ORACLE

Oracle on z/OS and Linux on System z
Tuning experiences

Thomas Niewel
Principal Sales Consultant

Agenda

- Tuning Why ?
- Statspack
- AWR Report
- ASH Report
- SQL Tuning tools
  - TKPROF
  - Explain Plan
  - Trace Analyzer

Why do we need to tune ?

- Users report „bad“ response times because of
  - CPU Time + Wait Time
  - Poor performing queries
    - SQL-Tuning
  - „bad“ database parameters
  - Bottlenecks in „system“
    (Operating system, WLM, IO/Subsystem etc.)

Statspack

Statspack – a short overview

- spcreate.sql - installs Statspack (run only once)
- statspack.snap - data capture (procedure)
- spreport.sql - reporting
- sprepsql.sql - report for a specific SQL statement
- spdoc.txt - user documentation
- sppurge.sql - delete Statspack data
- spdrop.sql - drop Statspack

Capturing data

- Prerequisites
  - spcreate.sql with user PERFSTAT
  - Statistics_level=typical(default) / all
- Use stored procedure statspack.snap
  - SQL> execute statspack.snap;

Capturing data

- Get a baseline for future comparisons
- Capture snapshots
  - across peak load
  - across batch window
  - The time between snapshots should be 10-30 minutes depending on workload
- Capture can be automated
  - Use OS utility e.g. cron
  - Use dbms_scheduler
    - spauto.sql shipped as example

Statspack report

- All data is held in an Oracle database
- Report between two snapshots (may include more snapshops)
  - cannot report across instance startup
- spreport.sql creates a report

Statspack Snapshot Level

- **Levels >= 0**
  - General performance statistics Statistics gathered:
    This level and any level greater than 0 collects general performance statistics, such as: wait statistics, system events,system statistics, rollback segment data, row cache, SGA, background events, session events, lock statistics, buffer pool statistics,latch statistics, resource limit, enqueue statistics, and statistics for each of the following, if enabled: automatic undo management, buffer cache advisory data, auto PGA memory management, Cluster DB statistics.

Statspack Snapshot Level

- **Levels >= 5 Additional data: SQL Statements**
  - This level includes all statistics gathered in the lower level(s), and additionally gathers the performance data on high resource usage SQL statements.
    In a level 5 snapshot (or above), note that the time required for the snapshot to complete is dependent on the shared_pool_size and on the number of SQL statements in the shared pool at the time the snapshot is taken: the larger the shared pool, the longer the time taken to complete the snapshot.

Statspack Snapshot Level

- **Levels >= 6 Additional data: SQL Plans and SQL Plan usage**
  - This level includes all statistics gathered in the lower level(s), and additionally gathers optimizer execution plans, and plan usage data for each of the high resource usage SQL statements captured. A level 6 snapshot gathers information which is invaluable when determining whether the execution plan used for a SQL statement has changed. Therefore level 6 snapshots should be used whenever there is the possibility a plan may change, such as after large data loads, or after gathering new optimizer statistics.To capture the plan for a SQL statement, the statement must be in the shared pool at the time the snapshot is taken, and must exceed one of the SQL thresholds. To gather plans for all statements in the shared pool, you can temporarily specify the executions threshold (i_executions_th) to be zero (0) for those snapshots. For information on how to do this, see the 'Changing the default values for Snapshot Level and SQL Thresholds' section of this document.

Statspack Snapshot Level

- **Levels >= 7 Additional data: Segment level statistics**
  - This level includes all statistics gathered in the lower level(s), and additionally gathers the performance data on highly used segments.

    A level 7 snapshot captures Segment-level statistics for segments which are heavily accessed or heavily contended for.

Statspack Snapshot Level

- **Levels >= 7 Additional data: Segment level statistics**
  - Segment-level statistics captured are:
    - logical reads
    - db block changes
    - physical reads
    - physical writes
    - physical reads direct
    - physical writes direct
    - global cache cr blocks served
    - global cache current blocks served
    - buffer busy waits
    - ITL waits
    - row lock waits

Statspack Snapshot Level

- **Levels >= 10 Additional statistics: Parent and Child latches**
  - This level includes all statistics gathered in the lower levels, and additionally gathers Parent and Child Latch information. Data gathered at this level can sometimes cause the snapshot to take longer to complete i.e. this level can be resource intensive, and should only be used when advised by Oracle personnel.
  - Example
    - execute statspack.snap(i_snap_level=>6)
    - SQL> execute statspack.snap - (i_snap_level=>10, i_modify_parameter=>'true');

Statspack report

```
SQL>  @spreport

       DB Id DB Name      Instance# Instance
  ----------- ---------- ---------- ----------
   1361567071 DB21               1   MAIL

  Completed Snapshots

  Instance   DB Name    SnapId      Snap Started      Snap Level
  ---------- ---------- ------ -------------------- ----------
  DB21       DB21            1  17 Aug 2003 10:00:16           5
                             2  17 Aug 2003 10:30:28           5
  Enter beginning Snap Id: 1
  Enter ending    Snap Id: 2

  Enter name of output file [sp_1_2] : <enter name or return>
```

Analyze of a Statspack report

- Top down analysis
- Summary page
  - Enviroment
  - Load profile
  - Instance efficiency
  - Shared pool usage
  - Top 5 Timed Events
- Top SQL

Environment section

```
STATSPACK report for

DB Name         DB Id    Instance     Inst Num Release     Cluster Host
------------ ----------- ------------ -------- ----------- ------- ------------
RECONPRD      1403107896 RECONPRD            1 9.2.0.8.0   NO      lin390t1

            Snap Id     Snap Time      Sessions Curs/Sess Comment
            ------- ------------------ -------- --------- -------------------
Begin Snap:       2 03-Mar-03 11:28:01       10       5.1
  End Snap:      31 04-Mar-03 11:58:04       17       5.5
   Elapsed:               30.05 (mins)

Cache Sizes (end)
~~~~~~~~~~~~~~~~~
               Buffer Cache:       256M      Std Block Size:         16K
           Shared Pool Size:        48M          Log Buffer:        128K
```

Load profile

- Contains a number of common ratios
- Allows characterization of the application
- Can point to problems
  - high hard parse rate
  - high I/O rate
  - high login rate

Load profile

- Useful if you have a comparable baseline (MAKE_BASELINE)
- What has changed?
  - txn/sec change implies changed workload
  - redo size/txn implies changed transaction mix
  - physical reads/txn implies changed SQL or plan

Load profile

Load Profile

	Per Second	Per Transaction
Redo size:	19,057.68	20,937.67
Logical reads:	2,408.15	2,645.70
Block changes:	98.64	108.37
Physical reads:	990.47	1,088.18
Physical writes:	6.92	7.61
User calls:	76.40	83.93
Parses:	7.08	7.78
Hard parses:	0.02	0.02
Sorts:	29.22	32.10
Logons:	24.73	27.17
Executes:	63.79	70.08
Transactions:	0.91	

% Blocks changed per Read:	4.10	Recursive Call %:	72.76
Rollback per transaction %:	36.52	Rows per Sort:	153.46

Instance Efficiency

- Gives an overview of how the instance is performing
- Can also be used with a comparable baseline
- Shared pool Statistics allow quick identification of cursor sharing problems

Instance Efficiency

```
Instance Efficiency Percentages (Target 100%)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

            Buffer Nowait %:   99.99       Redo NoWait %:   99.97
            Buffer  Hit   %:   59.00    In-memory Sort %:   99.99
            Library Hit   %:   99.94        Soft Parse %:   99.69
         Execute to Parse %:   88.89         Latch Hit %:   99.98
Parse CPU to Parse Elapsd %:   56.55     % Non-Parse CPU:   99.93

 Shared Pool Statistics        Begin   End
                               ------  ------
             Memory usage %:   38.86   66.81
    % SQL with executions>1:   43.41   87.22
  % Memory for SQL w/exec>1:   39.28   80.21
```

Top 5 Timed Events

- CPU time – real work
- Shows where Oracle sessions are waiting
- Compare wait time to elapsed time
- % Total wait time shows potential benefits
- Use as basis for drilldown

Event	Waits	Time (s)	% Total Ela Time
CPU time		78,588	50.24
enqueue	1,560,523	59,961	38.33
db file sequential read	1,635,253	6,324	4.04
db file scattered read	14,620,725	5,907	3.78
control file parallel write	32,816	1,396	.89

Top 5 Timed Events

- Sample drilldowns
  - CPU Time „on CPU“
    (CPU used by this session)
  - enqueue
    e.g TX Enqueue
  - db file sequential read
    Index Access
  - db file scattered read
    Scan Operations

Top SQL

- Helps to find problem statements
  - SQL ordered by Gets
  - SQL ordered by Reads
  - SQL ordered by Executions
  - SQL ordered by Parse Calls
  - Ratio elapsed to CPU time

I/O Statistics

- Help to find I/O problems
  - Tablespace I/O statistics
  - File I/O statistics

I/O Statistics

Tablespace

	Reads	Av Reads/s	Av Rd(ms)	Av Blks/Rd	Writes	Av Writes/s	Buffer Waits	Av Buf Wt(ms)
GAH_TS00_DT_MEDIUM								
	15,242,896	160	0.4	6.1	41,066	0	22,468	18.4
GAH_TS00_IX_ITEM								
	210,346	2	11.2	1.0	130,299	1	9	15.6
GAH_TS00_IX_MEDIUM								
	207,433	2	6.9	1.0	86,699	1	39	43.8
RECONPRD_TS00_TEMP								
	185,865	2	1.7	1.6	101,560	1	0	0.0
GAH_TS00_IX_ITEM_REF								
	155,027	2	8.4	1.0	34,867	0	1	0.0

Oracle 10g
Automatic Workload Repository (AWR)

Automatic Workload Repository (AWR)

- Automatically collects database instance statistics
  - An “automated” STATSPACK with less overhead and enhanced functionality
  - On by default in Oracle Database 10g
  - Stores data in SYSAUX tablespace
  - Provides DBA_HIST (historical) views

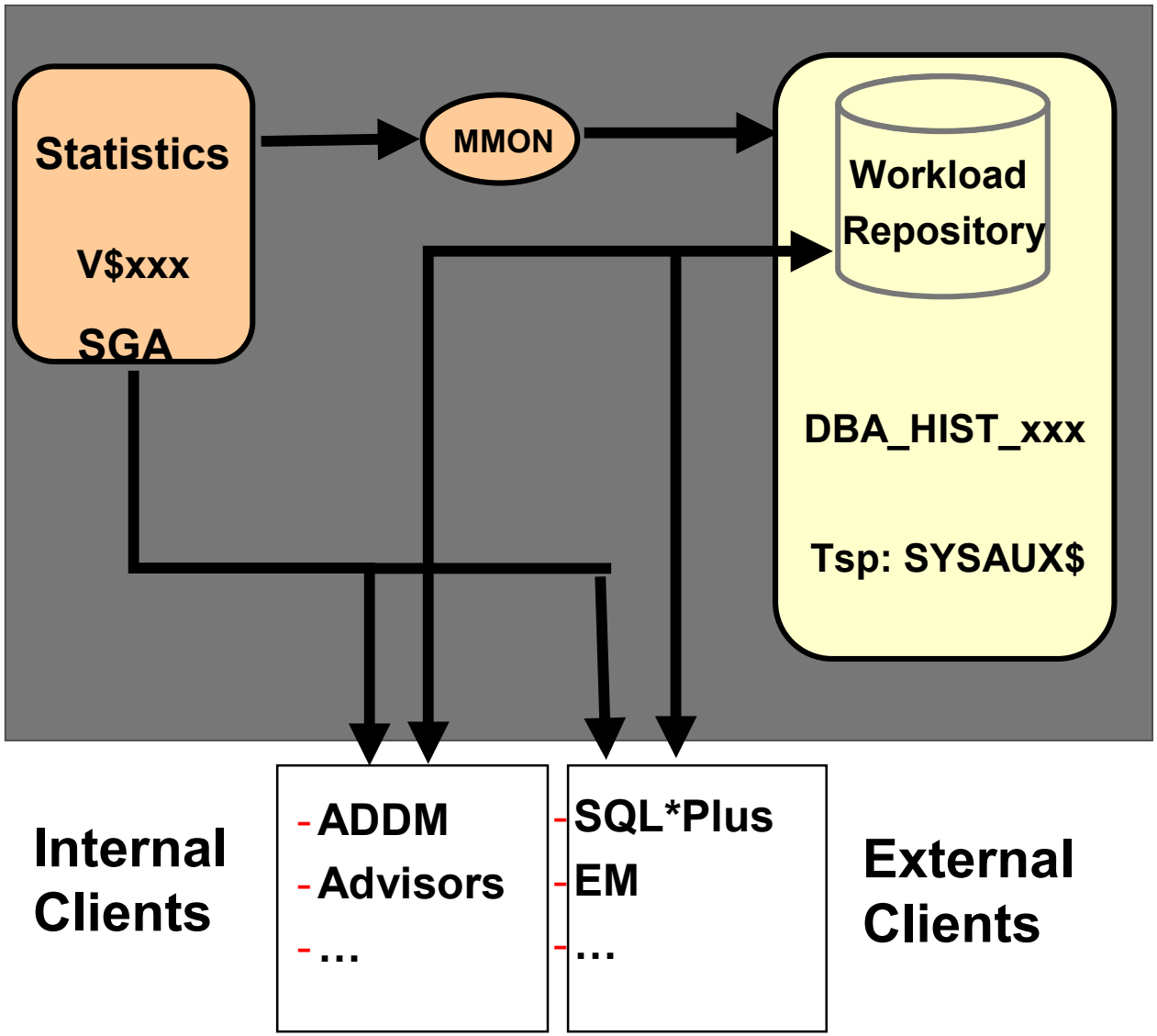
Automatic Workload Repository (AWR)

- Automatically collects database instance statistics
  - Licensed in the Diagnostics Packs
  - Captures statistical data
    - Used by
      - AWR-Reports
      - Oracle database advisors
      - self-management features
      - Coordinated across RAC instances

Automatic Workload Repository (AWR)

- Text and HTML Version available
- Reports can be generated / viewed by
  - OEM
  - Scripts
    - awrrpt.sql
    - awrrpti.sql
    - ashrpt.sql (10.2)
    - awrddrpt.sql (10.2)
    - awrsqrpt.sql
  - Contains the Statspack Information
    - ***Plus*** a lot of more Information

Automatic Workload Repository (AWR)



- Base Statistics, Metrics, SQL-Statistics,Active Session History
- Automatic Snapshots (Default 1h)
- "Historic" Data (Default 7 days)
- Automatic Space Management
- "Light Weight-Capture"

Oracle 10g SQL Statistics

- SQL_id – unique hash value
- SQL statement statistics
  - Wait class time
  - PLSQL time
  - Java time
- Sampled bind values (v$sql_bind_capture)
  - Default=900 seconds
- Efficient top SQL identification using Δs in the kernel, by 6 dimensions:
  - CPU
  - Elapsed
  - Parse
  - ...

Automatic Workload Repository (AWR)

- Creating Snapshots

```
BEGIN
DBMS_WORKLOAD_REPOSITORY.CREATE_SNAPSHOT ();
END;
/
```

- Dropping Snapshots

```
BEGIN
DBMS_WORKLOAD_REPOSITORY.DROP_SNAPSHOT_RANGE (low_snap_id => 22,
high_snap_id => 32, dbid => 3310949047);
END;
/
```

- Modifying Snapshot Settings

```
DBMS_WORKLOAD_REPOSITORY.MODIFY_SNAPSHOT_SETTINGS( retention => 43200,interval
   => 30, dbid => 3310949047);
END
```

- Dropping Baselines

```
BEGIN
DBMS_WORKLOAD_REPOSITORY.DROP_BASELINE (baseline_name => 'peak
   baseline',
cascade => FALSE, dbid => 3310949047);
END;
/
```

AWR report

WORKLOAD REPOSITORY report for

DB Name	DB Id	Instance	Inst Num	Release	Cluster	Host
DB009	2644935458	DE21001	1	10.1.0.3.21	NO	MVS09

	Snap Id	Snap Time	Sessions	Curs/Sess
Begin Snap:	6201	21-Mar-06 00:15:06	20	2.9
End Snap:	6202	21-Mar-06 00:30:43	22	2.9
Elapsed:		15.62 (mins)		
DB Time:		18.74 (mins)		

Cache Sizes (end)

Buffer Cache:	24M	Std Block Size:	4K
Shared Pool Size:	144M	Log Buffer:	256K

AWR report

Load Profile

	Per Second	Per Transaction
Redo size:	7,582.46	182,948.00
Logical reads:	108,175.09	2,610,024.44
Block changes:	43.51	1,049.89
Physical reads:	0.71	17.11
Physical writes:	0.58	14.11
User calls:	0.28	6.67
Parses:	0.79	19.00
Hard parses:	0.04	1.00
Sorts:	0.25	6.00
Logons:	0.01	0.33
Executes:	21.32	514.33
Transactions:	0.04	

% Blocks changed per Read:	0.04	Recursive Call %:	100.00
Rollback per transaction %:	0.00	Rows per Sort:	163.81

AWR report

Instance Efficiency Percentages (Target 100%)

~~~~~~

Buffer Nowait %:	100.00	Redo NoWait %:	100.00
Buffer Hit %:	100.00	In-memory Sort %:	100.00
Library Hit %:	99.40	Soft Parse %:	94.74
Execute to Parse %:	96.31	Latch Hit %:	100.00
Parse CPU to Parse Elapsd %:	100.00	% Non-Parse CPU:	99.97

Shared Pool Statistics	Begin	End
Memory usage %:	82.47	83.09
% SQL with executions>1:	90.00	90.39
% Memory for SQL w/exec>1:	87.31	87.02
~~~~~~

AWR report

Top 5 Timed Events

Event	Waits	Time (s)	% Total DB Time	Wait Class
CPU time		353	60.31	
enq: TX - row lock contention	60	193	32.95	Application
latch: cache buffers chains	82	9	1.60	Concurrency
class slave wait	1	5	.89	Other
Queue Monitor Task Wait	1	5	.89	Other

Wait Events DB/Inst: DB009/DE21001 Snaps: 6201-6202

-> s - second

-> cs - centisecond - 100th of a second

-> ms - millisecond - 1000th of a second

-> us - microsecond - 1000000th of a second

-> ordered by wait time desc, waits desc (idle events last)

AWR report

- AWR Wait Classes
  - **Administration** - **backups, index rebuilds,…**
  - **Application** - **row/table locks, user locks,..**
  - **Cluster** - **RAC waits,…**
  - **Commit** - **log file sync,…**
  - **Concurrency** - **buffer busy, latches,…**
  - **Configuration** - **free buffer waits,…**
  - **Idle** - **rdbms ipc msg, smon timer,…**
  - **Network** - **Oracle Net**
  - **Scheduler** - **Resource manager,…**
  - **System I/O** - **log file parallel write,…**
  - **User I/O** - **reads, direct writes,…**
  - **Other** - **Miscellaneous waits**

AWR report

Time Model Statistics DB/Inst: DB009/DE21001 Snaps: 6201-6202

-> ordered by Time (seconds) desc

Statistic Name	Time (seconds)	% Total DB Time
DB time	584.47	100.00
sql execute elapsed time	584.26	99.96
DB CPU	352.52	60.31
PL/SQL execution elapsed time	31.14	5.33
background elapsed time	11.57	1.98
background cpu time	.59	.10
connection management call elapsed time	.06	.01
parse time elapsed	.06	.01
hard parse elapsed time	.04	.01
PL/SQL compilation elapsed time	.03	.00
hard parse (sharing criteria) elapsed time	.00	.00
Java execution elapsed time	.00	.00
hard parse (bind mismatch) elapsed time	.00	.00

AWR report

SQL ordered by Elapsed Time DB/Inst: DB009/DE21001 Snaps: 6201-6202

-> Resources reported for PL/SQL code includes the resources used by all SQL statements called by the code.

-> % Total DB Time is the Elapsed Time of the SQL statement divided into the Total Database Time multiplied by 100

Elapsed Time (s)	CPU Time (s)	Executions	Elap per Exec (s)	% Total DB Time	SQL Id
141	132	4,370	0.0	24.1	31c7r47m8p1vt

Module: SQL*Plus

SQL ordered by CPU Time DB/Inst: DB009/DE21001 Snaps: 6201-6202

SQL ordered by Gets DB/Inst: DB009/DE21001 Snaps: 6201-6202

SQL ordered by Reads DB/Inst: DB009/DE21001 Snaps: 6201-6202

SQL ordered by Executions DB/Inst: DB009/DE21001 Snaps: 6201-6202

SQL ordered by Parse Calls DB/Inst: DB009/DE21001 Snaps: 6201-6202

SQL ordered by Sharable Memory DB/Inst: DB009/DE21001 Snaps: 6201-6202

SQL ordered by Version Count DB/Inst: DB009/DE21001 Snaps: 6201-6202

AWR report

Statistic	Total	per Second	per Trans
table scans (short tables)	4,378	20.2	486.4
transaction rollbacks	0	0.0	0.0
transaction tables consistent re	0	0.0	0.0
transaction tables consistent re	0	0.0	0.0
undo change vector size	502,320	2,313.2	55,813.3
user calls	60	0.3	6.7
user commits	9	0.0	1.0
user rollbacks	0	0.0	0.0
user I/O wait time	22	0.1	2.4
workarea executions - onepass	0	0.0	0.0
workarea executions - optimal	44	0.2	4.9
write clones created in foregrou	0	0.0	0.0
Cached Commit SCN referenced	0	0.0	0.0
Commit SCN cached	0	0.0	0.0
CPU used by this session	3366	150.3	7.3
CPU used when call started	3366	150.3	7.3
CR blocks created	10	0.1	1.1

AWR report

Tablespace IO Stats DB/Inst: DB009/DE21001 Snaps: 6201-6202

-> 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)
USER01	345107	334	1.0	1.0	0	0	0	0.0
USER02	34	0	2.9	2.6	8	0	3	10.0
UNDO1	0	0	0.0	.0	9	0	0	0.0

I/O Rules of thumb

- dbfile sequential read < 10ms
- dbfile scattered read 10 - 30ms (dependant on I/O-Size)
- log file parallel write < 5ms (into disk cache)
- dbfile parallel write < 5ms (into disk cache)
  - not as important in case of asynchronous I/O

AWR report

Enqueue Activity DB/Inst: ACCST/ACCST1 Snaps: 6177-6178

-> only enqueues with waits are shown

-> Enqueue stats gathered prior to 10g should not be compared with 10g data

-> ordered by Wait Time desc, Waits desc

Enqueue Type (Request Reason)

Requests	Succ Gets	Failed Gets	Waits	Wt Time (s)	Av Wt Time(ms)
WF-AWR Flush					
13	13	0	8	1	167.50
US-Undo Segment					
984	984	0	475	0	.55
PS-PX Process Reservation					
50	46	4	14	0	1.43
TT-Tablespace					
14	14	0	8	0	.00
HW-Segment High Water Mark					
506	506	0	6	0	.00

AWR report

Enqueue Activity DB/Inst: DB009/DE21001 Snaps: 6201-6202

No data exists for this section of the report.

Undo Segment Summary DB/Inst: DB009/DE21001 Snaps: 6201-6202

No data exists for this section of the report.

Undo Segment Stats DB/Inst: DB009/DE21001 Snaps: 6201-6202

No data exists for this section of the report.

AWR report

Latch Activity DB/Inst: ACCST/ACCST1 Snaps: 6177-6178

-> "Get Requests", "Pct Get Miss" and "Avg Slps/Miss" are statistics for willing-to-wait latch get requests

-> "NoWait Requests", "Pct NoWait Miss" are for no-wait latch get requests

-> "Pct Misses" for both should be very close to 0.0

Latch Name	Get Requests	Pct Get Miss	Avg Slps /Miss	Wait Time (s)	NoWait Requests	Pct NoWait Miss
ASM allocation	2	0.0	N/A	0	0	N/A
ASM map operation freeli	13	0.0	N/A	0	0	N/A
ASM map operation hash t	6,150	0.0	N/A	0	0	N/A
ASM network background l	177	0.0	N/A	0	0	N/A
AWR Alerted Metric Eleme	1,683	0.0	N/A	0	0	N/A
Consistent RBA	34	0.0	N/A	0	0	N/A
FAL request queue	9	0.0	N/A	0	0	N/A
FAL subheap alocation	9	0.0	N/A	0	0	N/A

AWR report

- AWR collects V$SEGSTAT statistics for hot segments (tables, indexes, etc.)
- Top segments are determined by
  - Logical and physical reads
  - Wait count (sum of ITL, row lock, buffer busy)
  - RAC interconnect activity
  - Size change over last snapshot period
  - Access of chained rows

AWR report

Segments by Logical Reads DB/Inst: DB009/DE21001 Snaps: 6201-6202

-> % Total shows % of logical reads for each top segment compared with total logical reads for all segments captured by the Snapshot

Owner	Tablespace Name	Object Name	Subobject Name	Obj. Type	Logical Reads	%Total
SCOTT	USER	EMP_TN		TABLE	23,435,072	99.96
SYS	SYSAUX	WRH$_SYSSTAT_PK	35458_6200	INDEX	736	.00
SYS	SYSAUX	WRH$_LATCH_PK	35458_6200	INDEX	656	.00
SYS	SYSAUX	WRH$_SQLBIND_PK	35458_6200	INDEX	656	.00
SYS	SYSAUX	WRH$_PARAMETER_PK	35458_6200	INDEX	496	.00

Segments by Physical Reads DB/Inst: DB009/DE21001 Snaps: 6201-6202

RAC-Statistics

```
Global Cache Load Profile
~~~~~~~~~~~~~~~~~~~~~~~~~                  Per Second       Per Transaction
                                        ---------------       ---------------
  Global Cache blocks received:                 1.20                  2.60
    Global Cache blocks served:                 1.40                  3.02
     GCS/GES messages received:                18.42                 39.74
         GCS/GES messages sent:                17.57                 37.91
            DBWR Fusion writes:                 0.00                  0.00
 Estd Interconnect traffic (KB)                27.88


Global Cache Efficiency Percentages (Target local+remote 100%)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Buffer access -  local cache %:  100.00
Buffer access - remote cache %:    0.00
Buffer access -         disk %:    0.00
```

RAC Statistics

```
Global Cache and Enqueue Services - Workload Characteristics
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                    Avg global enqueue get time (ms):      0.7

         Avg global cache cr block receive time (ms):      0.5
    Avg global cache current block receive time (ms):      0.6

           Avg global cache cr block build time (ms):      0.0
            Avg global cache cr block send time (ms):      0.0
     Global cache log flushes for cr blocks served %:      2.0
           Avg global cache cr block flush time (ms):      5.0

        Avg global cache current block pin time (ms):      0.0
       Avg global cache current block send time (ms):      0.0
Global cache log flushes for current blocks served %:      2.6
      Avg global cache current block flush time (ms):      5.0

Global Cache and Enqueue Services - Messaging Statistics
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                    Avg message sent queue time (ms):      0.4
             Avg message sent queue time on ksxp (ms):      0.3
                 Avg message received queue time (ms):      3.6
                    Avg GCS message process time (ms):      0.0
                    Avg GES message process time (ms):      0.0

                          % of direct sent messages:    55.19
                        % of indirect sent messages:    37.94
                     % of flow controlled messages:     6.88
          -------------------------------------------------------------
```

RAC Statistics

- Are there any GC events in the top 5 events ?
- Block Transfer Times
  - AVG global cache current Block Receive Time (ms)
    - < 5ms
  - Avg global cache cr block receive time (ms)
    - CR Block Receive Time=global cache cr block receive time / global cache cr blocks received
    - < 2ms
- Estd Interconnect traffic (KB)
  - 1GB Ethernet line 70 MB/sec

RAC Statistics

- Avg global cache current block pin time (ms) and Global cache transfer stats
  - congested Blocks
  - CPU Bottleneck ?
- Avg global cache current block flush time (ms): (> 2ms)
  - I/O Bottleneck ?

Workload Repository Compare Report

WORKLOAD REPOSITORY COMPARE PERIOD REPORT

Snapshot Set	DB Name	DB Id	Instance	Inst num	Release	Cluster	Host
First (1st)	ORAI1	1154309737	DE23	1	10.2.0.2.0	YES	MVS09
Second (2nd)	ORAI1	1154309737	DE23	1	10.2.0.2.0	YES	MVS09

Snapshot Set	Begin Snap Id	Begin Snap Time	End Snap Id	End Snap Time	Elapsed Time (min)	DB Time (min)	Avg Active Users
1st	3302	21-Mar-07 10:03:58	3303	21-Mar-07 10:11:13	15.01	14.90	2.06
2nd	3323	22-Mar-07 05:33:53	3324	22-Mar-07 05:52:12	15.00	5.30	0.73

Configuration Comparison

	1st	2nd	%Diff
Buffer Cache:	100M	100M	0.00
Std Block Size:	4K	4K	0.00
Shared Pool Size:	108M	108M	0.00
Log Buffer:	2,648K	2,648K	0.00
SGA Target:	0	0	0.00
PGA Aggregate Target:	25M	25M	0.00
Undo Management:	AUTO	AUTO	

Load Profile

	1st Per Sec	2nd Per Sec	%Diff	1st Per Txn	2nd Per Txn	%Diff
Redo size:	1,376,256.85	283,165.11	-79.42	3,121.06	3,112.23	-0.28
Logical reads:	17,048.22	5,194.35	-69.53	38.66	57.09	47.67
Block changes:	13,319.48	2,799.00	-78.99	30.21	30.21	0.00
Physical reads:	0.40	0.88	120.00	0.00	0.01	100.00
Physical writes:	114.82	23.56	-79.48	0.26	0.26	0.00
User calls:	1.14	0.93	-18.42	0.00	0.01	100.00
Parses:	38.65	6.14	-84.11	0.09	0.07	-22.22
Hard parses:	1.50	0.74	-50.67	0.00	0.01	100.00

Active Session History (ASH)

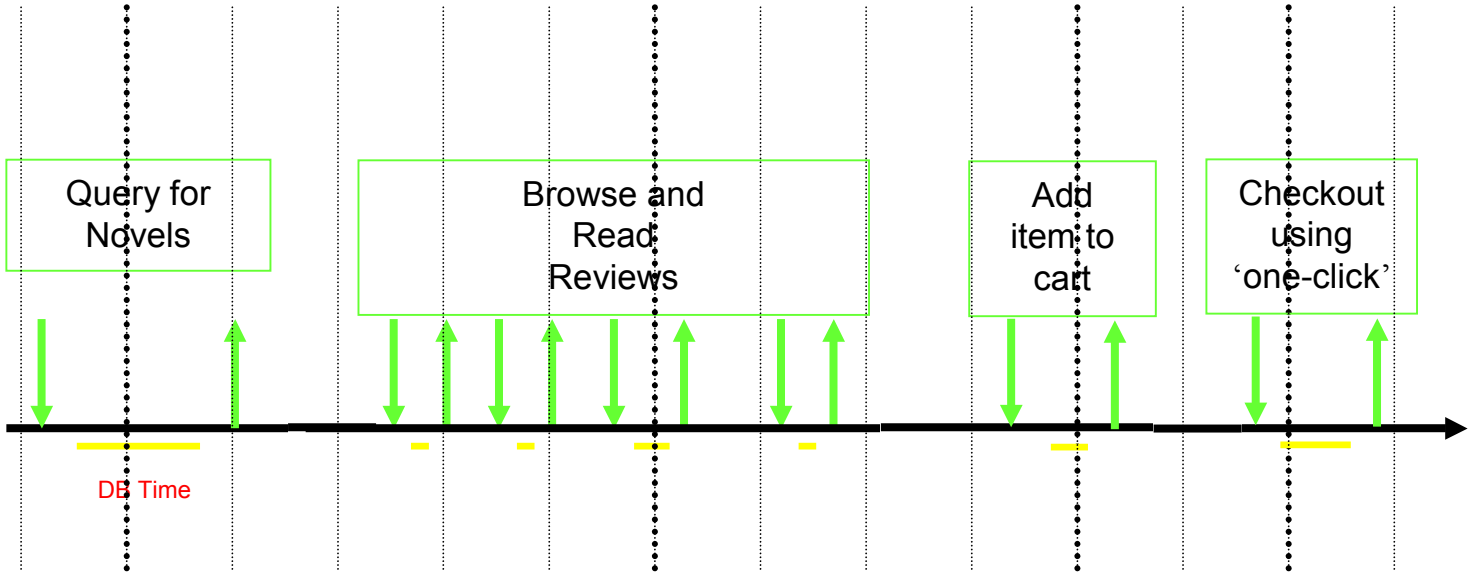
- Samples active sessions every second into memory (v$active_session_history)
- Direct access to kernel structures
- Selected samples flushed to AWR
- Data captured includes:
  - SID
  - SQL ID
  - Program, Module, Action
  - Wait event#
  - Object, File, Block
  - actual wait time (if captured while waiting)

Sampled history of v$session_wait

Active Session History

- Part of AWR
- Helps to analyze
  - Short term problems (minute history)
  - Isolation of the cause by SQL_ID, SESSION_ID, MODULE etc.
  - Blocking Sessions (Enqueue, buffer busy wait)
- Called by
  - ASH Report (ashrpt.sql or Enterprise Manager)
  - Hang Analyze

Active Session History (ASH)



Time	SID	Module	SQL ID	State	Event
7:38:26	213	Book by author	qa324jffritcf	WAITING	db file sequential read
7:42:35	213	Get review id	aferv5desfzs5	CPU	
7:50:59	213	Add to cart	hk32pekfcbdfr	WAITING	buffer busy wait
7:52:33	213	One click	abngldf95f4de	WAITING	log file sync

Active Session History - Examples

ASH Report For ORAI1/DE23

DB Name	DB Id	Instance	Inst num	Release	RAC	Host
ORAI1	1154309737	DE23	1	10.2.0.2.0	YES	MVS09

CPUs	SGA Size	Buffer Cache	Shared Pool	ASH Buffer Size
6	348M (100%)	124M (35.6%)	108M (31.0%)	5.4M (1.6%)

	Sample Time	Data Source
Analysis Begin Time:	21-Mar-07 08:09:37	V$ACTIVE_SESSION_HISTORY
Analysis End Time:	21-Mar-07 08:24:40	V$ACTIVE_SESSION_HISTORY
Elapsed Time:	15.1 (mins)	
Sample Count:	199	
Average Active Sessions:	0.22	
Avg. Active Session per CPU:	0.04	
Report Target:	None specified	

ASH Report

- Top Events
- Load Profile
- Top SQL
- Top PL/SQL
- Top Sessions
- Top Objects/Files/Latches
- Activity Over Time

Back to Top

Active Session History - Examples

Top Sessions

- '# Samples Active' shows the number of ASH samples in which the session was found waiting for that particular event. The percentage shown in this column is calculated with respect to wall clock time and not total database activity.
- 'XIDs' shows the number of distinct transaction IDs sampled in ASH when the session was waiting for that particular event
- For sessions running Parallel Queries, this section will NOT aggregate the PQ slave activity into the session issuing the PQ. Refer to the 'Top Sessions running PQs' section for such statistics.

Sid, Serial#	% Activity	Event	% Event	User	Program	# Samples Active	XIDs
527, 2769	51.59	enq: TX - row lock contention	51.59	KHESSEN	sqlplus.exe	178/905 [20%]	0
552, 1	24.64	control file sequential read	20.29	SYS	LMON	70/905 [8%]	0
		CPU + Wait for CPU	4.35			15/905 [2%]	0
543, 1	11.59	control file parallel write	10.72	SYS	CKPT	37/905 [4%]	0
523, 1968	4.06	CPU + Wait for CPU	2.32	SYSTEM	sqlplus.exe	8/905 [1%]	0
		db file sequential read	1.45			5/905 [1%]	0
540, 1	2.03	rdbms ipc reply	0.58	SYS	MMON	2/905 [0%]	0

Back to Top Sessions
Back to Top

Top Blocking Sessions

- Blocking session activity percentages are calculated with respect to waits on enqueues, latches and "buffer busy" only
- '% Activity' represents the load on the database caused by a particular blocking session
- '# Samples Active' shows the number of ASH samples in which the blocking session was found active.
- 'XIDs' shows the number of distinct transaction IDs sampled in ASH when the blocking session was found active.

Blocking Sid	% Activity	Event Caused	% Event	User	Program	# Samples Active	XIDs
519, 2346	51.59	enq: TX - row lock contention	51.59			0/905 [0%]	0

SQL tuning

- Prerequisites
  - Use Cost based optimizer (RBO desupported with Oracle 10g)
  - Use DBMS_STATS (important)
- Explain Query
  - Create Plan Table: UTLXPLAN
- Visualize Execution Plan
  - UTLXPLS
  - UTLXPLP
  - Package dbms_xplan

Note: Scripts are located in HLQ.SQL library (z/OS, V9.2)
$ORACLE_HOME/rdbms/admin

SQL tuning

```
SQL> explain plan for select a.* from scott.emp a, scott.dept b where a.deptno=b.deptno;

Explained.

SQL> save explain
Created file explain.sql
SQL> @?/rdbms/admin/utlxpls

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------

-----------------------------------------------------------------------
| Id  | Operation            |  Name       | Rows  | Bytes | Cost  |
-----------------------------------------------------------------------
|   0 | SELECT STATEMENT     |             |    14 |   560 |     2 |
|   1 |  NESTED LOOPS        |             |    14 |   560 |     2 |
|   2 |   TABLE ACCESS FULL  | EMP         |    14 |   518 |     2 |
|*  3 |   INDEX UNIQUE SCAN  | PK_DEPT     |     1 |     3 |       |
-----------------------------------------------------------------------

Predicate Information (identified by operation id):

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------

   3 - access("A"."DEPTNO"="B"."DEPTNO")
```

SQL tuning

- Explain Query – View Execution plan

```
explain plan for
     select * from scott.emp@DB2_for_ZOS where soundex(ename) = 'SMITH';
:
SELECT * FROM table(DBMS_XPLAN.DISPLAY('plan_table', null, 'ALL'));

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------

--------------------------------------------------------------------------------
| Id  | Operation           |  Name       | Rows  | Bytes | Cost  | Inst   |IN-OUT|
--------------------------------------------------------------------------------
|   0 | SELECT STATEMENT    |             |    20 |  1160 |    52 |        |      |
|*  1 |  FILTER             |             |       |       |       |        |      |
|   2 |   REMOTE            |             |       |       |       | DB2_F~ | R->S |
--------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   1 - filter(SOUNDEX("A1"."ENAME")='SMITH')

Remote SQL Information (identified by operation id):
----------------------------------------------------

   2 - SELECT "EMPNO", "ENAME", "JOB", "MGR", "HIREDATE", "SAL", "COMM", "DEPTNO"
            FROM "SCOTT"."EMP" (accessing 'DB2_FOR_ZOS.US.ORACLE.COM' )
```

SQL tuning

- View execution plan (SGA)

```
select * from table(dbms_xplan.display_cursor('31c7r47m8p1vt',null,'ALL'))

PLAN_TABLE_OUTPUT
------------------------------------------------------------------------------
SQL_ID  31c7r47m8p1vt, child number 0
-------------------------------------
SELECT ENAME FROM SCOTT.EMP_TN

Plan hash value: 343104048
-------------------------------------------------------------------------------
| Id  | Operation         | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |        |       |       |    16 (100)|          |
|   1 |  TABLE ACCESS FULL| EMP_TN |  5376 | 32256 |    16   (0)| 00:00:01 |
-------------------------------------------------------------------------------
Query Block Name / Object Alias (identified by operation id):
-------------------------------------------------------------
   1 - SEL$1 / EMP_TN@SEL$1

Column Projection Information (identified by operation id):
-----------------------------------------------------------

   1 - "ENAME"[VARCHAR2,10]
```

SQL tuning

View execution plan (AWR)

```
select * from table(dbms_xplan.display_AWR('31c7r47m8p1vt',null,null,'ALL'))

PLAN_TABLE_OUTPUT
------------------------------------------------------------------------------
SQL_ID 31c7r47m8p1vt
--------------------
SELECT ENAME FROM SCOTT.EMP_TN

Plan hash value: 343104048

-------------------------------------------------------------------------------
| Id  | Operation         | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |        |       |       |    16 (100)|          |
|   1 |  TABLE ACCESS FULL| EMP_TN |  5183 | 36281 |    16   (0)| 00:00:01 |
-------------------------------------------------------------------------------

Note
-----
   - dynamic sampling used for this statement


17 rows selected.
```

SQL tuning

- Optimizer features which help to improve execution plans
  - Function based indexes (very important)
    - SELECT * From emp where upper(ename) = ´SMITH´
  - Bitmap indexes (Useful in case of Read Only)
    - Useful for Low Cardinality columns
  - Parameter: Optimizer_index_cost_adj
    - Optimizer access path selection can be adjusted to be more index friendly
  - NLS_SORT
    - Index may not be used

SQL tuning

- SQLTRACE
  - Statistics_Level=TYPICAL(default) / ALL
  - Activate
    - Alter Session set SQL_trace=true
    - dbms_system.set_sql_trace_in_session
    - dbms_monitor.session_trace_enable (10.2)
  - Use TKPROF to show execution statistics
    - sys=no,explain=uid/pw

Bind Variable Peeking

- The feature is activated per default
- It can be controlled via the parameter _OPTIM_PEEK_USER_BINDS=TRUE/FALSE
- The execution plan depends on the bind values of the first parse. For the first parse of a statement the bind values are determined
- If histograms are created, then different plans for different values may be produced
  - DBMS_STATS
  - OTIMIZER_DYNAMIC_SAMPLING > 1
- CURSOR_SHARING=FORCE
  - FORCE uses the same plan whatever the circumstance

Bind Variable Peeking

- Visualize Bind Variables

```
select * from
table(dbms_xplan.display_cursor('f42w70mf0pc7q',0, 'ADVANCED'));
```

```
Plan hash value: 3281146378

-------------------------------------------------------------------------------
| Id  | Operation          | Name | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |      |       |       |     2 (100)|          |
|   1 |  SORT AGGREGATE    |      |     1 |     3 |            |          |
|*  2 |   TABLE ACCESS FULL| EMP  |     5 |    15 |     2   (0)| 00:00:01 |
-------------------------------------------------------------------------------

Peeked Binds (identified by position):
--------------------------------------

  1 - :B1 (NUMBER): 20
```

dbms_monitor

- Available with Oracle 10g
- Enables 10046 trace
- execute dbms_monitor.session_trace_enable(session_id=>141, serial_num=>21, waits=>true, binds=>true);
- Trace Analyzer
  - Can be used to analyze 10046 trace
  - Described in NOTE 224270.1

Operating System

- z/OS
  - Execution plan of a few SQL Queries
  - WLM
    - ENCLAVE(SESS) with respose time goals
    - Same importance / no discretionary goals
    - Service Class SYSOTHER used
- Linux on System/z
  - Execution plan of a few SQL Queries
  - I/O Subsystem
    - resoled by use of Async I/O


Q&A

ORACLE IS THE INFORMATION COMPANY


ORACLE®