



MAKLEE

software engineering
solutions

Oracle IO Performance

Christian Moser
VP Technology
Maklee Engineering
cmos@maklee.com



Preface

- The following slides contain tips to improve the throughput of the Oracle I/O sub-system.
- Most systems utilize only small portion of their I/O bandwidth.
 - Maklee specializes in tuning Oracle databases to achieve optimal throughput.
 - In a recent benchmark Maklee was able to complete a full table scan of a 1TB table in 56 seconds !!
- Some information has been censored to protect Maklee's proprietary knowledge.





Facts

- Reading from memory is fast
- Disk I/O is an extremely slow operation
- Storage hardware has improved over the past years
 - Faster disks
 - Faster controllers
 - Fibrechannel (2Gb, 4Gb, 8Gb etc)
 - Infiniband
 - Solid-state disks (FusionIO etc)
- But the best hardware does not help you unless you start using it to your advantage





IO Performance

- How to improve IO performance
 - Asynchronous IOs
 - Direct IOs
 - Consolidate IOs (larger and fewer)

- Golden rule
 - The fastest IOs are those avoided

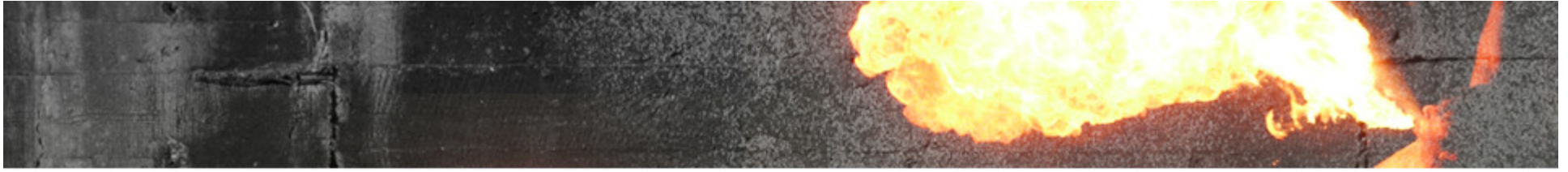




Oracle Considerations

- Tablespace, tables and indexes
 - Ensure that adequate initial and next extent sizes are specified during tablespace creation to avoid fragmentation and issues during space extension
 - Similar use larger initial and next extents for large tables and indexes
 - Default values are way too small
 - Might consider using “big” datafiles, which do not have a 32 GB limit on filesize





I/O Throughput



MAKLEE

Measure Oracle IO Throughput

- Oracle 11g includes IO calibration and also ORION (ORacle Input and Output Numbers) tool
- ORION can also be downloaded separately from OTN
 - <http://www.oracle.com/technetwork/topics/index-089595.html>



IO Calibration

- Run the following IO calibration query

```
set serveroutput on
declare
  lat integer;
  iops integer;
  mbps integer;
begin
  -- dbms_resource_manager.calibrate_io (<disks>, <max_latency>, iops,mbps,lat);
  dbms_resource_manager.calibrate_io (8, 10, iops, mbps, lat);

  dbms_output.put_line ('max_iops = ' || iops);
  dbms_output.put_line ('latency = ' || lat);
  dbms_output.put_line ('max_mbps = ' || mbps);
end;
/
```



IO Calibration (cont'd)

- Extract results

```
SQL> exec print_table('select * from dba_rsrc_io_calibrate');
```

```
START_TIME           : 28-NOV-10 11.31.14.142831 PM
END_TIME             : 28-NOV-10 11.46.42.929611 PM
MAX_IOPS              : 54839
MAX_MBPS              : 1957
MAX_PMBPS             : 1622
LATENCY               : 10
NUM_PHYSICAL_DISKS   : 8
```

- max_iops - max data block read req/sec
- max_mbps - max MB/sec of max-size read req
- max_pmbps - max MB/sec large IO req for single process



IO Calibration (cont'd)

- Check stats for each datafile

```
SQL> select file_id, file_name, file_no, asynch_io
       from dba_data_files, v$iostat_file where file_id=file_no;
```

FILE_ID	FILE_NAME	FILE_NO	ASYNCH_IO
1	+DATA/datafile/system.256.735392153	1	ASYNCH_ON
2	+DATA/datafile/sysaux.257.735392153	2	ASYNCH_ON
3	+DATA/datafile/undotbs1.258.735392153	3	ASYNCH_ON
4	+DATA/datafile/users.259.735392153	4	ASYNCH_ON
5	+DATA/undotbs3.dbf	5	ASYNCH_ON
6	+DATA_XP/db2/tb_test_small.dbf	6	ASYNCH_ON
7	+DATA_XP/db2/tb_test_large.dbf	7	ASYNCH_ON
8	+DATA_XP/db2/i_tb_test_small.dbf	8	ASYNCH_ON
9	+ARCH/db2/cmos_8k.dbf	9	ASYNCH_ON
10	+ARCH/db2/cmos_8k_.dbf	10	ASYNCH_ON
11	+DATA_XP/db2/i_tb_test_large.dbf	11	ASYNCH_ON
12	/cmos/db1/cmos_fs.dbf	12	ASYNCH_OFF



I/O Throughput - Example

- An XP storage array, connected via 8 Fibrechannel 4Gb ports
 - Theoretical maximum throughput is $8 \times 4\text{Gb/s} = 32\text{Gb/s} = 4\text{GB/s}$
- Measured maximum I/O throughput
 - 3 GB/sec using IOsize of 2 MB

```
# $ORACLE_HOME/bin/orion -run dss -simulate raid0 -duration 20  
-size_small 2048 -size_large 2048
```

```
ORION: ORacle IO Numbers -- Version 11.2.0.2.0
```

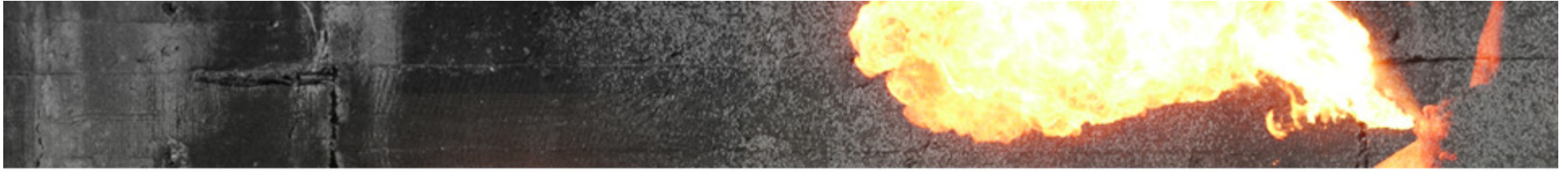
```
orion_20101202_1005
```

```
Calibration will take approximately 16 minutes.
```

```
Using a large value for -cache_size may take longer.
```

```
Maximum Large MBPS=2976.92 @ Small=0 and Large=448
```





ASM or Filesystem



MAKLEE

ASM or Filesystem?

- Compare performance of placing Oracle datafiles into ASM or onto filesystems
 - HP-UX 11.33
 - XP storage array
 - Physical LUN consist of RAID10 with 8 spindles
 - vxfs filesystem
 - Stripe 8 physical LUN's into logical volume
 - ASM
 - Striped over 8 physical LUN's

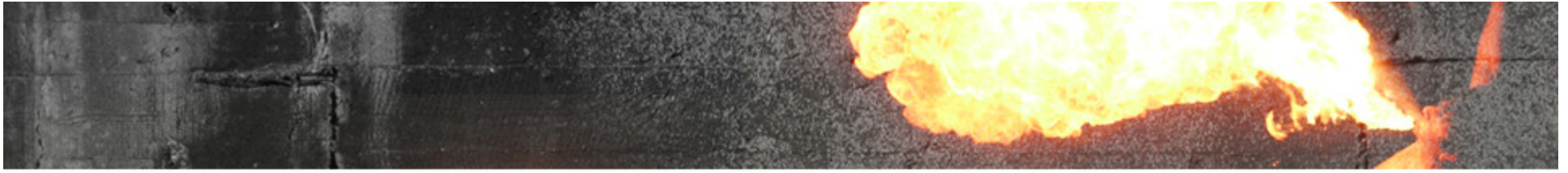


ASM or Filesystem? (cont'd)

IO Load	Filesystem	ASM
Large seq read	724 MB/sec	1495 MB/sec
Large random read	470 MB/sec	752 MB/sec
Large random write	1496 MB/sec	1491 MB/sec
Small random write	1097 MB/sec	1094 MB/sec

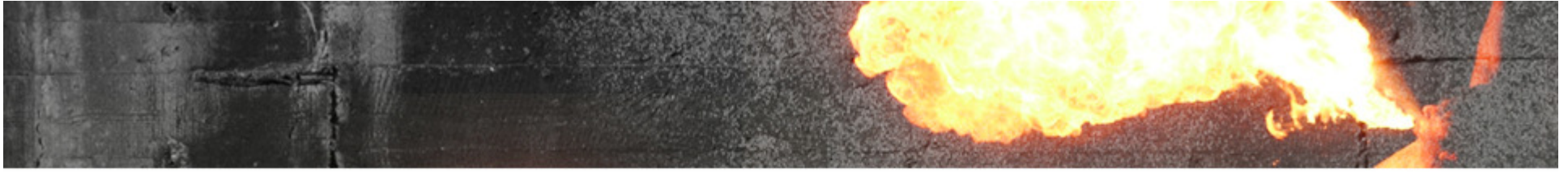
- Writes throughput is the same
 - No surprise here – issue write and forget, controller will handle
- Read via filesystem is significantly slower





The following slides demonstrate
the performance improvement
that could be achieved by tuning
the I/O sub-system





*Example 1 –
Tempfile access*



I/O Size to Tempfile

- If you cannot avoid writing data to tempfile and reading it back then make sure to use larger IOs

	Before Tuning	After Tuning
Direct path read & write temp (*)	34% of DB time	18% of DB time
Tempfile throughput (**)	96 MB/sec	140 MB/sec
Tempfile IOsize (**)	630 KB	2.5 MB

(*) less is better

(**) more is better



I/O size to Tempfile (cont'd)

- Before tuning

Top 5 Timed Foreground Events

Event	Waits	Time(s)	Avg wait (ms)	% DB time	Wait Class
direct path write temp	6,961	1,083	156	19.54	User I/O
DB CPU		986		17.79	
direct path write	28,215	855	30	15.42	User I/O
direct path read temp	10,754	785	73	14.16	User I/O
direct path read	841	400	476	7.22	User I/O



Filetype Name	Reads: Data	Reqs per sec	Data per sec	Writes: Data	Reqs per sec	Data per sec	Small Read	Large Read
Temp File	12.9G	156.67	96.5609	12.9G	156.68	96.5609		75.92
Data File	7.3G	68.18	54.2334	8.1G	242.17	60.4345	10.34	156.52
Log File	0M	0.00	0M	7M	3.15	.051067		
Control File	4M	1.85	.029181	2M	0.73	.014590	12.74	
TOTAL:	20.2G	226.70	150.823	21G	402.73	157.061	10.46	89.08



Tablespace	Reads	Av Reads/s	Av Rd(ms)	Av Blks/Rd	Writes	Av Writes/s	Buffer Waits	Av Buf Wt(ms)
TEMP_BIG	21,468	157	0.01	78.87	21,469	157	0	0.00
EXADATA_TEST_LARGE	5,084	37	65.64	185.62	33,202	242	22	0.45
SYSTEM	3,865	28	9.18	1.13	0	0	8	0.00
SYSAUX	625	5	17.28	2.03	2	0	0	0.00
UNDOTBS1	25	0	11.20	1.00	0	0	0	0.00
EXADATA_TEST_SMALL	2	0	5.00	1.00	0	0	0	0.00



I/O size to Tempfile (cont'd)

- After tuning

Top 5 Timed Foreground Events

Event	Waits	Time(s)	Avg wait (ms)	% DB time	Wait Class
DB CPU		996		18.55	
direct path read	847	996	1176	18.54	User I/O
direct path write	28,059	908	32	16.90	User I/O
direct path read temp	2,828	501	177	9.33	User I/O
direct path write temp	702	481	685	8.95	User I/O

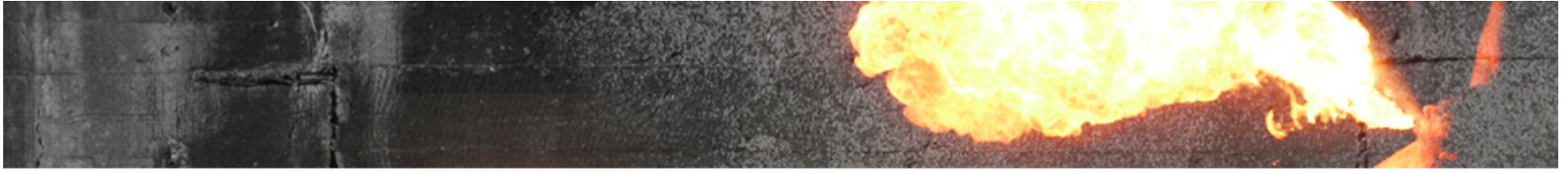


Filetype Name	Reads: Data	Reqs per sec	Data per sec	Writes: Data	Reqs per sec	Data per sec	Small Read	Large Read
Temp File	13.2G	55.73	140.311	13.2G	55.71	140.228		262.71
Data File	7.3G	90.11	76.9358	8.1G	343.83	85.8134	3.19	260.42
Log File	0M	0.00	0M	7M	4.32	.072512		
Control File	3M	2.10	.031076	1M	0.54	.010358	27.92	
TOTAL:	20.5G	147.95	217.278	21.3G	404.40	226.125	4.25	261.70



Tablespace	Reads	Av Reads/s	Av Rd(ms)	Av Blks/Rd	Writes	Av Writes/s	Buffer Waits	Av Buf Wt(ms)
EXADATA_TEST_LARGE	5,086	53	63.11	185.83	33,201	344	0	0.00
TEMP_BIG	5,374	56	0.02	322.14	5,366	56	0	0.00
SYSTEM	3,882	40	0.89	1.26	0	0	35	4.57
SYSAUX	534	6	2.92	2.01	1	0	0	0.00
UNDOTBS1	45	0	46.89	1.00	0	0	0	0.00
EXADATA_TEST_SMALL	2	0	5.00	1.00	0	0	0	0.00





*Example 2 –
Latency would impact your
throughput*



Slow Disk I/O

- Oracle
 - 235,027 reads * 62 blocks/read * 8K blocksize = 116 GB
 - IOsize is 496 KB (62 blocks * 8K)
- Disk Response time is way too long
 - Average time to read 496 KB is 56.82 msec

Tablespace IO Stats

• ordered by IOs (Reads + Writes) desc

Tablespace	Reads	Av Reads/s	Av Rd(ms)	Av Blks/Rd	Writes	Av Writes/s	Buffer Waits	Av Buf Wt(ms)
TEMP	1,773,780	3,199	0.56	1.00	28,495	51	0	0.00
CMOS_8K	235,027	424	56.82	62.01	0	0	19,801	12.57
EAGLE_IND1_CONSIGNMENT_LOC	79	0	6.58	1.00	59,285	107	0	0.00
SYSTEM	443	1	6.16	1.00	53	0	0	0.00
SYSAUX	125	0	6.80	1.00	291	1	0	0.00
UNDOTBS	0	0	0.00	0.00	144	0	8	0.00
DTM_TAB_LOC	0	0	0.00	0.00	4	0	0	0.00

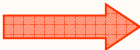


Slow Disk I/O (cont'd)

- Comparison
 - Below is the data from another large customer with similar storage array.
 - Average time to read 960 KB is only 16.67 msec
 - Almost 7x faster !!!

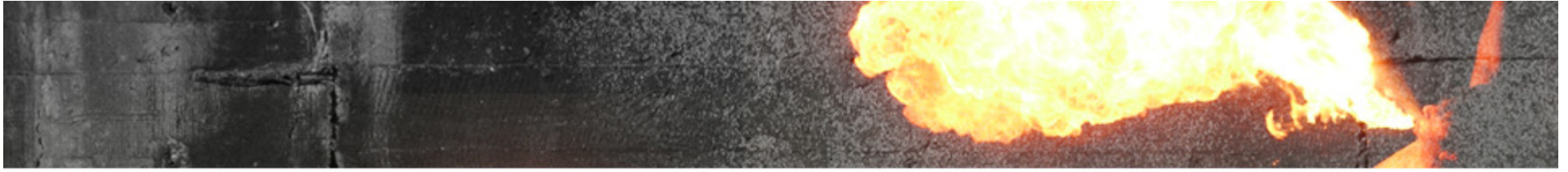
Tablespace IO Stats

• ordered by IOs (Reads + Writes) desc



Tablespace	Reads	Av Reads/s	Av Rd(ms)	Av Blks/Rd	Writes	Av Writes/s	Buffer Waits	Av Buf Wt(ms)
DLDW_STG_DATA	79,272	88	16.67	120.25	1,490,245	1,647	2,921	0.26
TEMP	548,084	606	0.03	18.93	439,971	486	0	0.00
UNDOTBS2	12	0	0.00	1.00	37,840	42	8,904	0.10





*Example 3 –
1 TB in less than 1 min*





Maximum Throughput

- System configuration
 - 2 node RAC configuration
 - Multiple Infiniband adapters per system
- Very large table
 - 1,440 million rows
 - 220 GB in size
- Test query
 - Full table scan
 - High degree of parallelism
 - Large IOs



Maximum Throughput (cont'd)

- Single instance
 - Query finishes in 22 sec
 - I/O rate is 10 GB/sec
- Both RAC instances
 - Query finishes now in 12.69 seconds
 - I/O rate is almost 18 GB/sec
- **With this configuration you can read a 1 TB table in less than 1 min**
- Executing multiple queries simultaneously can push the I/O throughput rate even higher
 - See next slide for results



Maximum Throughput (cont'd)

- Single instance

IOStat by Function (per Second)

- Total Reads includes all Functions: Buffer Cache, Direct Reads, ARCH, Data Pump, Others, RMAN, Recovery, Streams/AQ and XDB
- Total Writes includes all Functions: DBWR, Direct Writes, LGWR, ARCH, Data Pump, Others, RMAN, Recovery, Streams/AQ and XDB

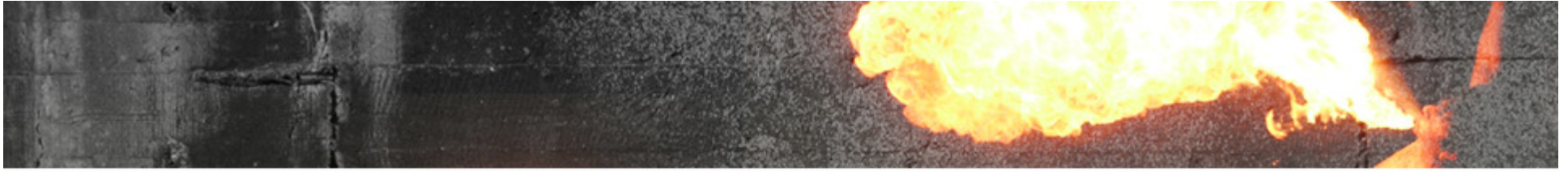
#	Reads MB/sec			Writes MB/sec				Reads requests/sec			Writes requests/sec			
	Total	Buffer Cache	Direct Reads	Total	DBWR	Direct Writes	LGWR	Total	Buffer Cache	Direct Reads	Total	DBWR	Direct Writes	LGWR
1	13,540.29	0.04	13,539.89	0.08	0.00	0.00	0.04	634.28	4.37	606.27	2.24	0.00	0.00	0.70
2	0.34	0.02	0.00	0.06	0.00	0.00	0.02	24.91	2.22	0.00	1.83	0.12	0.00	0.60
Sum	13,540.63	0.06	13,539.89	0.14	0.00	0.00	0.06	659.19	6.59	606.27	4.07	0.12	0.00	1.29
Avg	6,770.31	0.03	6,769.95	0.07	0.00	0.00	0.03	329.59	3.30	303.14	2.04	0.06	0.00	0.65

- Both RAC instances

IOStat by Function (per Second)

- Total Reads includes all Functions: Buffer Cache, Direct Reads, ARCH, Data Pump, Others, RMAN, Recovery, Streams/AQ and XDB
- Total Writes includes all Functions: DBWR, Direct Writes, LGWR, ARCH, Data Pump, Others, RMAN, Recovery, Streams/AQ and XDB

#	Reads MB/sec			Writes MB/sec				Reads requests/sec			Writes requests/sec			
	Total	Buffer Cache	Direct Reads	Total	DBWR	Direct Writes	LGWR	Total	Buffer Cache	Direct Reads	Total	DBWR	Direct Writes	LGWR
1	10,516.13	0.03	10,515.95	0.09	0.01	0.00	0.03	484.94	2.97	471.68	3.04	1.27	0.00	0.50
2	9,839.60	0.02	9,839.44	0.07	0.01	0.00	0.02	451.41	1.80	439.96	2.97	1.44	0.01	0.38
Sum	20,355.73	0.05	20,355.38	0.16	0.03	0.00	0.05	936.35	4.77	911.64	6.02	2.71	0.01	0.88
Avg	10,177.87	0.02	10,177.69	0.08	0.01	0.00	0.02	468.17	2.39	455.82	3.01	1.35	0.00	0.44

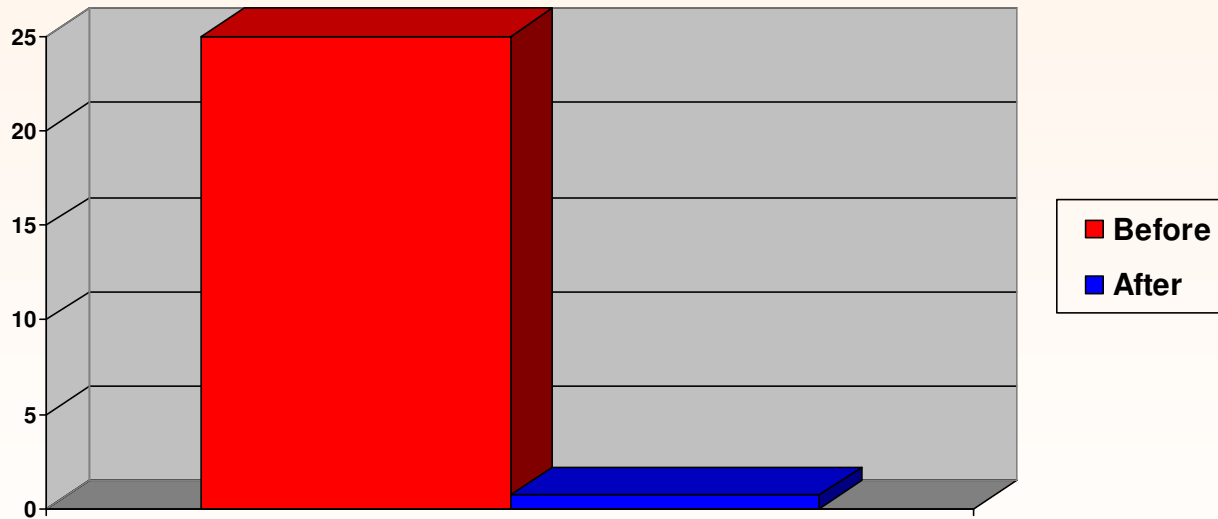


*Example 4 –
Fast table creation*



Rollup Table Creation

- Create table in parallel mode
- Avoid spilling hash-join to temp space
- Increase loader IO size limit



Elapsed time (in min) to create rollup table
Less is better



Rollup Table Creation (cont'd)

- Query creates rollup table using CTAS (create table as select)
- Source tables

Name	Size	Number of Rows
TABLE1	3.7 GB	30,977,309
TABLE2	1.6 GB	14,644,095
TABLE3	720 MB	13,300,427
TABLE4	7 KB	5

- Target table

Name	Size	Number of Rows
ROLLUP_TABLE	8.1 GB	29,683,110



Rollup Table Creation (cont'd)

- Reading source tables
 - Average IOSize 6.4 MB and throughput 110 MB/sec
- Writing target table to datafile
 - Average IOSize almost 24 MB and throughput almost 150 MB/sec

Filetype Name	Reads: Data	Reqs per sec	Data per sec	Writes: Data	Reqs per sec	Data per sec	Small Read	Large Read
Data File	6.1G	17.52	111.113	8.1G	7.51	147.173	1.96	32.74
Control File	4M	4.78	.071055	1M	0.96	.017763	0.45	
Log File	0M	0.00	0M	5M	6.18	.088819		
TOTAL:	6.1G	22.29	111.184	8.1G	14.66	147.280	0.83	32.74

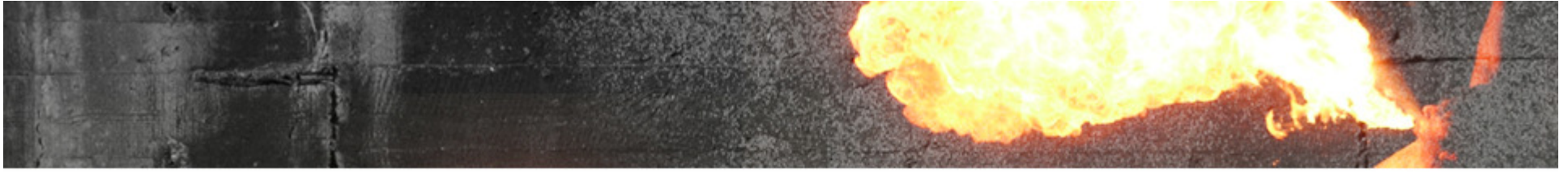
Tablespace	Reads	Av Reads/s	Av Rd(ms)	Av Blks/Rd	Writes	Av Writes/s	Buffer Waits	Av Buf Wt(ms)
CMOS_XP	968	17	0.04	825.90	352	6	43	2.33
UNDOTBS1	0	0	0.00	0.00	39	1	14	0.00
SYSTEM	0	0	0.00	0.00	27	0	242	0.12
SYSAUX	3	0	6.67	1.00	10	0	0	0.00
EXADATA_TEST_LARGE	1	0	0.00	1.00	1	0	0	0.00



Rollup Table Creation (cont'd)

Id	Operation	Name	Starts	E-Rows	A-Rows	A-Time	Buffers	Used-Mem
0	CREATE TABLE STATEMENT		1		32	00:00:44.64	24959	
1	PX COORDINATOR		1		32	00:00:44.64	24959	
2	PX SEND QC (RANDOM)	:TQ10006	0	14M	0	00:00:00.01	0	
3	LOAD AS SELECT		0		0	00:00:00.01	0	64M (0)
4	HASH UNIQUE		0	14M	0	00:00:00.01	0	317M (0)
5	PX RECEIVE		0	14M	0	00:00:00.01	0	
6	PX SEND HASH	:TQ10005	0	14M	0	00:00:00.01	0	
7	HASH UNIQUE		0	14M	0	00:00:00.01	0	317M (0)
* 8	HASH JOIN RIGHT OUTER		0	14M	0	00:00:00.01	0	35M (0)
9	PX RECEIVE		0	5	0	00:00:00.01	0	
10	PX SEND BROADCAST	:TQ10002	0	5	0	00:00:00.01	0	
11	PX BLOCK ITERATOR		0	5	0	00:00:00.01	0	
* 12	TABLE ACCESS FULL	TABLE4	0	5	0	00:00:00.01	0	
* 13	HASH JOIN		0	14M	0	00:00:00.01	0	90M (0)
14	JOIN FILTER CREATE	:BF0000	0	14M	0	00:00:00.01	0	
15	PX RECEIVE		0	14M	0	00:00:00.01	0	
16	PX SEND HASH	:TQ10003	0	14M	0	00:00:00.01	0	
* 17	HASH JOIN BUFFERED		0	14M	0	00:00:00.01	0	123M (0)
18	PX RECEIVE		0	13M	0	00:00:00.01	0	
19	PX SEND HASH	:TQ10000	0	13M	0	00:00:00.01	0	
20	PX BLOCK ITERATOR		0	13M	0	00:00:00.01	0	
* 21	TABLE ACCESS FULL	TABLE3	0	13M	0	00:00:00.01	0	
22	PX RECEIVE		0	14M	0	00:00:00.01	0	
23	PX SEND HASH	:TQ10001	0	14M	0	00:00:00.01	0	
24	PX BLOCK ITERATOR		0	14M	0	00:00:00.01	0	
* 25	TABLE ACCESS FULL	TABLE2	0	14M	0	00:00:00.01	0	
26	PX RECEIVE		0	29M	0	00:00:00.01	0	
27	PX SEND HASH	:TQ10004	0	29M	0	00:00:00.01	0	
28	JOIN FILTER USE	:BF0000	0	29M	0	00:00:00.01	0	
29	PX BLOCK ITERATOR		0	29M	0	00:00:00.01	0	
* 30	TABLE ACCESS FULL	TABLE1	0	29M	0	00:00:00.01	0	





*Example 5 –
Fast IOs to/from datafiles*



Producer Table

- Speed up a query where full table scans cannot be avoided since the whole table has to be processed
- Large IO sizes will increase throughput
- Moves a subset of columns from large source table into target table
 - PRODUCER_SOURCE 89 Mio rows, 13 GB size
 - PRODUCER_TRAGET 89 Mio rows, 6.4 GB size
- Elapsed time to process 89 Mio rows is less than 5 min

Id	Operation	Name	Starts	E-Rows	A-Rows	A-Time	Buffers
0	INSERT STATEMENT		1		0	00:04:50.02	1585K
1	LOAD AS SELECT		1		0	00:04:50.02	1585K
2	PX COORDINATOR		1		86M	00:01:54.92	17813
3	PX SEND QC (RANDOM)	:TQ10000	0	86M	0	00:00:00.01	0
4	PX BLOCK ITERATOR		0	86M	0	00:00:00.01	0
* 5	TABLE ACCESS FULL	APPOINTMENT_STAGE	0	86M	0	00:00:00.01	0



Producer Table (cont'd)

- Reading source table
 - Average IOsize almost 14 MB and throughput 49 MB/sec
- Writing target table to datafile
 - Average IOsize 12 MB and throughput 39 MB/sec

Top 5 Timed Foreground Events

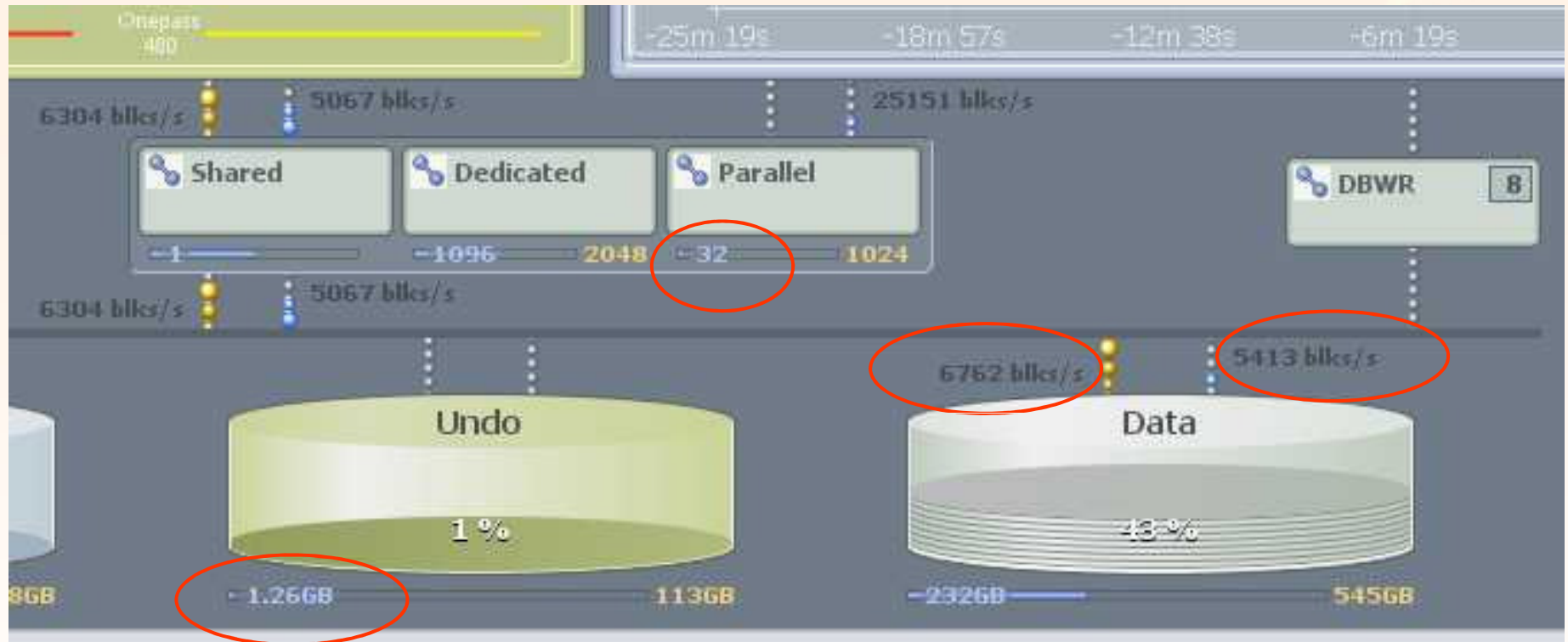
Event	Waits	Time(s)	Avg wait (ms)	% DB time	Wait Class
DB CPU		392		73.98	
direct path read	473	40	84	7.50	User I/O
control file sequential read	11,451	1	0	0.23	System I/O
cursor: pin S wait on X	30	1	22	0.12	Concurrency
db file sequential read	368	0	1	0.08	User I/O

Filetype Name	Reads: Data	Reqs per sec	Data per sec	Writes: Data	Reqs per sec	Data per sec	Small Read	Large Read
Data File	15.1G	4.00	49.0022	11.9G	4.63	38.6768	0.88	2076.23
Control File	195M	39.40	.617436	5M	0.97	.015831	0.01	
Log File	0M	0.04	0M	3M	1.09	.009499	0.00	
TOTAL:	15.3G	43.44	49.6197	11.9G	6.68	38.7021	0.04	2076.23

Tablespace	Reads	Av Reads/s	Av Rd(ms)	Av Blks/Rd	Writes	Av Writes/s	Buffer Waits	Av Buf Wt(ms)
EXADATA_TEST_LARGE	1,155	4	0.41	1714.95	975	3	0	0.00
SYSAUX	0	0	0.00	0.00	257	1	0	0.00
UNDOTBS1	0	0	0.00	0.00	213	1	2	0.00
SYSTEM	0	0	0.00	0.00	18	0	1	0.00



Producer Table (cont'd)



- 32 parallel jobs
- 1.26 GB undo space occupied
- 6762 blocks/sec reading from datafile
- 5413 blocks/sec writing to datafile