00	0		0	38
dc_users	10,119,305	0.00	0		15	6,827
global database name	8,789	0.00	0		0	2
outstanding_alerts	182	75.82	0		1	26

Discussion

Partitioning Options

Oracle Database Search Results: VLDB

☒ Oracle Database 11g Release 2 (11.2) documentation ☐ All of Oracle.com

Results 1 to 10 of about 90 for **VLDB**.

VLDB and Partitioning

A very large database has no minimum absolute size.

VLDB and Partitioning Guide • [Search this book](#) • [Hide this book](#) • [Contents](#) • [PDF](#)

CHAPTER Introduction to Very Large Databases

VLDB and Partitioning Guide • [Search this book](#) • [Hide this book](#) • [Contents](#) • [PDF](#)

SYNTAX ALTER TABLE

SQL Language Reference • [Search this book](#) • [Hide this book](#) • [Contents](#) • [PDF](#)

TASK Striping Using Oracle ASM

Oracle Automatic Storage Management (Oracle ASM) always stripes across all devices presented to it as a disk group.

VLDB and Partitioning Guide • [Search this book](#) • [Hide this book](#) • [Contents](#) • [PDF](#)

CHAPTER Storage Management for VLDBs

Storage performance in data warehouse environments often translates into I/O throughput (MB/s).

VLDB and Partitioning Guide • [Search this book](#) • [Hide this book](#) • [Contents](#) • [PDF](#)

Scalability and Manageability

A very important characteristic of a **VLDB** is its large size.

VLDB and Partitioning Guide • [Search this book](#) • [Hide this book](#) • [Contents](#) • [PDF](#)

1999



- Simple
 - Partition by Hash
 - Partition by Range
- Composite
 - Range - Hash

2001



- 9.0.#
 - Partition by List
- 9.2.0
 - Range – List Composite



2004



- 10gR1
 - Nothing New
- 10gR2
 - Nothing New

ORACLE
DATABASE **10^g**

- Partition by Interval
- Partition by Reference
- Partition by System
- Partition by Virtual Column
- New Composite Partitioning Options
 - Hash – Hash
 - List – Hash
 - List – List
 - List – Range
 - Range – Range

Optimizing Partitioning



- Subpartition
 - Have a thorough understanding of partition pruning
 - Partition-subpartition into as many buckets as logically possible
- Do not use one large tablespace
- Set tablespaces to READ ONLY when you can
- Set partitions to READ ONLY when you can
- Understand stats collection and collection bugs
- Make Instance ID part of the partition key
or
- Make Instance ID part of the primary key
- Do not make partition keys the leading column in an index

Partition Indexing and Constraints



- Local, not global
 - In 11gR2 set local indexes not used to UNUSABLE
- Be very careful using bitmap indexes
 - How much redo is generated by updating one byte in one row?
- Use Function Based Indexes to avoid indexing values that will never be used
- Make column NOT NULL if possible
- Do not make primary key columns NOT NULL
- Index foreign keys to minimize locking (perhaps)

Partition Indexing in 11gR2



```
CREATE TABLE range_part (  
  prof_history_id NUMBER(10),  
  record_date      DATE NOT NULL)  
PARTITION BY RANGE (record_date) (  
  PARTITION yr0 VALUES LESS THAN (TO_DATE('01-JAN-2010', 'DD-MON-YYYY')),  
  PARTITION yr1 VALUES LESS THAN (TO_DATE('01-JAN-2011', 'DD-MON-YYYY')),  
  PARTITION yr2 VALUES LESS THAN (TO_DATE('01-JAN-2012', 'DD-MON-YYYY')),  
  PARTITION yr9 VALUES LESS THAN (MAXVALUE));
```

```
CREATE INDEX ix_range_part_phid  
ON range_part(prof_history_id)  
LOCAL UNUSABLE;
```

```
ALTER INDEX ix_range_part_phid REBUILD PARTITION yr1;
```

```
ALTER INDEX ix_range_part_phid REBUILD PARTITION yr2;
```

```
ALTER INDEX ix_range_part_phid MODIFY PARTITION yr1 UNUSABLE;
```

Discussion

Cost Based Optimizer
`dbms_xplan.display`

Optimizer Plans (the way it was)



```
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_OFDA7485F_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_OFDA7485F_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_OFDA74860_6A66C42E	337K	3956K		142 (14)	00:00:01		
20	VIEW		337K	4286K		142 (14)	00:00:01		
21	TABLE ACCESS FULL	SYS_TEMP_OFDA74860_6A66C42E	337K	3956K		142 (14)	00:00:01		
22	VIEW		22M	277M		16808 (12)	00:00:42		
23	TABLE ACCESS FULL	SYS_TEMP_OFDA7485F_6A66C42E	22M	1111M		16808 (12)	00:00:42		
24	VIEW		22M	277M		16808 (12)	00:00:42		
25	TABLE ACCESS FULL	SYS_TEMP_OFDA7485F_6A66C42E	22M	1111M		16808 (12)	0		

Optimizer Plans (tuning gone terribly wrong)



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_OFDA7485B_6A66C42E	22M	1111M		16808 (12)	00:00:42		
17	VIEW		22M	277M		16808 (12)	00:00:42		
18	TABLE ACCESS FULL	SYS_TEMP_OFDA7485B_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_OFDA7485C_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_OFDA7485C_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_OFDA7485B_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_OFDA7485B_6A66C42E	22M	1111M		16808 (12)	0		

Optimizer Plans (making it better)



```
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'),
     mn 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, mn h1, mn 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

Discussion

Backups

How Long Will It Take?



- Backup Failures
- Backup of Redo Logs
- Backups of Data Tablespaces
- Block Change Tracking
- Hardware Vendor vs Oracle

Discussion

Resource Utilization

Rationing Resource



- Resource Manager
- Consumer Groups
- Services
 - You do not need RAC to take advantage of services
 - Map services to resource plans

Resource Utilization



```
SQL> SELECT resource_name, current_utilization, max_utilization, limit_value
       2 FROM v$resource_limit
       3 ORDER BY 1;
```

RESOURCE_NAME	CURRENT_UTILIZATION	MAX_UTILIZATION	LIMIT_VALU
branches	2	24	UNLIMITED
cmtcallbk	0	3	UNLIMITED
dml_locks	88	88	UNLIMITED
enqueue_locks	980	2195	62224
enqueue_resources	827	1320	UNLIMITED
gcs_resources	285534	791769	972477
gcs_shadows	194845	394603	972477
ges_big_msgs	68	337	UNLIMITED
ges_cache_ress	51272	57938	UNLIMITED
ges_locks	0	0	UNLIMITED
ges_procs	414	1124	2007
ges_reg_msgs	617	1482	UNLIMITED
ges_ress	0	0	UNLIMITED
ges_rsv_msgs	0	1	1000
k2q_locks	3	20	UNLIMITED
max_rollback_segments	234	236	65535
max_shared_servers	1	1	UNLIMITED
parallel_max_servers	647	990	3600
processes	417	1129	2000
sessions	393	1438	2500
sort_segment_locks	123	348	UNLIMITED
temporary_table_locks	0	5	UNLIMITED
transactions	4282503752	4294967295	UNLIMITED

Discussion

Parallel Access

Parallel Issues



- Minimize the urge to run everything in parallel
- In 11gR2 set
 - PARALLEL_DEGREE_LIMIT
 - Limit the degree of parallelism used to ensure parallel server processes do not flood the system
 - PARALLEL_DEGREE_POLICY
 - See Arup Nanda's article in Oracle Magazine
 - PARALLEL_FORCE_LOCAL
 - On a RAC cluster force parallel query slaves to stay on the local instance and not parallelize across nodes increasing interconnect traffic
 - To use AutomaticDOP you must run CALIBRATE_IO
 - FIFO queue can become a denial of service attack
- DBMS_STATS and a lot of code is often configured by developers with a parallel degree > 1
- Beware of end users with a little knowledge

Discussion

Metadata Staging

Staging Issues



- Loaders need to store information about what they do
 - Especially if updates or deletes take place
 - Especially if summarizations (aggregations) are run
- Materialized Views
- Use first level summaries to generate second level summaries: Do not go back to the raw data

Discussion

Developer Practices

I am not not afraid to show mine in public



RMOUTG
ROCKY MOUNTAIN OUTFITTERS GROUP

[illegible]

Presented: RMOUG 16 Feb 2012

Presented: RMOUG 16 Feb 2012

Presented: RMOUG 16 Feb 2012

```

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_PSCORE.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_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.GSCGMM 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 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)sunday, 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)sunday, 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_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.GSDSTATS 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 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

```

Why SQL Like This



- Tables with hundreds of columns
- SQL written by a framework rather than a human
- A design based on literally transcribing message streams into columns

Unnecessary SQL



```
PROCEDURE get_records(pCustID IN NUMBER) AUTHID DEFINER IS
  vCount PLS_INTEGER;
BEGIN
  SELECT COUNT(*)
  INTO vCount
  FROM customers
  WHERE cust_id = pCustID;

  IF vCount > 0 THEN
    process_customer_recs(pCustID);
  END IF;
END get_records;
/
```

```
PROCEDURE process_customer_recs(pCustID IN NUMBER) AUTHID DEFINER IS
  CURSOR cust_cur IS
  SELECT transno, trans_type, transdate
  FROM customers
  WHERE cust_id = pCustID;
BEGIN
  ... do stuff ...
END process_customer_recs;
/
```

Unnecessary Dynamic SQL



```
PROCEDURE gather_schema_stats(pschema IN VARCHAR2, pGlobalOption IN BOOLEAN := TRUE) IS
BEGIN
    IF pschema IS NOT NULL THEN
        DBMS_STATS.UNLOCK_SCHEMA_STATS (pschema);

        IF pGlobalOption = TRUE THEN
            EXECUTE IMMEDIATE 'BEGIN DBMS_STATS.GATHER_SCHEMA_STATS(OWNNAME => ''' ||
                pschema || ''', ' ||
                ' ESTIMATE_PERCENT => 25, METHOD_OPT => ''FOR ALL INDEXED COLUMNS SIZE 254'', DEGREE => 4, ' ||
                ' CASCADE => TRUE, granularity => ''GLOBAL'', OPTIONS => ''GATHER STALE'', no_invalidate => TRUE); END;';
        ELSE
            EXECUTE IMMEDIATE 'BEGIN DBMS_STATS.GATHER_SCHEMA_STATS(OWNNAME => ''' ||
                pschema || ''', ' ||
                ' ESTIMATE_PERCENT => 25, METHOD_OPT => ''FOR ALL INDEXED COLUMNS SIZE 254'', DEGREE => 4, ' ||
                ' CASCADE => TRUE, OPTIONS => ''GATHER STALE'', no_invalidate => TRUE); END;';
        END IF;

        add_to_log_table(clogInformation, 827533, 'Schema stats gathered for schema ' || upper(pschema));
        -- Lock schema stats
        DBMS_STATS.LOCK_SCHEMA_STATS (pschema);
    END IF;
EXCEPTION
    WHEN OTHERS THEN
        --log error in COMMON_LOGS table
        add_to_log_table(clogmajor, 827516, 'Error in Gather_Schema_stats for schema ' ||
            upper(pschema) || ' - message:' || SQLERRM);
END;
```

```
dbms_stats.gather_schema_stats(pschema, ESTIMATE_PERCENT => 25,
METHOD_OPT => 'FOR ALL INDEXED COLUMNS SIZE 254', DEGREE => 4,
CASCADE => TRUE, granularity => 'GLOBAL', OPTIONS => 'GATHER STALE', no_invalidate => TRUE);

dbms_stats.gather_schema_stats(pschema, ESTIMATE_PERCENT => 25,
METHOD_OPT => 'FOR ALL INDEXED COLUMNS SIZE 254', DEGREE => gcDegree,
CASCADE => TRUE, granularity => 'GLOBAL', OPTIONS => 'GATHER STALE', no_invalidate => TRUE);
```

Failure to Use Bind Variables



```
SQL> SELECT owner, COUNT(*)
  2  FROM dba_source
  3  WHERE owner NOT LIKE '%SYS%'
  4  AND UPPER(text) LIKE '%EXECUTE IMMEDIATE%'
  5  GROUP BY owner
  6  ORDER BY 1;
```

OWNER	COUNT (*)
ORACLE_OCM	8
OSSBACKEND	9
PMCM	54
SRE	21
WEBWIZ	20
ZZYZX	39

Bind Variables Demo



```
DECLARE
  x NUMBER(10);
BEGIN
  FOR i IN 1 .. 50000
  LOOP
    EXECUTE IMMEDIATE 'SELECT ' || i || ' INTO :b1 FROM dual'
    INTO x;
  END LOOP;
END;
/
```

```
DECLARE
  x NUMBER(10);
BEGIN
  FOR i IN 1 .. 50000
  LOOP
    EXECUTE IMMEDIATE 'SELECT :b1 FROM dual'
    INTO x
    USING i;
  END LOOP;
END;
/
```

Unnecessary Concatenation



```
FUNCTION delete_partitions_in_range(pschema    IN VARCHAR2,
                                   ptable      IN VARCHAR2,
                                   pretention  IN INTEGER,
                                   parttype    IN INTEGER) RETURN PLS_INTEGER

IS
    ... variable declarations ...

BEGIN
    --add starting processing
    IF ptable <> 'COMMON_LOGS' THEN
        add_to_log_table(cloginformation,
                        827600,
                        upper(pschema) || '.' || upper(ptable) ||
                        ' - Processing (Delete partitions in range)');

    END IF;
    ... code here ...
    LOOP
        BEGIN
            ... code here ...
        EXCEPTION
            WHEN OTHERS THEN
                add_to_log_table(clogmajor,
                                827604,
                                upper(pschema) || '.' || upper(ptable) ||
                                ' - Unable to Delete Partition ' ||
                                partTable(i).Partition_Name || ':' ||
                                SQLERRM);

                RAISE;
            END;
        END LOOP;

        --Add log entry to summarise the partitions added
        IF ((vSummaryMinDate < vmindate) AND (ptable <> 'COMMON_LOGS')) THEN
            add_to_log_table(cloginformation,
                            827605,
                            upper(pschema) || '.' || upper(ptable) ||
                            ' - Deleted partitions from ' ||
                            to_char(vSummaryMinDate, 'YYYY-MM-DD HH24:MI:SS') ||
                            ' to ' || to_char(vmindate, 'YYYY-MM-DD HH24:MI:SS'));

        END IF;
        RETURN partTable.count;
    END;
```

Replaced by a Variable Assignment



```
FUNCTION delete_partitions_in_range(pschema    IN VARCHAR2,
                                   ptable      IN VARCHAR2,
                                   pretention  IN INTEGER,
                                   parttype   IN INTEGER) RETURN PLS_INTEGER

IS
    ... variable declarations ...
    cObject VARCHAR2(61) := UPPER(pschema || '.' || ptable);
BEGIN
    --add starting processing
    IF ptable <> 'COMMON_LOGS' THEN
        add_to_log_table(cloginformation,
                        827600,
                        cObject ||
                        ' - Processing (Delete partitions in range)');
    END IF;
    ... code here ...
    LOOP
        BEGIN
            ... code here ...
        EXCEPTION
            WHEN OTHERS THEN
                add_to_log_table(clogmajor,
                                827604,
                                cObject ||
                                ' - Unable to Delete Partition ' ||
                                partTable(i).Partition_Name || ':' ||
                                SQLERRM);

                RAISE;
            END;
        END LOOP;

        --Add log entry to summarise the partitions added
        IF ((vSummaryMinDate < vmindate) AND (ptable <> 'COMMON_LOGS')) THEN
            add_to_log_table(cloginformation,
                            827605,
                            cObject ||
                            ' - Deleted partitions from ' ||
                            to_char(vSummaryMinDate, 'YYYY-MM-DD HH24:MI:SS') ||
                            ' to ' || to_char(vmindate, 'YYYY-MM-DD HH24:MI:SS'));
        END IF;
        RETURN partTable.count;
    END;
```

Deterministic Functions



- Indicates that the function returns the same result value whenever it is called with the same values for its parameters

```
SQL> SELECT owner, COUNT(*)
  2  FROM dba_arguments
  3  WHERE sequence = 0
  4  AND owner NOT LIKE '%SYS%'
  5  GROUP BY owner
  6  ORDER BY 1;
```

OWNER	COUNT (*)
PMCM	16
OSSBACKEND	3
OSS_DICTIONARY	8
SRE	6
WEBWIZ	2
ZZYZX	17

```
SQL> SELECT owner, COUNT(*)
  2  FROM dba_source
  3  WHERE owner NOT LIKE '%SYS%'
  4  AND UPPER(text) LIKE '%DETERMINISTIC%'
  5  GROUP BY owner
  6  ORDER BY 1;
```

OWNER	COUNT (*)
ZZYZX	1
SRE	79

```
CREATE OR REPLACE FUNCTION get_date_determ RETURN DATE DETERMINISTIC IS
BEGIN
    RETURN TRUNC(SYSDATE);
END get_date_determ;
/
```

Direct Path



- A direct load operation requires that the table being loaded is locked
- Queries are lock-free and are allowed while the object is being loaded
- The mode of the DML lock, and which DML locks are obtained depend upon the specification of the `OCI_ATTR_DIRPATH_PARALLEL` option, and if a partition or subpartition load is being done as opposed to an entire table load
- Parallel Insert automatically uses DIRECT PATH
- Whereas a normal insert will not block another insert when you use DIRECT PATH multiple loaders can create blockage (especially bad with RAC)

GV\$ Dynamic Performance Views



- V\$ is the local instance
- GV\$ view hit all instances
 - Creating delays and increased cache fusion interconnect traffic
- Do not query a gv\$ unless you really need to collect information from more than one node

```
SELECT name FROM v$database;
```

```
SELECT name FROM gv$database;
```

Long Operations



- DBMS_APPLICATION_INFO.SET_SESSION_LONGOPS

```
dbms_application_info.set_session_longops(  
  rindex      IN OUT BINARY_INTEGER,  
  slno        IN OUT BINARY_INTEGER,  
  op_name     IN      VARCHAR2(64) DEFAULT NULL,  
  target      IN      BINARY_INTEGER DEFAULT 0,  
  context     IN      BINARY_INTEGER DEFAULT 0,  
  sofar       IN      NUMBER DEFAULT 0,  
  totalwork   IN      NUMBER DEFAULT 0,  
  target_desc IN      VARCHAR2(32) DEFAULT 'unknown_target',  
  units       IN      VARCHAR2(32) DEFAULT NULL);
```

PLSQL Warnings



- PLW-07202
 - Bind type will result in conversion away from column type
- PLW-07203
 - Parameter '*string*' may benefit from use of the NOCOPY compiler hint
- PLW-07204
 - Conversion away from column type may result in sub-optimal query plan
- PLW-07205
 - SIMPLE_INTEGER is mixed with BINARY_INTEGER or PLS_INTEGER
- PLW-07206
 - Analysis suggests that the assignment to '*string*' may be unnecessary

ALTER SESSION SET PLSQL_WARNINGS='ENABLE:ALL';

Result Cache



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	932srzgt1krc33	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(016110006)	SELECT DISTINCT NE_TIMEZONE FR...
3,914,621	3,848,268	0.98	0.00	0.00	5tbzddgguu8cc	asci_56_RANAPProtocolStats(016110006)	SELECT SYS_VERSION FROM CMPM.T...
311,645	311,604	1.00	0.00	0.00	7gtztzv329wg0		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	fhzi09a7fmrnb	XML_V7I_IN_LP_DC(00811000V)	SELECT DBTIMEZONE, LENGTH(DBT...
64,632	64,542	1.00	0.00	0.00	15inrrb6016nd	XML_V7I_IN_LP_DC(00811000V)	SELECT SESSIONTIMEZONE, LENGT...

```
CREATE OR REPLACE FUNCTION rcache(p_srvr_id IN servers.srvr_id%TYPE) RETURN BOOLEAN
RESULT_CACHE RELIES_ON(servers) IS
```

```
    srvrow servers%ROWTYPE;
BEGIN
    SELECT *
    INTO srvrow
    FROM servers
    WHERE srvr_id = p_srvr_id;
    RETURN TRUE;
EXCEPTION
    WHEN OTHERS THEN
        RETURN FALSE;
END rcache;
/
```

```
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;
```

Sequence Optimization



- High volume insert intensive applications using sequence generated keys add extra overhead
- NOCACHE is the worst for performance
- CACHE NOORDER is the best for performance
- Services may help improve performance even for CACHE NOORDER sequences if the insert application uses a service with only one Preferred Instance
- This eliminates the Interconnect traffic and synchronization overhead required to coordinate the next value from amongst all the instances when using CACHE ORDER

Sequence Caching



```
SQL> SELECT sequence_owner, cache_size, COUNT(*)
  2  FROM dba_sequences
  3  WHERE sequence_owner IN ('ADMIN', 'PMCM', 'OPS', 'OSSBACKEND', 'OSS_DICTIONARY', 'SRE', 'WEBWIZ', 'ZZYZX')
  4  GROUP BY sequence_owner, cache_size
  5  ORDER BY 1,2;
```

SEQUENCE_OWNER	CACHE_SIZE	COUNT (*)
-----	-----	-----
ADMIN	20	1
PMCM	1000	2
OPS	0	1
OSSBACKEND	0	11
OSS_DICTIONARY	0	23
SRE	0	33
SRE	20	7
WEBWIZ	20	24
ZZYZX	0	21
ZZYZX	20	44

“You would be amazed what setting a sequence cache via alter sequence to 100,000 or more can do during a large load -- amazed.”

~ Tom Kyte

Sequence Ordering



```
SQL> SELECT sequence_owner, order_flag, COUNT(*)
  2  FROM dba_sequences
  3  WHERE sequence_owner IN ('ADMIN', 'PMCM', 'OPS', 'OSSBACKEND', 'OSS_DICTIONARY', 'SRE', 'WEBWIZ', 'ZZYZX')
  4  GROUP BY sequence_owner, order_flag
  5  ORDER BY 1,2;
```

SEQUENCE_OWNER	O	COUNT (*)
-----	-	-----
ADMIN	N	1
PMCM	N	2
OPS	N	1
OSSBACKEND	N	5
OSSBACKEND	Y	6
OSS_DICTIONARY	N	7
OSS_DICTIONARY	Y	16
SRE	N	39
SRE	Y	1
WEBWIZ	Y	24
ZZYZX	N	59
ZZYZX	Y	6

Discussion

Stats Collection

Stats Collection Issues



- RAC will parallelize across nodes
- You can not collect full stats ... it isn't going to happen
- BLOCK sample ... not row sample
- COPY_STATS
- SET_STATS

```
dbms_stats.copy_table_stats(  
  ownname      IN VARCHAR2,  
  tabname      IN VARCHAR2,  
  srcpartname  IN VARCHAR2,  
  dstpartname  IN VARCHAR2,  
  scale_factor IN NUMBER   DEFAULT 1,  
  flags        IN NUMBER   DEFAULT NULL,  
  force        IN BOOLEAN  DEFAULT FALSE);
```

```
dbms_stats.set_table_stats(  
  ownname      VARCHAR2,  
  tabname      VARCHAR2,  
  partname     VARCHAR2 DEFAULT NULL,  
  stattab      VARCHAR2 DEFAULT NULL,  
  statid       VARCHAR2 DEFAULT NULL,  
  numrows      NUMBER   DEFAULT NULL,  
  numblks      NUMBER   DEFAULT NULL,  
  avgrlen      NUMBER   DEFAULT NULL,  
  flags        NUMBER   DEFAULT NULL,  
  statown      VARCHAR2 DEFAULT NULL,  
  no_invalidate BOOLEAN  DEFAULT to_no_invalidate_type(get_param('NO_INVALIDATE')),  
  cachedblk    NUMBER   DEFAULT NULL,  
  cachehit     NUMBER   DEFAULT NULL,  
  force        BOOLEAN  DEFAULT FALSE);
```

Optimizer Bugs Can Bite



```
SELECT table_name, column_name, low_value, high_value
FROM dba_tab_cols
WHERE owner = 'EG' AND data_type = 'DATE' AND rownum < 6;
```

TABLE_NAME	COLUMN_NAME	LOW_VALUE	HIGH_VALUE
ALARM_DATA	LAST_DATA_DATE	786E0415010101	786E070A010101
ALARM_DATA	DATA_DATE	786E0415010101	786E0518010101
APSTATS	DATETIME	786E0709010101	786E0809100101

```
set serveroutput on
```

```
DECLARE
```

```
  d1 DATE;
```

```
  d2 DATE;
```

```
  r1 RAW(32) := '786E0415010101';
```

```
  r2 RAW(32) := '786E070A010101';
```

```
  r3 RAW(32) := '786E0415010101';
```

```
  r4 RAW(32) := '786E0518010101';
```

```
  r5 RAW(32) := '786E0709010101';
```

```
  r6 RAW(32) := '786E0809100101';
```

```
BEGIN
```

```
  dbms_stats.convert_raw_value(r1, d1);
```

```
  dbms_output.put_line(d1);
```

```
  dbms_stats.convert_raw_value(r2, d2);
```

```
  dbms_output.put_line(d2);
```

```
  dbms_stats.convert_raw_value(r3, d1);
```

```
  dbms_output.put_line(d1);
```

```
  dbms_stats.convert_raw_value(r4, d2);
```

```
  dbms_output.put_line(d2);
```

```
  dbms_stats.convert_raw_value(r5, d1);
```

```
  dbms_output.put_line(d1);
```

```
  dbms_stats.convert_raw_value(r6, d2);
```

```
  dbms_output.put_line(d2);
```

```
END;
```

```
/
```

```
21-APR-2010 00:00:00
```

```
10-JUL-2010 00:00:00
```

```
21-APR-2010 00:00:00
```

```
24-MAY-2010 00:00:00
```

```
09-JUL-2010 00:00:00
```

```
09-AUG-2010 15:00:00
```

Setting High and Low Values



```
DECLARE
  sr_arr  dbms_stats.statrec;
  new_low  DATE := TO_DATE('01-JAN-2002');
  new_high DATE := TO_DATE('31-DEC-2012');
  vals_arr dbms_stats.datearray;
BEGIN
  sr_arr.eavs := 0;
  sr_arr.chvals := NULL;
  sr_arr.epc := 2;

  vals_arr := dbms_stats.datearray(new_low, new_high);

  sr_arr.bkvals := dbms_stats.numarray(10000,1000000);
  dbms_stats.prepare_column_values(sr_arr, vals_arr);
  dbms_stats.set_column_stats(USER, 'PREPTEST', 'LAST_DDL_TIME', NULL, srec => sr_arr);
END;
/
```

Discussion

Storage

Storage Related Issues



- VLDBs tend to be DSS or DW systems
- A lot of total nonsense has been written about using big block sizes (32K)
 - Is it a RAC cluster?
 - Will it ever be a RAC cluster?
- One tablespace per day
 - You are not going to load a dev or test environment without Transportable Tablespaces
- Advanced Compression (255 column maximum width)
- AUD\$ Table
 - DBSNMP queries are not optimized
 - Storage defaults are not optimized

```
SQL> select pct_free, pct_used from dba_tables where table_name = 'AUD$';
```

PCT_FREE	PCT_USED
10	40

Buy the right hardware

- Bigger <> Better



**Runs Oracle
10x Faster***

The World's Fastest Database Machine

- Hardware by Sun
- Software by Oracle

* But you have to be willing to spend 50% less on hardware.

ORACLE®

10x faster based on comparing Oracle data warehouses on customer systems vs. Oracle Exadata Database Machines. Potential savings based on total hardware costs. Oracle Database and options licenses not included. Actual results and savings may vary.

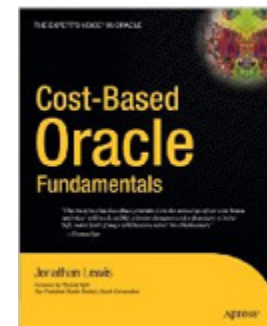
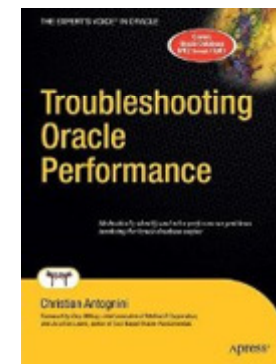
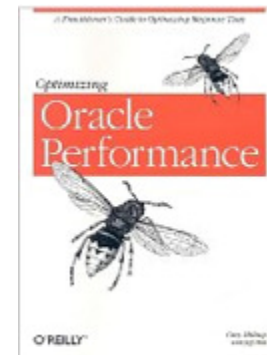
Copyright © 2010, Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates.

HOW: ExV2, 10x Faster, 2471

Resources



- <http://tahiti.oracle.com>
- <http://technet.oracle.com> - OTN Forums
- Indexing ... Richard Foote
- Books
 - Cary Millsap
 - Christian Antognini
 - Jonathan Lewis



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



All demos at morganslibrary.org

Thank you