```
To check INACTIVE sessions with HIGH DISK IO
=============================================

select p.spid,s.username, s.sid,s.status,t.disk_reads, s.last_call_et/3600
last_call_et_Hrs,
s.action,s.program,s.machine cli_mach,s.process cli_process,lpad(t.sql_text,30)
"Last SQL"
from gv$session s, gv$sqlarea t,v$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
t.disk_reads > 5000
and s.status='INACTIVE'
and s.process='11017'
order by S.PROGRAM;


To Analyze the DISK I/o's
==========================

prompt SESSIONS PERFORMING HIGH I/O > 50000
select p.spid, s.sid,s.process cli_process, s.status,t.disk_reads,
s.last_call_et/3600 last_call_et_Hrs,
s.action,s.program,lpad(t.sql_text,30) "Last SQL"
from v$session s, v$sqlarea t,v$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
t.disk_reads > 10000
order by t.disk_reads desc;


#####################33LATCH Free Sessions Monitoring and Troubleshooting
===========================================================================


LATCH COUNT ON DATABASE:
=========================

set pagesize 5000
set lines 180
set long 5000
col username for a15
col osuser for a15
col program for a20
col "LOGON_TIME" for a23
col "LAST_CALL_HRS" for 99999.999
col status for a8
col machine for a15
col SQL_TEXT for a90
col P1TEXT for a10
col P2TEXT for a10
col P3TEXT for a10
col p1 for 9999999999999
col p2 for 9999999999999
col p3 for 9999999999999
col event for a50
col "LATCH_NAME" for a20
select event,count(event) "LATCH_COUNT" from v$session_wait having count(event)> 2
```

```
and event like '%latch%' group by event;
```

LATCH SESSIONS DETAIL:
=========================

```
col event for a10
select s.sid,username,osuser,program,machine,status,to_char(logon_time,'DD-MON-YYYY
HH24:MI:SS') "LOGON_TIME",last_call_et/3600 "LAST_CALL_HRS",sw.event from v$session
s,v$session_wait sw where s.sid=sw.sid and sw.event like '%latch%';
```

SQL_TEXT OF LATCH SESSIONS:
==============================

```
select s.sid,username,sql_text,sw.event,l.name "LATCH_NAME" from v$session
s,v$session_wait sw,v$sqltext sq,v$latch l where s.sid=sw.sid and sq.address =
s.sql_address and l.latch# = sw.p2 and sw.event like '%latch%' order by
s.sid,piece;
```

DB File Scattered read Monitoring and Troubleshooting
===

Note:
This event signifies that the user process is reading buffers into the SGA buffer cache and is waiting for a physical I/O call to return. A db file scattered read issues a scatter-read to read the data into multiple discontinuous memory locations. A scattered read is usually a multiblock read. It can occur for a fast full scan (of an index) in addition to a full table scan.

The db file scattered read wait event identifies that a full table scan is occurring. When performing a full table scan into the buffer cache, the blocks read are read into memory locations that are not physically adjacent to each other. Such reads are called scattered read calls, because the blocks are scattered throughout memory. This is why the corresponding wait event is called 'db file scattered read'. Multiblock (up to DB_FILE_MULTIBLOCK_READ_COUNT blocks) reads due to full table scans into the buffer cache show up as waits for 'db file scattered read'.

DB_FILE_SCATTERED_READ COUNT ON DATABASE:
==

```
set pagesize 5000
set lines 185
set long 5000
col username for a15
col osuser for a15
col program for a20
col "LOGON_TIME" for a23
col status for a8
col machine for a15
col SQL_TEXT for a90
col EVENT for a50
col P1TEXT for a10
col P2TEXT for a10
col P3TEXT for a10
col p1 for 9999999999999
col p2 for 9999999999999
col p3 for 9999999999999
col "LAST_CALL_HRS" for 99999.999
col STATE for a12
select event,count(event) "DB_FILE_SCATTERED_READ_COUNT" from v$session_wait having
count(event)>= 1 and event like '%scattered%' group by event;
```

DB_FILE_SCATTERED_READ SESSIONS DETAIL:
==

```
col event for a25
select s.sid,username,osuser,program,machine,status,to_char(logon_time,'DD-MON-YYYY
HH24:MI:SS') "LOGON_TIME",last_call_et/3600 "LAST_CALL_HRS",sw.event from
v$session s,v$session_wait sw where s.sid=sw.sid and sw.event like '%scattered%';
```

DB_FILE_SCATTERED_READ_WAIT_DETAIL:
====================================

```
select sid,EVENT,P1TEXT,P1,P2TEXT,P2,P3TEXT,P3,WAIT_TIME,SECONDS_IN_WAIT,STATE from
v$session_wait where event like '%scattered%';
```

SQL_TEXT OF DB_FILE_SCATTERED_READ SESSIONS:
===

```
select sw.sid,username,sql_text "SQL_TEXT",sw.event from v$session s,v$session_wait
sw,v$sqltext sq where s.sid=sw.sid and sq.address = s.sql_address and sw.event like
'%scattered%' order by sw.sid,piece;
```

USE THE BELOW SQL_FILE TO IDENTIFY THE SEGMENT:
===

```
set linesize 150
set pagesize 5000
col owner for a15
col segment_name for a30
SELECT owner,segment_name,segment_type FROM dba_extents WHERE file_id=&file AND
&block_id BETWEEN block_id AND block_id + blocks -1 ;
```

###################Troubleshooting BUFFER BUSY WAITS
===

Note:
This wait indicates that there are some buffers in the buffer cache that multiple processes are attempting to access concurrently. Query V$WAITSTAT for the wait statistics for each class of buffer. Common buffer classes that have buffer busy waits include data block, segment header, undo header, and undo block.

BUFFER BUSY WAITS COUNT ON DATABASE:
=====================================

```
set pagesize 5000
set lines 180
set long 5000
col username for a15
col osuser for a15
col program for a20
col "LOGON_TIME" for a23
col status for a8
col machine for a15
col SQL_TEXT for a90
col EVENT for a50
col P1TEXT for a10
col P2TEXT for a10
col P3TEXT for a10
col p1 for 9999999999999
col p2 for 9999999999999
col p3 for 9999999999999
col "LAST_CALL_HRS" for 99999.999


select event,count(event) "BUFFER_BUSY_WAITS/LOCK_COUNT" from v$session_wait having
count(event)>= 1 and event like '%buffer busy waits%' group by event;


BUFFER_BUSY_WAITS SESSIONS DETAIL:
===================================

col event for a10
select s.sid,username,osuser,program,machine,status,to_char(logon_time,'DD-MON-YYYY
HH24:MI:SS') "LOGON_TIME",last_call_et/3600 "LAST_CALL_HRS",sw.event from
v$session s,v$session_wait sw where s.sid=sw.sid and sw.event like '%buffer busy
waits%';


SQL_TEXT OF BUFFER_BUSY_WAITS SESSIONS:
==========================================

col "EVENT" for a25
select s.sid,username "USERNAME",sql_text "SQL_TEXT",sw.event "EVENT" from
v$session s,v$session_wait sw,v$sqltext sq where s.sid=sw.sid and
sq.address = s.sql_address and sw.event like '%buffer busy waits% order by
sw.sid,piece';


TYPE_OF_SEGMENT_CONTENDED_FOR
===============================

SELECT class, count FROM V$WAITSTAT WHERE count > 0 ORDER BY count DESC;


USE THE BELOW SQL_FILE TO IDENTIFY THE SEGMENT
===================================================

set linesize 150
set pagesize 5000
col owner for a15
col segment_name for a30
SELECT owner,segment_name,segment_type FROM dba_extents WHERE file_id=&file AND
```

```
&block_id BETWEEN block_id AND block_id + blocks -1 ;



###################Oracle Wait Event Analysis
===============================================


Wait event
===========
column seq# format 99999
column EVENT format a30
column p2 format 999999
column STATE format a10
column WAIT_T format 9999
select SID,SEQ#,EVENT,P1,P2,WAIT_TIME WAIT_T,SECONDS_IN_WAIT,STATE
from v$session_wait
where sid = '&sid' ;

WAIT EVENT DETAILS COMPLETE


Wait event List in DB
======================

select event,count(event) "EVENT_COUNT" from v$session_wait group by event order by
event;
To Find Wait Events for a given Session


column seq# format 99999
column EVENT format a30
column p2 format 9999
column STATE format a10
column WAIT_T format 9999
select SID,SEQ#,EVENT,P1,P2,WAIT_TIME WAIT_T,SECONDS_IN_WAIT,STATE
from gv$session_wait
where sid =  '&sid' ;


To Find Wait Event details of a specific wait event
====================================================

column seq# format 99999
column EVENT format a30
column p2 format 9999
column STATE format a10
column WAIT_T format 9999
select SID,SEQ#,EVENT,P1,P2,WAIT_TIME WAIT_T,SECONDS_IN_WAIT,STATE
from gv$session_wait
where event like '%cursor: pin S%';


Count of sessions ordered by wait event associated
===================================================

SELECT count(*), event FROM v$session_wait WHERE wait_time = 0 AND event NOT IN
('smon
timer','pmon timer','rdbms ipc message','SQL*Net message from client') GROUP BY
```

```
event ORDER BY 1
DESC;
```

To find Wait event Most of the time the session waited for
==

```
select event,TOTAL_WAITS ,TOTAL_TIMEOUTS,TIME_WAITED from gv$session_event where
sid=54
order by TIME_WAITED
```

To find the list of wait events and count of associated sessions
==

```
select count(sid),event from v$session_wait group by event order by 1;
```

No of events with sid's
==========================

```
prompt Sessions Wait Event Summary
select EVENT,COUNT(SID)
from v$session_wait
GROUP BY EVENT;
```

Obtaining a parameter defined
==============================

```
col value for a10
col description for a30
select name,value,description from v$parameter where name like '%timed_statistics
%';
```

Wait events

```
set linesize 152
set pagesize 80
column EVENT format a30
select *  from  v$system_event
where  event like '%wait%';
```

Sessions waiting "sql*net message from client"

```
prompt Sessions having Wait Event "sql*net message from client"
select program,module,count(s.sid) from v$session s, v$session_Wait w
where w.sid=s.sid and w.event='SQL*Net message from client' group by program,module
having
count(s.sid)>5 order by count(s.sid);
```

Sessions having Wait Event "sql*net message from client" from more than 1Hour

```
select program,module,count(s.sid) from v$session s, v$session_Wait w
```

```
where w.sid=s.sid
and s.last_call_et > 3600
and w.event='SQL*Net message from client' group by program,module  having
count(s.sid)>5 order by count(s.sid);
```

Sessions having Wait Event "sql*net message from client"

```
select
s.sid,s.process,S.STATUS,s.program,s.module,s.sql_hash_value,s.last_call_et/3600
Last_Call_Et_HRS
from v$session s, v$session_Wait w
where w.sid=s.sid and w.event='SQL*Net message from client'
and s.module='&Module_name'
order by 6 desc;
```

Segment Statistics
=====================

```
select
object_name,
statistic_name,
value
from
V$SEGMENT_STATISTICS
where object_name ='SOURCE$';
```

```
select    statistic_name,  count(object_name)  from V$SEGMENT_STATISTICS
where STATISTIC_NAME like 'physical%'
group by statistic_name;
```

```
select distinct(STATISTIC_NAME) from v$SEGMENT_STATISTICS;
```

V$SYSTEM_EVENT

This view contains information on total waits for an event.
Note that the TIME_WAITED and AVERAGE_WAIT columns will contain
a value of zero on those platforms that do not support a fast timing mechanism.
If you are running on one of these platforms and you want this column to reflect
true wait times, you must set TIMED_STATISTICS to TRUE in the parameter file;
doing this will have a small negative effect on system performance.

Buffer Busy waits
======================

```
SELECT * FROM v$event_name WHERE name = 'buffer busy waits';
```

```
SELECT   sid, event, state, seconds_in_wait, wait_time, p1, p2, p3
FROM     v$session_wait
WHERE    event = 'buffer busy waits'
ORDER BY sid;
```

```
select * from v$waitstat;


SELECT   sid, event, state, seconds_in_wait, wait_time, p1, p2, p3
FROM     v$session_wait
WHERE    event = 'buffer busy waits'
ORDER BY sid;


Segment details from File number
=================================

SELECT owner, segment_name, segment_type
FROM   dba_extents
WHERE  file_id = &absolute_file_number
AND    &block_number BETWEEN block_id AND block_id + blocks -1;


Direct path write


SELECT * FROM v$event_name WHERE name = 'direct path write';


SELECT tablespace_name, file_id "AFN", relative_fno "RFN"
FROM   dba_data_files
WHERE  file_id = 201;
SELECT tablespace_name, file_id "AFN", relative_fno "RFN"
FROM   dba_data_files
WHERE  file_id = 201;


Total waits/time waited/max wait for a session


SELECT   event, total_waits, time_waited, max_wait
FROM     v$session_event
WHERE    sid = 47
ORDER BY event;


SELECT   A.name, B.value
FROM     v$statname A, v$sesstat B
WHERE    A.statistic# = 12
AND      B.statistic# = A.statistic#
AND      B.sid = 47;
Sessions Ordered by Wait event in Database


set lines 150
set pages 500
col event for a50
select event,count(event) "EVENT_COUNT" from v$session_event group by event order
by event;



###################333LOCKED OBJECTS ANALYSIS
```

==

Finding all the locked objects in db currently
==

```
select oracle_username, os_user_name,locked_mode,object_name,object_type from
v$locked_object a, dba_objects b where a.object_id=b.object_id;
```

Finding the locking session id using the object name
==

```
select OBJECT_ID,SESSION_ID,ORACLE_USERNAME from v$locked_object where OBJECT_ID in
(select object_id from dba_objects where object_name='%IVFC_ITEM2VFC%');
```

```
select OBJECT_ID,SESSION_ID,ORACLE_USERNAME from v$locked_object where OBJECT_ID in
(select object_id from dba_objects where object_name in
('L4REPL_FILI_ART_DAG','L4REPL_ART_ACTI_DAG','L4REPL_FILI_FASG_ACTI_DAG'));
```

Identifying count of locks for a object
==

```
select b.object_name,count(*) from V$LOCKED_OBJECT a,DBA_OBJECTS b where
a.object_id=b.object_id
group by b.object_name having count(*) > 1 order by 2;
```

With Lock mode
================

```
select  oracle_username, object_name,decode(a.locked_mode,
0, 'None',           /* Mon Lock equivalent */
1, 'Null',           /* N */
2, 'Row-S (SS)',     /* L */
3, 'Row-X (SX)',     /* R */
4, 'Share',          /* S */
5, 'S/Row-X (SSX)',  /* C */
6, 'Exclusive',      /* X */
to_char(a.locked_mode)) mode_held
from V$LOCKED_OBJECT a,DBA_OBJECTS b where a.object_id = b.object_id;
```

Identifying locking session
============================

```
col object_name for a30 trunc
set lines 100
set pages 500
select a.object_id,a.session_id,b.object_name from v$locked_object a,dba_objects b
where a.object_id = b.object_id order by 3;
```

Identifying Session/Process details which is locking
===

```
select a.session_id,a.process
clientprocess,a.oracle_username,a.os_user_name,b.owner,b.object_name,b.status from
v$locked_object a,dba_objects b where a.object_id=b.object_id and
a.session_id=&sid;
```

Example

```
select SID, STATUS, LAST_CALL_ET, SERIAL# from v$session where SID in (select
SESSION_ID
    from v$locked_object where OBJECT_ID in (select OBJECT_ID from dba_objects
where
    OBJECT_NAME='MSC_ST_SALES_ORDERS'));
```

Killing a Session/Process Forcefully
=====================================

```
alter system kill session 'sid,serial#';
```

OR

```
alter system kill session 'sid,serial#' immediate;
```

If session Marked for kill, then identify and kill the associated Server Process of the session using any of the session details script.

```
Kill -9 <SPID>
```

######################To check the Library Cache Lock contention
==

Note:
Library Cache contention is a serious issue. In most cases it would be good to analyze what is holding the library cache lock and killing it will resolve the issue. Library cache events can even bring the database to a hang state. It�s a good idea to identify and kill appropriately as early as possible. But do not kill any mandatory processes or sessions as it may lead to an outage. Contact Oracle support for critical issues.

Library cache resource types waited for over the life of the instance
==

```
set linesize 152
column average_wait format 9999990.00


select     substr(e.event, 1, 40) event,
e.time_waited,
e.time_waited / decode(
e.event,
'latch free', e.total_waits,
decode(e.total_waits - e.total_timeouts,0, 1,e.total_waits - e.total_timeouts))
average_wait
from     sys.v$system_event e,
sys.v$instance i
where     e.event like '%library cache%';
```

Detect sessions waiting for a Library Cache Locks
==

```
select sid Waiter, p1raw,
substr(rawtohex(p1),1,30) Handle,
substr(rawtohex(p2),1,30) Pin_addr
from v$session_wait where wait_time=0 and event like '%library cache%';
```

Sessions waiting for lib cache in RAC
==

```
select a.sid Waiter,b.SERIAL#,a.event,a.p1raw,
substr(rawtohex(a.p1),1,30) Handle,
substr(rawtohex(a.p2),1,30) Pin_addr
from v$session_wait a,v$session b where a.sid=b.sid
and a.wait_time=0 and a.event like 'library cache%';
```

Objects locked by Library Cache based on sessions detected above
===

```
select to_char(SESSION_ID,'999') sid ,
substr(LOCK_TYPE,1,30) Type,
substr(lock_id1,1,23) Object_Name,
substr(mode_held,1,4) HELD, substr(mode_requested,1,4) REQ,
lock_id2 Lock_addr
from dba_lock_internal
where
mode_requested<>'None'
and mode_requested<>mode_held
and session_id in ( select sid
from v$session_wait where wait_time=0
and event like '%library cache%') ;
```

Detect Library Cache holders that sessions are waiting for
===

```
select sid Holder ,KGLPNUSE Sesion , KGLPNMOD Held, KGLPNREQ Req
from x$kglpn , v$session
```

```
where KGLPNHDL in (select p1raw from v$session_wait
where wait_time=0 and event like '%library cache%')
and KGLPNMOD <> 0
and v$session.saddr=x$kglpn.kglpnuse ;
```

Sessions holding the lib cache in RAC
==

```
select a.sid Holder ,a.SERIAL#,b.INST_ID,b.KGLPNUSE Sesion , b.KGLPNMOD Held,
b.KGLPNREQ Req
from x$kglpn b , v$session a
where b.KGLPNHDL in (select p1raw from v$session_wait
where wait_time=0 and event like 'library cache%')
and b.KGLPNMOD <> 0
and a.saddr=b.kglpnuse ;
```

What are the holders waiting for?
====================================

```
select sid,substr(event,1,30),wait_time
from v$session_wait
where sid in (select sid from x$kglpn , v$session
where KGLPNHDL in (select p1raw from v$session_wait
where wait_time=0 and event like 'library cache%')
and KGLPNMOD <> 0
and v$session.saddr=x$kglpn.kglpnuse );
```

Note:
Sometimes using the library query below to identify the holding sessions can cause temp tablespace to run out of space�.

```
ORA-01652: unable to extend temp segment by 256 in tablespace TEMP
Current SQL statement for this session:
select to_char(SESSION_ID,'999') sid ,
substr(LOCK_TYPE,1,30) Type,
substr(lock_id1,1,23) Object_Name,
substr(mode_held,1,4) HELD, substr(mode_requested,1,4) REQ,
lock_id2 Lock_addr
from dba_lock_internal
where
mode_requested<>'None'
and mode_requested<>mode_held
and session_id in ( select sid
from v$session_wait where wait_time=0
and event like 'library cache%')
```

Solution: Henceforth, Please use queries using x$kglob or x$kgllk or x$kglpn.. or tweak the following sql (picked up from metalink) to gather lib cache lock event details

ReWritten SQL

```
select /*+ ordered */ w1.sid  waiting_session,
h1.sid  holding_session,
w.kgllktype lock_or_pin,
w.kgllkhdl address,
decode(h.kgllkmod,  0, 'None', 1, 'Null', 2, 'Share', 3, 'Exclusive',
'Unknown') mode_held,
decode(w.kgllkreq,  0, 'None', 1, 'Null', 2, 'Share', 3, 'Exclusive',
'Unknown') mode_requested
from dba_kgllock w, dba_kgllock h, v$session w1, v$session h1
where
(((h.kgllkmod != 0) and (h.kgllkmod != 1)
and ((h.kgllkreq = 0) or (h.kgllkreq = 1)))
and
(((w.kgllkmod = 0) or (w.kgllkmod= 1))
and ((w.kgllkreq != 0) and (w.kgllkreq != 1))))
and  w.kgllktype  =  h.kgllktype
and  w.kgllkhdl  =  h.kgllkhdl
and  w.kgllkuse     =   w1.saddr
and  h.kgllkuse     =   h1.saddr;


Library cache pin sessions
==========================

SELECT s.sid,
waiter.p1raw w_p1r,
holder.event h_wait,
holder.p1raw h_p1r,
holder.p2raw h_p2r,
holder.p3raw h_p2r,
count(s.sid) users_blocked,
sql.hash_value
FROM
v$sql sql,
v$session s,
x$kglpn p,
v$session_wait waiter,
v$session_wait holder
WHERE
s.sql_hash_value = sql.hash_value and
p.kglpnhdl=waiter.p1raw and
s.saddr=p.kglpnuse and
waiter.event like 'library cache pin' and
holder.sid=s.sid
GROUP BY
s.sid,
waiter.p1raw ,
holder.event ,
holder.p1raw ,
holder.p2raw ,
holder.p3raw ,
sql.hash_value;


Library Cache lock Query
==========================
select decode(lob.kglobtyp,
0, 'NEXT OBJECT ',
1, 'INDEX ',
2, 'TABLE ',
3, 'CLUSTER ',
4, 'VIEW ',
5, 'SYNONYM ',
6, 'SEQUENCE ',
7, 'PROCEDURE ',
8, 'FUNCTION ',
9, 'PACKAGE ',
11, 'PACKAGE BODY ',
12, 'TRIGGER ',
13, 'TYPE ',
14, 'TYPE BODY ',
19, 'TABLE PARTITION ',
20, 'INDEX PARTITION ',
21, 'LOB ',
22, 'LIBRARY ',
23, 'DIRECTORY ',
24, 'QUEUE ',
28, 'JAVA SOURCE ',
29, 'JAVA CLASS ',
30, 'JAVA RESOURCE ',
32, 'INDEXTYPE ',
33, 'OPERATOR ',
34, 'TABLE SUBPARTITION ',
35, 'INDEX SUBPARTITION ',
40, 'LOB PARTITION ',
41, 'LOB SUBPARTITION ',
42, 'MATERIALIZED VIEW ',
43, 'DIMENSION ',
44, 'CONTEXT ',
46, 'RULE SET ',
47, 'RESOURCE PLAN ',
48, 'CONSUMER GROUP ',
51, 'SUBSCRIPTION ',
52, 'LOCATION ',
55, 'XML SCHEMA ',
56, 'JAVA DATA ',
57, 'SECURITY PROFILE ',
59, 'RULE ',
62, 'EVALUATION CONTEXT ',
'UNDEFINED ') object_type,
lob.kglnaobj object_name,
pn.kglpnmod lock_mode_held,
pn.kglpnreq lock_mode_requested,
ses.sid,
ses.serial#,
ses.username
from sys.v$session_wait vsw,
sys.x$kglob lob,
sys.x$kglpn pn,
sys.v$session ses
where vsw.event = 'library cache lock '
and vsw.p1raw = lob.kglhdadr
and lob.kglhdadr = pn.kglpnhdl
and pn.kglpnmod != 0
and pn.kglpnuse = ses.saddr
/
```

Detect Library Cache holders that sessions are waiting for
===

```
set pagesize 40
select x$kglpn.inst_id,sid Holder ,KGLPNUSE Sesion , KGLPNMOD Held, KGLPNREQ Req
from x$kglpn , gv$session
where KGLPNHDL in (select p1raw from gv$session_wait
where wait_time=0 and event like 'library cache%')
and KGLPNMOD <> 0
and gv$session.saddr=x$kglpn.kglpnuse ;
PROMPT Detect Library Cache holders that sessions are waiting for


Detect sessions waiting for a Library Cache Locks
==================================================

select sid Waiter,  p1raw,
substr(rawtohex(p1),1,30) Handle,
substr(rawtohex(p2),1,30) Pin_addr
from gv$session_wait where wait_time=0 and event like 'library cache%';


Sessions waiting on Library Cache events
==========================================

set pagesize 80
undefine spid
col spid for a8
col INST_ID for 99
col sid for 99999
set linesize 140
col action format a20
col logon_time format a15
col module format a13
col cli_process format a7
col cli_mach for a15
col status format a10
col username format a10
col event for a20
col program for a20
col "Last SQL" for a30
col last_call_et_hrs for 999.99
col sql_hash_value for 9999999999999
select p.INST_ID,p.spid,s.sid, s.serial#, s.status,s.last_call_et/3600
last_call_et_hrs ,
s.process cli_process,s.machine cli_mach,sw.event,
s.action,s.module,s.program,s.sql_hash_value,t.disk_reads,lpad(t.sql_text,30) "Last
SQL"
from gv$session s, gv$sqlarea t,gv$process p,gv$session_wait sw
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
s.sid=sw.sid and
sw.event like '%library cache%'
order by p.spid;


set lines 152
col sid for a9999999999999
col name for a40
select a.sid,b.name,a.value,b.class
```

```
from gv$sesstat a , gv$statname b
where a.statistic#=b.statistic#
and  name like '%library cache%';


select sid
from gv$session_wait where wait_time=0
and event like 'library cache%';
```

Objects waiting for Library Cache lock
==

```
col type for a20
set linesize 150
set pagesize 80
col OBJECT_NAME for a20
col LOCK_ADDR for a20


select to_char(SESSION_ID,'999') sid ,
substr(LOCK_TYPE,1,30) Type,
substr(lock_id1,1,23) Object_Name,
substr(mode_held,1,4) HELD, substr(mode_requested,1,4) REQ,
lock_id2 Lock_addr
from dba_lock_internal
where
mode_requested<>'None'
and mode_requested<>mode_held
and session_id in ( select sid
from gv$session_wait where wait_time=0
and event like 'library cache%') ;
```

This script shows the Library cache resource types waited for over the life of the instance
==

```
set linesize 152
column average_wait format 9999990.00
col event for a30
col HANDLE for a20
col PIN_ADDR for a20
col TYPE for a20
col OBJECT_NAME for a35
col LOCK_ADDR for a30


select     substr(e.event, 1, 40) event,
e.time_waited,
e.time_waited / decode(
e.event,
'latch free', e.total_waits,
decode(e.total_waits - e.total_timeouts,0, 1,e.total_waits - e.total_timeouts))
average_wait
from     sys.v$system_event e,
sys.v$instance i
where     e.event like 'library cache%';
```

Count of sessions waiting
===========================

```
select count(SESSION_ID) total_waiting_sessions
from dba_lock_internal
where
mode_requested<>'None'
and mode_requested<>mode_held
and session_id in ( select sid
from v$session_wait where wait_time=0
and event like 'library cache%') ;
```

What are the holders waiting for?
===================================

```
select sid,substr(event,1,30) event,wait_time
from v$session_wait
where sid in (select sid from x$kglpn , v$session
where KGLPNHDL in (select p1raw from v$session_wait
where wait_time=0 and event like 'library cache%')
and KGLPNMOD <> 0
and v$session.saddr=x$kglpn.kglpnuse );
```

Process ids/sid/pid
====================

```
col spid for a6
col sid for 99999
set linesize 152
set pagesize 80
col action format a20
col logon_time format a15
col program for a10
col terminal for a10
col module format a13
col cli_process format a7
col cli_mach for a10
col status format a10
col username format a10
col  last_call_et for 999.999
select  a.sid  ,
a.serial#  ,
a.username ,
a.status  ,
a.machine cli_mach,
a.terminal ,
a.program ,
a.module ,
a.action ,
a.sql_hash_value ,
to_char(a.logon_time,'DD-Mon-YYYY HH24:MI:SS') logon_time ,
round((a.last_call_et/60),2) last_call_et,
a.process cli_process,
b.spid  spid,
b.event event,
b.state  state
from   gv$session a, gv$process b,  gv$session_wait  sw
```

```
where  a.paddr=b.addr and a.inst_id=b.inst_id
and a.sid    in (583,743,669,766
)
and a.inst_id=sw.inst_id
and a.sid=sw.sid;


Session details
=================

set linesize 150
col action format a25
col logon_time format a16
col module format a13
col program for a15
col process format a7
col status format a10
col username format a10
col last_call_et for 9999.99
select s.sid,p.spid,s.program, s.serial#, s.status, s.username, s.action,
to_char(s.logon_time, 'DD-MON-YY, HH24:MI') logon_time,
s.module,s.last_call_et/3600 last_call_et,s.process
from gv$session s, gv$process p
where p.addr=s.paddr and s.sid= '&sid';


Last SQL
col username for a10
col "Last SQL" for a65
col process for a10
set pagesize 48
select s.username, s.sid,s.process, s.status,t.sql_text "Last SQL"
from gv$session s, gv$sqlarea t
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
s.sid = '&sid';


Wait event
===========

column seq# format 99999
column EVENT format a30
column p2 format 999999
column STATE format a10
column WAIT_T format 9999
select SID,SEQ#,EVENT,P1,P2,WAIT_TIME WAIT_T,SECONDS_IN_WAIT,STATE
from v$session_wait
where sid in (159,610);


Last SQL Multiple inputs
col username for a10
col "Last SQL" for a65
col process for a10
set pagesize 48
select s.username, s.sid,s.process, s.status,t.sql_text "Last SQL"
from gv$session s, gv$sqlarea t
where s.sql_address =t.address and
```

```
s.sql_hash_value =t.hash_value and
s.sid in (190,224,217,306);
```

Active transactions multiple inputs
==

```
select username,s.sid,
t.used_ublk,t.used_urec
from v$transaction t,v$session s
where t.addr=s.taddr and
s.sid in (159,610);
```

Session details thru server process id
=======================================

```
undefine spid
col spid for a10
set linesize 150
col action format a10
col logon_time format a16
col module format a13
col cli_process format a7
col cli_mach for a15
col status format a10
col username format a10
col last_call_et for 9999.99
select p.spid,s.sid, s.serial#, s.status, s.username, s.action,
to_char(s.logon_time, 'DD-MON-YY, HH24:MI') logon_time,
s.module,s.last_call_et/3600 last_call_et,s.process cli_process,s.machine cli_mach
from gv$session s, gv$process p
where p.addr=s.paddr and p.spid= '&spid';
```

Session details thru Client process name
==

```
undefine spid
col spid for a6
col sid for 99999
set linesize 140
col action format a20
col logon_time format a15
col module format a13
col cli_process format a7
col cli_mach for a10
col status format a10
col username format a10
select p.spid,s.sid, s.serial#, s.status, s.username, s.action,
to_char(s.logon_time, 'DD-MON-YY, HH24:MI') logon_time,
s.module,s.last_call_et/3600,s.process cli_process,s.machine cli_mach
from gv$session s, gv$process p
where p.addr=s.paddr and s.process = '&cli_process';
```

Details thru multiple inputs of SIDs
====================================

```
undefine spid
col last_call_et for 9999999
set pagesize 40
col spid for a6
col sid for 99999
set linesize 150
col action format a20
col logon_time format a15
col module format a13
col cli_process format a7
col cli_mach for a10
col status format a10
col last_call_et for 999.99
col username format a10
select p.spid,s.sid, s.serial#, s.status, s.username, s.action,
to_char(s.logon_time, 'DD-MON-YY, HH24:MI') logon_time,
s.module,s.last_call_et/3600 last_call_et,s.process cli_process,s.machine cli_mach
from gv$session s, gv$process p
where p.addr=s.paddr and s.sid in (51,146,407,397,389,377,345,302,239,214)
order by sid;
```

This script points to the session that is holding a library cache lock for any object. A typical session will wait on a library cache lock when when it is trying to modify the object definition. The object may be a package/procedure or just a table or index definition

```
SELECT a.KGLPNMOD, a.KGLPNREQ, b.username, c.KGLNAOBJ, c.KGLOBTYP
FROM
x$kglpn a,
v$session b,
x$kglob c
WHERE
a.KGLPNUSE = b.saddr and
upper(c.KGLNAOBJ)  like upper('%&obj_name%') and
a.KGLPNHDL = c.KGLHDADR
/
Exit
```

#######################To check for Various Locking/Blocking Issues
==

Different Lock Modes with explanation
==

```
0, 'None',
1, 'Null',
2, 'Row-S (SS)',
3, 'Row-X (SX)',
4, 'Share',
5, 'S/Row-X (SSX)',
6, 'Exclusive'
```

```
EXCLUSIVE (X)              Lock allows queries but nothing else.
SHARE (S)                  Lock allows queries but not updates.
ROW SHARE (RS)             Lock allows concurrent process access to table. Resource
may not be locked exclusively.
ROW EXCLUSIVE (RX)         Same as row share but no share mode locking. Updates,
deletes and inserts use this lock mode.
SHARE ROW EXCLUSIVE (SRX) Lock used to do selective updates and to allow other
processes to look at rows in table but not                          lock table in
share mode or to update any rows. (Never used).
```

To check for blocking sessions
================================

Indentifying blocking session tree

```
@$ORACLE_HOME/rdbms/admin/catblock.sql


@$ORACLE_HOME/rdbms/admin/utllockt.sql
```

From v$lock

```
select  *  from v$lock  where block>0;


select * from v$lock where request>0;
```

From gv$lock global view

```
select * from gv$lock where block>0;


select * from gv$lock where request>0;
```

From dba_locks

```
select distinct BLOCKING_OTHERS from dba_locks;
```

To find Session information
============================

```
col program for a15F
col machine for a15
col terminal for a15
set lines 152
select s.sid,
s.serial#,
'*'||s.process||'*'  Client,
p.spid Server,
s.sql_address,
s.sql_hash_value,
s.username,
s.action,
s.program || s.module,
s.terminal,
s.machine,
s.status,
--s.last_call_et
```

```
s.last_call_et/3600
from gv$session s, gv$process p
where p.addr=s.paddr and
s.sid=nvl('&sid',s.sid) and
p.spid=nvl('&spid',p.spid) and
nvl(s.process,-1) = nvl('&ClientPid',nvl(s.process,-1));
```

```
select p.username pu,s.username su,s.status stat,s.sid ssid,s.serial#
sser,lpad(p.spid,7) pid,
s.process client_process,s.machine client_machine from v$process p, v$session s
where    p.addr=s.paddr  and s.username is not null
and      s.sid=&BSID;
```

Blockers-Waiters Info

```
col SESS for a12
set lines 132
SELECT DECODE(request,0,'Holder: ','Waiter: ')||sid sess,
id1, id2, lmode, request, type
FROM V$LOCK
WHERE (id1, id2, type) IN
(SELECT id1, id2, type FROM V$LOCK WHERE request>0)
ORDER BY id1, request;
select distinct holding_session from dba_waiters where holding_session not in
(select
waiting_session from dba_waiters);
```

```
select s.sid, s.serial#, s.action, s.module, s.status, s.last_call_et,s.logon_time
from v$session s
where s.sid in (select sid from v$lock where block >0)
order by sid;
```

Blocking session
=================

```
SELECT blocking_sid, num_blocked
FROM ( SELECT blocking_sid, SUM(num_blocked) num_blocked
FROM ( SELECT l.id1, l.id2,
MAX(DECODE(l.block, 1, i.instance_name||'-'||l.sid,
2, i.instance_name||'-'||l.sid, 0 )) blocking_sid,
SUM(DECODE(l.request, 0, 0, 1 )) num_blocked
FROM gv$lock l, gv$instance i
WHERE ( l.block!= 0 OR l.request > 0 ) AND
l.inst_id = i.inst_id
GROUP BY l.id1, l.id2)
GROUP BY blocking_sid
ORDER BY num_blocked DESC)
WHERE num_blocked != 0
```

Blocker-waiter
==============

```
set linesize 80
col sess for a15
SELECT DECODE(request,0,'Holder: ','Waiter: ')||trim(sid) sess,
id1, id2, lmode, request, type
FROM V$LOCK
WHERE (id1, id2, type) IN
(SELECT id1, id2, type FROM V$LOCK WHERE request>0)
ORDER BY id1, request;


SELECT DECODE(request,0,'Holder: ','Waiter: ')||sid sess,inst_id,
id1, id2, lmode, request, type
FROM GV$LOCK
WHERE (id1, id2, type) IN
(SELECT id1, id2, type FROM GV$LOCK WHERE request>0)
ORDER BY id1, request;
```

Blocking sessions in RAC
==========================

```
set linesize 999
select
INST_ID, SID, SERIAL#, MODULE, ACTION
from gv$session
where
SID in (select sid from gv$lock where block=1);


select * from gv$lock where (ID1,ID2) in
(select ID1,ID2 from gv$lock where request>0);


select DECODE(request,0,'Holder: ','Waiter: ')||sid sess,id1, id2, lmode, request,
type,inst_id FROM gV$LOCK WHERE (id1, id2, type) IN (SELECT id1, id2, type FROM
gV$LOCK WHERE
request>0) ORDER BY id1, request;


select * from GV_$GLOBAL_BLOCKED_LOCKS;


select
INST_ID, SID, SERIAL#, MODULE, ACTION, status
from gv$session
where
(SID, inst_id)  in (select sid, inst_id  from gv$lock where block=1);
```

###############To Check the Timing Details of SID and event waiting
==

```
select a.sid, a.serial#, a.status, a.program, b.event,to_char(a.logon_time, 'dd-
mon-yy hh24:mi') LOGON_TIME,
to_char(Sysdate, 'dd-mon-yy--hh24:mi') CURRENT_TIME, (a.last_call_et/3600) "Hrs
connected" from v$session a,
v$session_wait b where a.sid=&sid and a.sid=b.sid;
```

```
Identifying Sever Machine and Client Machine names of a session
===============================================================

col server_machine for a15
col client_machine for a15
select p.spid server_pid, s.sid oracle_sid, s.machine client_machine,i.host_name
server_machine
from gv$session s, gv$process p, gv$instance I
where
p.inst_id = s.inst_id and
p.addr = s.paddr and
i.inst_id = s.inst_id and
p.spid = '&server_pid';



##############To check for Long Running Sessions
===================================================



SESSIONS ACTIVE FOR MORE THAN AN HOUR:


set pagesize 5000
set lines 150
col username for a15
col osuser for a15
col program for a20
col logon_time for a20
col status for a8
col machine for a15
col sql_text for a100
col EVENT for a30
col P1TEXT for a10
col P2TEXT for a10
col P3TEXT for a10
col p1 for 9999999999999
col p2 for 9999999999999
col p3 for 9999999999999
col LAST_CALL_ET_HRS for 999999.99


select sid,serial#,username,osuser,program,machine,status,to_char(logon_time,'DD-
MON-YYYY HH24:MI:SS') "LOGON_TIME",last_call_et/3600 " LAST_CALL_ET_HRS" from
v$session where status='ACTIVE' AND username is not null and last_call_et/3600 >1
order by 9 desc;



SQL_TEXT OF LONG RUNNING SESSIONS:


select sid,sql_text from v$sqltext , v$session where v$sqltext.address =
v$session.sql_address and sid in (select sid from v$session where status='ACTIVE'
AND username is not null and last_call_et/3600 >1) order by sid,piece;
```

WAIT_EVENT OF LONG RUNNING SESSIONS:

```
select sid,EVENT,P1TEXT,P1,P2TEXT,P2,P3TEXT,P3,WAIT_TIME,SECONDS_IN_WAIT,STATE from
v$session_wait where sid in (select sid from v$session where status='ACTIVE' AND
username is not null and last_call_et/3600 >1);
```

Active transaction

```
select username,s.sid,
t.used_ublk,t.used_urec
from v$transaction t,v$session s
where t.addr=s.taddr and
s.sid= '&sid' ;
```

Checking for active transactions SID

```
select username,t.used_ublk,t.used_urec from v$transaction t,v$session s where
t.addr=s.taddr and s.sid='&sessionid';
```

Checking rollback/Undo segment info used by SID

```
column rr heading 'RB Segment' format a18
column us heading 'Username' format a15
column os heading 'OS User' format a10
column te heading 'Terminal' format a10
SELECT r.name rr, nvl(s.username,'no transaction') us,s.sid, s.osuser os,
s.terminal te, rs.rssize,
rs.xacts, rs.rssize/1048576 Rssize
FROM v$lock  l, v$session  s,v$rollname  r , v$rollstat rs
WHERE l.sid = s.sid(+) AND trunc(l.id1/65536) = r.usn AND l.type = 'TX' AND
l.lmode = 6   AND r.usn=rs.usn  and s.sid in (&sid_list_comma_sep);
```

###################Session information on known SID basis
===

```
set echo off
set linesize 132
set verify off
set feedback off
set serveroutput on;
declare
  SID             number        := 0 ;
  SERIAL          number        := 0 ;
  username        varchar(20)   := '';
  Status          varchar(8)    := '';
  machine         varchar(10)   := '';
  terminal        varchar(25)   := '';
  program         varchar(30)   := '';
  Module          varchar(30)   := '';
  Action          varchar(20)   := '';
  sql_hash_value  number        := 0 ;
  logontime       varchar(30)   := '';
  last_call_et    number        := 0 ;
  proc            number        := 0 ;
  spid            number        := 0 ;
  event           varchar(30)   := '';
  state           varchar(30)   := '';
  sql_text        varchar(1000) := '';
cursor cur1 is
select
    a.sid  sid,
    a.serial#  serial,
    a.username username,
    a.status status ,
    a.machine machine,
    a.terminal terminal,
    a.program program,
    a.module module,
    a.action action,
    a.sql_hash_value sql_hash_value,
    to_char(a.logon_time,'DD-Mon-YYYY HH:MI:SS') logontime,
    a.last_call_et last_call_et,
    a.process proc,
    b.spid spid,
    sw.event event,
    sw.state state
  from gv$session a, gv$process b,  gv$session_wait  sw
  where a.paddr=b.addr
    and a.inst_id=b.inst_id
    and a.sid ='&sid'
    and a.inst_id=sw.inst_id
    and a.sid=sw.sid;
begin

DBMS_OUTPUT.PUT_LINE('------------------------------------------------------------
----');
  DBMS_OUTPUT.PUT_LINE(' Database session detail for the shadow process ');

DBMS_OUTPUT.PUT_LINE('------------------------------------------------------------
----');
  for m in cur1 loop
    DBMS_OUTPUT.ENABLE(50000);
    DBMS_OUTPUT.PUT_LINE('  ');
    DBMS_OUTPUT.PUT_LINE('SID............ : ' || m.sid);
    DBMS_OUTPUT.PUT_LINE('SERIAL#........ : ' || m.serial);
    DBMS_OUTPUT.PUT_LINE('USERNAME....... : ' || m.username);
    DBMS_OUTPUT.PUT_LINE('STATUS......... : ' || m.status);
    DBMS_OUTPUT.PUT_LINE('Machine........ : ' || m.machine);
    DBMS_OUTPUT.PUT_LINE('Terminal....... : ' || m.terminal);
    DBMS_OUTPUT.PUT_LINE('Program........ : ' || m.program);
    DBMS_OUTPUT.PUT_LINE('Module......... : ' || m.module);
    DBMS_OUTPUT.PUT_LINE('Action......... : ' || m.action);
    DBMS_OUTPUT.PUT_LINE('SQL Hash Value. : ' || m.sql_hash_value);
    DBMS_OUTPUT.PUT_LINE('Logon Time..... : ' || m.logontime);
    DBMS_OUTPUT.PUT_LINE('Last Call Et... : ' || m.last_call_et);
    DBMS_OUTPUT.PUT_LINE('Process ID..... : ' || m.proc);
    DBMS_OUTPUT.PUT_LINE('SPID........... : ' || m.spid);
    DBMS_OUTPUT.PUT_LINE('Session Waiting for event:'||m.event);
    DBMS_OUTPUT.PUT_LINE('Session state ...........:'||m.state);
    for rec in ( select distinct(sql_text) sql_text from v$session s,v$sql v where
        s.sql_hash_value=v.hash_value and s.sql_address=v.address and s.sid=m.sid)
    loop
      dbms_output.put_line(substr('SQL_TEXT is..........:'||rec.sql_text,1,255));
    end loop;

DBMS_OUTPUT.PUT_LINE('------------------------------------------------------------------
-------');
    DBMS_OUTPUT.PUT_LINE(' ');
  end loop;
end;
/


--       dbms_output.put_line(SUBSTR('SQL_TEXT is..........:'||
rec.sql_text,1,255));
```

Another version

```
set echo off
set linesize 132
set verify off
set feedback off
set serveroutput on;
declare
  SID             number        := 0 ;
  SERIAL          number        := 0 ;
  username        varchar(20)   := '';
  Status          varchar(8)    := '';
  machine         varchar(10)   := '';
  terminal        varchar(25)   := '';
  program         varchar(30)   := '';
  Module          varchar(30)   := '';
  Action          varchar(20)   := '';
  sql_hash_value  number        := 0 ;
  logontime       varchar(30)   := '';
  last_call_et    number        := 0 ;
  proc            number        := 0 ;
  spid            number        := 0 ;
  event           varchar(30)   := '';
  state           varchar(30)   := '';
  sql_text        varchar(1000) := '';
cursor cur1 is
select
    a.sid  sid,
    a.serial#  serial,
    a.username username,
    a.status status ,
    a.machine machine,
    a.terminal terminal,
    a.program program,
    a.module module,
    a.action action,
    a.sql_hash_value sql_hash_value,
    to_char(a.logon_time,'DD-Mon-YYYY HH:MI:SS') logontime,
    a.last_call_et last_call_et,
    a.process proc,
    b.spid spid,
    b.event event,
    b.state state
  from gv$session a, gv$process b,  gv$session_wait  sw
  where a.paddr=b.addr
    and a.inst_id=b.inst_id
    and b.spid='&1'
    and a.inst_id=sw.inst_id
    and a.sid=sw.sid;
begin

DBMS_OUTPUT.PUT_LINE('-------------------------------------------------------------
----');
  DBMS_OUTPUT.PUT_LINE(' Database session detail for the shadow process ');

DBMS_OUTPUT.PUT_LINE('-------------------------------------------------------------
----');
  for m in cur1 loop
    DBMS_OUTPUT.ENABLE(50000);
    DBMS_OUTPUT.PUT_LINE('  ');
    DBMS_OUTPUT.PUT_LINE('SID............ : ' || m.sid);
    DBMS_OUTPUT.PUT_LINE('SERIAL#........ : ' || m.serial);
    DBMS_OUTPUT.PUT_LINE('USERNAME....... : ' || m.username);
    DBMS_OUTPUT.PUT_LINE('STATUS......... : ' || m.status);
    DBMS_OUTPUT.PUT_LINE('Machine........ : ' || m.machine);
    DBMS_OUTPUT.PUT_LINE('Terminal....... : ' || m.terminal);
    DBMS_OUTPUT.PUT_LINE('Program........ : ' || m.program);
    DBMS_OUTPUT.PUT_LINE('Module......... : ' || m.module);
    DBMS_OUTPUT.PUT_LINE('Action......... : ' || m.action);
    DBMS_OUTPUT.PUT_LINE('SQL Hash Value. : ' || m.sql_hash_value);
    DBMS_OUTPUT.PUT_LINE('Logon Time..... : ' || m.logontime);
    DBMS_OUTPUT.PUT_LINE('Last Call Et... : ' || m.last_call_et);
    DBMS_OUTPUT.PUT_LINE('Process ID..... : ' || m.proc);
    DBMS_OUTPUT.PUT_LINE('SPID........... : ' || m.spid);
    DBMS_OUTPUT.PUT_LINE('Session Waiting for event:'||m.event);
    DBMS_OUTPUT.PUT_LINE('Session state ...........:'||m.state);

DBMS_OUTPUT.PUT_LINE('-------------------------------------------------------------
-------');
    DBMS_OUTPUT.PUT_LINE(' ');
  end loop;
end;
/
```

###############Queries to get the SESSION INFORMATION
==

Checking Timing details, Client PID of associated oracle SID
===

```
set head off
set verify off
set echo off
set pages 1500
set linesize 100
set lines 120
prompt
prompt Details of SID / SPID / Client PID
prompt ==================================
select /*+ CHOOSE*/
select
'Session  Id.............................................: '||s.sid,
'Serial Num..............................................: '||s.serial#,
'User Name ..............................................: '||s.username,
'Session Status .........................................: '||s.status,
'Client Process Id on Client Machine ....................: '||'*'||s.process||'*'
Client,
'Server Process ID ......................................: '||p.spid Server,
'Sql_Address ............................................: '||s.sql_address,
'Sql_hash_value .........................................: '||s.sql_hash_value,
'Schema Name ..... ......................................: '||s.SCHEMANAME,
'Program  ...............................................: '||s.program,
'Module .................................................: '|| s.module,
'Action .................................................: '||s.action,
'Terminal ...............................................: '||s.terminal,
'Client Machine .........................................: '||s.machine,
'LAST_CALL_ET ...........................................: '||s.last_call_et,
'S.LAST_CALL_ET/3600 ....................................: '||s.last_call_et/3600
from v$session s, v$process p
where p.addr=s.paddr and
s.sid=nvl('&sid',s.sid) and
p.spid=nvl('&spid',p.spid) and
nvl(s.process,-1) = nvl('&ClientPid',nvl(s.process,-1));
```

To Find Session Information Details based on SID or SPID or CLIENTPID
==

```
col program for a15F
col machine for a15
col terminal for a15
set lines 152
select s.sid,
s.serial#,
'*'||s.process||'*'  Client,
p.spid Server,
s.sql_address,
s.sql_hash_value,
s.username,
s.action,
s.program || s.module,
s.terminal,
s.machine,
s.status,
--s.last_call_et
s.last_call_et/3600
from gv$session s, gv$process p
where p.addr=s.paddr and
s.sid=nvl('&sid',s.sid) and
p.spid=nvl('&spid',p.spid) and
nvl(s.process,-1) = nvl('&ClientPid',nvl(s.process,-1));
```

```
Checking Timing details, Client PID of associated oracle SID
==============================================================

undefine spid
set pagesize 40
col INST_ID for 99
col spid for a10
set linesize 150
col action format a10
col logon_time format a16
col module format a13
col cli_process format a7
col cli_mach for a15
col status format a10
col username format a10
col last_call_et for 9999.99
col sql_hash_value for 9999999999999
select p.INST_ID,p.spid,s.sid, s.serial#, s.status, s.username, s.action,
to_char(s.logon_time, 'DD-MON-YY, HH24:MI') logon_time,
s.module,s.program,s.last_call_et/3600 last_call_et ,s.process
cli_process,s.machine
cli_mach,s.sql_hash_value
from gv$session s, gv$process p
where p.addr=s.paddr and p.spid in(&SPID);


Checking Timing Details of SID and event waiting for
====================================================

select a.sid, a.serial#, a.status, a.program, b.event,to_char(a.logon_time, 'dd-
mon-yy hh24:mi') LOGON_TIME,
to_char(Sysdate, 'dd-mon-yy--hh24:mi') CURRENT_TIME, (a.last_call_et/3600) "Hrs
connected" from v$session a,
v$session_wait b where a.sid in(&SIDs) and a.sid=b.sid;


Checking for active transactions SID
=======================================

select username,t.used_ublk,t.used_urec from v$transaction t,v$session s where
t.addr=s.taddr and s.sid in(&SIDs);




Checking what is the Last SQL (input multiple sids)
=====================================================

undefine sid
col "Last SQL" for a70
select s.username, s.sid, s.serial#,t.sql_text "Last SQL"
from gv$session s, gv$sqlarea t
where s.sql_address =t.address and
```

```
s.sql_hash_value =t.hash_value and
s.sid in (&SIDs);
```

All Active and Inactive connections
===================================

```
col program for a15F
col machine for a15
col terminal for a15
set lines 152
select s.sid,
s.serial#,
'*'||s.process||'*'  Client,
p.spid Server,
s.sql_address,
s.sql_hash_value,
s.username,
s.action,
s.program || s.module,
s.terminal,
s.machine,
s.status,
--s.last_call_et
s.last_call_et/3600
from gv$session s, gv$process p
where p.addr=s.paddr and s.type != 'BACKGROUND';


col program for a15F
col machine for a15
col terminal for a15
set lines 152


select s.sid,
s.serial#,
'*'||s.process||'*'  Client,
p.spid Server,
s.sql_address,
s.sql_hash_value,
s.username,
s.action,
s.program || s.module,
s.terminal,
s.machine,
s.status,
--s.last_call_et
s.last_call_et/3600
from gv$session s, gv$process p
where p.addr=s.paddr and
s.sid=nvl('&sid',s.sid) and
p.spid=nvl('&spid',p.spid) and
s.status='ACTIVE' and
--(s.last_call_et/3600)<1 and
nvl(s.process,-1) = nvl('&ClientPid',nvl(s.process,-1));
```

```
Active sessions
================

select p.spid "Thread", s.sid "SID-Top Sessions",
substr(s.osuser,1,15) "OS User", substr(s.program,1,25) "Program Running"
from v$process p, v$session s
where p.addr=s.paddr
order by substr(s.osuser,1,15);




Session details from Session long ops
=======================================

select SID,SERIAL#,OPNAME,SOFAR,TOTALWORK,START_TIME,LAST_UPDATE_TIME,username from
v$session_longops where sid=&SID and serial#=&SERIAL




To list the nodes
====================

set head off
set verify off
set echo off
set pages 1500
set linesize 70
prompt
prompt Environment sketch
prompt ==================================
select /*+ CHOOSE*/
'NODE_NAME.................: '||NODE_NAME,
'CREATION_DATE.............: '||CREATION_DATE,
'CREATED_BY ...............: '||CREATED_BY,
'SUPPORT_CP ...............: '||SUPPORT_CP,
'SUPPORT_FORMS ............: '||SUPPORT_FORMS,
'SUPPORT_WEB ..............: '||SUPPORT_WEB,
'SUPPORT_ADMIN ............: '||SUPPORT_ADMIN,
'STATUS ...................: '||STATUS,
'HOST.DOMAIN ..... ........: '||HOST||'.'||DOMAIN,
'SUPPORT_DB  ..............: '||SUPPORT_DB
from  apps.fnd_nodes;


Session details thru SPID
============================

select sid, serial#, USERNAME, STATUS, OSUSER, PROCESS,
MACHINE, MODULE, ACTION, to_char(LOGON_TIME,'yyyy-mm-dd hh24:mi:ss')
from v$session where paddr in (select addr from v$process where spid = '11533')



Checking  Timing details, Client PID of associated oracle SID
================================================================
set head off
set verify off
set echo off
set pages 1500
set linesize 100
set lines 120
prompt
prompt Details of SID / SPID / Client PID
prompt ==================================
select /*+ CHOOSE*/
'Session  Id.............................................: '||s.sid,
'Serial Num..............................................: '||s.serial#,
'User Name ..............................................: '||s.username,
'Session Status .........................................: '||s.status,
'Client Process Id on Client Machine ....................: '||'*'||s.process||'*'
Client,
'Server Process ID ......................................: '||p.spid Server,
'Sql_Address ............................................: '||s.sql_address,
'Sql_hash_value .........................................: '||s.sql_hash_value,
'Schema Name ..... ......................................: '||s.SCHEMANAME,
'Program  ...............................................: '||s.program,
'Module .................................................: '|| s.module,
'Action .................................................: '||s.action,
'Terminal ...............................................: '||s.terminal,
'Client Machine .........................................: '||s.machine,
'LAST_CALL_ET ...........................................: '||s.last_call_et,
'S.LAST_CALL_ET/3600 ....................................: '||s.last_call_et/3600
from v$session s, v$process p
where p.addr=s.paddr and
s.sid=nvl('&sid',s.sid) and
p.spid=nvl('&spid',p.spid) and
nvl(s.process,-1) = nvl('&ClientPid',nvl(s.process,-1));
```

To list count of connections from other machines

```
select count(1),machine from gv$session where inst_id=2 group by machine;
```

To get total count of sessions and processes

```
select count(*) from v$session;
```

```
select count(*) from v$process;
```

```
select (select count(*) from v$session) sessions, (select count(*) from v$process)
processes from dual;
```

To find sqltext thru sqladdress

```
select sql_address from v$session where sid=1999;
select sql_text from v$sqltext where ADDRESS='C00000027FF00AF0' order by PIECE;
```

To find sqltext for different sql hashvalue

```
select hash_value,sql_text from v$sql where hash_value in (1937378691,1564286875,
248741712,2235840973,2787402785)
```

To list long running forms user sessions
==

```
select s.sid,s.process,p.spid,s.status ,s.action,s.module, (s.last_call_et/3600)
from
v$session s, v$process p where round(last_call_et/3600) >4 and action like '%FRM%'
and
p.addr=s.paddr ;
```

To list inactive Sessions respective username
===

```
SELECT username,count(*) num_inv_sess
FROM v$session
where last_call_et > 3600
and username is not null
AND STATUS='INACTIVE'
group by username
order by num_inv_sess DESC;
```

```
SELECT count(*) FROM v$session where last_call_et > 43200 and username is not null
AND
STATUS='INACTIVE';
SELECT count(*) FROM v$session where last_call_et > 3600 and username is not null
AND
STATUS='INACTIVE';
```

To find session id with set of SPIDs
======================================

```
select sid from v$session, v$process where addr=paddr and spid in
('11555','26265','11533');
```

To find Sql Text given SQLHASH & SQLADDR
===

```
select piece,sql_text from v$sqltext where HASH_VALUE = &hash and ADDRESS ='&addr'
order by piece;
select piece,sql_text from v$sqltext where  ADDRESS ='&addr' order by piece;
```

To find Undo Generated For a given session
===

```
select  username,
t.used_ublk ,t.used_urec
from    gv$transaction t,gv$session s
where   t.addr=s.taddr and
s.sid='&sessionid';
```

APPS 11i**

To Find Forms User Session Details Given ClientProcess id
===

```
SELECT /*+ ORDERED FULL(fl) FULL(vp) USE_HASH(fl vp) */
( SELECT SUBSTR ( fu.user_name, 1, 20 )
FROM apps.fnd_user fu
WHERE fu.user_id = fl.user_id
) user_name,
TO_CHAR ( fl.start_time, 'DD-MON-YYYY HH24:MI' ) login_start_time,
SUBSTR ( fl.process_spid, 1, 6 ) spid,
SUBSTR ( TO_CHAR ( fl.pid ), 1, 3 ) pid,
SUBSTR ( vs.process, 1, 8 ) f60webmx,
SUBSTR ( TO_CHAR ( rf.audsid ), 1, 6 ) audsid,
SUBSTR ( TO_CHAR ( vs.sid ), 1, 3 ) sid,
SUBSTR ( TO_CHAR ( vs.serial#), 1, 7 ) serial#,
SUBSTR ( vs.module || ' - ' ||
( SELECT SUBSTR ( ft.user_form_name, 1, 40 )
FROM apps.fnd_form_tl ft
WHERE ft.application_id = rf.form_appl_id
AND ft.form_id = rf.form_id
and ft.language='US'
), 1, 40 ) form
FROM apps.fnd_logins fl,
gv$process vp,
apps.fnd_login_resp_forms rf,
gv$session vs
--fnd_form_tl ft
WHERE fl.end_time IS NULL
AND fl.start_time > sysdate - 31 /* login within last 7 days */
AND fl.login_type = 'FORM'
AND fl.process_spid = vp.spid
AND fl.pid = vp.pid
AND fl.login_id = rf.login_id
AND rf.end_time IS NULL
AND rf.audsid = vs.audsid
AND vs.process='&1'
ORDER BY
user_name,
login_start_time,
spid,
pid,
f60webmx,
sid,
serial#;
```

Checking Timing Details of SID and event waiting for
===

```
select a.sid, a.serial#, a.status, a.program, b.event,to_char(a.logon_time, 'dd-
mon-yy hh24:mi')
LOGON_TIME,
to_char(Sysdate, 'dd-mon-yy--hh24:mi') CURRENT_TIME, (a.last_call_et/3600) "Hrs
connected" from
v$session a,
v$session_wait b where a.sid=&sid and a.sid=b.sid;
```

Checking for active transactions SID
=======================================

```
select username,t.used_ublk,t.used_urec from v$transaction t,v$session s where
t.addr=s.taddr and
s.sid='&sessionid';
SQL> SQL> Enter value for sessionid: 219
old   1: select username,t.used_ublk,t.used_urec from v$transaction t,v$session s
where
t.addr=s.taddr and s.sid='&sessionid'
new   1: select username,t.used_ublk,t.used_urec from v$transaction t,v$session s
where
t.addr=s.taddr and s.sid='219';
```

Checking rollback/Undo segment info used by SID
==

```
column rr heading 'RB Segment' format a18
column us heading 'Username' format a15
column os heading 'OS User' format a10
column te heading 'Terminal' format a10
SELECT r.name rr, nvl(s.username,'no transaction') us,s.sid, s.osuser os,
s.terminal te, rs.rssize,
rs.xacts, rs.rssize/1048576 Rssize
FROM v$lock  l, v$session  s,v$rollname  r , v$rollstat rs
WHERE l.sid = s.sid(+) AND trunc(l.id1/65536) = r.usn AND l.type = 'TX' AND
l.lmode = 6   AND r.usn=rs.usn  and s.sid in (&sid_list_comma_sep);
```

Checking what is the Last SQL
===============================

```
undefine sid
col "Last SQL" for a70
select s.username, s.sid, t.sql_text "Last SQL"
from gv$session s, gv$sqlarea t
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
s.sid = '&sid';
```

Killing inactive sessions for more than 48hrs
===

```
set heading off
set feedback off
spool /PENVI/applcsf/prevent/scripts/kill_session.sql
SELECT 'ALTER SYSTEM KILL SESSION '||''''||sid ||','|| serial#||''''||' immediate;'
FROM v$session
where last_call_et > 43200 and username is not null AND STATUS='INACTIVE';
spool off
exit
```

Session details complete (Input sid)
=====================================

```
set echo off
set linesize 132
set verify off
set feedback off
set serveroutput on;
declare
SID number := 0 ;
inst_id number := 0 ;
SERIAL number := 0 ;
username varchar(20) := '';
Status varchar(8) := '';
machine varchar(10) := '';
terminal varchar(25) := '';
program varchar(30) := '';
Module varchar(30) := '';
Action varchar(20) := '';
sql_hash_value number := 0 ;
logontime varchar(30) := '';
last_call_et number := 0 ;
proc number := 0 ;
spid number := 0 ;
event varchar(30) := '';
state varchar(30) := '';
sql_texts varchar(1000) := '';
undo_size varchar (100) := 'N/A';
cursor cur1 is
select a.inst_id,a.sid sid,
a.serial# serial,
a.username username,
a.status status ,
a.machine machine,
a.terminal terminal,
a.program program,
a.module module,
a.action action,
a.sql_hash_value sql_hash_value,
to_char(a.logon_time,'DD-Mon-YYYY HH24:MI:SS') logontime,
round((a.last_call_et/60),2) last_call_et,
a.process proc,
b.spid spid,
event event,
state state
from gv$session a, gv$process b, gv$session_wait sw
where a.paddr=b.addr and a.inst_id=b.inst_id
and a.sid in (75)
and a.inst_id=sw.inst_id
and a.sid=sw.sid;
begin
for m in cur1
loop
DBMS_OUTPUT.ENABLE(25000);
DBMS_OUTPUT.PUT_LINE(' ');
DBMS_OUTPUT.PUT_LINE('INSTANCE ID...................... : ' || m.inst_id );
DBMS_OUTPUT.PUT_LINE('SID.............................. : ' || m.sid );
DBMS_OUTPUT.PUT_LINE('SERIAL#.......................... : ' || m.serial );
DBMS_OUTPUT.PUT_LINE('USERNAME......................... : ' || m.username );
DBMS_OUTPUT.PUT_LINE('STATUS........................... : ' || m.status );
DBMS_OUTPUT.PUT_LINE('Client Machine...................: ' || m.machine );
DBMS_OUTPUT.PUT_LINE('Terminal......................... : ' || m.terminal);
DBMS_OUTPUT.PUT_LINE('Program.......................... : ' || m.program );
DBMS_OUTPUT.PUT_LINE('Module........................... : ' || m.module );
DBMS_OUTPUT.PUT_LINE('Action........................... : ' || m.action );
DBMS_OUTPUT.PUT_LINE('SQL Hash Value................... : ' || m.sql_hash_value );
DBMS_OUTPUT.PUT_LINE('Logon Time....................... : ' || m.logontime );
DBMS_OUTPUT.PUT_LINE('Last Call Et..................... : ' || m.last_call_et||'
'||'min' );
DBMS_OUTPUT.PUT_LINE('ClientPID........................ : ' || m.proc );
DBMS_OUTPUT.PUT_LINE('ServerPID........................ : ' || m.spid );
DBMS_OUTPUT.PUT_LINE('Session Waiting for Event.........: ' || m.event );
DBMS_OUTPUT.PUT_LINE('Session state ....................: ' || m.state);
for rec_undo in (select nvl(t.used_ublk,0)||' '||'Blocks' undo_size from v$session
s,v$transaction t where
s.taddr=t.addr(+) and
s.sid=m.sid )
loop
dbms_output.put_line('Undo Generation for sid is.........: ' ||rec_undo.undo_size);
end loop;
dbms_output.put_line('SQL_TEXT is..........:');
for rec in ( select sql_text sql_texts from v$session s,v$sqltext v where
s.sql_hash_value=v.hash_value and
s.sql_address=v.address and s.sid=m.sid order by piece)
loop
dbms_output.put_line(' '||rec.sql_texts);
end loop;
for n in ( select t.DISK_READS DISK_READS from gv$session s, gv$sqlarea t
where s.sql_hash_value =t.hash_value and s.sid=m.sid)
loop
dbms_output.put_line('Disk reads due to above SQL execution ' || n.DISK_READS);
end loop;
DBMS_OUTPUT.PUT_LINE(' ' );
DBMS_OUTPUT.PUT_LINE(':------------------------------------------------: ' );
DBMS_OUTPUT.PUT_LINE(' ' );


end loop;
end;
```

Session details complete (Input SPID)
==

```
set echo off
set linesize 132
set verify off
set feedback off
set serveroutput on;
declare
inst_id number := 0 ;
SID number := 0 ;
SERIAL number := 0 ;
username varchar(20) := '';
Status varchar(8) := '';
machine varchar(10) := '';
terminal varchar(25) := '';
program varchar(30) := '';
Module varchar(30) := '';
Action varchar(20) := '';
sql_hash_value number := 0 ;
logontime varchar(30) := '';
last_call_et number := 0 ;
proc number := 0 ;
spid number := 0 ;
event varchar(30) := '';
state varchar(30) := '';
sql_texts varchar(1000) := '';
undo_size varchar (100) := 'N/A';
cursor cur1 is
select a.inst_id, a.sid sid,
a.serial# serial,
a.username username,
a.status status ,
a.machine machine,
a.terminal terminal,
a.program program,
a.module module,
a.action action,
a.sql_hash_value sql_hash_value,
to_char(a.logon_time,'DD-Mon-YYYY HH24:MI:SS') logontime,
round((a.last_call_et/60),2) last_call_et,
a.process proc,
b.spid spid,
event event,
state state
from gv$session a, gv$process b, gv$session_wait sw
where a.paddr=b.addr and a.inst_id=b.inst_id
and b.spid in ( '&spid')
and a.inst_id=sw.inst_id
and a.sid=sw.sid;
begin
for m in cur1
loop
DBMS_OUTPUT.ENABLE(25000);
DBMS_OUTPUT.PUT_LINE(' ');
DBMS_OUTPUT.PUT_LINE('INSTANCE ID...................... : ' || m.inst_id );
DBMS_OUTPUT.PUT_LINE('SID.............................. : ' || m.sid );
DBMS_OUTPUT.PUT_LINE('SERIAL#.......................... : ' || m.serial );
DBMS_OUTPUT.PUT_LINE('USERNAME......................... : ' || m.username );
DBMS_OUTPUT.PUT_LINE('STATUS........................... : ' || m.status );
DBMS_OUTPUT.PUT_LINE('Client Machine....................: ' || m.machine );
DBMS_OUTPUT.PUT_LINE('Terminal......................... : ' || m.terminal);
DBMS_OUTPUT.PUT_LINE('Program.......................... : ' || m.program );
DBMS_OUTPUT.PUT_LINE('Module........................... : ' || m.module );
DBMS_OUTPUT.PUT_LINE('Action........................... : ' || m.action );
DBMS_OUTPUT.PUT_LINE('SQL Hash Value................... : ' || m.sql_hash_value );
DBMS_OUTPUT.PUT_LINE('Logon Time...................... : ' || m.logontime );
DBMS_OUTPUT.PUT_LINE('Last Call Et.................... : ' || m.last_call_et||'
'||'min' );
DBMS_OUTPUT.PUT_LINE('ClientPID....................... : ' || m.proc );
DBMS_OUTPUT.PUT_LINE('ServerPID....................... : ' || m.spid );
DBMS_OUTPUT.PUT_LINE('Session Waiting for Event..........: ' ||m.event );
DBMS_OUTPUT.PUT_LINE('Session state .....................: ' ||m.state);
for rec_undo in (select nvl(t.used_ublk,0)||' '||'Blocks' undo_size from v$session
s,v$transaction t where
s.taddr=t.addr(+) and
s.sid=m.sid )
loop
dbms_output.put_line('Undo Generation for sid is.........: ' ||rec_undo.undo_size);
end loop;
dbms_output.put_line('SQL_TEXT is..........:');
for rec in ( select sql_text sql_texts from v$session s,v$sqltext v where
s.sql_hash_value=v.hash_value and
s.sql_address=v.address and s.sid=m.sid order by piece)
loop
dbms_output.put_line(' '||rec.sql_texts);
end loop;
for n in ( select t.DISK_READS DISK_READS from gv$session s, gv$sqlarea t
where s.sql_hash_value =t.hash_value and s.sid=m.sid)
loop
dbms_output.put_line('Disk reads due to above SQL execution ' || n.DISK_READS);
end loop;
DBMS_OUTPUT.PUT_LINE(' ' );
DBMS_OUTPUT.PUT_LINE(':-------------------------------------------------: ' );
DBMS_OUTPUT.PUT_LINE(' ' );
end loop;end;
```

Count of JDBC thin client sessions grouped by status
===

```
col program for a15F
col machine for a15
col terminal for a15
set lines 152
select count(s.sid),s.status from gv$session s, gv$process p where p.addr=s.paddr
and s.program || s.module like ('%JDBC Thin Client%') group by status;
```

JDBC Session count
========================

```
select count(s.sid) from gv$session s where s.program || s.module like ('%JDBC Thin
Client%');
```

Inactive sessions count
========================

```
SELECT username,count(*) num_inv_sess
FROM v$session
where last_call_et > 1800
and username is not null
and module like '%JDBC Thin Client%'
AND STATUS='INACTIVE'
```

```
group by username
order by num_inv_sess DESC;


SELECT username,count(*) num_inv_sess
FROM v$session
where last_call_et > 43200
and username is not null
and module like '%JDBC Thin Client%'
AND STATUS='INACTIVE'
group by username
order by num_inv_sess DESC;
```

JDBC Sessions count
====================

```
SELECT username,count(*) sess
FROM v$session
where username is not null
and module like '%JDBC Thin Client%'
group by username
order by sess DESC;
```

Machine wise count
=======================

```
select MACHINE, PROCESS,COUNT(*)
from V$SESSION
where program like '%JDBC%'
and username = 'APPS'
and process is not null
group by MACHINE, PROCESS
order by MACHINE ;
```

Inactive sessions count
==========================

```
SELECT count(*),module FROM v$session where last_call_et > 43200 and username is
not null AND
STATUS='INACTIVE' group by module;


SELECT username,status,count(*)sesion FROM v$session where username is not null and
module
like '%JDBC Thin Client%' group by username ,status;


select count(status) Count, status, machine, program from v$session where program
like '%JDBC%'
group by status, machine,program having status = 'INACTIVE' order by 1;
```

ACTIVE / INACTIVE Sessions
===============================

```
set linesize 132
set pagesize 100
col machine format a15
col OSuser format a12
col program format a30
SQL> select count(*) from v$session;


SQL> select count(*) from v$session where status='INACTIVE';


SQL> select count(*) from v$session where status='ACTIVE';


SQL> select machine, osuser, program, count(*) from v$session
group by machine, osuser, program order by 4 desc;


SQL> select count(status) Count, status, machine, program from v$session
where program like '%JDBC%' group by status, machine, program;


SQL> select count(status) Count, status, machine, module from v$session
where program = 'JDBC Thin Client' group by status, machine, module;
```

Logon time of JDBC
=======================

```
SQL> SELECT serial#, substr(program,1,20) program, status,
to_char(logon_time,'DD-MON-YY HH24:SS') Login_Time,
to_char(sysdate-last_call_et/86400,'DD-MON-YY HH24:SS') Last_Activity FROM
v$session
WHERE program like 'JDBC%' order by 4;
```

Session distribution
========================

```
select to_char(sysdate,'DD/MM HH24:MI') "DATE",inst_id,count(inst_id)
total_ses,sum(decode(status,'INACTIVE',1,0) ) inactive_ses from gv$session group by
inst_id;
```

Program grouped by count of user connection
==

```
select unique s.program,s.osuser ,count(1) from v$session s, v$process p where
s.username
is not null and s.paddr = p.addr and s.status='INACTIVE' group by
s.program,s.osuser;
```

TOTAL Sessions/Inactive Sessions
=================================

```
select to_char(sysdate,'DD/MM HH24:MI') "DATE", inst_id, count(inst_id) total_ses,
sum(decode(status,'INACTIVE',1,0) )inactive_ses from gv$session group by inst_id
```

Thru Os user
==================

```
col program for a15F
col machine for a15
col terminal for a15
set lines 152


select s.sid,
s.serial#,
'*'||s.process||'*'  Client,
p.spid Server,
s.sql_address,
s.sql_hash_value,
s.username,
s.action,
s.program || s.module,
s.terminal,
s.machine,
s.status,
--s.last_call_et
s.last_call_et/3600
from gv$session s, gv$process p
where p.addr=s.paddr
and s.osuser='&osuser';
```

Session accessing an object
===============================

```
select b.sql_text,a.sid,a.serial#,b.users_executing,
b.rows_processed,a.last_call_et/3600 Hrs from v$sqlarea b,v$session a where
b.sql_text like '&object_name' and a.sql_address=b.address;




select b.sql_text,a.sid,a.serial#,b.users_executing,
b.rows_processed,a.last_call_et/3600 Hrs from gv$sqlarea b,gv$session a where
sql_text like '%DTEA_PA_REPORTING_AGT_HISTO%' and a.sql_address=b.address;


Select
a.session_id, b.sql_text,
count(*)
from
v$active_session_history a,v$sql b
where
a.session_state= 'ON CPU' and
a.SAMPLE_TIME > sysdate - (120/(24*60)) and a.sql_id=b.sql_id and b.sql_text like
'%WF_ITEM_ATTRIBUTE_VALUES%'
group by a.session_id,b.sql_text
order by
count(*) desc;
```

Listing out details of program thru SQLID
==

```
Select
a.session_id, b.sql_text,a.program,a.module,a.action
from
v$active_session_history a,v$sql b
where
a.sql_id=b.sql_id and b.sql_id like '%fk9qzystpcazs%';
```

How to find apps user when you know the o/s pid in 11i for Forms users (f60webmx 100% CPU)
===

You have to pass the UNIX process id to this script

```
 column "User Name" format a20
 column "ClPID" format a8
 select
 d.user_name "User Name",
 b.sid SID,b.serial# "Serial#", c.spid "srvPID", a.SPID "ClPID",
 to_char(START_TIME,'DD-MON-YY HH:MM:SS') "STime"
 from
 fnd_logins a, v$session b, v$process c, fnd_user d
 where
 b.paddr = c.addr
 and a.pid=c.pid
 and a.spid = b.process
 and d.user_id = a.user_id
 and (d.user_name = 'USER_NAME' OR 1=1)
 and a.SPID = &PID;
```

Details of the Session information
==

```
select ' Sid, Serial#, Aud sid : '|| s.sid||' , '||s.serial#||' , '||
s.audsid||chr(10)|| '     DB User / OS User : '||s.username||
'   /   '||s.osuser||chr(10)|| '    Machine - Terminal : '||
s.machine||'  -  '|| s.terminal||chr(10)|| s.module||'-'||s.action ||
'        OS Process Ids : '||
s.process||' (Client)  '||p.spid||' - '||p.pid||' (Server)'|| chr(10)||
'   Client Program Name : '||s.program "Session Info"
from v$process p,v$session s
where p.addr = s.paddr
and s.sid = '&sid';
```

To identify the Largest Archivelog generators
==

```
select s.sid, username,t.used_ublk,t.used_urec from v$transaction t,v$session s
```

```
where
t.addr=s.taddr order by 3;
```

Checking rollback/Undo segment info used by SID (single - multiple)
==

```
column rr heading 'RB Segment' format a18
column us heading 'Username' format a15
column os heading 'OS User' format a10
column te heading 'Terminal' format a10
SELECT r.name rr, nvl(s.username,'no transaction') us,s.sid, s.osuser os,
s.terminal te, rs.rssize,
rs.xacts, rs.rssize/1048576 Rssize
FROM v$lock  l, v$session  s,v$rollname  r , v$rollstat rs
WHERE l.sid = s.sid(+) AND trunc(l.id1/65536) = r.usn AND l.type = 'TX' AND
l.lmode = 6   AND r.usn=rs.usn  and s.sid in (&sid_list_comma_sep);
```

To find Sqltext based on the SID
===================================

```
select sql_text from v$sqltext a,v$session b where a.address=b.sql_address and
b.sid=&sidnumber order by piece;


select b.sid, b.serial#,b.username,a.sql_text, a.address, a.hash_value from
v$sqltext a,v$session b where a.address=b.sql_address and b.sid='&SID';
```

############SQL Script to pull out complete Session details information on a single page
==

Procedure to use

```
                1) Choose a handy location to save this file in Unix server
                2) Open a new vi file
                     vi sid.sql
                3) Copy the whole content below and save in the sid.sql file
                4) Go to sqlplus as sysdba from the same location where the file is
saved
                5) Call this file
                   @sid.sql
                6) Input required information (Either sid or serverprocess id or
clientprocess id). This will give complete session details.
```

###Actual Script#######

Sid.sql

```
col sid format 9999
col username format a10
col osuser format a10
col program format a25
col process format 9999999
col spid format 999999
col logon_time format a13


set lines 150


set heading off
set verify off
set feedback off


undefine sid_number
undefine spid_number
rem accept sid_number number prompt "pl_enter_sid:"


col sid NEW_VALUE sid_number noprint
col spid NEW_VALUE spid_number noprint




         select  s.sid   sid,
                p.spid  spid
--              ,decode(count(*), 1,'null','No Session Found with this info') " "
         FROM v$session s,
              v$process p
         WHERE s.sid LIKE NVL('&sid', '%')
         AND p.spid LIKE NVL ('&OS_ProcessID', '%')
         AND s.process LIKE NVL('&Client_Process', '%')
         AND s.paddr = p.addr
--       group by s.sid, p.spid;


PROMPT Session and Process Information
PROMPT -------------------------------


col event for a30


select '    SID                         : '||v.sid      || chr(10)||
       '    Serial Number               : '||v.serial#  || chr(10) ||
       '    Oracle User Name            : '||v.username         || chr(10) ||
       '    Client OS user name         : '||v.osuser   || chr(10) ||
       '    Client Process ID           : '||v.process  || chr(10) ||
       '    Client machine Name         : '||v.machine  || chr(10) ||
       '    Oracle PID                  : '||p.pid      || chr(10) ||
       '    OS Process ID(spid)         : '||p.spid     || chr(10) ||
       '    Session''s Status           : '||v.status   || chr(10) ||
       '    Logon Time                  : '||to_char(v.logon_time, 'MM/DD
HH24:MIpm')   || chr(10) ||
       '    Program Name                : '||v.program  || chr(10)
from v$session v, v$process p
where v.paddr = p.addr
and v.serial# > 1
and p.background is null
and p.username is not null
and sid = &sid_number
order by logon_time, v.status, 1
/




PROMPT Sql Statement
PROMPT --------------


select sql_text
from v$sqltext , v$session
where v$sqltext.address = v$session.sql_address
and sid = &sid_number
order by piece
/


PROMPT
PROMPT Event Wait Information
PROMPT ----------------------


select '   SID '|| &sid_number ||' is waiting on event  : ' || x.event || chr(10)
||
       '   P1 Text                      : ' || x.p1text || chr(10) ||
       '   P1 Value                     : ' || x.p1 || chr(10) ||
       '   P2 Text                      : ' || x.p2text || chr(10) ||
       '   P2 Value                     : ' || x.p2 || chr(10) ||
       '   P3 Text                      : ' || x.p3text || chr(10) ||
       '   P3 Value                     : ' || x.p3
from v$session_wait x
where x.sid= &sid_number
/


PROMPT
PROMPT Session Statistics
PROMPT ------------------


select        '     '|| b.name  ||'            : '||decode(b.name, 'redo size',
round(a.value/1024/1024,2)||' M', a.value)
from v$session s, v$sesstat a, v$statname b
where a.statistic# = b.statistic#
and name in ('redo size', 'parse count (total)', 'parse count (hard)', 'user
commits')
and s.sid = &sid_number
and a.sid = &sid_number
--order by b.name
order by decode(b.name, 'redo size', 1, 2), b.name
/


COLUMN USERNAME FORMAT a10
COLUMN status FORMAT a8
column RBS_NAME format a10
```

```
PROMPT
PROMPT Transaction and Rollback Information
PROMPT ------------------------------------


select        '    Rollback Used                : '||t.used_ublk*8192/1024/1024 ||'
M'          || chr(10) ||
              '    Rollback Records             : '||t.used_urec        ||
chr(10)||
              '    Rollback Segment Number      : '||t.xidusn           ||
chr(10)||
              '    Rollback Segment Name        : '||r.name             ||
chr(10)||
              '    Logical IOs                  : '||t.log_io           ||
chr(10)||
              '    Physical IOs                 : '||t.phy_io           ||
chr(10)||
              '    RBS Startng Extent ID        : '||t.start_uext       ||
chr(10)||
              '    Transaction Start Time       : '||t.start_time       ||
chr(10)||
              '    Transaction_Status           : '||t.status
FROM v$transaction t, v$session s, v$rollname r
WHERE t.addr = s.taddr
and r.usn = t.xidusn
and s.sid = &sid_number
/


PROMPT
PROMPT Sort Information
PROMPT ----------------


column username format a20
column user format a20
column tablespace format a20


SELECT        '    Sort Space Used(8k block size is asssumed    : '||
u.blocks/1024*8 ||' M'             || chr(10) ||
              '    Sorting Tablespace                           : '||u.tablespace
|| chr(10)||
              '    Sort Tablespace Type                 : '||u.contents ||
chr(10)||
              '    Total Extents Used for Sorting               : '||u.extents
FROM v$session s, v$sort_usage u
WHERE s.saddr = u.session_addr
AND s.sid = &sid_number
/



set heading on
set verify on
clear column
```

```
###############################Sql queries to check ACTIVE / INACTIVE Sessions
================================================================================



Total Count of sessions


select count(s.status) TOTAL_SESSIONS
from gv$session s;


Total Count of Inactive sessions


select count(s.status) INACTIVE_SESSIONS
from gv$session s, v$process p
where
p.addr=s.paddr and
s.status='INACTIVE';


SESSIONS WHICH ARE IN INACTIVE STATUS FROM MORE THAN 1HOUR
select count(s.status) "INACTIVE SESSIONS > 1HOUR "
from gv$session s, v$process p
where
p.addr=s.paddr and
s.last_call_et > 3600 and
s.status='INACTIVE';


COUNT OF ACTIVE SESSIONS


select count(s.status) ACTIVE_SESSIONS
from gv$session s, v$process p
where
p.addr=s.paddr and
s.status='ACTIVE';


TOTAL SESSIONS COUNT ORDERED BY PROGRAM


col program for a30
select s.program,count(s.program) Total_Sessions
from gv$session s, v$process p
where  p.addr=s.paddr
group by s.program;


TOTAL COUNT OF SESSIONS ORDERED BY MODULE


col module  for a30
prompt TOTAL SESSIONS
select s.module,count(s.sid) Total_Sessions
from gv$session s, v$process p
```

```
where  p.addr=s.paddr
group by s.module;
```

TOTAL COUNT OF SESSIONS ORDERED BY ACTION

```
col action for a30
prompt TOTAL SESSIONS
select s.action,count(s.sid) Total_Sessions
from gv$session s, v$process p
where  p.addr=s.paddr
group by s.action;
```

INACTIVE SESSIONS

```
prompt INACTIVE SESSIONS
select p.spid, s.sid,s.last_call_et/3600 last_call_et
,s.status,s.action,s.module,s.program
from gv$session s, v$process p
where
p.addr=s.paddr and
s.status='INACTIVE';
```

INACTIVE

```
prompt INACTIVE SESSIONS
select count(s.status) INACTIVE
from gv$session s, gv$sqlarea t,v$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
s.status='INACTIVE';
```

INACTIVE PROGRAMS

```
col module for a40
prompt INACTIVE SESSIONS
col INACTIVE_PROGRAMS FOR A40
select distinct (s.program) INACTIVE_PROGRAMS,s.module
from gv$session s, v$process p
where  p.addr=s.paddr and
s.status='INACTIVE';
```

INACTIVE PROGRAMS with disk reads

```
prompt INACTIVE SESSIONS
select distinct (s.program) INACTIVE_PROGRAMS,SUM(T.DISK_READS)
from gv$session s, gv$sqlarea t,v$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
```

```
p.addr=s.paddr and
s.status='INACTIVE'
GROUP BY S.PROGRAM;
```

INACTIVE SESSIONS COUNT WITH PROGRAM

```
col program for a30
prompt TOTAL INACTIVE SESSIONS
col INACTIVE_PROGRAMS FOR A40
select s.program,count(s.program) Total_Inactive_Sessions
from gv$session s,v$process p
where     p.addr=s.paddr  AND
s.status='INACTIVE'
group by s.program
order by 2 desc;
```

TOTAL INACTIVE SESSIONS MORE THAN 1HOUR

```
col program for a30
col INACTIVE_PROGRAMS FOR A40
select s.program,count(s.program) Inactive_Sessions_from_1Hour
from gv$session s,v$process p
where     p.addr=s.paddr  AND
s.status='INACTIVE'
and s.last_call_et > (3600)
group by s.program
order by 2 desc;
```

TOTAL INACTIVE SESSIONS GROUP BY MODULE

```
col program for a60
COL MODULE FOR A30
prompt TOTAL SESSIONS
col INACTIVE_PROGRAMS FOR A40
select s.module,count(s.module) Total_Inactive_Sessions
from gv$session s,v$process p
where     p.addr=s.paddr  AND
s.status='INACTIVE'
group by s.module;
```

INACTIVE SESSION DETAILS MORE THAN 1 HOUR

```
set pagesize 40
col INST_ID for 99
col spid for a10
set linesize 150
col PROGRAM for a10
col action format a10
col logon_time format a16
col module format a13
col cli_process format a7
col cli_mach for a15
col status format a10
col username format a10
col last_call_et_Hrs for 9999.99
col sql_hash_value for 9999999999999col username for a10
set linesize 152
set pagesize 80
col "Last SQL" for a60
col elapsed_time for 999999999999
select p.spid, s.sid,s.last_call_et/3600 last_call_et_Hrs
,s.status,s.action,s.module,s.program,t.disk_reads,lpad(t.sql_text,30) "Last SQL"
from gv$session s, gv$sqlarea t,gv$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
s.status='INACTIVE'
and s.last_call_et > (3600)
order by last_call_et;
```

INACTIVE PROGRAM --ANY--

```
select p.spid, s.sid,s.last_call_et/3600 last_call_et_Hrs
,s.status,s.action,s.module,s.program,t.disk_reads,lpad(t.sql_text,30) "Last SQL"
from gv$session s, gv$sqlarea t,gv$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
s.status='INACTIVE'
And s.program='&PROGRAM_NAME'
order by last_call_et;
```

INACTIVE MODULES --ANY--

```
select p.spid, s.sid,s.last_call_et/3600 last_call_et_Hrs
,s.status,s.action,s.module,s.program,t.disk_reads,lpad(t.sql_text,30) "Last SQL"
from gv$session s, gv$sqlarea t,gv$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr
And s.module like '%order_cleanup_hazmat_v3.sql'
order by last_call_et;
```

INACTIVE JDBC SESSIONS

```
set pagesize 40
col INST_ID for 99
col spid for a10
set linesize 150
col PROGRAM for a10
col action format a10
col logon_time format a16
col module format a13
col cli_process format a7
col cli_mach for a15
col status format a10
col username format a10
col last_call_et for 9999.99
col sql_hash_value for 9999999999999col username for a10
set linesize 152
set pagesize 80
col "Last SQL" for a60
col elapsed_time for 999999999999
select p.spid, s.sid,s.last_call_et/3600 last_call_et ,s.status,s.action,
s.module,s.program,t.disk_reads,lpad(t.sql_text,30) "Last SQL"
from gv$session s, gv$sqlarea t,gv$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
s.status='INACTIVE'
and s.program='JDBC Thin Client'
and s.last_call_et > 3600
order by last_call_et;
```

COUNT OF INACTIVE SESSIONS MORE THAN ONE HOUR

```
SELECT COUNT(P.SPID)
from gv$session s, gv$sqlarea t,gv$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
s.status='INACTIVE'
and s.program='JDBC Thin Client'
and s.last_call_et > 3600
order by last_call_et;
```

FORMS
TOTAL FORM SESSIONS

```
SELECT COUNT(S.SID) INACTIVE_FORM_SESSIONS FROM V$SESSION S
WHERE S.STATUS='INACTIVE' and
s.action like ('%FRM%');
```

FORMS SESSIONS DETAILS

```
col "Last SQL" for a30
select p.spid,s.sid,s.status,s.last_call_et/3600 last_call_et_hrs ,
s.sid,t.disk_reads, t.elapsed_time,lpad(t.sql_text,30) "Last SQL"
from gv$session s, gv$sqlarea t,gv$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
s.action like ('FRM%') and
s.last_call_et > 3600
order by spid;
```

```
col machine for a15
col "Last SQL" for a30
```

```
select p.spid,s.sid,s.status,s.last_call_et/3600 last_call_et_hrs ,
S.ACTION,s.process Client_Process,s.machine
from gv$session s, gv$sqlarea t,gv$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
s.action like ('FRM%') and
s.last_call_et > 3600;
order by 4;
```

INACTIVE FORMS SESSIONS DETAILS

```
col program for a15
col last_call_et for 999.99
select p.spid, s.sid, s.process,s.last_call_et/3600 last_call_et
,s.status,s.action,s.module,s.program,t.disk_reads,lpad(t.sql_text,30) "Last SQL"
from gv$session s, gv$sqlarea t,gv$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
s.status='INACTIVE'
and s.action like 'FRM:%'
and s.last_call_et > 3600
order by last_call_et desc;
```

UNIQUE SPID

```
select unique(p.spid)
from gv$session s, gv$sqlarea t,gv$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
s.status='INACTIVE'
and s.action like 'FRM:%'
and s.last_call_et > 3600;
```

COUNT FORMS

```
select COUNT(p.spid)
from gv$session s, gv$sqlarea t,gv$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
s.status='INACTIVE'
and s.action like 'FRM:%'
and s.last_call_et > 3600;
```

ZERO HASH VALUE

```
select COUNT(p.spid)
from gv$session s,gv$process p
```

```
where
p.addr=s.paddr and
s.status='INACTIVE'
and s.action like 'FRM:%'
and s.last_call_et > 3600
AND S.SQL_HASH_VALUE=0;
```

INACTIVE FORM BY NAME

```
select count(s.sid) from v$session S
where s.action like ('%&ACTION%')
AND S.STATUS='INACTIVE';
```

GROUP BY ACTION

```
SELECT S.ACTION,COUNT(S.SID) FROM V$SESSION S
WHERE S.STATUS='INACTIVE' and
s.action like ('%FRM%')
group by s.action;
```

FROM A SPECIFIC USERNAME

```
SET LINSIZE 152
col spid for a10
col process_spid for a10
col user_name for a20
col form_name for a20
select a.pid,a.spid,a.process_spid, c.user_name,to_char(a.start_time,'DD-MON-YYYY
HH24:MI:SS') "START_TIME" ,
d.user_form_name "FORM_NAME"
from apps.fnd_logins a, apps.fnd_login_resp_forms b, apps.fnd_user c,
apps.fnd_form_tl d
where
a.login_id=b.login_id
and c.user_name like 'JROMO'
and a.user_id=c.user_id
and trunc(b.start_time) >trunc(sysdate -11)
and trunc(b.end_time) is null
and b.form_id=d.form_id
and d.language='US';
```

INACTIVE FORM

```
set pagesize 40
col INST_ID for 99
col spid for a10
set linesize 150
col PROGRAM for a10
col action format a10
col logon_time format a16
col module format a13
col cli_process format a7
col cli_mach for a15
col status format a10
col username format a10
col last_call_et for 9999.99
col sql_hash_value for 9999999999999col username for a10
set linesize 152
set pagesize 80
col "Last SQL" for a30
col elapsed_time for 999999999999
select p.spid, s.sid,s.process cli_process,s.last_call_et/3600 last_call_et ,
s.status,s.action,s.module,s.program,t.disk_reads,lpad(t.sql_text,30) "Last SQL"
from gv$session s, gv$sqlarea t,gv$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
s.status='INACTIVE'
and s.action like ('FRM%')
and s.last_call_et > (3600*3)
order by last_call_et;
```

INACTIVE FORM SESSIONS

```
col cli_proc for a9
COL AUDSID FOR A6
COL PID FOR A6
COL SID FOR A5
COL FORM_NAME FOR A25
COL USER_NAME FOR A15
col last_call_et for 9999.99
SELECT
-- /*+ ORDERED FULL(fl) FULL(vp) USE_HASH(fl vp) */
( SELECT SUBSTR ( fu.user_name, 1, 20 )
FROM apps.fnd_user fu
WHERE fu.user_id = fl.user_id
) user_name,vs.status,
TO_CHAR ( fl.start_time, 'DD-MON-YYYY HH24:MI' ) login_start_time,
TO_CHAR ( fl.end_time, 'DD-MON-YYYY HH24:MI' ) login_end_time,
vs.last_call_et/3600 last_call_et,
SUBSTR ( fl.process_spid, 1, 6 ) spid,
SUBSTR ( vs.process, 1, 8 ) cli_proc,
SUBSTR ( TO_CHAR ( vs.sid ), 1, 3 ) sid,
SUBSTR ( TO_CHAR ( vs.serial#), 1, 7 ) serial#,
SUBSTR ( TO_CHAR ( rf.audsid ), 1, 6 ) audsid,
SUBSTR ( TO_CHAR ( fl.pid ), 1, 3 ) pid,
SUBSTR ( vs.module || ' - ' ||
( SELECT SUBSTR ( ft.user_form_name, 1, 40 )
FROM apps.fnd_form_tl ft
WHERE ft.application_id = rf.form_appl_id
AND ft.form_id        = rf.form_id
AND ft.language       = USERENV('LANG')
), 1, 40 ) form_name
```

```
FROM apps.fnd_logins           fl,
gv$process            vp,
apps.fnd_login_resp_forms rf,
gv$session            vs
WHERE fl.start_time   > sysdate - 7 /* login within last 7 days */
AND fl.login_type   = 'FORM'
AND fl.process_spid = vp.spid
AND fl.pid          = vp.pid
AND fl.login_id     = rf.login_id
AND rf.end_time    IS NULL
AND rf.audsid       = vs.audsid
and vs.status='INACTIVE'
ORDER BY
vs.process,
fl.process_spid;
```

ACTIVE

```
prompt ACTIVE SESSIONS
select count(s.status) ACTIVE
from gv$session s, gv$sqlarea t,v$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr and
s.status='ACTIVE';
```

MODULE

```
set pagesize 40
col INST_ID for 99
col spid for a10
set linesize 150
col PROGRAM for a10
col action format a10
col logon_time format a16
col module format a13
col cli_process format a7
col cli_mach for a15
col status format a10
col username format a10
col last_call_et for 9999.99
col sql_hash_value for 9999999999999col username for a10
set linesize 152
set pagesize 80
col "Last SQL" for a30
col elapsed_time for 999999999999
select p.spid, s.sid,s.process cli_process,s.last_call_et/3600 last_call_et ,
s.status,s.action,s.module,s.program,t.disk_reads,lpad(t.sql_text,30) "Last SQL"
from gv$session s, gv$sqlarea t,gv$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr
and s.MODULE like ('&MODULE_NAME_1HR%')
and s.last_call_et > ('&TIME_HRS' * 3600)
order by last_call_et;
```

```
select p.spid, s.sid,s.process cli_process,s.last_call_et/3600 last_call_et ,
s.status,s.action,s.module,s.program
from gv$session s, gv$sqlarea t,gv$process p
where s.sql_address =t.address and
p.addr=s.paddr
and s.MODULE like ('%TOAD%')
Order by last_call_et;
```

TOAD SESSIONS

```
select p.spid, s.sid,s.process cli_process,s.last_call_et/3600 last_call_et ,
s.status,s.action,s.module,s.program
from gv$session s, gv$process p
where
p.addr=s.paddr
and s.MODULE like ('%TOAD%')
Order by last_call_et;
```

CLIENT MACHINE SESSIONS COUNT

```
select count(s.process) TOTAL from v$session S
where s.machine like ('%&CLIENT_MACHINE%');
```

```
select count(s.process) INACTIVE from v$session S
where s.machine like ('%&CLIENT_MACHINE%')
and s.status='INACTIVE';
```

hash value=0

```
select count(s.process) from v$session S
where s.machine like ('%&CLIENT_MACHINE%')
AND S.SQL_HASH_VALUE=0;
```

```
select count(s.process) from v$session S
where s.machine like ('%&CLIENT_MACHINE%')
AND S.SQL_HASH_VALUE=0
AND S.LAST_CALL_ET > 3600;
```

Unique Actions

```
col module for a40
prompt INACTIVE SESSIONS
col INACTIVE_PROGRAMS FOR A40
select distinct (s.program) INACTIVE_PROGRAMS,s.module
from gv$session s, gv$sqlarea t,v$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
```

```
s.machine like ('%&CLIENT_MACHINE%') AND
p.addr=s.paddr and
s.status='INACTIVE';


GROUP BY  program


col program for a60
prompt TOTAL SESSIONS
col INACTIVE_PROGRAMS FOR A40
select s.program,count(s.program) Total_Inactive_Sessions
from gv$session s, gv$sqlarea t,v$process p
where s.sql_address =t.address and
s.sql_hash_value =t.hash_value and
p.addr=s.paddr  AND
s.machine like ('%&CLIENT_MACHINE%') AND
s.status='INACTIVE'
group by s.program;
```

To check the sessions with most Active I/O
==

```
SELECT DISTINCT wait_class
FROM gv$event_name
ORDER BY 1;


SELECT sql_id, COUNT(*)
FROM gv$active_session_history ash, gv$event_name evt
WHERE ash.sample_time > SYSDATE - 1/24
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;
```

To get the Most Active SQL in the previous hour
===

```
SELECT sql_id,COUNT(*),ROUND(COUNT(*)/SUM(COUNT(*)) OVER(), 2) PCTLOAD
FROM gv$active_session_history
WHERE sample_time > SYSDATE - 1/24
AND session_type = 'BACKGROUND'
GROUP BY sql_id
ORDER BY COUNT(*) DESC;


SELECT sql_id,COUNT(*),ROUND(COUNT(*)/SUM(COUNT(*)) OVER(), 2) PCTLOAD
FROM gv$active_session_history
WHERE sample_time > SYSDATE - 1/24
AND session_type = 'FOREGROUND'
GROUP BY sql_id
ORDER BY COUNT(*) DESC;
```

To check the Active Waiting sessions

```
Select
sysdate sample_time,
s.sid               "SESSION_ID"    ,
decode(w.WAIT_TIME, 0,'WAITING','ON CPU') "SESSION_STATE",
s.serial#         "SESSION_SERIAL#",
s.user#           "USER_ID",
s.sql_address     "SQL_ADDRESS",
s.sql_hash_value      "SQL_HASH" ,
s.command         "SQL_OPCODE"  ,
s.type            "SESSION_TYPE"  ,
w.event           "EVENT",
w.p1              "P1"          ,
w.p2              "P2"        ,
w.p3              "P3"         ,
s.ROW_WAIT_OBJ#   "CURRENT_OBJ#",
s.ROW_WAIT_FILE#  "CURRENT_FILE#",
s.ROW_WAIT_BLOCK# "CURRENT_BLOCK#",
s.program         "PROGRAM",
s.module     "MODULE",
s.action    "ACTION"
from
v$session s ,
v$session_wait w
where
s.sid != ( select distinct sid from v$mystat  where rownum < 2 ) and
w.sid = s.sid and
(  (
/* status Active - seems inactive & "on cpu"=> not on CPU */
w.wait_time != 0  and  /* on CPU  */
s.status='ACTIVE'  /* ACTIVE */
)
or
lower(w.event)  not in   /* waiting and the wait event is not idle */
(
'queue monitor wait',
'null event',
'pl/sql lock timer',
'px deq: execution msg',
'px deq: table q normal',
'px idle wait',
'sql*net message from client',
'sql*net message from dblink',
'dispatcher timer',
'lock manager wait for remote message',
'pipe get',
'pmon timer',
'queue messages',
'rdbms ipc message',
'slave wait',
'smon timer',
'virtual circuit status',
'wakeup time manager',
'i/o slave wait',
'jobq slave wait',
'queue monitor wait',
'SQL*Net message from client'
)
);
```

################Top Session per User with CPU Status
==

```
select
/* if  sid not found in v$session then  disconnected */

select decode(nvl(to_char(s.sid),-1),-1,'DISCONNECTED','CONNECTED')
"STATUS",
topsession.session_id             "SESSION_ID",
u.name  "NAME",
topsession.program                  "PROGRAM",
max(topsession.CPU)              "CPU",
max(topsession.WAITING)       "WAITING",
max(topsession.IO)                  "IO",
max(topsession.TOTAL)          "TOTAL"
from (select
ash.session_id,
ash.session_serial#,
ash.user_id,
ash.program,
sum(decode(ash.session_state,'ON CPU',1,0))     "CPU",
sum(decode(ash.session_state,'WAITING',1,0))    -
sum(decode(ash.session_state,'WAITING',
decode(en.wait_class,'User I/O',1, 0 ), 0))    "WAITING" ,
sum(decode(ash.session_state,'WAITING',
decode(en.wait_class,'User I/O',1, 0 ), 0))    "IO" ,
sum(decode(session_state,'ON CPU',1,1))     "TOTAL"
from v$active_session_history ash,
v$event_name en
where en.event# = ash.event#
group by session_id,user_id,session_serial#,program
order by sum(decode(session_state,'ON CPU',1,1))) topsession,
v$session s,
user$ u
where
u.user# =topsession.user_id and
topsession.session_id         = s.sid         (+) and
topsession.session_serial# = s.serial#   (+)
group by  topsession.session_id, topsession.session_serial#, topsession.user_id,
topsession.program, s.username,s.sid,s.paddr,u.name
order by max(topsession.TOTAL) desc;
/* outer join to v$session because the session might be disconnected */
```

To get the Top Session Userwise
================================

```
select
/* if  sid not found in v$session then  disconnected */
```

```
decode(nvl(to_char(s.sid),-1),-1,'DISCONNECTED','CONNECTED')
"STATUS",
topsession.session_id             "SESSION_ID",
u.name  "NAME",
topsession.program                  "PROGRAM",
max(topsession.CPU)              "CPU",
max(topsession.WAITING)       "WAITING",
max(topsession.IO)                  "IO",
max(topsession.TOTAL)           "TOTAL"
from (   previous query   ) topsession,
v$session s,
user$ u
where
u.user# =topsession.user_id and
/* outer join to v$session because the session might be disconnected */
topsession.session_id         = s.sid         (+) and
topsession.session_serial# = s.serial#   (+)
group by  topsession.session_id, topsession.session_serial#, topsession.user_id,
topsession.program, s.username,s.sid,s.paddr,u.name
order by max(topsession.TOTAL) desc;
```

To check the Top Sessions
==========================

```
select
ash.session_id,
ash.session_serial#,
ash.user_id,
ash.program,
sum(decode(ash.session_state,'ON CPU',1,0))     "CPU",
sum(decode(ash.session_state,'WAITING',1,0))    -
sum(decode(ash.session_state,'WAITING',
decode(en.wait_class,'User I/O',1, 0 ), 0))    "WAITING" ,
sum(decode(ash.session_state,'WAITING',
decode(en.wait_class,'User I/O',1, 0 ), 0))    "IO" ,
sum(decode(session_state,'ON CPU',1,1))     "TOTAL"
from v$active_session_history ash,
v$event_name en
where en.event# = ash.event# and rownum<11
group by session_id,user_id,session_serial#,program
order by sum(decode(session_state,'ON CPU',1,1));
```

To get the Top SQL from ASH
===========================

```
select
ash.SQL_ID ,
sum(decode(ash.session_state,'ON CPU',1,0))     "CPU",
sum(decode(ash.session_state,'WAITING',1,0))    -
sum(decode(ash.session_state,'WAITING', decode(en.wait_class, 'User I/O',1,0),0))
"WAIT" ,
sum(decode(ash.session_state,'WAITING', decode(en.wait_class, 'User I/O',1,0),0))
"IO" ,
sum(decode(ash.session_state,'ON CPU',1,1))     "TOTAL"
from v$active_session_history ash,
v$event_name en
where SQL_ID is not NULL  and en.event#=ash.event# and rownum<11
group by sql_id
order by sum(decode(session_state,'ON CPU',1,1)) desc;
```

```
Top Waiting Session in last 5 minutes
========================================

Select
session_id,
count(*)
from
v$active_session_history
where
session_state='WAITING'  and
SAMPLE_TIME >  SYSDATE - (60/(24*60)) and rownum<11
group by
session_id
order by
count(*) desc;



To check Top CPU Session in last 5 minutes
============================================

elect
session_id,
count(*)
from
v$active_session_history
where
session_state= 'ON CPU' and
SAMPLE_TIME > sysdate - (120/(24*60)) and rownum<11
group by session_id
order by
count(*) desc;



Scripts to check Rollback Segments information
================================================

Rollback segment Information


SELECT segment_name, tablespace_name, status
        FROM sys.dba_rollback_segs;
SELECT segment_name, tablespace_name, (bytes)/1024/1024, blocks, extents
        FROM sys.dba_segments
   WHERE segment_type = 'ROLLBACK';


SELECT name, xacts "ACTIVE TRANSACTIONS" FROM v$rollname, v$rollstat WHERE status =
'PENDING OFFLINE' AND v$rollname.usn = v$rollstat.usn;


SELECT segment_name, tablespace_name, owner
       FROM sys.dba_rollback_segs;


SELECT segment_name, segment_type, tablespace_name
     FROM sys.dba_segments
```

```
WHERE segment_type = 'DEFERRED ROLLBACK';
```

Shrinking Rollback segment command

```
ALTER ROLLBACK SEGMENT rbs1 SHRINK TO 100K;
```

```
select count(*) from dba_extents where tablespace_name='RBSTS';
```

Shrinking all rollback Segments

```
spool shrink_em.sql
select 'alter rollback segment '||segment_name||' shrink to 2;' from
dba_rollback_segs where tablespace_name='RBSTS';
spool off
@shrink_em.sql
```

Number of rollback extents

```
select count(*) from dba_extents where tablespace_name='RBSTS';
```

Finding Rollback Segment Size

```
SQL> select segment_name,sum(bytes) from dba_segments where
> tablespace_name
> = 'RBS' and segment_name
> = 'RBS17' group by segment_name;
>
> SEGMENT_NAME SUM(BYTES)
> ------------------------ ----------
> RBS17 22364160
```

Finding Rollback Segment Optimal Size

```
> SQL> select rs.optsize, rs.extents
> 2 from dba_rollback_segs drs,
> 3 v$rollstat rs
> 4 where drs.segment_name = 'RBS17'
> 5 and drs.segment_id = rs.usn;
>
> OPTSIZE EXTENTS
> ---------- ----------
> 22020096 21
```

Shrinking Rollback Segment

```
> SQL> alter rollback segment RBS17 shrink to 10M;
```

```
>
> Rollback segment altered.
```

Script to shrink all rollback Segments

```
-- Script: shrink_rollback_segs.sql
-- Purpose:            to shrink all online rollback segments back to optimal
-------------------------------------------------------------------------------
@save_sqlplus_settings


set pagesize 0
set termout off


spool shrink_rollback_segs.tmp
select
  'alter rollback segment ' || segment_name || ' shrink;'
from
  sys.dba_rollback_segs
where
  status = 'ONLINE'
/
spool off


@shrink_rollback_segs.tmp


host rm -f shrink_rollback_segs.tmp                -- for Unix
host del shrink_rollback_segs.tmp   -- for others
```

Finding Current Optimal and Suggested Optimal

```
column name format a30 heading "Rollback Segment"
column optsize format 99999999999 heading "Current Optimal"
column new_opt format 99999999999 heading "Suggested Optimal"


select
  n.name,
  s.optsize,
  ( ceil(s.extents * (s.optsize + s.aveshrink)/(s.rssize + p.value))
    * (s.rssize + p.value)
    / s.extents
  ) - p.value  new_opt
from
  ( select
      optsize,
      avg(rssize)     rssize,
      avg(extents)    extents,
      max(wraps)      wraps,
      max(shrinks)    shrinks,
      avg(aveshrink)  aveshrink
    from
      sys.v_$rollstat
    where
      optsize is not null and
      status = 'ONLINE'
    group by
      optsize
  )  s,
  ( select
      kvisval  value
    from
      sys.x_$kvis
    where
      kvistag = 'kcbbkl' )  p,
  sys.v_$rollstat  r,
  sys.v_$rollname  n
where
  s.shrinks > 1 and
  s.shrinks > s.wraps / ceil(s.optsize / ((s.rssize + p.value) / s.extents)) and
  r.optsize = s.optsize and
  r.status = 'ONLINE' and
  n.usn = r.usn
/
```

Generating Shrink commands (Examples)

```
select
b.segment_name,b.tablespace_name,a.extents,a.rssize,a.xacts,a.optsize,a.shrinks,a.w
raps,a.status from v$rollstat a, dba_rollback_segs b where b.segment_id = a.usn;


select 'alter rollback segment ' || segment_name || ' shrink;' from
sys.dba_rollback_segs where status = 'ONLINE';


SQL> select 'alter rollback segment ' || segment_name || ' shrink;' from
sys.dba_rollback_segs where status = 'ONLINE';


'ALTERROLLBACKSEGMENT'||SEGMENT_NAME||'SHRINK;'
-------------------------------------------------------------
alter rollback segment SYSTEM shrink;
alter rollback segment R01 shrink;
alter rollback segment R02 shrink;
alter rollback segment R03 shrink;
alter rollback segment R04 shrink;


SQL> alter rollback segment R01 shrink;


Rollback segment altered.


SQL> alter rollback segment R02 shrink;


Rollback segment altered.


SQL> alter rollback segment R03 shrink;
```

```
Rollback segment altered.


SQL> alter rollback segment R04 shrink;


Rollback segment altered.


Enter value for tbs: RBS1
old  11:                                     dba_data_files where tablespace_name
in ('&tbs')) where
new  11:                                     dba_data_files where tablespace_name
in ('RBS1')) where
Enter value for tbs: RBS1
old  12:                                     tablespace_name in ('&tbs')
new  12:                                     tablespace_name in ('RBS1')


Used Space(MB) allocated size(MB) maximum allowable (MB) effectivefree(MB)     %
FREE
-------------- ------------------ ---------------------- -----------------
----------
           200                500                    500               300
60





SQL> SELECT segment_name, tablespace_name, (bytes)/1024/1024, blocks, extents
        FROM sys.dba_segments
   WHERE segment_type = 'ROLLBACK';  2    3


SEGMENT_NAME
TABLESPACE_NAME           (BYTES)/1024/1024     BLOCKS    EXTENTS
--------------------------------------------------------------------------------
------------------------- ----------------- ---------- ----------
SYSTEM
SYSTEM                             1.328125        170         17
R0
2        256          2
R01
RBS1                                     50       6400         10
R02
50       6400         10
R03
50       6400         10
R04
50       6400         10


6 rows selected.


SQL> SELECT segment_name, tablespace_name, status
```

```
        FROM sys.dba_rollback_segs;  2

SEGMENT_NAME                   TABLESPACE_NAME           STATUS
------------------------------ ------------------------- ----------------
SYSTEM                         SYSTEM                    ONLINE
R0                                                       OFFLINE
R01                            RBS1                      ONLINE
R02                                                      ONLINE
R03                                                      ONLINE
R04                                                      ONLINE


SQL> SELECT segment_name, tablespace_name, (bytes)/1024/1024, blocks, extents
        FROM sys.dba_segments
   WHERE segment_type = 'ROLLBACK';  2    3


SEGMENT_NAME
TABLESPACE_NAME           (BYTES)/1024/1024     BLOCKS    EXTENTS
--------------------------------------------------------------------------------
------------------------- ----------------- ---------- ----------
SYSTEM
SYSTEM                             1.328125        170         17
R0
2        256           2
R01
RBS1                                     50       6400         10
R02
345      44160         69
R03
50       6400          10
R04
50       6400          10


6 rows selected.
SQL> SELECT name, xacts "ACTIVE TRANSACTIONS" FROM v$rollname, v$rollstat WHERE
status = 'PENDING OFFLINE' AND v$rollname.usn = v$rollstat.usn;


no rows selected


SQL> SELECT segment_name, segment_type, tablespace_name
     FROM sys.dba_segments
WHERE segment_type = 'DEFERRED ROLLBACK';  2    3


no rows selected


SQL> select
b.segment_name,b.tablespace_name,a.extents,a.rssize,a.xacts,a.optsize,a.shrinks,a.w
raps,a.status from v$rollstat a, dba_rollback_segs b where b.segment_id = a.usn;


SEGMENT_NAME                   TABLESPACE_NAME              EXTENTS     RSSIZE
XACTS Current Optimal    SHRINKS      WRAPS STATUS
```

```
------------------------------ ------------------------- ---------- ----------
---------- -------------- ---------- ---------- --------------
SYSTEM                         SYSTEM                            17    1384448
0                          0           0 ONLINE
R01                            RBS1                              10   52420608
0                          0         360 ONLINE
R02                                                              69  361750528
0                          0         703 ONLINE
R03                                                              10   52420608
0                          0         710 ONLINE
R04

   10   52420608          0                           0        356 ONLINE
```

```
Segments Unable to extend/Segments reached max extents
=======================================================
SET pages 2000
SET verify OFF
SET heading off
set lines 150
undef warning
def Warning=5
spool alter_chunk.sql
 -- Alter tables and table partitions in ts with a size of > 1GB to next 1MB
 SELECT 'alter ' ||
 decode(a.segment_type, 'TABLE PARTITION', 'TABLE', a.segment_type)  || ' '
 || a.owner || '.' || a.segment_name || ' storage (pctincrease 0 next 1M);'
 FROM dba_segments a
 WHERE
   a.next_extent * (DECODE(
                   POWER(1+(a.pct_increase/100),(&&warning+1))-1,
0,(&&warning+1),POWER(1+(a.pct_increase/100),(&&warning+1))-1)/
           (DECODE(a.pct_increase,0,100,a.pct_increase)/100))
  >
  (SELECT max(bytes) FROM dba_free_space b
   WHERE a.tablespace_name = b.tablespace_name
   AND b.tablespace_name != 'SYSTEM')
 AND a.segment_type in ('TABLE PARTITION', 'TABLE')
 AND a.next_extent > 1024000
 AND a.owner not in ('SYS','SYSTEM')
 AND  1073741824 <= (
   SELECT SUM(bytes) FROM dba_data_files f
   WHERE f.tablespace_name = a.tablespace_name
   AND f.tablespace_name != 'SYSTEM')
 GROUP BY a.segment_type, a.owner, a.segment_name
 UNION ALL
 -- Alter tables and table partitions in ts with a size of <1GB to next 128KB
 SELECT 'alter ' ||
 decode(a.segment_type, 'TABLE PARTITION', 'TABLE', a.segment_type)  || ' '
 || a.owner || '.' || a.segment_name || ' storage (pctincrease 0 next
128K);'
 FROM dba_segments a
 WHERE
   a.next_extent * (DECODE(
                   POWER(1+(a.pct_increase/100),(&&warning+1))-1,
0,(&&warning+1),POWER(1+(a.pct_increase/100),(&&warning+1))-1)/
           (DECODE(a.pct_increase,0,100,a.pct_increase)/100))
  >
  (SELECT max(bytes) FROM dba_free_space b
   WHERE a.tablespace_name = b.tablespace_name
   AND b.tablespace_name != 'SYSTEM')
 AND a.segment_type in ('TABLE PARTITION', 'TABLE')
 AND a.next_extent > 128000
 AND a.owner not in ('SYS','SYSTEM')
 AND  1073741824 > (
   SELECT SUM(bytes) FROM dba_data_files f
   WHERE f.tablespace_name = a.tablespace_name
   AND f.tablespace_name != 'SYSTEM')
 GROUP BY a.segment_type, a.owner, a.segment_name
 UNION ALL
 -- Alter indexes or IOTs (partitioned or not) in ts with a size of > 1GB to next 1
MB
 SELECT 'alter ' || DECODE(i.index_type, 'IOT - TOP', 'TABLE', 'index') ||
        ' ' || DECODE(i.index_type, 'IOT - TOP', i.table_owner, a.owner) ||
'.' ||
        DECODE(i.index_type, 'IOT - TOP', i.table_name, a.segment_name) ||
        ' storage (pctincrease 0 next 1M);'
 FROM dba_segments a, dba_indexes i
 WHERE a.segment_name = i.index_name
 AND a.owner = i.owner
 AND a.next_extent * (DECODE(
                   POWER(1+(a.pct_increase/100),(&&warning+1))-1,
0,(&&warning+1),POWER(1+(a.pct_increase/100),(&&warning+1))-1)/
           (DECODE(a.pct_increase,0,100,a.pct_increase)/100))
  >
  (SELECT max(bytes) FROM dba_free_space b
   WHERE a.tablespace_name = b.tablespace_name
   AND b.tablespace_name != 'SYSTEM')
 AND a.segment_type in ('INDEX PARTITION', 'INDEX')
 AND a.next_extent > 1024000
 AND a.owner not in ('SYS','SYSTEM')
 AND  1073741824 <= (
   SELECT SUM(bytes) FROM dba_data_files f
   WHERE f.tablespace_name = a.tablespace_name
   AND f.tablespace_name != 'SYSTEM')
 GROUP BY i.index_type, i.table_owner, a.owner, i.table_name, a.segment_name
 UNION ALL
 -- Alter indexes or IOTs (partitioned or not) in ts with a size of <1GB to next
128 KB
 SELECT 'alter ' || DECODE(i.index_type, 'IOT - TOP', 'TABLE', 'index') ||
        ' ' || DECODE(i.index_type, 'IOT - TOP', i.table_owner, a.owner) ||
'.' ||
        DECODE(i.index_type, 'IOT - TOP', i.table_name, a.segment_name) ||
        ' storage (pctincrease 0 next 128K);'
 FROM dba_segments a, dba_indexes i
 WHERE a.segment_name = i.index_name
 AND a.owner = i.owner
 AND a.next_extent * (DECODE(
                   POWER(1+(a.pct_increase/100),(&&warning+1))-1,
0,(&&warning+1),POWER(1+(a.pct_increase/100),(&&warning+1))-1)/
           (DECODE(a.pct_increase,0,100,a.pct_increase)/100))
  >
  (SELECT max(bytes) FROM dba_free_space b
   WHERE a.tablespace_name = b.tablespace_name
   AND b.tablespace_name != 'SYSTEM')
 AND a.segment_type in ('INDEX PARTITION', 'INDEX')
 AND a.next_extent > 128000
 AND a.owner not in ('SYS','SYSTEM')
 AND  1073741824 > (
   SELECT SUM(bytes) FROM dba_data_files f
   WHERE f.tablespace_name = a.tablespace_name
   AND f.tablespace_name != 'SYSTEM')
 GROUP BY i.index_type, i.table_owner, a.owner, i.table_name, a.segment_name
 UNION ALL
 -- Alter Lobs segments to next 1 MB
 SELECT 'alter table ' || l.owner || '.' || l.table_name || ' modify lob ('||
l.column_name || ')  (storage (pctincrease 0 next 1M));'
 FROM dba_segments a, dba_lobs l
 WHERE
   a.segment_name = l.segment_name
 AND
   a.next_extent * (DECODE(
                   POWER(1+(a.pct_increase/100),(&&warning+1))-1,
0,(&&warning+1),POWER(1+(a.pct_increase/100),(&&warning+1))-1)/
           (DECODE(a.pct_increase,0,100,a.pct_increase)/100))
  >
  (SELECT max(bytes) FROM dba_free_space b
   WHERE a.tablespace_name = b.tablespace_name
   AND b.tablespace_name != 'SYSTEM')
 AND a.segment_type IN ('LOBSEGMENT')
 AND a.next_extent > 1024000
 AND a.owner not in ('SYS','SYSTEM');
spool off;
```

IDENTIFYING the Activities done in the Tablespace
===

Find the creation dates of datafiles for the reported tablespace

```
set pagesize 100
 select FILE#||'   '||CREATION_TIME||'    '||TS#||'    '||BYTES||'    '||NAME from
v$datafile where TS# = (select TS# from v$tablespace  where name = '&TS_name');
```

To find which users are having this tablespace as their default ts

```
Select USERNAME,ACCOUNT_STATUS,DEFAULT_TABLESPACE,TEMPORARY_TABLESPACE from
dba_users where DEFAULT_TABLESPACE='&Ts_name';
```

To find Which user using how many segments from this ts

```
select OWNER,SEGMENT_NAME,TABLESPACE_NAME,BYTES from dba_segments where
SEGMENT_NAME='&segment_name' order by bytes;
```

To check Tablespaces Approaching Max Extents

==

Approaching Max extents different formula (less than 500 extents to extend)

```
select owner, SEGMENT_NAME, segment_type, tablespace_name,
NEXT_EXTENT,PCT_INCREASE, INITIAL_EXTENT, EXTENTS,
MAX_EXTENTS  from dba_segments  where tablespace_name= 'APPLSYSD' and extents-
max_extents<500;
```

```
select count(*) from dba_segments  where tablespace_name= 'APPLSYSD' and extents-
max_extents<506;
```

Approaching Max extents different Formula 2*extents>Maxextents

```
select owner, SEGMENT_NAME, segment_type, tablespace_name,
NEXT_EXTENT,PCT_INCREASE, INITIAL_EXTENT, EXTENTS,MAX_EXTENTS  from dba_segments
where tablespace_name='&tbs' and  2*extents>max_extents;
```

Less than 25 extents able to extend

```
select owner, segment_name, segment_type, initial_extent, next_extent, extents,
max_extents, pct_increase from dba_segments where tablespace_name='&TS_NAME' and
max_extents-extents<25;
```

Checking total space available within TS (Effective MB)

```
select tablespace_name, sum(bytes/1024/1024) "mb" from dba_free_space
group by tablespace_name having tablespace_name=upper('&tname');
```

To get the Segment Information
==============================

```
select owner, SEGMENT_NAME, segment_type, tablespace_name,
NEXT_EXTENT,PCT_INCREASE, INITIAL_EXTENT, EXTENTS,
MAX_EXTENTS  from dba_segments  where tablespace_name= 'APPLSYSD' ;
```

To check if tablespace is dictionary managed or locally managed
==

```
select tablespace_name, extent_management from dba_tablespaces
where tablespace_name = <Tablespace_name> order by 1
```

To check the Segments Approaching Max extent in specific tablespaces
===

```
set linesize 160
set pagesize 9999
alter session set nls_date_format='dd-mon-RR hh24:mi:ss';
col tbs format a12
col owner format a8
col segment format a30
col type format a12


select tablespace_name tbs,owner, segment_name segment, segment_type type,
initial_extent "init B",next_extent "next B",extents EXTENTS, pct_increase pctinc
from dba_segments
where tablespace_name in ('SHARED','BISD','GMDD','GMDX','WSHD','GLD')
and (next_extent/1024/1024)> (select max(bytes/1024/1024) from dba_free_space where
tablespace_name in ('SHARED','BISD','GMDD','GMDX','WSHD','GLD'))
order by tablespace_name,segment_name;


select tablespace_name tbs,owner, segment_name segment, segment_type type,
initial_extent "init B",next_extent "next B",extents EXTENTS, pct_increase pctinc
from dba_segments
where tablespace_name in ('APPS_TS_SEED')
and (next_extent/1024/1024)> (select max(bytes/1024/1024) from dba_free_space where
tablespace_name in ('APPS_TS_SEED'))
order by tablespace_name,segment_name;
```

To identify the Segments Unable to extend
==
Segments in TS unable to extend

```
set linesize 150
select tablespace_name||'   '||owner||'  '||segment_name||'  '||next_extent||'
'||segment_type||'  '||pct_increase from
dba_segments
where tablespace_name ='&tbs' and next_extent > (select max(BYTES) from
dba_free_space where
tablespace_name ='&tbs')
order by next_extent;


set linesize 150
select a.tablespace_name||'   '||a.owner||'  '||a.segment_name||'  '||
a.next_extent||'   '||a.segment_type from
dba_segments a,dba_free_space b
where a. tablespace_name=b.tablespace_name and a.next_extent > (select max(BYTES)
from  dba_free_space where
tablespace_name =a.tablespace_name)
order by next_extent;
```

To check the SEGMENTS APPROCHING MAX EXTENTS
==

Identify all objects in the instance that do NOT have maxextents set to �UNLIMITED�

Applicable to all database versions -

```
col segment_name format a30
col owner format a20
col segment_type format a20
set lines 120
select s.segment_name, s.owner, s.segment_type, s.max_extents
from dba_segments s , dba_tablespaces t
where
s.tablespace_name=t.tablespace_name and
t.extent_management <> 'LOCAL' and
s.segment_type not in ('CACHE','DEFERRED ROLLBACK','SPACE HEADER') and
s.owner not in ('SYS','SYSTEM') and
s.max_extents <> 2147483645
order by s.owner,s.segment_type;
```

Script to generate �alter table� and �alter index� statements to set maxextents to unlimited

```
set lines 120
set heading off
spool maxextent_fix.sql
select
'ALTER '||s.segment_type||' '||s.owner||'.'||s.segment_name||' storage (maxextents
unlimited);' "Alter statement"
from dba_segments s , dba_tablespaces t
where
s.tablespace_name=t.tablespace_name and
t.extent_management<>'LOCAL' and
s.segment_type in ('TABLE','INDEX') and
s.owner not in ('SYS') and
s.max_extents<>2147483645
order by s.owner,s.segment_type
spool off
```

Checking for the segment which is approaching max extents (max extents reached)

```
col segment_name for a30
col owner for a15
set lines 200
select segment_name||'  '||segment_type||'  '||owner||'  '||extents||'  '||
max_extents||'  '||next_extent||'  '||pct_increase from
dba_segments where extents=max_extents and tablespace_name='&tablespace';
```

Checking storage parameters of a particular segment

```
col segment_name for a30
col owner for a15
set lines 200
select segment_name||'  '||segment_type||'  '||owner||'  '||extents||'  '||
max_extents||'  '||next_extent||'  '||pct_increase from
dba_segments where segment_name='WF_LOCAL_ROLES' and tablespace_name='APPLSYSD';
```

Segments reached maximum extent count in DB

```
col segment_name for a30
col owner for a15
set lines 200
select tablespace_name,segment_name||'  '||segment_type||'  '||owner||'  '||
extents||'  '||max_extents||'  '||next_extent||'  '||pct_increase from
dba_segments where extents=max_extents;
```

Segments approaching Max extents

```
select owner, SEGMENT_NAME, segment_type, tablespace_name,
NEXT_EXTENT,PCT_INCREASE, INITIAL_EXTENT, EXTENTS,MAX_EXTENTS  from dba_segments
where tablespace_name='&tbs' and  2*extents>max_extents;
```

Finding out table name

```
select table_name, column_name from dba_lobs where segment_name = '&Segment_name';
```

Modifying LOB Segments (EXAMPLES)

```
alter table <table> modify lob (<lobcolumn>) (storage (maxextents unlimited))
alter table P1CMGT.ASSET modify lob(CONTENT_BINARY) (storage (maxextents
unlimited));
alter table APPLSYS.FND_LOBS modify lob(FILE_DATA) (storage (maxextents
unlimited));
alter table APPLSYS.FND_LOBS allocate extent;
alter table JTF.JTF_DIAGNOSTIC_LOG modify lob(REPORT) (storage (maxextents
unlimited));
alter table JTF.JTF_DIAGNOSTIC_LOG modify lob(REPORT) (storage (maxextents
unlimited));
```

Modifying Normal Extents (EXAMPLES)

```
alter table GL.GL_DAILY_RATES storage (maxextents unlimited);
alter table GL.GL_DAILY_RATES allocate extent;
alter index APPLSYS.WF_USER_ROLE_ASSIGNMENTS_U1 storage (maxextents unlimited);
alter index APPLSYS.WF_USER_ROLE_ASSIGNMENTS_N1 storage (maxextents unlimited);
alter table APPLSYS.WF_LOCAL_USER_ROLES storage (maxextents unlimited);
alter table APPLSYS.WF_LOCAL_ROLES storage (maxextents unlimited);
alter index CRP.CRP_RESOURCE_PLAN_N1 storage (maxextents unlimited);
alter index CRP.CRP_RESOURCE_PLAN_N2 storage (maxextents unlimited);
alter index CRP.CRP_RESOURCE_PLAN_N3 storage (maxextents unlimited);
```

Allocating a new extent (EXAMPLES)

```
alter index CRP.CRP_RESOURCE_PLAN_N1 allocate extent;
alter index CRP.CRP_RESOURCE_PLAN_N2 allocate extent;
alter index CRP.CRP_RESOURCE_PLAN_N3 allocate extent;
alter index CRP.CRP_RESOURCE_PLAN_U1 allocate extent;
```

Scripts related to TEMP TABLESPACE
=====================================

To check instance-wise total allocated, total used TEMP for both rac and non-rac

```
set lines 152
col FreeSpaceGB format 999.999
col UsedSpaceGB format 999.999
col TotalSpaceGB format 999.999
col host_name format a30
col tablespace_name format a30
select tablespace_name,
(free_blocks*8)/1024/1024 FreeSpaceGB,
(used_blocks*8)/1024/1024 UsedSpaceGB,
(total_blocks*8)/1024/1024 TotalSpaceGB,
i.instance_name,i.host_name
from gv$sort_segment ss,gv$instance i where ss.tablespace_name in (select
tablespace_name from dba_tablespaces where contents='TEMPORARY') and
i.inst_id=ss.inst_id;
```

Total Used and Total Free Blocks

```
select inst_id, tablespace_name, total_blocks, used_blocks, free_blocks  from
gv$sort_segment;
```

Another Query to check TEMP USAGE

```
col name for a20
SELECT d.status "Status", d.tablespace_name "Name", d.contents "Type",
d.extent_management
"ExtManag",
TO_CHAR(NVL(a.bytes / 1024 / 1024, 0),'99,999,990.900') "Size (M)",
TO_CHAR(NVL(t.bytes,
0)/1024/1024,'99999,999.999') ||'/'||TO_CHAR(NVL(a.bytes/1024/1024,
0),'99999,999.999') "Used (M)",
TO_CHAR(NVL(t.bytes / a.bytes * 100, 0), '990.00') "Used %"
FROM sys.dba_tablespaces d, (select tablespace_name, sum(bytes) bytes from
dba_temp_files group by
tablespace_name) a,
(select tablespace_name, sum(bytes_cached) bytes from
v$temp_extent_pool group by tablespace_name) t
WHERE d.tablespace_name = a.tablespace_name(+) AND d.tablespace_name =
t.tablespace_name(+)
AND d.extent_management like 'LOCAL' AND d.contents like 'TEMPORARY';
```

Temporary Tablespace groups

```
SELECT * FROM DATABASE_PROPERTIES where PROPERTY_NAME='DEFAULT_TEMP_TABLESPACE';
```

```
select tablespace_name,contents from dba_tablespaces where tablespace_name like
'%TEMP%';
```

```
select * from dba_tablespace_groups;
```

Block wise Check

```
select TABLESPACE_NAME, TOTAL_BLOCKS, USED_BLOCKS, MAX_USED_BLOCKS,
MAX_SORT_BLOCKS, FREE_BLOCKS from V$SORT_SEGMENT;
```

```
select sum(free_blocks) from gv$sort_segment where tablespace_name = 'TEMP';
```

To Check Percentage Usage of Temp Tablespace

```
select (s.tot_used_blocks/f.total_blocks)*100 as "percent used"
from (select sum(used_blocks) tot_used_blocks
from v$sort_segment where tablespace_name='TEMP') s,
(select sum(blocks) total_blocks
from dba_temp_files where tablespace_name='TEMP') f;
```

To check Used Extents ,Free Extents available in Temp Tablespace

```
SELECT tablespace_name, extent_size, total_extents, used_extents,free_extents,
max_used_size FROM v$sort_segment;
```

To list all tempfiles of Temp Tablespace

```
col file_name for a45
select tablespace_name,file_name,bytes/1024/1024,maxbytes/1024/1024,autoextensible
from dba_temp_files  order by file_name;
```

```
SELECT d.tablespace_name tablespace , d.file_name filename, d.file_id fl_id,
d.bytes/1024/1024
size_m
, NVL(t.bytes_cached/1024/1024, 0) used_m, TRUNC((t.bytes_cached / d.bytes) * 100)
pct_used
```

```
FROM
sys.dba_temp_files d, v$temp_extent_pool t, v$tempfile v
WHERE (t.file_id (+)= d.file_id)
AND (d.file_id = v.file#);
```

Additional checks

```
select distinct(temporary_tablespace) from dba_users;
```

```
select username,default_tablespace,temporary_tablespace from dba_users order by
temporary_tablespace;
```

```
SELECT * FROM DATABASE_PROPERTIES where PROPERTY_NAME='DEFAULT_TEMP_TABLESPACE';
```

Changing the default temporary Tablespace

```
SQL> alter database default temporary tablespace TEMP;
```

```
Database altered.
```

To add tempfile to Temp Tablespace

```
alter tablespace  temp  add tempfile '&tempfilepath' size 1800M;
```

```
alter tablespace temp add tempfile '/m001/oradata/SID/temp02.dbf' size 1000m;
```

```
alter tablespace TEMP add tempfile '/SID/oradata/data02/temp04.dbf' size 1800M
autoextend on maxsize 1800M;
```

To resize the tempfile in Temp Tablespace

```
alter database tempfile '/u02/oradata/TESTDB/temp01.dbf' resize 250M
```

```
alter database tempfile '/SID/oradata/data02/temp12.dbf' autoextend on maxsize
1800M;
```

```
alter tablespace TEMP add tempfile '/SID/oradata/data02/temp05.dbf' size 1800m
reuse;
```

To find Sort Segment Usage by Users

```
select username,sum(extents) "Extents",sum(blocks) "Block"
```

```
from v$sort_usage
group by username;
```

To find Sort Segment Usage by a particular User

```
SELECT s.username,s.sid,s.serial#,u.tablespace, u.contents, u.extents, u.blocks
FROM v$session s, v$sort_usage u
WHERE s.saddr=u.session_addr
order by u.blocks desc;
```

To find Total Free space in Temp Tablespace

```
select 'FreeSpace  ' || (free_blocks*8)/1024/1024 ||' GB'  from v$sort_segment
where tablespace_name='TEMP';
```

```
select tablespace_name , (free_blocks*8)/1024/1024  FreeSpaceInGB,
(used_blocks*8)/1024/1024  UsedSpaceInGB,
(total_blocks*8)/1024/1024  TotalSpaceInGB
from v$sort_segment where tablespace_name like '%TEMP%'
```

To find Total Space Allocated for Temp Tablespace

```
select 'TotalSpace ' || (sum(blocks)*8)/1024/1024 ||' GB'  from dba_temp_files
where tablespace_name='TEMP';
```

Get 10 sessions with largest temp usage

```
cursor bigtemp_sids is
select * from (
select s.sid,
s.status,
s.sql_hash_value sesshash,
u.SQLHASH sorthash,
s.username,
u.tablespace,
sum(u.blocks*p.value/1024/1024) mbused ,
sum(u.extents) noexts,
nvl(s.module,s.program) proginfo,
floor(last_call_et/3600)||':'||
floor(mod(last_call_et,3600)/60)||':'||
mod(mod(last_call_et,3600),60) lastcallet
from v$sort_usage u,
v$session s,
v$parameter p
where u.session_addr = s.saddr
and p.name = 'db_block_size'
group by s.sid,s.status,s.sql_hash_value,u.sqlhash,s.username,u.tablespace,
nvl(s.module,s.program),
floor(last_call_et/3600)||':'||
floor(mod(last_call_et,3600)/60)||':'||
mod(mod(last_call_et,3600),60)
order by 7 desc,3)
where rownum < 11;
```

Displays the amount of IO for each tempfile

```
SELECT SUBSTR(t.name,1,50) AS file_name,
f.phyblkrd AS blocks_read,
f.phyblkwrt AS blocks_written,
f.phyblkrd + f.phyblkwrt AS total_io
FROM   v$tempstat f,v$tempfile t
WHERE  t.file# = f.file#
ORDER BY f.phyblkrd + f.phyblkwrt DESC;
```

```
select * from (SELECT u.tablespace, s.username, s.sid, s.serial#, s.logon_time,
program, u.extents, ((u.blocks*8)/1024) as MB,
i.inst_id,i.host_name
FROM gv$session s, gv$sort_usage u ,gv$instance i
WHERE s.saddr=u.session_addr and u.inst_id=i.inst_id  order by MB DESC) a where
rownum<10;
```

Check for ORA-1652

```
show parameter background
```

```
cd <background dump destination>
```

```
ls -ltr|tail
```

```
view <alert log file name>
```

shift + G ---> to get the tail end...

?ORA-1652 ---- to search of the error...

shift + N ---- to step for next reported error...

I used these queries to check some settings:

```
-- List all database files and their tablespaces:
select  file_name, tablespace_name, status
,bytes   /1000000  as MB
,maxbytes/1000000  as MB_max
from dba_data_files ;
```

```
-- What temporary tablespace is each user using?:
select username, temporary_tablespace, default_tablespace from dba_users ;


-- List all tablespaces and some settings:
select tablespace_name, status, contents, extent_management
from dba_tablespaces ;

TABLESPACE_NAME                CONTENTS  EXTENT_MAN STATUS
------------------------------ --------- ---------- ---------
SYSTEM                         PERMANENT DICTIONARY ONLINE
TOOLS                          PERMANENT DICTIONARY ONLINE
TEMP                           TEMPORARY DICTIONARY OFFLINE
TMP                            TEMPORARY LOCAL      ONLINE
```

Now, the above query and the storage clause of the old 'create tablespace TEMP' command seem to tell us the tablespace only allows temporary objects, so it should be safe to assume that no one created any tables or other permanent objects in TEMP by mistake, as I think Oracle would prevent that. However, just to be absolutely certain, I decided to double-check. Checking for any tables in the tablespace is very easy:

```
-- Show number of tables in the TEMP tablespace - SHOULD be 0:
select count(*)  from dba_all_tables
where tablespace_name = 'TEMP' ;
```

Checking for any other objects (views, indexes, triggers, pl/sql, etc.) is trickier, but this query seems to work correctly - note that you'll probably need to connect internal in order to see the sys_objects view:

```
-- Shows all objects which exist in the TEMP tablespace - should get
-- NO rows for this:
column owner        format a20
column object_type  format a30
column object_name  format a40
select
o.owner  ,o.object_name
,o.object_type
from sys_objects s
,dba_objects o
,dba_data_files df
where df.file_id = s.header_file
and o.object_id = s.object_id
and df.tablespace_name = 'TEMP' ;
```

Identifying WHO is currently using TEMP Segments

10g onwards

```
SELECT sysdate,a.username, a.sid, a.serial#, a.osuser,
(b.blocks*d.block_size)/1048576 MB_used, c.sql_text
FROM v$session a, v$tempseg_usage b, v$sqlarea c,
     (select block_size from dba_tablespaces where tablespace_name='TEMP') d
    WHERE b.tablespace = 'TEMP'
    and a.saddr = b.session_addr
    AND c.address= a.sql_address
    AND c.hash_value = a.sql_hash_value
    AND (b.blocks*d.block_size)/1048576 > 1024
    ORDER BY b.tablespace, 6 desc;
```

Queries related to Undo tablespace
=======================================

To check retention guarantee for undo tablespace

```
select tablespace_name,status,contents,logging,retention from dba_tablespaces where
tablespace_name like '%UNDO%';
```

To show ACTIVE/EXPIRED/UNEXPIRED Extents of Undo Tablespace

```
select     tablespace_name,
status,
count(extent_id) "Extent Count",
sum(blocks) "Total Blocks",
sum(blocks)*8/(1024*1024) total_space
from     dba_undo_extents
group by    tablespace_name, status;
```

Extent Count and Total Blocks

```
set linesize 152
col tablespace_name for a20
col status for a10
```

```
select tablespace_name,status,count(extent_id) "Extent Count",
sum(blocks) "Total Blocks",sum(bytes)/(1024*1024*1024) spaceInGB
from   dba_undo_extents
where  tablespace_name in ('&undotbsp')
group by  tablespace_name,status;
```

To show UndoRetention Value

```
Show parameter undo_retention;
```

Undo retention in hours

```
col "Retention" for a30
col name for a30
col value for a50
select name "Retention",value/60/60 "Hours" from v$parameter where name like
'%undo_retention%';
```

To check space related statistics of UndoTablespace from stats$undostat of 90 days

```
select UNDOBLKS,BEGIN_TIME,MAXQUERYLEN,UNXPSTEALCNT,EXPSTEALCNT,NOSPACEERRCNT from
stats$undostat where BEGIN_TIME between sysdate-90 and sysdate and UNXPSTEALCNT >
0;
```

To check space related statistics of UndoTablespace from v$undostat

```
select
sum(ssolderrcnt) "Total ORA-1555s",
round(max(maxquerylen)/60/60) "Max Query HRS",
sum(unxpstealcnt) "UNExpired STEALS",
sum(expstealcnt) "Expired STEALS"
from v$undostat
order by begin_time;
```

Date wise occurrence of ORA-1555

```
select to_char(begin_time, 'mm/dd/yyyy hh24:mi') "Int. Start",
ssolderrcnt "ORA-1555s", maxquerylen "Max Query",
unxpstealcnt "UNExp SCnt",UNXPBLKRELCNT "UnEXPblks", expstealcnt "Exp
SCnt",EXPBLKRELCNT "ExpBlks",
NOSPACEERRCNT nospace
from v$undostat where ssolderrcnt>0
order by begin_time;
```

Total number of ORA-1555s since instance startup

```
select 'TOTAL # OF ORA-01555 SINCE INSTANCE STARTUP : '|| to_char(startup_time,'DD-
MON-YY HH24:MI:SS')
from v$instance;
```

To check for Active Transactions

```
set head on
select usn,extents,round(rssize/1048576)
rssize,hwmsize,xacts,waits,optsize/1048576 optsize,shrinks,wraps
from v$rollstat where xacts>0
order by rssize;
```

Undo Space Utilization by each Sessions

```
set lines 200
col sid for 99999
col username for a10
col name for a15
select  s.sid,s.serial#,username,s.machine,
t.used_ublk ,t.used_urec,rn.name,(t.used_ublk *8)/1024/1024 SizeGB
from    v$transaction t,v$session s,v$rollstat rs, v$rollname rn
where   t.addr=s.taddr and rs.usn=rn.usn and rs.usn=t.xidusn and rs.xacts>0;
```

List of long running queries since instance startup

```
set head off
select 'LIST OF LONG RUNNING - QUERY SINCE INSTANCE STARTUP' from dual;
set head on
select     *
from
(select to_char(begin_time, 'DD-MON-YY hh24:mi:ss') BEGIN_TIME ,
round((maxquerylen/3600),1) Hours
from v$undostat
order by maxquerylen desc)
where    rownum < 11;
```

Undo Space used by all transactions

```
set lines 200
col sid for 99999
col username for a10
col name for a15
select  s.sid,s.serial#,username,s.machine,
t.used_ublk ,t.used_urec,rn.name,(t.used_ublk *8)/1024/1024 SizeGB
from    v$transaction t,v$session s,v$rollstat rs, v$rollname rn
where   t.addr=s.taddr and rs.usn=rn.usn and rs.usn=t.xidusn and rs.xacts>0;
```

List of All active Transactions

```
select  sid,username,
t.used_ublk ,t.used_urec
from    v$transaction t,v$session s
where   t.addr=s.taddr;
```

To list all Datafile of UndoTablespace

```
select tablespace_name,file_name,file_id,autoextensible,bytes/1048576
Mbytes, maxbytes/1048576 maxMbytes
from dba_data_files
where tablespace_name like '%UNDO%'
or tablespace_name like '%RBS%'
order by tablespace_name,file_name;
```

```
select tablespace_name,file_name,file_id,autoextensible,bytes/1048576
Mbytes, maxbytes/1048576 maxMbytes
from dba_data_files
where tablespace_name like '%UNDOTBS2%'
order by tablespace_name,file_name;


col file_name for a40
set pagesize 100
select tablespace_name,file_name,file_id,autoextensible,bytes/1048576
Mbytes, maxbytes/1048576 maxMbytes
from dba_data_files
where tablespace_name like '%APPS_UNDOTS1%'
order by tablespace_name,file_name;


select file_name,tablespace_name,bytes/1024/1024,maxbytes/1024/1024,autoextensible
from dba_data_files where file_name like '%undo%' order by file_name;
```

To check when a table is last analysed

```
select
OWNER,TABLE_NAME,TABLESPACE_NAME,STATUS,LAST_ANALYZED,PARTITIONED,DEPENDENCIES,DROP
PED from dba_tables where TABLE_NAME like 'MLC_PICK_LOCKS_DETAIL';


select
OWNER,TABLE_NAME,TABLESPACE_NAME,LAST_ANALYZED,PARTITIONED,DEPENDENCIES,DROPPED
from dba_tables where TABLE_NAME like 'APPS.XLA_AEL_GL_V';
```

To list all Undo datafiles with status and size

```
show parameter undo
show parameter db_block_size
col tablespace_name form a20
col file_name form a60
set lines 120
select tablespace_name, file_name, status, bytes/1024/1024 from dba_data_files
where tablespace_name=(select tablespace_name from dba_tablespaces where
contents='UNDO');
```

Total undo space

```
select    sum(bytes)/1024/1024/1024 GB from dba_data_files  where
tablespace_name='&Undo_TB_Name';
```

Undo Tablespace

```
select tablespace_name from dba_tablespaces where tablespace_name like '%UNDO%';
```

To find MaxQueryLength from stats$undostat

```
Select Max(MAXQUERYLEN) from stats$undostat;
```

```
*select max(maxquerylen) from v$undostat;
```

```
*select begin_date,u.maxquerylen from
(select to_char(begin_time,'DD-MON-YYYY:HH24-MI-SS') begin_date,maxquerylen
from v$undostat order by maxquerylen desc) u  where rownum<11;
```

```
*select begin_date,u.maxquerylen from
(select maxquerylen,to_char(begin_time,'DD-MON-YYYY:HH24-MI-SS') begin_date from
v$undostat order by maxquerylen DESC) u  where rownum<26 order by begin_date ASC,
maxquerylen DESC;
```

```
*select begin_date,u.maxquerylen from
(select maxquerylen,to_char(begin_time,'DD-MON-YYYY:HH24-MI-SS') begin_date from
v$undostat order by maxquerylen DESC) u  where rownum<26 order by  maxquerylen
DESC;
```

```
*select sum(u.maxquerylen)/25 AvgUndoRetTime
from (select maxquerylen from v$undostat order by maxquerylen desc) u  where
rownum<26;
*select sum(u.maxquerylen)
from (select maxquerylen from v$undostat order by maxquerylen desc) u  where
rownum<26;
```

DBA_UNDO_EXTENTS

```
set linesize 152
col tablespace_name for a20
col status for a10
select tablespace_name,status,count(extent_id) "Extent Count",
sum(blocks) "Total Blocks",sum(bytes)/(1024*1024*1024) spaceInGB
from   dba_undo_extents
group by  tablespace_name, status
order by tablespace_name;
```

Mapping Undo Segments to usernames

```
select  s.sid,s.serial#,username,s.machine,
t.used_ublk ,t.used_urec,(rs.rssize)/1024/1024 MB,rn.name
from    v$transaction t,v$session s,v$rollstat rs, v$rollname rn
where   t.addr=s.taddr and rs.usn=rn.usn and rs.usn=t.xidusn and rs.xacts>0;
```

Total Undo Statistics

```
alter session set nls_date_format='dd-mon-yy hh24:mi';
set lines 120
set pages 2000
select BEGIN_TIME,  END_TIME, UNDOBLKS, TXNCOUNT , MAXQUERYLEN  , UNXPSTEALCNT ,
EXPSTEALCNT , SSOLDERRCNT , NOSPACEERRCNT
from v$undostat;
```

Total Undo Statistics since specified year

```
select 'TOTAL STATISTICS SINCE Jan 01, 2005 - STATSPACK' from dual;
set head on
set lines 152
column undotsn format 999 heading 'Undo|TS#';
column undob format 9,999,999,999 heading 'Undo|Blocks';
column txcnt format 9,999,999,999,999 heading 'Num|Trans';
column maxq format 999,999 heading 'Max Qry|Len (s)';
column maxc format 9,999,999 heading 'Max Tx|Concurcy';
column snol format 9,999 heading 'Snapshot|Too Old';
column nosp format 9,999 heading 'Out of|Space';
column blkst format a13 heading 'uS/uR/uU/|eS/eR/eU' wrap;
column unst format 9,999 heading 'Unexp|Stolen' newline;
column unrl format 9,999 heading 'Unexp|Relesd';
column unru format 9,999 heading 'Unexp|Reused';
column exst format 9,999 heading 'Exp|Stolen';
column exrl format 9,999 heading 'Exp|Releas';
column exru format 9,999 heading 'Exp|Reused';
select undotsn
, sum(undoblks) undob
, sum(txncount) txcnt
, max(maxquerylen) maxq
, max(maxconcurrency) maxc
, sum(ssolderrcnt) snol
, sum(nospaceerrcnt) nosp
, sum(unxpstealcnt)
||'/'|| sum(unxpblkrelcnt)
||'/'|| sum(unxpblkreucnt)
||'/'|| sum(expstealcnt)
||'/'|| sum(expblkrelcnt)
||'/'|| sum(expblkreucnt) blkst
from stats$undostat
where dbid in (select dbid from v$database)
and instance_number in (select instance_number from v$instance)
and end_time > to_date('01012005 00:00:00', 'DDMMYYYY HH24:MI:SS')
and begin_time < (select sysdate from dual)
group by undotsn;
```

```
*SELECT (SUM(undoblks))/ SUM ((end_time - begin_time) * 86400) FROM v$undostat;
```

Checking for Recent ORA-1555

```
show parameter background
```

```
cd <background dump destination>
ls -ltr|tail
```

```
view <alert log file name>
```

shift + G ---> to get the tail end...

?ORA-1555 ---- to search of the error...

shift + N ---- to step for next reported error...

Rollback segment queries

Wraps

```
select name,extents,rssize/1048576 rssizeMB ,xacts,writes/1024/1024,optsize/1048576
optsize,
shrinks,wraps,extends,aveshrink/1048576,waits,rs.status,rs.curext
from v$rollstat rs, v$rollname rn  where  rn.usn=rs.usn
order by wraps;
```

Wraps column as high values for the all segments size of rollback segments are small for long running queries and transactions by increasing the rollback segments size we can avoid the ORA-01555 errors

Undo Contention

Rollback Segment Contention

```
prompt   If any ratio is > .01 then more rollback segments are needed
```

```
column "total_waits" format 999,999,999
column "total_timeouts" format 999,999,999
column "Ratio" format 99.99999
select name, waits, gets, waits/gets "Ratio"
from v$rollstat a, v$rollname b
where a.usn = b.usn;
```

```
Sample Output:
REM NAME                              WAITS             GETS     Ratio
REM ------------------------------ ----------         ---------- ---------
REM SYSTEM                                 0                269    .00000
REM R01                                    0                304    .00000
REM R02                                    0               2820    .00000
REM R03                                    0                629    .00000
REM R04                                      1                511    .00196
REM R05                                      0                513    .00000
REM R06                                      1                503    .00199
REM R07                                      0                301    .00000
REM R08                                      0                299    .00000
```

Looking at the tcl script to see what sql gets performed to determine rollback segment contention

```
select count from v$waitstat where class = 'system undo header';
select count from v$waitstat where class = 'system undo block';
select count from v$waitstat where class = 'undo header';
select count from v$waitstat where class = 'undo block';
```

Rollback Segment Information

```
set lines 152
col segment_type  for a10
col tablespace_name for a20
select owner,tablespace_name,extents,next_extent/1024
next_extnentKB,max_extents,pct_increase
from dba_segments
where segment_type='ROLLBACK';
```

```
* set lines 152
col name for a15
select name,extents,rssize/1048576 rssizeMB ,xacts,writes/1024/1024,optsize/1048576
optsize,
shrinks,wraps,aveshrink/1048576,waits,rs.status,rs.curext
from v$rollstat rs, v$rollname rn  where  rn.usn=rs.usn and rs.xacts>0;
```

```
* select name,extents,rssize/1048576 rssizeMB
,xacts,writes/1024/1024,optsize/1048576 optsize,
shrinks,wraps,extends,aveshrink/1048576,waits,rs.status,rs.curext
from v$rollstat rs, v$rollname rn  where  rn.usn=rs.usn
order by wraps;
```

```
* select name,extents,optsize/1048576 optsize,
shrinks,wraps,aveshrink/1048576,aveactive,rs.status,rs.curext
from v$rollstat rs, v$rollname rn  where  rn.usn=rs.usn;
```

```
* select  sum(rssize)/1024/1024/1024 sizeGB from v$rollstat;
```

```
* select sum(xacts) from v$rollstat;
select  sum(rssize)/1024/1024/1024 sizeGB from v$rollstat where xacts=0;
select  sum(rssize)/1024/1024/1024 sizeGB from v$rollstat where xacts>0;
select sum(xacts) from v$rollstat;
```

```
* select
```

```
tablespace_name,segment_name,initial_extent,next_extent,min_extents,max_extents,sta
tus
from dba_rollback_segs
where status='ONLINE';


* select
tablespace_name,file_name,bytes/1024/1024,maxbytes/1024/1024,autoextensible
from dba_data_files where file_name like '%&filename%';


* select sum(bytes)/1024/1024 from dba_free_space where tablespace_name='&tbs';
```

Optimize Oracle UNDO Parameters

Actual Undo Size

```
SELECT SUM(a.bytes/1024/1024/1024) "UNDO_SIZE"
FROM v$datafile a,
v$tablespace b,
dba_tablespaces c
WHERE c.contents = 'UNDO'
AND c.status = 'ONLINE'
AND b.name = c.tablespace_name
AND a.ts# = b.ts#;


UNDO_SIZE
----------
209715200
```

Undo Blocks per Second

```
SELECT MAX(undoblks/((end_time-begin_time)*3600*24))
"UNDO_BLOCK_PER_SEC"
FROM v$undostat;


UNDO_BLOCK_PER_SEC
------------------
3.12166667
```

Undo Segment Summary for DB

```
Undo Segment Summary for DB: S901  Instance: S901  Snaps: 2 -3
-> Undo segment block stats:
-> uS - unexpired Stolen,   uR - unexpired Released,   uU - unexpired reUsed
-> eS - expired   Stolen,   eR - expired   Released,   eU - expired   reUsed


Undo           Undo        Num  Max Qry     Max Tx Snapshot Out of uS/uR/uU/
TS#          Blocks      Trans  Len (s)   Concurcy  Too Old  Space eS/eR/eU
---- -------------- ---------- -------- ---------- -------- ------ -------------
1         20,284      1,964        8         12        0      0 0/0/0/0/0/0
```

Undo Segment Stats for DB

```
Undo Segment Stats for DB: S901  Instance: S901  Snaps: 2 -3
-> ordered by Time desc


Undo       Num Max Qry   Max Tx  Snap   Out of uS/uR/uU/
End Time           Blocks    Trans Len (s)    Concy Too Old  Space eS/eR/eU
------------ ------------ -------- ------- -------- ------- ------ -------------
12-Mar 16:11       18,723    1,756       8       12       0      0 0/0/0/0/0/0
12-Mar 16:01        1,561      208       3       12       0      0 0/0/0/0/0/0
```

Undo Segment Space Required = (undo_retention_time * undo_blocks_per_seconds)

As an example, an UNDO_RETENTION of 5 minutes (default) with 50 undo blocks/second (8k blocksize)
will generate:
Undo Segment Space Required = (300 seconds * 50 blocks/ seconds * 8K/block) = 120 M

```
select tablespace_name,file_name,file_id,autoextensible,bytes/1048576
Mbytes, maxbytes/1048576 maxMbytes
from dba_data_files
where tablespace_name like '%UNDO%'
or tablespace_name like '%RBS%'
or tablespace_name like '%ROLLBACK%'
order by tablespace_name,file_name;
```

```
select a.owner,a.tablespace_name,b.status, a.extents,a.next_extent/1024
next_extnentKB,a.max_extents,a.pct_increase from dba_segments a,dba_tablespaces b
where segment_type='ROLLBACK' and a.tablespace_name=b.tablespace_name;
```

```
select tablespace_name,status from dba_tablespaces where
tablespace_name='ROLLBACK';
```

Actual Undo Size

```
SELECT SUM(a.bytes/1024/1024) "UNDO_SIZE"
FROM v$datafile a,
v$tablespace b,
dba_tablespaces c
WHERE c.contents = 'UNDO'
AND c.status = 'ONLINE'
AND b.name = c.tablespace_name
AND a.ts# = b.ts#;
```

```
UNDO_SIZE
```

```
----------
209715200
```

Undo Blocks per Second

```
SELECT MAX(undoblks/((end_time-begin_time)*3600*24))
"UNDO_BLOCK_PER_SEC"
FROM v$undostat;
```

```
UNDO_BLOCK_PER_SEC
------------------
3.12166667
```

DB Block Size

```
SELECT TO_NUMBER(value) "DB_BLOCK_SIZE [KByte]"
FROM v$parameter
WHERE name = 'db_block_size';
```

```
DB_BLOCK_SIZE [Byte]
--------------------
4096
```

Optimal Undo Retention

```
209'715'200 / (3.12166667 * 4'096) = 16'401 [Sec]
```

Using Inline Views, you can do all in one query!

```
SELECT d.undo_size/(1024*1024) "ACTUAL UNDO SIZE [MByte]",
SUBSTR(e.value,1,25) "UNDO RETENTION [Sec]",
ROUND((d.undo_size / (to_number(f.value) *
g.undo_block_per_sec))) "OPTIMAL UNDO RETENTION [Sec]"
FROM (
SELECT SUM(a.bytes) undo_size
FROM v$datafile a,
v$tablespace b,
dba_tablespaces c
WHERE c.contents = 'UNDO'
AND c.status = 'ONLINE'
AND b.name = c.tablespace_name
AND a.ts# = b.ts#
) d,
v$parameter e,
v$parameter f,
(
SELECT MAX(undoblks/((end_time-begin_time)*3600*24))
undo_block_per_sec
FROM v$undostat
) g
WHERE e.name = 'undo_retention'
AND f.name = 'db_block_size'
/
```

```
ACTUAL UNDO SIZE [MByte]
------------------------
200
```

```
UNDO RETENTION [Sec]
--------------------
10800
```

```
OPTIMAL UNDO RETENTION [Sec]
----------------------------
16401
```

Calculate Needed UNDO Size for given Database Activity

If you are not limited by disk space, then it would be better to choose the UNDO_RETENTION time that is best for you (for FLASHBACK, etc.). Allocate the appropriate size to the UNDO tablespace according to the database activity:

Again, all in one query:

```
SELECT d.undo_size/(1024*1024) "ACTUAL UNDO SIZE [MByte]",
SUBSTR(e.value,1,25) "UNDO RETENTION [Sec]",
(TO_NUMBER(e.value) * TO_NUMBER(f.value) *
g.undo_block_per_sec) / (1024*1024)
"NEEDED UNDO SIZE [MByte]"
FROM (
SELECT SUM(a.bytes) undo_size
FROM v$datafile a,
v$tablespace b,
dba_tablespaces c
WHERE c.contents = 'UNDO'
AND c.status = 'ONLINE'
AND b.name = c.tablespace_name
AND a.ts# = b.ts#
) d,
v$parameter e,
v$parameter f,
(
SELECT MAX(undoblks/((end_time-begin_time)*3600*24))
undo_block_per_sec
FROM v$undostat
) g
WHERE e.name = 'undo_retention'
AND f.name = 'db_block_size'
/
```

```
ACTUAL UNDO SIZE [MByte]
------------------------
200
UNDO RETENTION [Sec]
--------------------
10800
NEEDED UNDO SIZE [MByte]
------------------------
131.695313
```

Checking when tables are last analyzed

```
select
OWNER,TABLE_NAME,TABLESPACE_NAME,STATUS,LAST_ANALYZED,PARTITIONED,DEPENDENCIES,DROP
PED from
dba_tables where TABLE_NAME like 'MLC_END_USER_REGISTRATION';
```

```
DECLARE
v_table_space_name      VARCHAR2(30);
v_table_space_size_in_MB        NUMBER(9);
v_auto_extend      BOOLEAN;
v_undo_retention      NUMBER(9);
v_retention_guarantee    BOOLEAN;
v_undo_info_return    BOOLEAN;
BEGIN
v_undo_info_return := dbms_undo_adv.undo_info(v_table_space_name,
v_table_space_size_in_MB, v_auto_extend, v_undo_retention, v_retention_guarantee);
dbms_output.put_line(�UNDO Tablespace Name: � || v_table_space_name);
dbms_output.put_line(�UNDO Tablespace size (MB) : � ||
TO_CHAR(v_table_space_size_in_MB));
dbms_output.put_line(�If UNDO tablespace is auto extensible above size indicates
max possible size of the undo tablespace�);
dbms_output.put_line(�UNDO tablespace auto extensiable is : �|| CASE WHEN
v_auto_extend THEN  �ON� ELSE �OFF� END);
dbms_output.put_line(�Undo Retention (Sec): � || v_undo_retention);
dbms_output.put_line(�Retention : �||CASE WHEN v_retention_guarantee THEN
�Guaranteed � ELSE �NOT Guaranteed� END);
END;
```

undo_autotune

This function is used to find auto tuning of undo retention is ENABLED or NOT.

```
Set serverout on
declare
v_autotune_return Boolean := null;
v_autotune_enabled boolean := null;
begin
v_autotune_return:= dbms_undo_adv.undo_autotune(v_autotune_enabled);
dbms_output.put_line(CASE WHEN v_autotune_return THEN 'Information is available :'
ELSE 'Information is NOT available :' END||
CASE WHEN v_autotune_enabled THEN 'Auto tuning of undo retention is ENABLED' ELSE
'Auto tuning of undo retention is NOT enabled' END);
end;
/
```

```
select dbms_undo_adv.longest_query from dual


select dbms_undo_adv.required_retention from dual




select dbms_undo_adv.best_possible_retention from dual


select  dbms_undo_adv.required_undo_size(1800) from dual


DECLARE
v_undo_health_return number;
v_retention number;
v_utbsize number;
v_problem VARCHAR2(1024);
v_recommendation VARCHAR2(1024);
v_rationale VARCHAR2(1024);
BEGIN
v_undo_health_return :=  dbms_undo_adv.undo_health(problem => v_problem,
recommendation => v_recommendation,
rationale => v_rationale,
retention => v_retention,
utbsize => v_utbsize);
dbms_output.put_line(�Problem : �||v_problem);
dbms_output.put_line(�Recommendation= : �||v_recommendation);
dbms_output.put_line(�Rationale : �||v_retention);
dbms_output.put_line(�Retention : �||v_retention);
dbms_output.put_line(�UNDO tablespace size : �||v_utbsize);
END;


undo_advisor


It uses oracle�s advisor framework to find out problem and provide recommendations.


DECLARE
v_undo_advisor_return VARCHAR2(100);
BEGIN
v_undo_advisor_return := dbms_undo_adv.undo_advisor(instance => 1);
dbms_output.put_line(v_undo_advisor_return);
END;



To find Percentage Usage of Undo Tablespace which considers Expired Space
===========================================================================



SELECT d.tablespace_name, round(((NVL(f.bytes,0) + (a.maxbytes - a.bytes))/1048576+
u.exp_space),2)
```

```
as max_free_mb, round(((a.bytes - (NVL(f.bytes,0)+
(1024*1024*u.exp_space)))*100/a.maxbytes),2)
used_pct FROM   sys.dba_tablespaces d, (select tablespace_name, sum(bytes) bytes,
sum(greatest(maxbytes,bytes)) maxbytes from sys.dba_data_files group by
tablespace_name) a,
(select tablespace_name, sum(bytes) bytes from sys.dba_free_space group by
tablespace_name) f ,
(select tablespace_name , sum(blocks)*8/(1024)  exp_space from
dba_undo_extents where status NOT IN ('ACTIVE','UNEXPIRED')  group by
tablespace_name) u
WHERE d.tablespace_name = a.tablespace_name(+) AND d.tablespace_name =
f.tablespace_name(+)
AND d.tablespace_name=u.tablespace_name  AND d.contents = 'UNDO' AND
u.tablespace_name = (select UPPER(value)
from v$parameter where name = 'undo_tablespace');
```

Tablespace Monitoring script
==============================

```
select d.tablespace_name,
round((d.max - u.bytes)/1024/1024,2) as max_free_mbytes,
round(u.bytes*100/d.max,2) as used_pct
from sys.SM$TS_USED u,
(select tablespace_name,
sum(decode(MAXBYTES,0,bytes,maxbytes)) max
from  sys.dba_data_files
group by tablespace_name) d
where u.tablespace_name = d.tablespace_name and
round(u.bytes*100/d.max,2)>70;
```

Oracle Tablespace Maintenance scripts excluding Undo and Temp tablespaces
==

To Find % of free space left in Tablespace

```
select
(BYTES/1024)/1024 "Used Space(MB)",
total  "allocated size(MB)",
maxi "maximum allowable (MB)",
maxi-(BYTES/1024)/1024 "effectivefree(MB)",
--maxi-total "free(MB)",
round(((maxi-(BYTES/1024)/1024)/maxi)*100,2) "% FREE"
from
SM$TS_USED,(select sum((BYTES/1024)/1024)
total,sum((decode(MAXBYTES,0,bytes,maxbytes)/1024)/1024)  maxi from
dba_data_files where tablespace_name in ('&tbs')) where
tablespace_name in ('&tbs');
```

To list all the datafiles of a given tablespace

```
col file_name for a60
set lines 170
set pages 200
select tablespace_name,file_name,bytes/1024/1024,maxbytes/1024/1024,autoextensible
from dba_data_files
where tablespace_name='&tablespace_name' order by file_name ;
```

To list a specific datafile space details using file_id

```
col file_name for a40
set pagesize 100
select tablespace_name,file_name,bytes/1024/1024,maxbytes/1024/1024,autoextensible
from dba_data_files
where file_id=128 order by file_name;
```

Space left to extend for autoextensible files in a mount point

(replace File_name variable to the associated mountpoint)

```
col file_name for a40
set lines 170 pages 0
BREAK on REPORT
compute SUM of space_left_to_extend on report
select tablespace_name,file_name,bytes/1024/1024,maxbytes/1024/1024,autoextensible,
(maxbytes/1024/1024)-(bytes/1024/1024) space_left_to_extend from dba_data_files
where file_name like '%/u01/oradata/%' and autoextensible='YES' order by file_name;
```

To check autoextendable files of a mountpoint

```
col file_name for a40
set pagesize 100
select tablespace_name,file_name,bytes/1024/1024 ,maxbytes/1024/1024,autoextensible
from dba_data_files
where file_name like '%/tbarti/oradata%' and autoextensible='YES' order by
file_name;
```

To check Current size of an autoextendable file

```
col file_name for a40
set pagesize 100
select SUM(bytes/1024/1024) from dba_data_files
where file_name like '%/tmzg1s/oradata%' and autoextensible='YES';
```

To check Total space that is supposed to be extendable for autoextensible files of
a mountpoint

```
col file_name for a40
set pagesize 100
```

```
select (SUM(maxbytes/1024/1024)-SUM(bytes/1024/1024)) EXTENSIBLESIZE from
dba_data_files
where file_name like '%/tmzg1s/oradata%' and autoextensible='YES';
```

To check files in a Mountpoint

```
col file_name for a40
set pagesize 100
select tablespace_name,file_name,bytes/1024/1024,maxbytes/1024/1024,autoextensible
from dba_data_files
where file_name like '%/tnhh1o/oradata/data01/orabpel.dbf_back%' order by file_name
;
```

```
col file_name for a40
set pagesize 100
select tablespace_name,file_name,bytes/1024/1024,maxbytes/1024/1024,autoextensible
from dba_data_files
where file_name like '%/tmzg1s/oradata%' and autoextensible='YES' order by
file_name ;
```

```
col file_name for a40
set pagesize 100
select count(file_name) from dba_data_files
where file_name like '%/dcybxi/oradata/%' and autoextensible='YES';
```

Checking total space available within TS (Effective MB)

```
select tablespace_name, sum(bytes/1024/1024) "mb" from dba_free_space
group by tablespace_name having tablespace_name=upper('&tname');
```

Tablespace with less than 25% free space

```
set linesize 100 pagesize 30 feedback 1 echo off wrap on
column tblspace format a12 heading "Tablespace"
column "Used Space(MB)" format 999999999
column "allocated size(MB)" format 999999999
column "maximum allowable (MB)" format 999999999
column "effective free(MB)" format 999999999
column "%FREE" format 999.99
select
a.tsdf tblspace,
b.used_space "Used Space(MB)",
nvl(a.file_space,0) "allocated size(MB)",
a.extn_space  as "maximum allowable (MB)",
(a.extn_space-b.used_space) as "effective free(MB)",
round((((a.extn_space-b.used_space)/a.extn_space)*100),2) as "%FREE"
from
(select tablespace_name tsdf,sum(bytes)/1048576
file_space,sum(decode(maxbytes,0,bytes,maxbytes))/1048576 extn_space from
dba_data_files group by tablespace_name) a, (select tablespace_name tsfs,
(bytes/1048576) used_space from sm$ts_used) b where
```

```
a.tsdf = b.tsfs (+) and
a.tsdf like upper('%%')
and round((((a.extn_space-b.used_space)/a.extn_space)*100),2) <= 25 and
(a.extn_space-b.used_space)<=10240
order by a.tsdf;
```

To add a datafile to a given tablespace manually

```
alter tablespace &tablespace_name add datafile '&filefullpath' size 200M autoextend
on next 20M maxsize 1800M;
```

```
alter tablespace INDEX1 add datafile
'/oradata12/ora816/SLA2/database/index1_26.dbf' size 1024M;
```

To resize a datafile of a given tablespace

For autoextensible file

```
alter database datafile '&filefullpath' autoextend on next 20M maxsize 1800M;
```

For static file without autoextension

```
alter database datafile '/SID/oradata/data01/a_txn_data01.dbf' resize 1800M;
```

Example

```
alter database datafile '/SID/oradata/data01/statsdata02.dbf' resize 1800M;
alter database datafile '/SID/oradata/data01/a_nolog01.dbf' autoextend on maxsize
120M;
alter database datafile '/SID/oradata/data03/ard198.dbf' autoextend off;
```

To find Non extendable datafiles

```
select file_name,bytes/1024/1024,maxbytes/1024/1024 from dba_data_files where
autoextensible='NO' order by file_name;
```

To enable autoextension

```
alter database datafile '<datafile>' autoextend on maxsize 1800m;
```

To backup the Control file after making a change / adding datafiles

```
alter database backup controlfile to trace;
```

Tablespaces with less than 25% of free space

```
set linesize 100 pagesize 30 feedback 1 echo off wrap on
column tblspace format a12 heading "Tablespace"
column "Used Space(MB)" format 999999999
column "allocated size(MB)" format 999999999
column "maximum allowable (MB)" format 999999999
column "effective free(MB)" format 999999999
column "%FREE" format 999.99
select
a.tsdf tblspace,
b.usedspace "Used Space(MB)",
nvl(a.filespace,0) "allocated size(MB)",
a.extnspace  as "maximum allowable (MB)",
(a.extnspace-b.usedspace) as "effective free(MB)",
round((((a.extnspace-b.usedspace)/a.extnspace)*100),2) as "%FREE"
from
(select tablespace_name tsdf,sum(bytes)/1048576
filespace,sum(decode(maxbytes,0,bytes,maxbytes))/1048576 extnspace from
dba_data_files group by tablespace_name) a,(select tablespace_name tsfs,
(bytes/1048576) usedspace from sm$ts_used) b where a.tsdf = b.tsfs (+) and a.tsdf
like upper('%%') and round((((a.extnspace-b.usedspace)/a.extnspace)*100),2) <= 25
order by a.tsdf;
and (a.extnspace-b.usedspace)<=1843
```

To check space for all the tablespaces

```
select TABLESPACE_NAME, (BYTES/1024)/1024 "Used Space(MB)",
total  "allocated size(MB)",
maxi "maximum allowable (MB)",
maxi-(BYTES/1024)/1024 "effectivefree(MB)",
--maxi-total "free(MB)",
round(((maxi-(BYTES/1024)/1024)/maxi)*100,2) "% FREE"
from
SM$TS_USED,(select sum((BYTES/1024)/1024)
total,sum((decode(MAXBYTES,0,bytes,maxbytes)/1024)/1024)  maxi from
dba_data_files where tablespace_name =tablespace_name) where
tablespace_name =tablespace_name;
```

To find size of Objects more than 2GB in a specific tablespace

```
select owner||'  ,  '||SEGMENT_NAME||'  , '||((BYTES/1024)/1024)/1024 from
dba_segments where tablespace_name='&tablespace_name' and
(((BYTES/1024)/1024)/1024)>2;
```

```
select owner||'  ,  '||SEGMENT_NAME||'  , '||((BYTES/1024)/1024)/1024 from
dba_segments where tablespace_name='&tablespace_name' and segment_name like
'%DTEA_PA_REPORTING_AGT_HISTO%';
```

To find LOBs

```
select OWNER, TABLE_NAME, COLUMN_NAME from dba_lobs where segment_NAME in (select *
from
(select SEGMENT_NAME from dba_segments where TABLESPACE_NAME='SYSAUX' and
segment_type =
'LOB PARTITION' order by 1 desc) where rownum<45);
```

Find List of Files - for Table spaces

```
select file_name,status,FILE_ID,bytes/1024/1024,maxbytes/1024/1024,autoextensible
from dba_data_files where tablespace_name=upper('&tablespace_name') order by
FILE_ID;
```

FIND FREE,UTILIZED,ALLOCATED

```
prompt Checking tablespace free space
SELECT
d.tablespace_name,
(a.bytes - NVL(f.bytes,0))/1024/1024 "Used Space(MB)",
a.bytes/1024/1024 "allocated size(MB)",
a.maxbytes/1024/1024 "maximum allowable (MB)",
(NVL(f.bytes,0) + (a.maxbytes - a.bytes))/1024/1024 "effective free(MB)",
100-round(((a.bytes - NVL(f.bytes,0))*100/a.maxbytes),2) "%FREE"
FROM
sys.dba_tablespaces d,
(select tablespace_name, sum(bytes) bytes,
sum(greatest(maxbytes,bytes)) maxbytes
from sys.dba_data_files group by tablespace_name) a,
(select tablespace_name,
sum(bytes) bytes
from sys.dba_free_space group by tablespace_name) f
WHERE d.tablespace_name = a.tablespace_name(+)
AND d.tablespace_name = f.tablespace_name(+)
AND NOT (d.extent_management = 'LOCAL' AND d.contents = 'TEMPORARY')
AND d.tablespace_name=upper('&TABLESPACE_NAME');
```

Checking Mb required to add tablespace

```
SELECT
d.tablespace_name,
a.maxbytes/1024/1024 "maximum allowable (MB)",(a.maxbytes/1024/1024)*0.15 "15% of
Max Allowable",
(NVL(f.bytes,0) + (a.maxbytes - a.bytes))/1024/1024 "effective free(MB)",
(((a.maxbytes/1024/1024)*0.15)-((NVL(f.bytes,0) + (a.maxbytes -
a.bytes))/1024/1024)) "Mb to add to maintain 15% free",
100-round(((a.bytes - NVL(f.bytes,0))*100/a.maxbytes),2) "%FREE"
FROM
sys.dba_tablespaces d,
(select tablespace_name, sum(bytes) bytes,
sum(greatest(maxbytes,bytes)) maxbytes
from sys.dba_data_files group by tablespace_name) a,
(select tablespace_name,
sum(bytes) bytes
from sys.dba_free_space group by tablespace_name) f
WHERE d.tablespace_name = a.tablespace_name(+)
AND d.tablespace_name = f.tablespace_name(+)
AND NOT (d.extent_management = 'LOCAL' AND d.contents = 'TEMPORARY')
AND d.tablespace_name=upper('&TABLESPACE_NAME');
```

```
set lines 152
col  file_name for a50
col TBS_NAME for a15
col "% FREE" FOR 999.99
select tablespace_name TBS_NAME,
(BYTES/1024)/1024 "Used(MB)",
total  "allocated size(MB)",
maxi "max allowable(MB)",
maxi-(BYTES/1024)/1024 "effect_free(MB)",
--maxi-total "free(MB)",
round(((maxi-(BYTES/1024)/1024)/maxi)*100,2) "% FREE"
from
SM$TS_USED,(select sum((BYTES/1024)/1024)
total,sum((decode(MAXBYTES,0,bytes,maxbytes)/1024)/1024)  maxi from
dba_data_files where tablespace_name in UPPER('&tablespace')) where
tablespace_name in upper('&tablespace');
```

```
select tablespace_name from dba_tablespaces where tablespace_name like
'%&TABLESPACE_NAME%';
```

FIND 25% LESS FREE SPACE

```
prompt Checking tablespace free space size less than 15% of total size
SELECT
d.tablespace_name,
(a.bytes - NVL(f.bytes,0))/1024/1024 "Used Space(MB)",
a.bytes/1024/1024 "allocated size(MB)",
a.maxbytes/1024/1024 "maximum allowable (MB)",
(NVL(f.bytes,0) + (a.maxbytes - a.bytes))/1024/1024 "effective free(MB)",
100-round(((a.bytes - NVL(f.bytes,0))*100/a.maxbytes),2) "%FREE"
FROM
sys.dba_tablespaces d,
(select tablespace_name, sum(bytes) bytes,
sum(greatest(maxbytes,bytes)) maxbytes
from sys.dba_data_files group by tablespace_name) a,
(select tablespace_name,
sum(bytes) bytes
from sys.dba_free_space group by tablespace_name) f
WHERE d.tablespace_name = a.tablespace_name(+)
AND d.tablespace_name = f.tablespace_name(+)
and (100-round(((a.bytes - NVL(f.bytes,0))*100/a.maxbytes),2)) <= 75
AND NOT (d.extent_management = 'LOCAL' AND d.contents = 'TEMPORARY');
```

To check freespace in the tablespace

```
select
a.tsdf tblspace,
a.file_space totspace,
nvl(b.free_space,0) tfsspace,
a.extn_space extspace,
round(((a.extn_space - (a.file_space - nvl(b.free_space,0)))/a.extn_space)*100,2)
pctfrees
from
(select tablespace_name tsdf,sum(bytes)/1048576
file_space,sum(decode(maxbytes,0,bytes,maxbytes))/1048576 extn_space
from dba_data_files group by tablespace_name) a,
(select tablespace_name tsfs,sum(bytes)/1048576 free_space,max(bytes)/1048576
max_chunk from dba_free_space group by
tablespace_name) b
where
a.tsdf = b.tsfs (+) and
a.tsdf like upper('%&TableSpace%')
and round(((a.extn_space - (a.file_space -
nvl(b.free_space,0)))/a.extn_space)*100,2) < &PcFree
order by a.tsdf;
```

```
set linesize 100 pagesize 30 feedback 1 echo off wrap on
column tblspace format a15 heading "Tablespace"
column totspace format 999999999 heading "Allocated MB"
column tfsspace format 999999999 heading "Free MB"
column extspace format 999999999 heading "AutoExt MB"
column pctfrees format 999.99 heading "% Free"
select
a.tsdf tblspace,
a.file_space totspace,
nvl(b.free_space,0) tfsspace,
a.extn_space extspace,
round(((a.extn_space - (a.file_space - nvl(b.free_space,0)))/a.extn_space)*100,2)
pctfrees
from
(select tablespace_name tsdf,sum(bytes)/1048576
file_space,sum(decode(maxbytes,0,bytes,maxbytes))/1048576 extn_space
from dba_data_files group by tablespace_name) a,
(select tablespace_name tsfs,sum(bytes)/1048576 free_space,max(bytes)/1048576
max_chunk from
dba_free_space group by
tablespace_name) b
where
a.tsdf = b.tsfs (+) and
a.tsdf like upper('%')
and round(((a.extn_space - (a.file_space -
nvl(b.free_space,0)))/a.extn_space)*100,2) < 25
order by a.tsdf;
```

```
SELECT
d.tablespace_name,
(a.bytes - NVL(f.bytes,0))/1024/1024 "Used Space(MB)",
a.bytes/1024/1024 "allocated size(MB)",
a.maxbytes/1024/1024 "maximum allowable (MB)",
(NVL(f.bytes,0) + (a.maxbytes - a.bytes))/1024/1024 "effective free(MB)",
100-round(((a.bytes - NVL(f.bytes,0))*100/a.maxbytes),2) "%FREE"
FROM
sys.dba_tablespaces d,
(select tablespace_name, sum(bytes) bytes,
sum(greatest(maxbytes,bytes)) maxbytes
from sys.dba_data_files group by tablespace_name) a,
(select tablespace_name,
sum(bytes) bytes
from sys.dba_free_space group by tablespace_name) f
WHERE d.tablespace_name = a.tablespace_name(+)
AND d.tablespace_name = f.tablespace_name(+)
AND NOT (d.extent_management = 'LOCAL' AND d.contents = 'TEMPORARY')
AND round(((NVL(f.bytes,0) + (a.maxbytes - a.bytes))/1048576),2) <2000
AND round(((a.bytes - NVL(f.bytes,0))*100/a.maxbytes),2) <75
order by 6 desc;
```

```
select TABLESPACE_NAME,sum(bytes)/1024/1024 from dba_free_space where
tablespace_name='&TABLESPACE_NAME';
```

To add DATAFILE in a Tablespace

```
alter tablespace &tablespace_name add datafile '&filefullpath' size 200M autoextend
on next 20M maxsize 1800M;
alter tablespace &tablespace_name add datafile '&filefullpath' size 2200M;
alter tablespace &tablespace_name add datafile '&filefullpath' size 1000M;
alter database datafile '&datafilename' autoextend on  maxsize 1800M;
ALTER DATABASE DATAFILE '&datafile' RESIZE 1000M;
alter database datafile '&filefullpath' autoextend on  maxsize 1800M;
alter database datafile '&filefullpath' autoextend on  maxsize 2000M;
alter database datafile '&filefullpath' autoextend on  maxsize 5000M;
alter database datafile '&filefullpath' autoextend on  maxsize 2048M;
alter database datafile '&filefullpath' resize 1800M;
```

To check the Count of Datafiles in a tablespace

```
select count(*) from dba_data_files  where tablespace_name='&tablespace_name';
```

To check the Count of datafiles which can still extend

```
select count(*) from dba_data_files  where maxbytes>=bytes and
tablespace_name='&tablespace_name';
```

```
select sum(maxbytes)/1024/1024 from dba_data_files  where
tablespace_name='&tablespace_name';
```

```
set lines 152
col  file_name for a70
select
(BYTES/1024)/1024 "Used Space(MB)",
total  "allocated size(MB)",
maxi "maximum allowable (MB)",
maxi-(BYTES/1024)/1024 "effectivefree(MB)",
--maxi-total "free(MB)",
round(((maxi-(BYTES/1024)/1024)/maxi)*100,2) "% FREE"
from
SM$TS_USED,(select sum((BYTES/1024)/1024)
total,sum((decode(MAXBYTES,0,bytes,maxbytes)/1024)/1024)  maxi from
dba_data_files where tablespace_name in UPPER('&&tbs')) where
tablespace_name in upper('&tbs');
```

```
select file_name,bytes/1024/1024,maxbytes/1024/1024,autoextensible
from dba_data_files where tablespace_name=upper('&tbs') order by file_name;
```

Changing maxsize for all the files whose size is set more than maxsize

```
set linesize 152
select 'alter  database datafile ' || ''''|| a.file_name||'''' || ' autoextend on
maxsize ' || a.maxsize || 'M;'
from
(select file_name,bytes/1024/1024 maxsize,maxbytes/1024/1024  ,autoextensible
from dba_data_files where tablespace_name in ('')
and maxbytes<bytes  and autoextensible='YES' order by file_name) a ;
```

Changing bytes for all the files which have size less than 20MB

```
set linesize 152
select 'alter  database datafile ' || ''''|| file_name||'''' || ' size 50M ;'
from dba_data_files
where bytes/1024/1024< 20 order by file_name ;
```

Percentage allocated in the tablespace

```
select (1-(sum(dfs.bytes)/sum(dbf.bytes)))*100  perallocated
from   dba_data_files dbf,
dba_free_space dfs
where  dbf.tablespace_name=dfs.tablespace_name
and    dbf.tablespace_name='&tablespace_name';
```

Freespace in tablespaces in MB

```
set lines 152
set pages 1000
```

```
select     utbs.tablespace_name,
round(utbs.mb) "Allocated Used/Unused MB",
round(Ftbs.mb) "Allocated_Free MB",
round((100/utbs.mb)*Ftbs.mb) "%Allocated_Free MB",
round(utbs.Maxmb-utbs.mb) "Space_AutoExtensible MB"
from
(select ddf.tablespace_name,sum(ddf.bytes)/1048576 MB,sum(ddf.maxbytes)/1048576
MaxMB
from dba_data_files ddf
group by ddf.tablespace_name) Utbs,
(select dfs.tablespace_name,sum(dfs.bytes)/1048576 MB
from dba_free_space dfs
group by dfs.tablespace_name) Ftbs
where utbs.tablespace_name=ftbs.tablespace_name
and (100*Ftbs.mb)/utbs.mb<25
order by (100*Ftbs.mb)/utbs.mb desc
/
```

2)

```
set linesize 100 pagesize 30 feedback 1 echo off wrap on
column tblspace format a12 heading "Tablespace"
column totspace format 999999999 heading "Allocated MB"
column tfsspace format 999999999 heading "Free MB"
column extspace format 999999999 heading "AutoExt MB"
column pctfrees format 999.99 heading "% Free"
select
a.tsdf tblspace,
a.file_space totspace,
nvl(b.free_space,0) tfsspace,
a.extn_space extspace,
round(((a.extn_space - (a.file_space - nvl(b.free_space,0)))/a.extn_space)*100,2)
pctfrees
from
(select tablespace_name tsdf,sum(bytes)/1048576
file_space,sum(decode(maxbytes,0,bytes,maxbytes))/1048576 extn_space
from dba_data_files group by tablespace_name) a,
(select tablespace_name tsfs,sum(bytes)/1048576 free_space,max(bytes)/1048576
max_chunk from dba_free_space group by
tablespace_name) b
where
a.tsdf = b.tsfs (+) and
a.tsdf like upper('%&TableSpace%')
and round(((a.extn_space - (a.file_space -
nvl(b.free_space,0)))/a.extn_space)*100,2) < &PcFree
order by a.tsdf;
```

To Find DB/Instance Name in RAC
================================

```
sql> show parameter cluster;


col host_name for a20
select inst_id,instance_name,host_name,instance_role from gv$instance;
select instance_name from v$instance;


To check from the operating system

# lsnodes



To get the NLS PARAMETER details
================================

set linesize 150
set pagesize 1000
col name for a50
col parameter for a30
col value for a30
col member for a50
col DATAFILE_NAME for a70
col TEMPFILE_NAME for a70
col CONTROLFILE_NAME for a70
col LOGFILE_MEMBER for a70


select * from NLS_DATABASE_PARAMETERS;


To check the Instance Uptime
================================

select instance_name||', up since '||
           to_char(startup_time,'DD-MON-YYYY HH24:MI:SS') start_time
    from gv$instance;


To Identify Space occupied and original space utilized of a database
====================================================================

set numf 999999.99
select sum(bytes)/(1024*1024*1024) "Total Allocated GB",Total_Consumed_GB from
dba_data_files,
(select sum(bytes)/(1024*1024*1024) Total_Consumed_GB from dba_segments) group by
Total_Consumed_GB;


To get the DATABASE SCHEMA SIZE from the database
===================================================

set linesize 150
set pagesize 5000
col owner for a15
col segment_name for a30
col segment_type for a20
col TABLESPACE_NAME for a30
clear breaks
clear computes
compute sum of SIZE_IN_GB on report
break on report
select OWNER,sum(bytes)/1024/1024/1000 "SIZE_IN_GB" from dba_segments group by
```

```
owner order by owner;
```

To get the LOGFILES INFORMATION from the database
==

```
col member for a65
select lf.MEMBER,l.GROUP#,THREAD#,SEQUENCE#,MEMBERS,bytes/1024/1024 "BYTES IN
MB",ARCHIVED,l.status from v$log l,v$logfile lf where l.GROUP#=lf.GROUP#
```

To get the CONTROLFILES INFORMATION from the database
===

```
col name for a90
select * from v$controlfile
```

To get the TEMPFILES SIZE from the database
===

```
select tablespace_name,file_name,bytes/1024/1024 "SIZE_IN_MB" from dba_temp_files
order by file_name asc;
```

To get the TABLESPACE SIZE from the database
==

```
select tablespace_name,sum(bytes)/1024/1024 "SIZE_IN_MB" from dba_data_files group
by tablespace_name;
```

To get the datafile sizes from the database
==

```
col file_name for a70
col tablespace_name for a30
clear breaks
clear computes
compute sum of SIZE_IN_MB on report
break on report
select tablespace_name,file_name,AUTOEXTENSIBLE,INCREMENT_BY,MAXBYTES/1024/1024
"MAX in MB",bytes/1024/1024 "SIZE_IN_MB" from dba_data_files order by
tablespace_name;
```

To get information about the database
======================================

```
SET LINESIZE 150
SET PAGESIZE 50000
cOL HOST_NAME FOR A25
col LOGINS FOR A20
col STATUS for A15
col "STARTUP_TIME" FOR A30
col INSTANCE_NAME for a20
col VERSION for a20
select INSTANCE_NAME,HOST_NAME,VERSION,LOGINS,STATUS,to_char(STARTUP_TIME,'DD-MON-
YYYY DAY HH24:MI:SS')
```

```
"STARTUP_TIME" FROM v$instance;




col name for a30
col "CREATED" for a25
col LOG_MODE for a15
col OPEN_MODE for a15
col DATABASE_ROLE for a15
select NAME,to_char(CREATED,'DD-MON-YYYY HH24:MI:SS')
"CREATED",LOG_MODE,OPEN_MODE,DATABASE_ROLE from v$database;
```

############33Procedure for generating the DDL statements of the database objects.
==

```
There are two parameters used in the specified object,

1. Object Type        ==> eg: FUNCTION,PROCEDURE,PACKAGE,TABLE,VIEW etc.,
2. Directory Name  ==> Specify the directory name where the output to be stored.

Usage -  DS_GET_DDL_STATEMENT(<Object type>,<Directory Name>)

Objects to be created:
----------------------
 CREATE TABLE "SMF_APEXRPS"."DS_ERROR_TABLE"
   (    "COMNAME" VARCHAR2(200 BYTE),
    "COMDDDL" LONG
   );


Create or Replace
PROCEDURE DS_GET_DDL_STATEMENT(P_OBJECTTYPE VARCHAR2,PDIR VARCHAR2)
AS
V_DDL LONG;
V_ERRMSG VARCHAR2(500);
vInHandle  utl_file.file_type;
CURSOR C1 IS
SELECT * FROM USER_OBJECTS WHERE object_type=P_OBJECTTYPE;
BEGIN
FOR I IN C1 LOOP
select
dbms_metadata.get_ddl(P_OBJECTTYPE,I.OBJECT_NAME)  INTO V_DDL
from dual;
  vInHandle := utl_file.fopen(PDIR, I.OBJECT_NAME||'.txt', 'W');
  IF utl_file.is_open(vInHandle) THEN
   utl_file.put_line(vInHandle, V_DDL, FALSE);
    utl_file.fflush(vInHandle);
    utl_file.fclose_all;
 END IF;
END LOOP;
EXCEPTION WHEN OTHERS THEN
V_ERRMSG:=SQLERRM;
INSERT INTO DS_ERROR_TABLE VALUES('ERROR',V_ERRMSG);
```

```
END;
```
Posted by Suresh Kumar at 11:08 PM No comments: Links to this post
Labels: DB Scripts
Monday, January 2, 2012Provides a report on the top segments (in bytes) grouped by Segment Type

```
-- +==================================================================+
-- PURPOSE: Provides a report on the top segments (in bytes) grouped by Segment
Type
-- +==================================================================+

SET LINESIZE 155
SET PAGESIZE 9999
SET VERIFY   OFF

BREAK ON segment_type SKIP 1
COMPUTE SUM OF bytes ON segment_type

COLUMN segment_type        FORMAT A20                HEADING 'Segment Type'
COLUMN owner               FORMAT A15                HEADING 'Owner'
COLUMN segment_name        FORMAT A30                HEADING 'Segment Name'
COLUMN partition_name      FORMAT A30                HEADING 'Partition Name'
COLUMN tablespace_name     FORMAT A20                HEADING 'Tablespace Name'
COLUMN bytes               FORMAT 9,999,999,999,999  HEADING 'Size (in bytes)'
COLUMN extents             FORMAT 999,999,999        HEADING 'Extents'

SELECT
    a.segment_type      segment_type
  , a.owner             owner
  , a.segment_name      segment_name
  , a.partition_name    partition_name
  , a.tablespace_name   tablespace_name
  , a.bytes             bytes
  , a.extents           extents
FROM
    (select
         b.segment_type
       , b.owner
       , b.segment_name
       , b.partition_name
       , b.tablespace_name
       , b.bytes
       , b.extents
     from
         dba_segments b
     order by
         b.bytes desc
    ) a
WHERE
    rownum < 101
ORDER BY
    segment_type, bytes desc, owner, segment_name
/
```
Posted by Suresh Kumar at 5:32 PM No comments: Links to this post
Labels: DB Scripts
To check for index fragmentation

```
 -- +--------------------------------------------------------------------------------
+
-- | PURPOSE  : To check for index fragmentation. As a rule of thumb if 10-15%  |
-- |            of the table data changes, then you should consider rebuilding the
```

```
index  |
-- +--------------------------------------------------------------------------------
+

ANALYZE INDEX &&index_name VALIDATE STRUCTURE;

COL name         HEADING 'Index Name'          FORMAT a30
COL del_lf_rows  HEADING 'Deleted|Leaf Rows'   FORMAT 99999999
COL lf_rows_used HEADING 'Used|Leaf Rows'      FORMAT 99999999
COL ibadness     HEADING '% Deleted|Leaf Rows' FORMAT 999.99999

SELECT
    name
  , del_lf_rows
  , lf_rows - del_lf_rows lf_rows_used
  , TO_CHAR( del_lf_rows /(DECODE(lf_rows,0,0.01,lf_rows))*100,'999.99999')
ibadness
FROM   index_stats
/

prompt
prompt Consider rebuilding any index if % of Deleted Leaf Rows is > 20%
prompt

undefine index_name
```

Posted by Suresh Kumar at 5:30 PM No comments: Links to this post
Labels: DB Scripts
Report free space fragmentation

```
-- +------------------------------------------------------------------------------+
-- | PURPOSE  : Report free space fragmentation.
|
-- |           THIS SCRIPT MUST BE RUN AS THE SYS USER!!!                 |
-- +------------------------------------------------------------------------------+

connect / as sysdba

CREATE OR REPLACE VIEW free_space (
    tablespace
  , pieces
  , free_bytes
  , free_blocks
  , largest_bytes
  , largest_blks
  , fsfi
  , data_file
  , file_id
  , total_blocks
)
AS
SELECT
    a.tablespace_name
  , COUNT(*)
  , SUM(a.bytes)
  , SUM(a.blocks)
  , MAX(a.bytes)
  , MAX(a.blocks)
  , SQRT(MAX(a.blocks)/SUM(a.blocks))*(100/SQRT(SQRT(count(a.blocks))))
  , UPPER(b.file_name)
  , MAX(a.file_id)
  , MAX(b.blocks)
FROM
    sys.dba_free_space  a
  , sys.dba_data_files  b
WHERE
    a.file_id = b.file_id
GROUP BY
    a.tablespace_name,  b.file_name
/

CLEAR COLUMNS
SET LINESIZE  120
SET PAGESIZE  9999
SET FEEDBACK  off
SET VERIFY    off

BREAK ON tablespace SKIP 2 ON REPORT

COMPUTE SUM OF  total_blocks  ON tablespace
COMPUTE SUM OF  free_blocks   ON tablespace
COMPUTE SUM OF  free_blocks   ON report
COMPUTE SUM OF  total_blocks  ON report

COLUMN tablespace     HEADING "Tablespace"    FORMAT a15
COLUMN file_id        HEADING File#           FORMAT 99999
COLUMN pieces         HEADING Frag            FORMAT 9999
COLUMN free_bytes     HEADING 'Free Byte'
COLUMN free_blocks    HEADING 'Free Blk'      FORMAT 999,999,999
COLUMN largest_bytes  HEADING 'Biggest Bytes'
COLUMN largest_blks   HEADING 'Biggest Blks'  FORMAT 999,999,999
COLUMN data_file      HEADING 'File Name'     FORMAT a45
COLUMN total_blocks   HEADING 'Total Blocks'  FORMAT 999,999,999


SELECT
    tablespace
  , data_file
  , pieces
  , free_blocks
  , largest_blks
  , file_id
  , total_blocks
FROM
    free_space
/


DROP VIEW free_space
/
```

Posted by Suresh Kumar at 4:44 PM No comments: Links to this post
Labels: DB Scripts
List all currently connected user sessions ordered by current PGA size

```
--
+----------------------------------------------------------------------------------
---+
-- | PURPOSE  : List all currently connected user sessions ordered by current PGA
```

```
size  |
--
+------------------------------------------------------------------------------
---+

SET LINESIZE 145
SET PAGESIZE 9999

COLUMN sid                     FORMAT 99999          HEADING 'SID'
COLUMN serial_id               FORMAT 999999         HEADING 'Serial#'
COLUMN session_status          FORMAT a9             HEADING 'Status'
JUSTIFY right
COLUMN oracle_username         FORMAT a12            HEADING 'Oracle User'
JUSTIFY right
COLUMN os_username             FORMAT a9             HEADING 'O/S User'
JUSTIFY right
COLUMN os_pid                  FORMAT 9999999        HEADING 'O/S PID'
JUSTIFY right
COLUMN session_program         FORMAT a18            HEADING 'Session Program'
TRUNC
COLUMN session_machine         FORMAT a8             HEADING 'Machine'
JUSTIFY right TRUNC
COLUMN session_pga_memory      FORMAT 9,999,999,999  HEADING 'PGA Memory'
COLUMN session_pga_memory_max  FORMAT 9,999,999,999  HEADING 'PGA Memory Max'
COLUMN session_uga_memory      FORMAT 9,999,999,999  HEADING 'UGA Memory'
COLUMN session_uga_memory_max  FORMAT 9,999,999,999  HEADING 'UGA Memory MAX'

prompt
prompt +----------------------------------------------------+
prompt | User Sessions Ordered by Current PGA Size          |
prompt +----------------------------------------------------+

SELECT
    s.sid                sid
  , s.serial#            serial_id
  , lpad(s.status,9)     session_status
  , lpad(s.username,12)  oracle_username
  , lpad(s.osuser,9)     os_username
  , lpad(p.spid,7)       os_pid
  , s.program            session_program
  , lpad(s.machine,8)    session_machine
  , sstat1.value         session_pga_memory
  , sstat2.value         session_pga_memory_max
  , sstat3.value         session_uga_memory
  , sstat4.value         session_uga_memory_max
FROM
    v$process  p
  , v$session  s
  , v$sesstat  sstat1
  , v$sesstat  sstat2
  , v$sesstat  sstat3
  , v$sesstat  sstat4
  , v$statname statname1
  , v$statname statname2
  , v$statname statname3
  , v$statname statname4
WHERE
      p.addr (+)            = s.paddr
  AND s.sid                 = sstat1.sid
  AND s.sid                 = sstat2.sid
  AND s.sid                 = sstat3.sid
  AND s.sid                 = sstat4.sid
  AND statname1.statistic#  = sstat1.statistic#
  AND statname2.statistic#  = sstat2.statistic#
  AND statname3.statistic#  = sstat3.statistic#
  AND statname4.statistic#  = sstat4.statistic#
  AND statname1.name        = 'session pga memory'
  AND statname2.name        = 'session pga memory max'
  AND statname3.name        = 'session uga memory'
  AND statname4.name        = 'session uga memory max'
ORDER BY session_pga_memory DESC
/
```

Posted by Suresh Kumar at 4:42 PM No comments: Links to this post
Labels: DB Scripts
List all currently connected user sessions ordered by Logical I/O

```
--
+----------------------------------------------------------------------------+
-- | PURPOSE  : List all currently connected user sessions ordered by Logical - I/O
|
--
+----------------------------------------------------------------------------+

SET LINESIZE 145
SET PAGESIZE 9999

COLUMN sid               FORMAT 99999            HEADING 'SID'
COLUMN serial_id         FORMAT 999999           HEADING 'Serial#'
COLUMN session_status    FORMAT a9               HEADING 'Status'          JUSTIFY
right
COLUMN oracle_username   FORMAT a12              HEADING 'Oracle User'     JUSTIFY
right
COLUMN os_username       FORMAT a9               HEADING 'O/S User'        JUSTIFY
right
COLUMN os_pid            FORMAT 9999999          HEADING 'O/S PID'         JUSTIFY
right
COLUMN session_program   FORMAT a18              HEADING 'Session Program' TRUNC
COLUMN session_machine   FORMAT a8               HEADING 'Machine'         JUSTIFY
right TRUNC
COLUMN logical_io        FORMAT 999,999,999,999  HEADING 'Logical I/O'
COLUMN physical_reads    FORMAT 999,999,999,999  HEADING 'Physical Reads'
COLUMN physical_writes   FORMAT 999,999,999,999  HEADING 'Physical Writes'

prompt
prompt +------------------------------------------------------+
prompt | User Sessions Ordered by Logical I/O                 |
prompt +------------------------------------------------------+

SELECT
    s.sid                sid
  , s.serial#            serial_id
  , lpad(s.status,9)     session_status
  , lpad(s.username,12)  oracle_username
  , lpad(s.osuser,9)     os_username
  , lpad(p.spid,7)       os_pid
  , s.program            session_program
  , lpad(s.machine,8)    session_machine
  , sstat1.value
    + sstat2.value         logical_io
  , sstat3.value         physical_reads
  , sstat4.value         physical_writes
FROM
    v$process  p
  , v$session  s
  , v$sesstat  sstat1
  , v$sesstat  sstat2
  , v$sesstat  sstat3
  , v$sesstat  sstat4
  , v$statname statname1
  , v$statname statname2
  , v$statname statname3
  , v$statname statname4
WHERE
      p.addr (+)            = s.paddr
  AND s.sid                 = sstat1.sid
  AND s.sid                 = sstat2.sid
  AND s.sid                 = sstat3.sid
  AND s.sid                 = sstat4.sid
  AND statname1.statistic#  = sstat1.statistic#
  AND statname2.statistic#  = sstat2.statistic#
  AND statname3.statistic#  = sstat3.statistic#
  AND statname4.statistic#  = sstat4.statistic#
  AND statname1.name        = 'db block gets'
  AND statname2.name        = 'consistent gets'
  AND statname3.name        = 'physical reads'
  AND statname4.name        = 'physical writes'
ORDER BY logical_io DESC
/
```

Posted by Suresh Kumar at 4:38 PM No comments: Links to this post
Labels: DB Scripts
List all currently connected user sessions ordered by CPU time.

```
-- +----------------------------------------------------------------------------+
-- | DATABASE : Oracle |
-- | PURPOSE : List all currently connected user sessions ordered by CPU time. |
-- +----------------------------------------------------------------------------+

SET LINESIZE 145
SET PAGESIZE 9999

COLUMN sid FORMAT 99999 HEADING 'SID'
COLUMN serial_id FORMAT 999999 HEADING 'Serial#'
COLUMN session_status FORMAT a9 HEADING 'Status' JUSTIFY right
COLUMN oracle_username FORMAT a12 HEADING 'Oracle User' JUSTIFY right
COLUMN os_username FORMAT a9 HEADING 'O/S User' JUSTIFY right
COLUMN os_pid FORMAT 9999999 HEADING 'O/S PID' JUSTIFY right
COLUMN session_program FORMAT a20 HEADING 'Session Program' TRUNC
COLUMN session_machine FORMAT a14 HEADING 'Machine' JUSTIFY right TRUNC
COLUMN cpu_value FORMAT 999,999,999,999 HEADING 'CPU'

prompt
prompt +------------------------------------------------------+
prompt | User Sessions Ordered by CPU |
prompt +------------------------------------------------------+

SELECT
s.sid sid
, s.serial# serial_id
, lpad(s.status,9) session_status
, lpad(s.username,12) oracle_username
, lpad(s.osuser,9) os_username
, lpad(p.spid,7) os_pid
, s.program session_program
, lpad(s.machine,14) session_machine
, sstat.value cpu_value
FROM
v$process p
, v$session s
, v$sesstat sstat
, v$statname statname
WHERE
p.addr (+) = s.paddr
AND s.sid = sstat.sid
AND statname.statistic# = sstat.statistic#
AND statname.name = 'CPU used by this session'
ORDER BY cpu_value DESC
/
```