Sifting through the ASHes
Performance Analysis with the Oracle 10g Active Session History

Graham Wood
Graham.Wood@oracle.com

Oracle Corporation

ORACLE

Agenda

- Introduction
- What is ASH
- Querying ASH data
- Comparison of ASH and Statspack/AWR
- Comparison of ASH and SQL trace/tkprof
- EM use of ASH data
- Conclusions

Oracle Statistics

- Instance level statistics (AWR, Statspack)
  - Too little detail ⇒ Stop short of complete diagnosis
  - Can be collected automatically
- Trace level statistics (sql_trace)
  - Too much detail ⇒ Intrusive. Hard to see big-picture
  - Must be enabled manually
  - Need prior knowledge that problem exists

Oracle Statistics

- Solution: Active Session History
  - Sample session activity in the system including:
    - Session id
    - Wait event
    - SQL id
    - Object
  - Always on for first fault analysis
  - Just right!

Active Session History

- Sampled, detailed,non-intrusive activity data
- Part of Oracle 10g
- On by default
- Licensed as part of the Diagnostic pack

Active Session History (ASH)

- Samples ‘Active’ sessions every second
  - Like doing “select * from v$session_wait” w/o SQL
- Writes into ASH buffer in SGA memory
  - 2MB per CPU, ≤ 5% shared_pool, 2% sga_target
- ‘Active’ == Non-idle sessions
  - Waiting on non-idle event or on CPU
- Data volume based on activity
  - 10,000 sessions => 200 active sessions
  - Design goal: one hour activity held in memory

Active Session History (ASH)

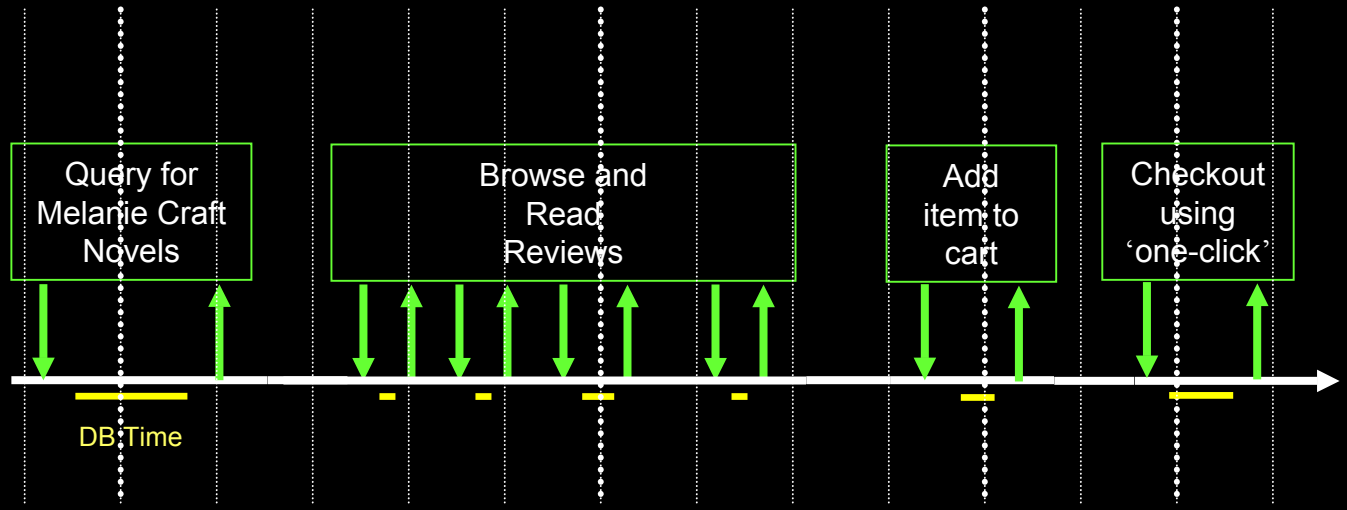
```
SQL> select * from  v$sgastat where name like 'ASH buffers';

POOL         NAME                            BYTES
------------ -------------------------- ----------
shared pool  ASH buffers                  65011712

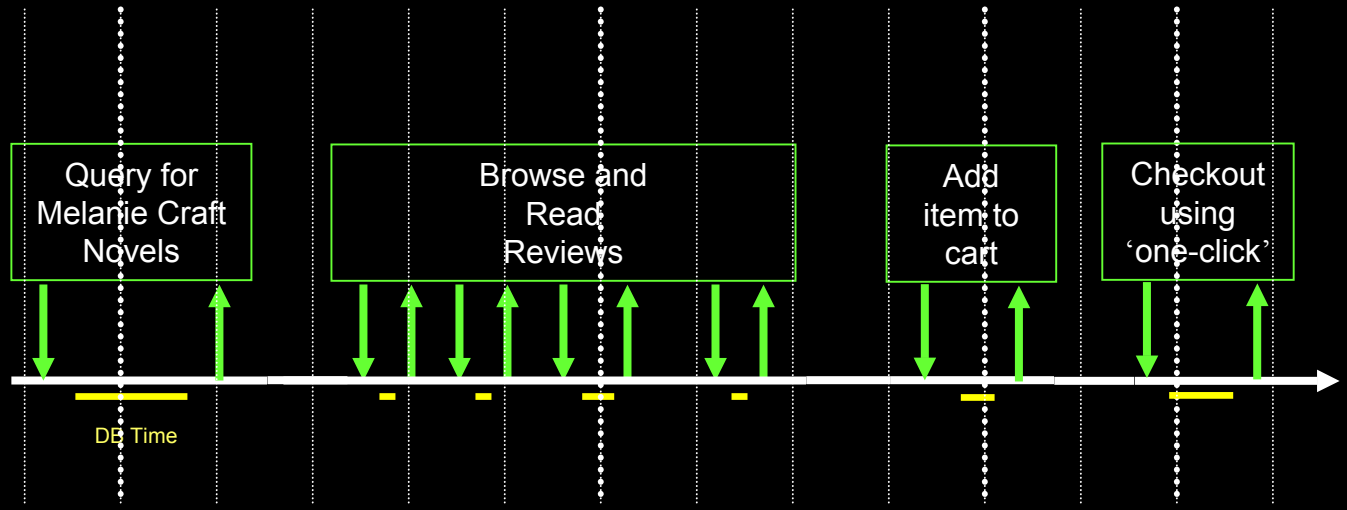
SQL> select min(sample_time), max(sample_time) from
   v$active_session_history;

MIN(SAMPLE_TIME)
---------------------------------------------------------------------------
MAX(SAMPLE_TIME)
---------------------------------------------------------------------------
20-FEB-05 10.31.38.615 PM
21-FEB-05 02.39.28.950 AM
```

Active Session History (ASH)



Active Session History (ASH)

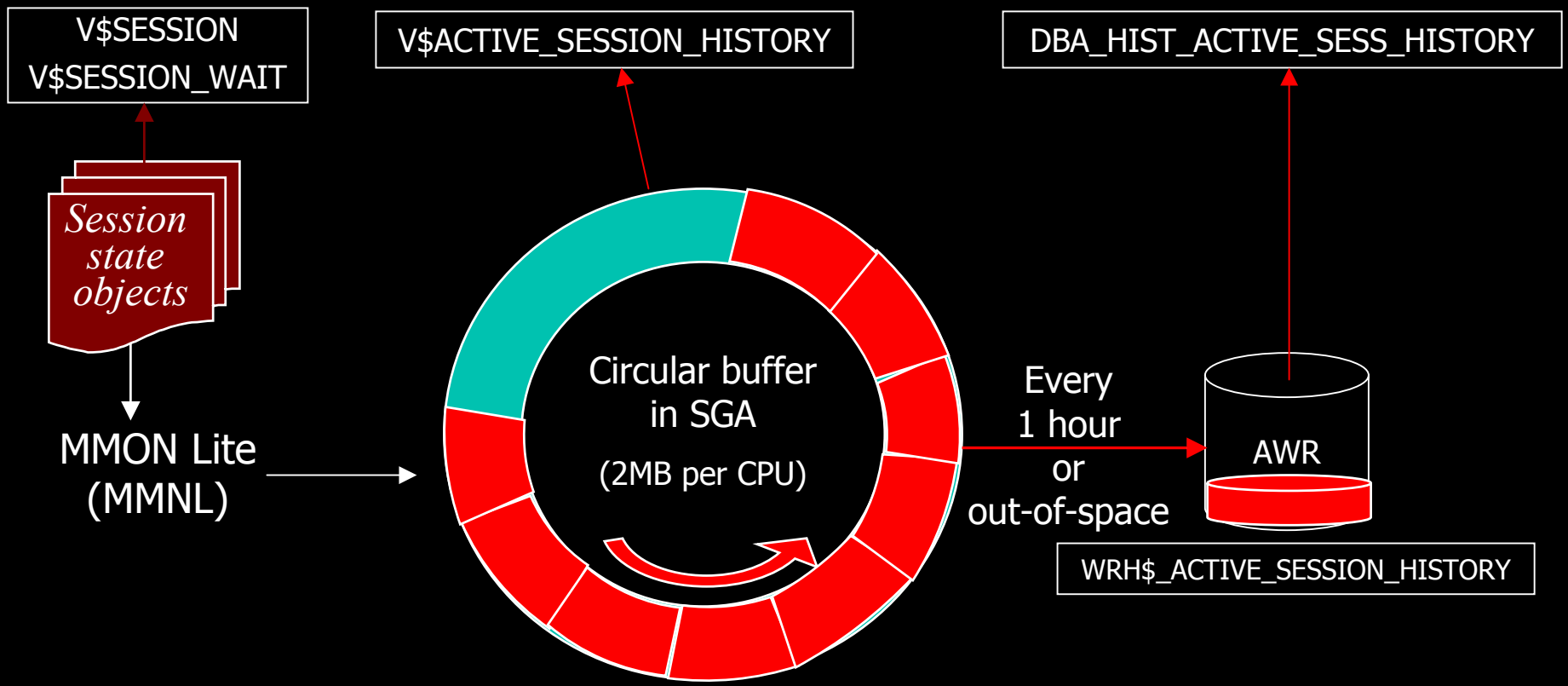


Time	SID	Module	SQL ID	State	Event
7:38:26	213	Book by author	qa324jffritcf	WAITING	db file sequential read
7:42:35	213	Get review id	aferv5desfzs5	CPU	
7:50:59	213	Add to cart	hk32pekfcbdfr	WAITING	buffer busy wait
7:52:33	213	One click	abngldf95f4de	WAITING	log file sync

ASH: On disk

- Captured as part of AWR snapshots
  - DBA_HIST_ACTIVE_SESS_HISTORY
- Takes samples from in-memory ASH
  - 10 second samples
- On-demand flush if required
  - Whenever circular buffer is 66% full
  - No missed data
- Seven days history by default
  - Table is partitioned for easy purging

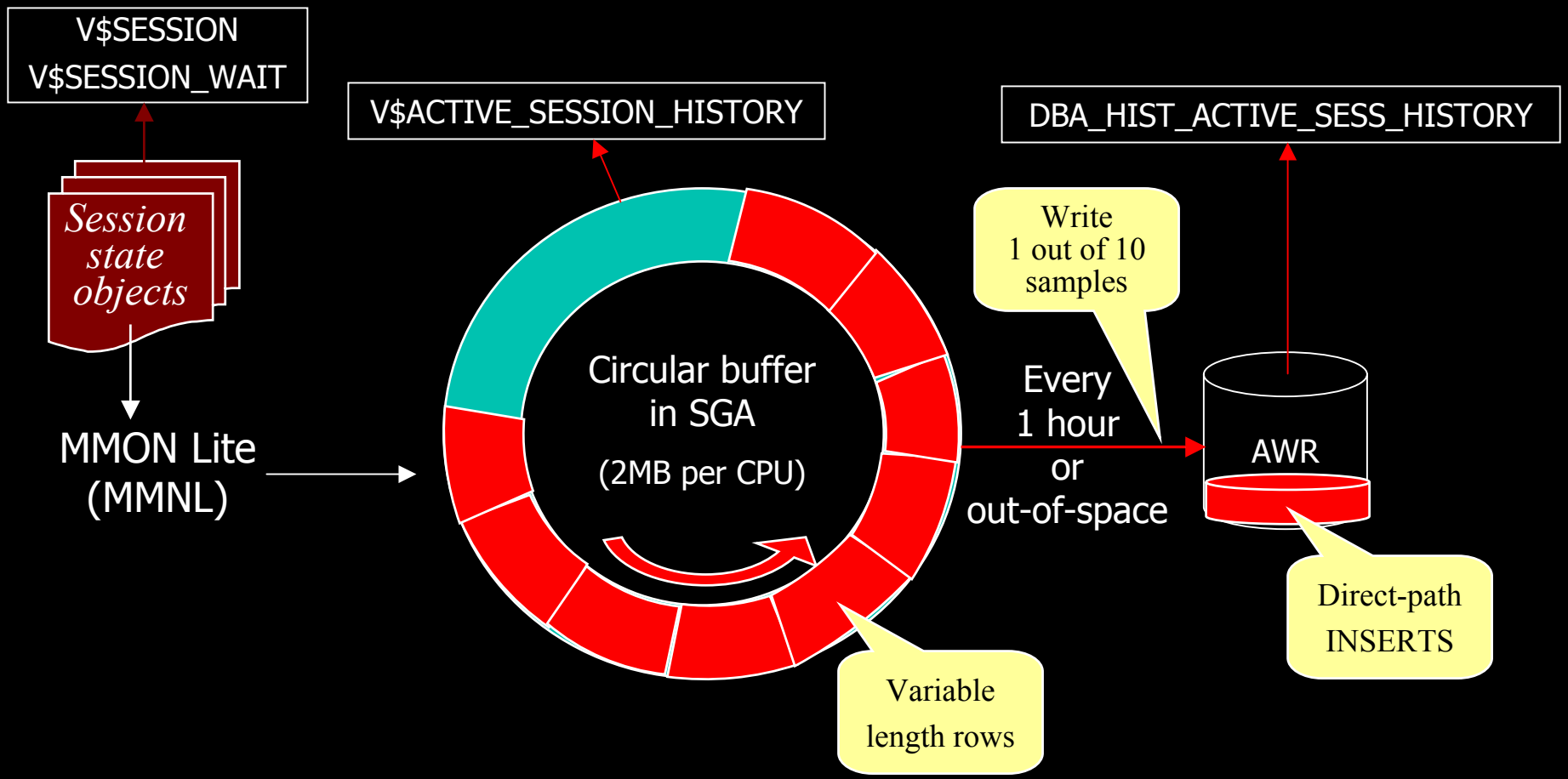
Active Session History



ASH: Challenges – Space

- Memory Usage
  - Module, Action, Client_id (~50%)
  - Variable length rows
- Disk Usage
  - Write 1 out of every 10 samples
- Log generation
  - Direct-path INSERTS

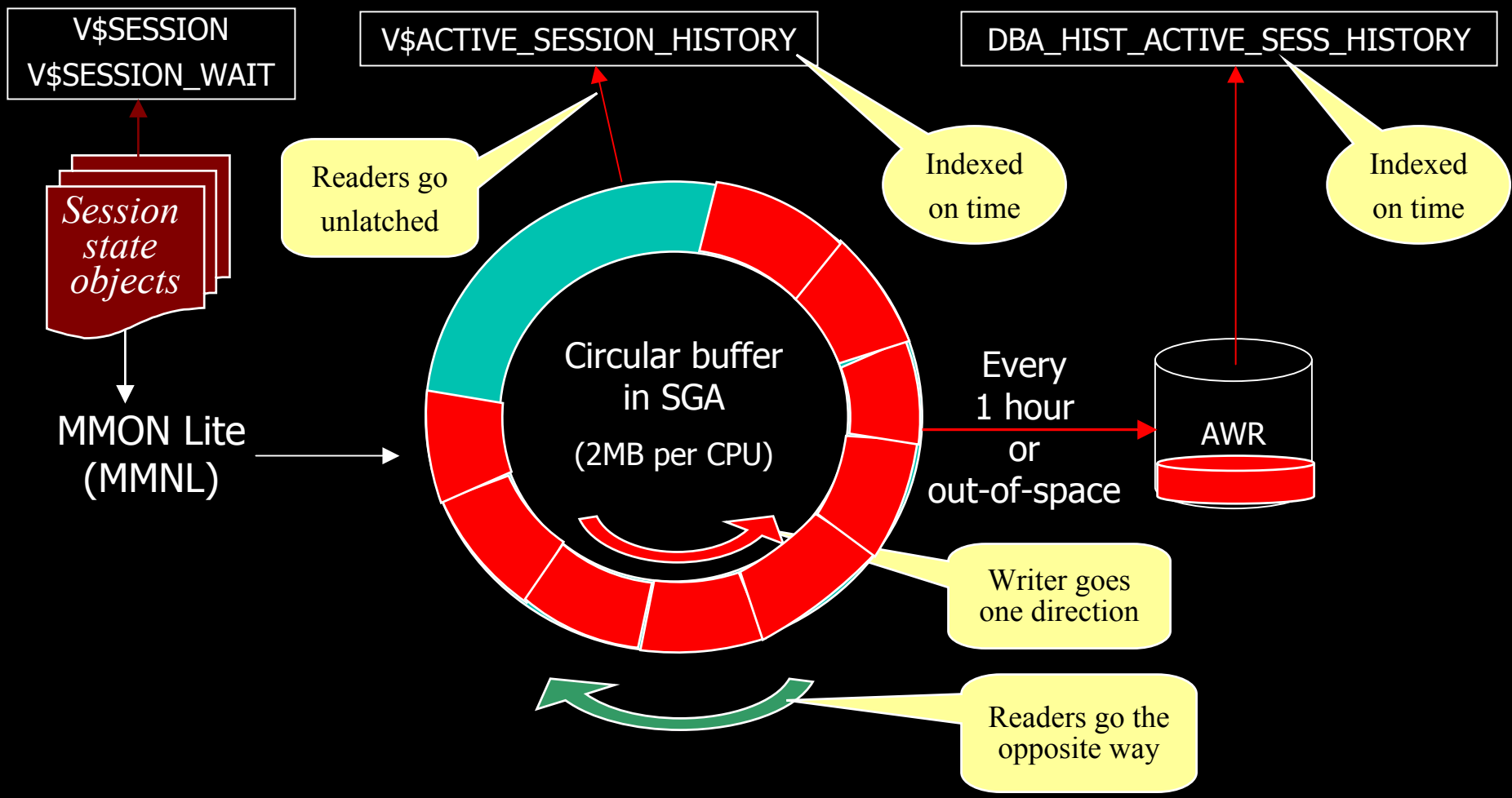
Active Session History



ASH: Challenges – Time

- Reader-Writer Concurrency
  - No Consistent-Read requirement
  - 1 Writer – Multiple Readers
  - Readers go unlatched
- Indexed on time
  - Both V$ view and DBA_HIST view

Active Session History



What you can do with it

- STATISTICAL analysis of where time was being spent by many different dimension.
  - What events were taking most time?
  - What was a session doing?
  - What does a SQL statement wait for?
- Can decide on dimension after the event!

ASH: Dimensions

- Session
- Waits
  - Event, P1, P2, P3
- SQL
  - Sql_id, Opcode,Plan_hash
- Objects
  - Object#, File#, Block#
- Application
  - Program, Module, Action, Client_id, Service
- Combinations of the above, CUBEs, ROLLUPs, …

Accessing ASH data

- Dump to trace file
- V$ACTIVE_SESSION_HISTORY
- DBA_HIST_ACTIVE_SESS_HISTORY
- ASH report
- EM Diagnostic Pack

Dumping ASH to file

```
>oradebug setmypid
>oradebug dump ashdump 10

>alter session set events 'immediate
  trace name ashdump level 10';
```

- 10 ==> minutes of history you want to dump
- Generated file can be loaded into database using supplied control file `rdbms/demo/ashldr.ctl`

V$ACTIVE_SESSION_HISTORY

- Gives most recent data first
- Control C or ‘set pause on’ is your friend
- Simpleash.sql

ASH: desc v$active_session_history

	Name	Null?	Type
	SAMPLE_ID		NUMBER
	SAMPLE_TIME		TIMESTAMP(3)
Session	SESSION_ID		NUMBER
Session	SESSION_SERIAL#		NUMBER
Session	USER_ID		NUMBER
Session	SESSION_TYPE		VARCHAR2(10)
Session	SESSION_STATE		VARCHAR2(7)
Session	QC_SESSION_ID		NUMBER
Session	QC_INSTANCE_ID		NUMBER
Wait	EVENT		VARCHAR2(64)
Wait	EVENT_ID		NUMBER
Wait	EVENT#		NUMBER
Wait	SEQ#		NUMBER
Wait	P1		NUMBER
Wait	P2		NUMBER
Wait	P3		NUMBER
SQL	SQL_ID		VARCHAR2(13)
SQL	SQL_CHILD_NUMBER		NUMBER
SQL	SQL_PLAN_HASH_VALUE		NUMBER
SQL	SQL_OPCODE		NUMBER
Object	CURRENT_OBJ#		NUMBER
Object	CURRENT_FILE#		NUMBER
Object	CURRENT_BLOCK#		NUMBER
Application	PROGRAM		VARCHAR2(48)
Application	MODULE		VARCHAR2(48)
Application	ACTION		VARCHAR2(32)
Application	CLIENT_ID		VARCHAR2(64)
Application	SERVICE_HASH		NUMBER
	WAIT_TIME		NUMBER
	TIME_WAITED		NUMBER

How to Sift the ASHes

- “group by”s and “count(*)”s
  - Proxy for non-idle elapsed time
  - Proportions of actual time spent
- Can analyze any time slice
- More samples ⇒ More accurate results

ASH: Top SQL

- ```
  select sql_id, count(*),
         round(count(*)
           /sum(count(*)) over (), 2) pctload
    from  v$active_session_history
   where  sample_time > sysdate - 1/24/60
     and  session_type <> 'BACKGROUND'
  group by sql_id
  order by count(*) desc;
  ```

- Returns most active SQL in the past minute
  ```

ASH: Top SQL

SQL_ID	COUNT(*)	PCTLOAD
25wtt4ycbtkyz	456	32.95
7umwqvcy7tusf	123	8.89
01vunx6d35khz	119	8.6
bdyq2uph07cmp	102	7.37
9y4f9n5hr23yr	73	5.27
0bnc9a5kkf4wn	57	4.12
bv1gns48hgxpk	57	4.12
gq82c5361nxbq	57	4.12
djzkbxr7cm122	57	4.12
b2bakhq4w7rbv	57	4.12
8jydryyvdwcqp	57	4.12
69x6zf5myht7s	57	4.12
2ccawhzy8b7ua	57	4.12
4z5z7xb2g04m6	55	3.97

ASH: Top IO SQL

```
select ash.sql_id, count(*)
from   v$active_session_history ash,
       v$event_name evt
where  ash.sample_time > sysdate – 1/24/60
  and  ash.session_state = ‘WAITING’
  and  ash.event_id = evt.event_id
  and  evt.wait_class = ‘User I/O’
group by sql_id
order by count(*) desc;
```

- Returns SQL spending most time doing I/Os
- Similarly, can do Top Sessions, Top Files, Top Objects

DBA_HIST_ACTIVE_SESS_HISTORY

- Similar to in-memory ASH but adds
  - DB_ID
  - INSTANCE_NUMBER
  - SNAP_ID
- One sample every 10 seconds

ASH data gotcha's

- Samples are a proxy for time not for counts
- Times are sampled times, not statistically valid for avg, min, max
- Beware of Obj#, File#, Block# (not cleared)
- Temp file numbers
- Wait time vs Time waited
- SQL*Forms RPC bug# 4137362
- Time period of data available in V$ACTIVE_SESSION_HISTORY is variable

ASH: Bad SQL

```
select
e.event,
e.total_waits - nvl(b.total_waits,0) total_waits,
e.time_waited - nvl(b.time_waited,0) time_waited
from
v$active_session_history b,
v$active_session_history e,
stats$snapshot sn
Where snap_time > sysdate-&1
And e.event not like '%timer'
And e.event not like '%message%'
And e.event not like '%slave wait%'
And e.snap_id = sn.snap_id
And b.snap_id = e.snap_id-1
And b.event = e.event
And e.total_timeouts > 100
And (e.total_waits - b.total_waits > 100
   or e.time_waited - b.time_waited > 100)
;
```

ASH: Bad SQL

```
select sum(a.time_waited) total_time
from    v$active_session_history a,
        v$event_name b
where   a.event# = b.event# and
        sample_time > '21-NOV-04 12:00:00 AM' and
        sample_time < '21-NOV-04 05:00:00 AM' and
        b.wait_class = 'User I/O'
```

ASH: Bad SQL

```
select sum(a.time_waited) total_time
from    v$active_session_history a,
        v$event_name b
where   a.event# = b.event# and
        sample_time > '21-NOV-04 12:00:00 AM' and
        sample_time < '21-NOV-04 05:00:00 AM' and
        b.wait_class = 'User I/O'
```

- Total time spent waiting on IO?

ASH: Bad SQL

```
select sum(a.time_waited) total_time
from    v$active_session_history a,
        v$event_name b
where   a.event# = b.event# and
        sample_time > '21-NOV-04 12:00:00 AM' and
        sample_time < '21-NOV-04 05:00:00 AM' and
        b.wait_class = 'User I/O'
```

- Total time spent waiting on IO?
- Totals **sampled** IO times

ASH: Bad SQL

```
select sum(a.time_waited) total_time
from    v$active_session_history a,
        v$event_name b
where   a.event# = b.event# and
        sample_time > '21-NOV-04 12:00:00 AM' and
        sample_time < '21-NOV-04 05:00:00 AM' and
        b.wait_class = 'User I/O'
```

- Total time spent waiting on IO?
- Totals sampled IO times
- Assumes that 5 hours history in memory

ASH: Bad SQL

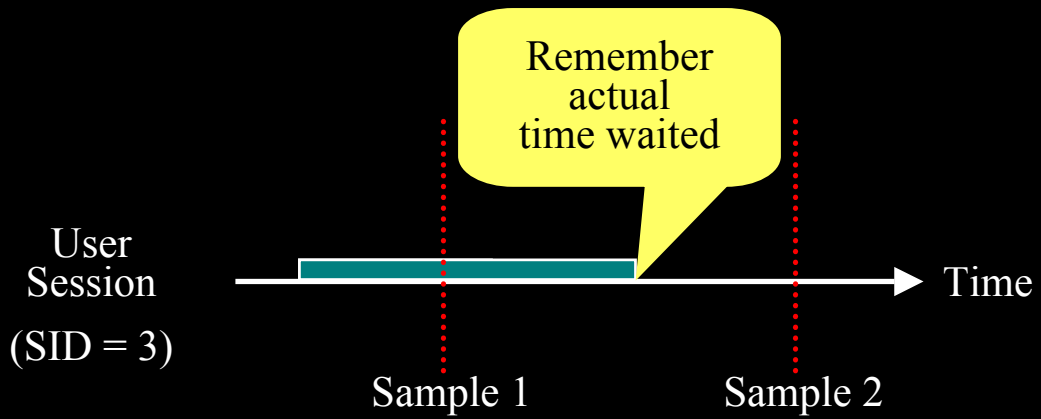
```
select  sess_id,username,program,wait_event,sess_time,
   round(100*(sess_time/total_time),2) pct_time_waited
from
(select a.session_id sess_id,
decode(session_type,'background',session_type,c.username) username,
        a.program program,
        b.name wait_event,
        sum(a.time_waited) sess_time
from    sys.v_$active_session_history a,
        sys.v_$event_name b,
        sys.dba_users c
where   a.event# = b.event# and
        a.user_id = c.user_id and
        sample_time > '21-NOV-04 12:00:00 AM' and
        sample_time < '21-NOV-04 05:00:00 AM' and
        b.wait_class = 'User I/O'
group by a.session_id,
        decode(session_type,'background',session_type,c.username),
        a.program,
        b.name),
(select sum(a.time_waited) total_time
```

ORACLE

ASH: WAIT_TIME vs TIME_WAITED

- WAIT_TIME
  - Same as V$SESSION_WAIT
  - 0 ⇒ 'WAITING'
    any other value ⇒ 'ON CPU'
- TIME_WAITED
  - Actual time waited for that event
  - Updated later upon event completion

ASH: TIME_WAITED



After Sample 1

Sample	Session	State	Event	Wait_time	Time_waited
1	3	WAITING	db file scattered read	0	0

After Sample 2, Sample 1 is updated

Sample	Session	State	Event	Wait_time	Time_waited
1	3	WAITING	db file scattered read	0	5ms

ASH vs AWR/Statpack

	ASH	AWR
Instance wide data	Yes	Yes
Time based data	Yes	Yes
Counts/occurrence data	No	Yes
Analyze any time period	Yes	No
Detailed session level data	Yes	No
Individual Wait event data	Yes	No
Sampled data	Yes	No
Time based analysis	Yes	Yes

ASH vs AWR

Top 5 Timed Events

Event	Waits	Time (s)	% Total DB Time	Wait Class
log file sync	990,495	233,649	43.79	Commit
latch: library cache	642,247	157,188	29.46	Concurrency
latch: cache buffers chains	133,136	39,747	7.45	Concurrency
latch: library cache pin	84,638	22,998	4.31	Concurrency
latch free	61,709	20,079	3.76	Other

Top Foreground Events

log file sync	46.01%	Commit
latch: library cache	23.13%	Concurrency
latch: cache buffers chains	6.50%	Concurrency
latch free	4.63%	Other
latch: library cache pin	2.99%	Concurrency

D E M O N S T R A T I O N

ASH Report

ASH vs SQLtrace/tkprof

	ASH	SQLtrace
Parse/Exec/Fetch breakdown	No	Yes
Time based data	Yes	Yes
Counts/occurrence data	No	Yes
Detailed session level data	Yes	Yes
Individual Wait event data	Yes	Yes
Complete trace of operations	No	Yes
Always on	Yes	No
Bind variables available	No	Yes

D E M O N S T R A T I O N

ASH Session Report

D E M O N S T R A T I O N

EM Diagnostic Pack

Top SQL

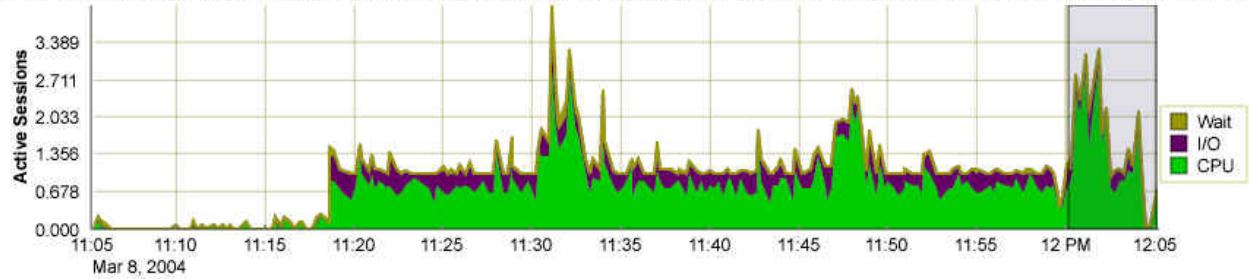
Spot SQL | Period SQL

Spot SQL shows all the sql statements that have been active in a recent 5 minute interval.

View Data: Real Time: 15 Second Refresh

Spot Interval Selection

Drag the shaded box to select the 5 minute interval for which you want to view details in the section below. Use the active sessions data to help with your selection.



Detail for Selected 5 minute Interval

Start Time **Mar 8, 2004 12:00:16 PM**

All SQL

Run SQL Tuning Advisor | Create SQL Tuning Set

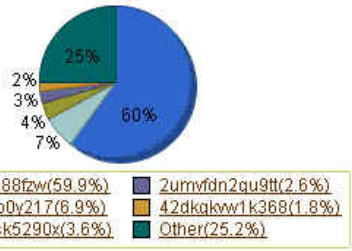
Select All | Select None Previous 1-10 of 64 Next 10

Select	SQL ID	SQL Type	Activity (%)	CPU (%)	Wait (%)
☐	8jwydvpq88fzw	SELECT	59.90	52.89	98.33
☐	d92h3rip0y217	PL/SQL EXECUTE	6.94	8.21	0.00
☐	6q766vsk5290x	PL/SQL EXECUTE	3.60	4.26	0.00
☐	2umvfdn2qu9tt	SELECT	2.57	3.04	0.00
☐	42dkqkvw1k368	SELECT	1.80	2.13	0.00
☐	1vqstvfdfbq9v	SELECT	1.54	1.82	0.00
☐	g541afnx1yqnp	SELECT	1.54	1.82	0.00
☐	5bxv6qrm6m1j7	SELECT	1.29	1.52	0.00
☐	257rmrxqvaj4z	SELECT	1.29	1.52	0.00
☐	748xwx1cjn66g	SELECT	1.03	1.22	0.00

Previous 1-10 of 64 Next 10

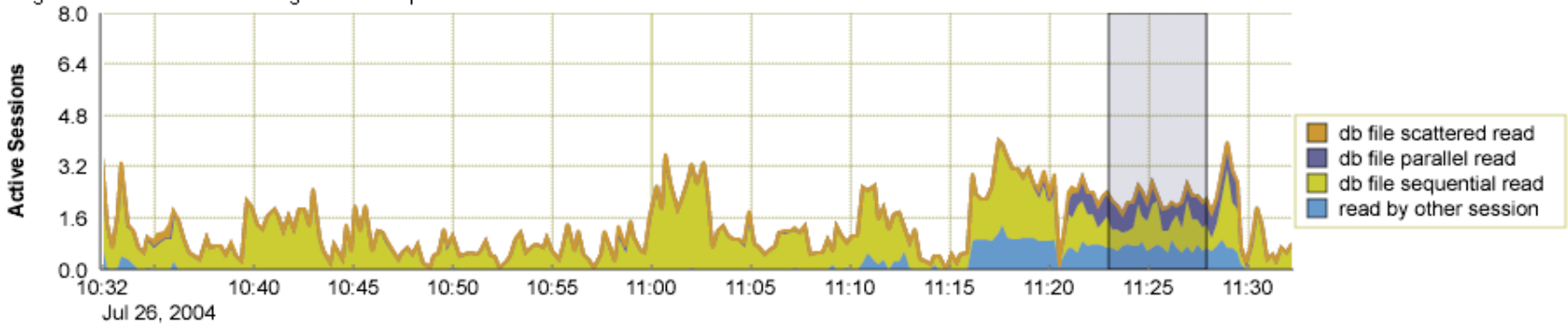
Run SQL Tuning Advisor | Create SQL Tuning Set

Top SQL (ordered by Activity)



Active Sessions Waiting: User I/O

Drag the shaded box to change the time period for the detail section below.



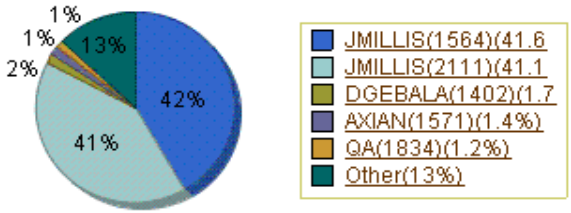
Detail for Selected 5 minute Interval

Start Time **Jul 26, 2004 11:23:01 AM**

Overview | Top SQL | **Top Sessions** | Top Files | Top Objects

Top Waiting Sessions

Total Sample Count: 654



Wait Events for Top Sessions

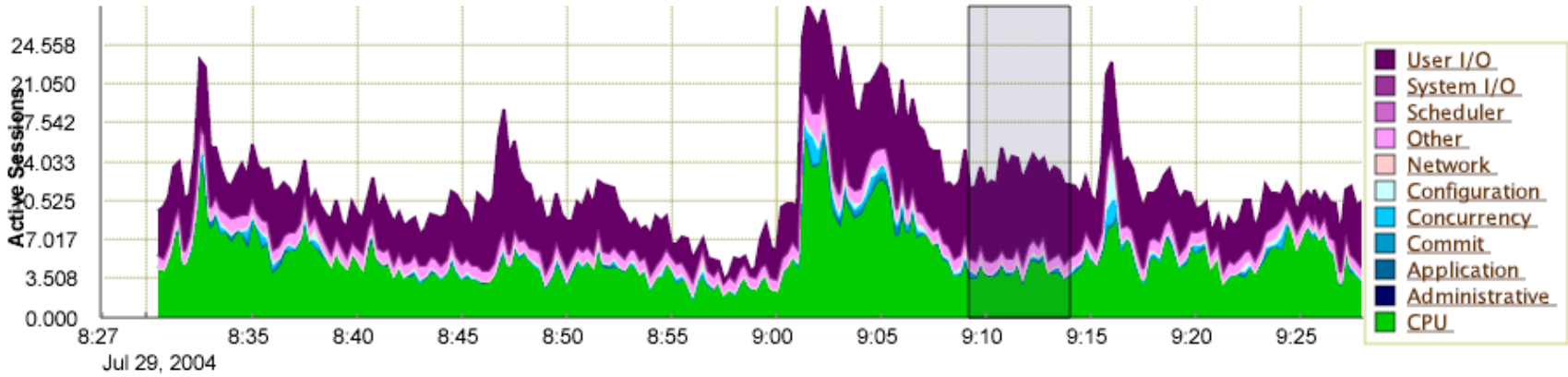
User Name	Session ID	Wait Event	Activity (%)	Program
JMILLIS	2111	db file parallel read	29.97	JDBC Thin Client
JMILLIS	1564	read by other session	28.29	JDBC Thin Client
JMILLIS	1564	db file sequential read	13.30	JDBC Thin Client
JMILLIS	2111	db file sequential read	7.49	JDBC Thin Client
JMILLIS	2111	read by other session	3.36	JDBC Thin Client
DGEBALA	1402	db file sequential read	1.68	httpd@web37 (TNS V1-V3)
AXIAN	1571	db file sequential read	1.38	httpd@web37 (TNS V1-V3)
QA	1834	db file sequential read	1.22	JDBC Thin Client
JMILLIS	2111	db file scattered read	.31	JDBC Thin Client

Overview | Top SQL | **Top Sessions** | Top Files | Top Objects

View Data Real Time: 15 Second Refresh

Activity Details

Drag the shaded box to change the time period for the detail section below.



Detail for Selected 5 Minute Interval

Start Time **Jul 29, 2004 9:09:07 AM**

View Top Sessions

Top SQL

Previous 1-10 of 21 Next 10

Activity (%)	SQL ID	SQL Type
18.25	28xhgw36vw2rm	SELECT
8.34	fz3w2067jab2q	SELECT
7.79	gkmd7xwuz1na0	SELECT
7.41	fnfunb6waj1rk	PL/SQL EXECUTE
6.61	dyqvdq0b0c5nw	SELECT
4.97	bpduxqup84anp	SELECT
1.83	4dqasqjrvudqn	UNKNOWN
1.22	azmrqs0hnwqg1	SELECT
1.03	3rgub5pf6afpb	PL/SQL EXECUTE
.99	4z5kf34u2pbvu	PL/SQL EXECUTE

Previous 1-10 of 21 Next 10

Total Sample Count: 3,118

Top Sessions

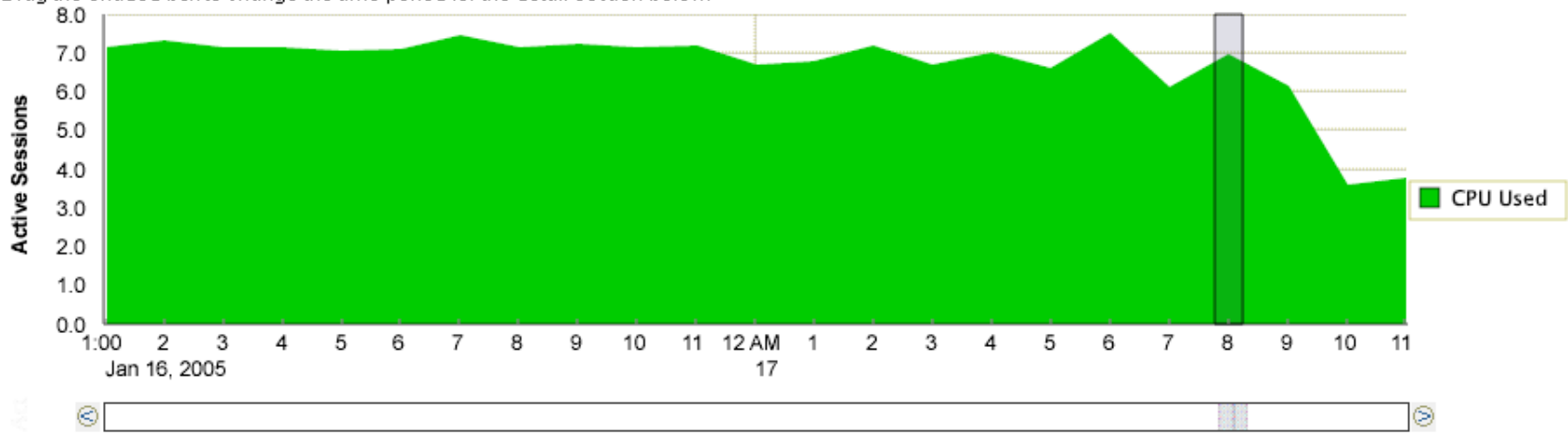
Previous 1-10 of 33 Next 10

Activity (%)	Session ID	User Name	Program
7.59	2098	PYERRAMI	oracle@stdisqa28.idc.oracle.com (TNS V1-V3)
7.56	1665	JMILLIS	JDBC Thin Client
7.41	1356	JMILLIS	JDBC Thin Client
7.30	1493	AFOTHERG	
6.85	1568	OPS$CRACICOT	oracletoolsdev@aimewebdev (TNS V1-V3)
5.75	1828	EMEIJER	oracle@dwsun42 (TNS V1-V3)
2.06	2312	SYS	oracle@dbs30 (LGWR)
1.66	1960	APPSNLS	oracle@tms.us.oracle.com (TNS V1-V3)
1.24	2313	SYS	oracle@dbs30 (DBW0)
1.00	2085	VENYUKOV	oracle@ap106fam (TNS V1-V3)

Previous 1-10 of 33 Next 10

Active Sessions Working: CPU Used

Drag the shaded box to change the time period for the detail section below.



Detail for Selected 30 Minute Interval

Start Time **Jan 17, 2005 7:50:27 AM**

Top Working SQL

Schedule SQL Tuning Advisor | Create SQL Tuning Set

Activity (%)	SQL Hash Value	SQL Type
61.11	2k9mkn75qnh51	SELECT
18.22	4w18h93xrxbq5	PL/SQL EXECUTE
4.07	4dqasqjrvudqn	PL/SQL EXECUTE
3.04	f5b199t06yqc1	SELECT
2.65	7qfqqwwtmxn1m	SELECT
2.52	3p2qkvqxswkjw	SELECT
2.39	f7mbp773x08uh	SELECT
2.33	2ppkfr98u3ht3	SELECT
2.13	d9bqjab5ty8pu	PL/SQL EXECUTE
1.55	5zbx5nbjvrvb0	UNKNOWN

Total Sample Count: 1,548

Top Working Sessions

Activity (%)	Session ID	User Name	Program
14.32	1175	VBHAVE	oracle@ap6038rt (TNS V1-V3)
14.32	1367	VBHAVE	oracle@ap6038rt (TNS V1-V3)
14.32	1535	VBHAVE	oracle@ap6038rt (TNS V1-V3)
14.23	1622	VBHAVE	oracle@ap6038rt (TNS V1-V3)
14.14	1273	VBHAVE	oracle@ap6038rt (TNS V1-V3)
13.87	1157	VBHAVE	oracle@ap6038rt (TNS V1-V3)
4.23	1843	NITIGUPT	
3.78	2176	ASRAJ	
3.69	886	VSHASTRY	
3.06	2263	PCSADMIN	? @ap601utl (TNS V1-V3)

Total Sample Count: 1,110

Additional Monitoring Links

- Period SQL
- Instance Activity
- Snapshots

ession Details: SM (2035)

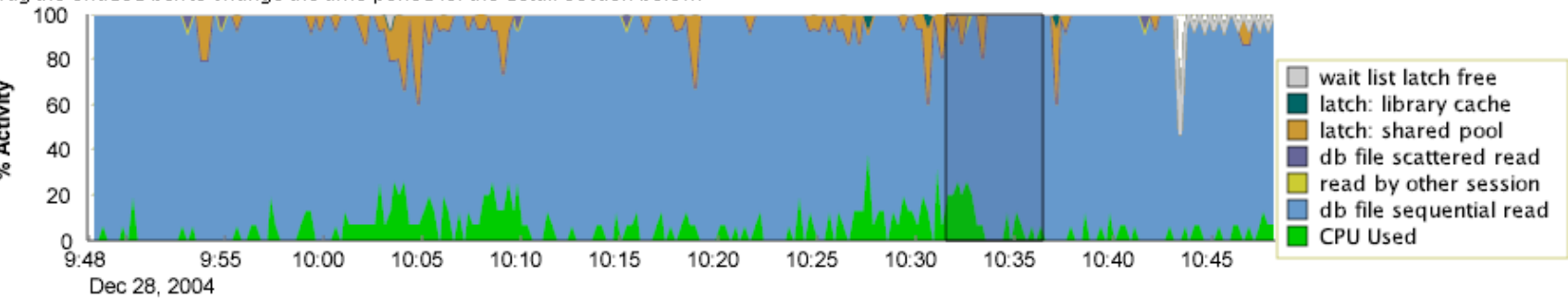
ollected From Target **Dec 28, 2004 10:47:40 AM**

View Data Real Time: 15 Second Refresh Refresh

Kill Session Enable SQL Trace Disable SQL Trace

General Activity Statistics Open Cursors Blocking Tree Wait Event History

rag the shaded box to change the time period for the detail section below.



Detail for Selected 5 Minute Interval

Start Time **Dec 28, 2004 10:31:33 AM** View Show Aggregated Data

Activity (%)	SQL ID	SQL Command	Plan Hash Value	Module	Action	Client ID
90.03	043s6zq4w225k	SELECT	2203460254	01@D:\bug_scr2_month_latest.sql		
3.78	5tjfyxq1k2kaj	SELECT	4165208437	01@D:\bug_scr2_month_latest.sql		
2.41		UNKNOWN	0	01@D:\bug_scr2_month_latest.sql		
2.06	043s6zq4w225k	SELECT	0	01@D:\bug_scr2_month_latest.sql		
1.03	9ca3dqwzztcdr	PL/SQL EXECUTE	0	01@D:\bug_scr2_month_latest.sql		
.69	fruc8ywyrpanq	SELECT	1371748965	01@D:\bug_scr2_month_latest.sql		

General Activity Statistics Open Cursors Blocking Tree Wait Event History

Kill Session Enable SQL Trace Disable SQL Trace

SQL Details: 21zvrn2udt6gz

Switch to SQL ID Go

View Data Real Time: 15 Second Refresh Run SQL Tuning Advisor

Text

```
SELECT h.rptno||'-xx-'||
h.cs_priority||'-xx-'||
h.status||'-xx-'||
          h.CATEGORY||'-xx-'||
replace(h.subject,chr(10))||'-xx-...
```

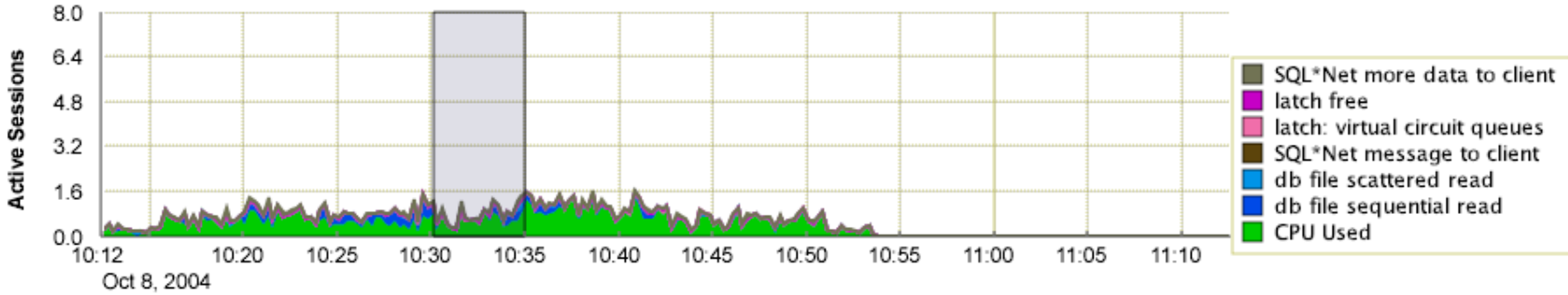
Details

Select the plan hash value to see the details below. Plan Hash Value 4047790200

Execution Statistics | **Session Activity** | Execution Plan | Tuning Information

Execution Activity

The following chart displays the users executing of the wait events and CPU of the SQL statement in the last hour. Move the slider to view the session activity detail information.



Detail for Selected 5 Minute Interval

Start Time **Oct 8, 2004 10:30:08 AM**

Activity (%)	SID	User	Program	Service	Plan Hash Value
34.21	1266	JPAL	sqlplus@ap672wgs (TNS V1-V3)	SYS$USERS	4047790200
32.71	1469	JPAL	sqlplus@ap672wgs (TNS V1-V3)	SYS$USERS	4047790200
28.57	1795	JPAL	sqlplus@ap672wgs (TNS V1-V3)	SYS$USERS	4047790200

ASH: What new in 10gR2

- Blocking sid (maybe in 10.1.0.5)
- XID

Conclusion

- ASH data always available
- Allows instance wide performance analysis
- Allows detailed session level performance analysis
- But it is sampled data, so use statistical analysis techniques


QUESTIONS
ANSWERS