## Audit

### Create

<https://docs.oracle.com/database/121/SQLRF/statements_5001.htm#BGEGGCBI>

CREATE AUDIT POLICY policy

[ privilege\_audit\_clause ] [ action\_audit\_clause ] [ role\_audit\_clause ]

[ WHEN 'audit\_condition' EVALUATE PER {STATEMENT | SESSION | INSTANCE } ]

[ CONTAINER = { ALL | CURRENT } ] ;

EXAMPLE: CREATE AUDIT POLICY table\_pol PRIVILEGES CREATE ANY TABLE, DROP ANY TABLE;

### View

#### Audit policies

SELECT \* FROM AUDIT\_UNIFIED\_POLICIES; -- danh sách Audit đã tạo.

SELECT \* FROM AUDIT\_UNIFIED\_ENABLED\_POLICIES; -- danh sách Audit đang áp dụng.

#### Audit record

SELECT OS\_USERNAME, USERHOST, DBUSERNAME, CLIENT\_PROGRAM\_NAME, EVENT\_TIMESTAMP, ACTION\_NAME, OS\_PROCESS, OBJECT\_SCHEMA, OBJECT\_NAME, SQL\_TEXT, SQL\_BINDS, AUTHENTICATION\_TYPE

FROM UNIFIED\_AUDIT\_TRAIL

WHERE -- OBJECT\_NAME = 'WFINSTRUMENTTABLE' AND

EVENT\_TIMESTAMP > '06-JAN-24'

ORDER BY EVENT\_TIMESTAMP DESC;

SELECT OS\_USER, HOST\_NAME, USERID, CLIENT\_PROGRAM\_NAME, EVENT\_TIMESTAMP, OBJ\_OWNER, OBJ\_NAME, SQL\_TEXT, SQL\_BINDS, UNIFIED\_AUDIT\_POLICIES

FROM AUDSYS.AUD$UNIFIED

WHERE obj\_owner != 'SYS'

AND (TRANSACTION\_ID IS NULL OR TRANSACTION\_ID='0000000000000000')

ORDER BY EVENT\_TIMESTAMP DESC;

## roles and privileges

### Create role and Grant, Revoke

### Get detail

DEFINE LV\_USERNAME = 'PTF\_CHUONGLN';

SELECT \* FROM DBA\_ROLE\_PRIVS

WHERE GRANTEE = '&LV\_USERNAME';

SELECT \* FROM DBA\_TAB\_PRIVS

WHERE GRANTEE = '&LV\_USERNAME';

SELECT \* FROM DBA\_SYS\_PRIVS

WHERE GRANTEE = '&LV\_USERNAME';

## Index

### Unusable index

SELECT INDEX\_NAME, PARTITION\_NAME, STATUS

FROM DBA\_IND\_PARTITIONS IDXP

WHERE IDXP.STATUS != 'USABLE'

UNION ALL

SELECT INDEX\_NAME, TABLE\_NAME, STATUS

FROM DBA\_INDEXES IDX

WHERE IDX.PARTITIONED = 'NO'

AND IDX.STATUS NOT IN ('N/A','VALID');

KQ: No rows selected

## Object

### Invalid object

SELECT OWNER, OBJECT\_TYPE, OBJECT\_NAME, STATUS

FROM DBA\_OBJECTS

WHERE STATUS = 'INVALID';

KQ: No rows selected

## Job/ Task

### Auto task

SELECT \* FROM DBA\_autotask\_client;

--Các job cần thực hiện enable. Check job chạy fail.

## LOCK CONFLICT

### List blocking session

SELECT 'Alter System Kill Session ''' || SID || ',' || S.SERIAL# || ',@' || INST\_ID || ''';' AS "Kill Scripts",

SID, USERNAME, SERIAL#, PROCESS,

NVL(SQL\_ID,0), BLOCKING\_SESSION, WAIT\_CLASS,

EVENT, SECONDS\_IN\_WAIT

FROM GV$SESSION S

WHERE BLOCKING\_SESSION\_STATUS = 'VALID' OR

SID IN ( SELECT BLOCKING\_SESSION

FROM GV$SESSION

WHERE BLOCKING\_SESSION\_STATUS = 'VALID'

);

### Check blocking session tree

SELECT LEVEL, LPAD(' ',(LEVEL - 1) \* 2,' ') || NVL(S.USERNAME,'(oracle)') AS USERNAME,

S.OSUSER, S.SID, S.SERIAL#,

S.LOCKWAIT, S.STATUS, S.MODULE,

S.MACHINE, S.PROGRAM,

TO\_CHAR(S.LOGON\_TIME,'DD-MON-YYYY HH24:MI:SS') AS LOGON\_TIME

FROM V$SESSION S

WHERE LEVEL > 1 OR

EXISTS ( SELECT 1

FROM V$SESSION

WHERE BLOCKING\_SESSION = S.SID

)

CONNECT BY PRIOR S.SID = S.BLOCKING\_SESSION

START WITH S.BLOCKING\_SESSION IS NULL;

## OTHER

### Find current running sqls

SELECT SESION.SID, SESION.USERNAME, OPTIMIZER\_MODE,

HASH\_VALUE, ADDRESS, CPU\_TIME,

ELAPSED\_TIME, SQL\_TEXT

FROM V$SQLAREA SQLAREA,

V$SESSION SESION

WHERE SESION.SQL\_HASH\_VALUE = SQLAREA.HASH\_VALUE AND

SESION.SQL\_ADDRESS = SQLAREA.ADDRESS AND

SESION.USERNAME IS NOT NULL;

### Find active sessions in oracle database

**set echo off**  
**set linesize 95**  
**set head on**  
**set feedback on**  
**col sid head "Sid" form 9999 trunc**  
**col serial# form 99999 trunc head "Ser#"**  
**col username form a8 trunc**  
**col osuser form a7 trunc**  
**col machine form a20 trunc head "Client|Machine"**  
**col program form a15 trunc head "Client|Program"**  
**col login form a11**  
**col "last call" form 9999999 trunc head "Last Call|In Secs"**  
**col status form a6 trunc**  
SELECT SID,

SERIAL#,

SUBSTR(USERNAME,1,10) USERNAME,

SUBSTR(OSUSER,1,10) OSUSER,

SUBSTR(PROGRAM || MODULE,1,15) PROGRAM,

SUBSTR(MACHINE,1,22) MACHINE,

TO\_CHAR(LOGON\_TIME,'ddMon hh24:mi') LOGIN,

LAST\_CALL\_ET "last call"

FROM GV$SESSION

WHERE STATUS = 'ACTIVE'

ORDER BY 1  
**/**

### Find sessions generating undo

**select a.sid, a.serial#, a.username, b.used\_urec used\_undo\_record, b.used\_ublk used\_undo\_blocks**  
**from v$session a, v$transaction b**  
**where a.saddr=b.ses\_addr ;**

### Find the temp usage of sessions

SELECT B.TABLESPACE,

ROUND(((B.BLOCKS \* P.VALUE) / 1024 / 1024),2) || 'M' AS TEMP\_SIZE,

A.INST\_ID AS INSTANCE, A.SID || ',' || A.SERIAL# AS SID\_SERIAL,

NVL(A.USERNAME,'(oracle)') AS USERNAME,

A.PROGRAM, A.STATUS, A.SQL\_ID

FROM GV$SESSION A,

GV$SORT\_USAGE B,

GV$PARAMETER P

WHERE P.NAME = 'db\_block\_size' AND

A.SADDR = B.SESSION\_ADDR AND

A.INST\_ID = B.INST\_ID AND

A.INST\_ID = P.INST\_ID

ORDER BY TEMP\_SIZE DESC

### Find sessions generating lot of redo

**set lines 2000**  
**set pages 1000**  
**col sid for 99999**  
**col name for a09**  
**col username for a14**  
**col PROGRAM for a21**  
**col MODULE for a25**  
SELECT S.SID, SN.SERIAL#, N.NAME,

ROUND(VALUE / 1024 / 1024,2) REDO\_MB,

SN.USERNAME, SN.STATUS, SUBSTR(SN.PROGRAM,1,21) "program",

SN.TYPE, SN.MODULE, SN.SQL\_ID

FROM V$SESSTAT S

JOIN V$STATNAME N ON N.STATISTIC# = S.STATISTIC#

JOIN V$SESSION SN ON SN.SID = S.SID

WHERE N.NAME LIKE 'redo size' AND

S.VALUE != 0

ORDER BY REDO\_MB DESC

### Get size of the database

**col "Database Size" format a20**  
**col "Free space" format a20**  
**col "Used space" format a20**  
SELECT ROUND(SUM(USED.BYTES) / 1024 / 1024 / 1024) || ' GB' "Database Size",

ROUND(SUM(USED.BYTES) / 1024 / 1024 / 1024) - ROUND(FREE.P / 1024 / 1024 / 1024) || ' GB' "Used space",

ROUND(FREE.P / 1024 / 1024 / 1024) || ' GB' "Free space"

FROM ( SELECT BYTES

FROM V$DATAFILE

UNION ALL

SELECT BYTES

FROM V$TEMPFILE

UNION ALL

SELECT BYTES

FROM V$LOG

) USED,

( SELECT SUM(BYTES) AS P

FROM DBA\_FREE\_SPACE

) FREE

GROUP BY FREE.P

### Monitor tablespace usage

**set feedback off**  
**set pagesize 70;**  
**set linesize 2000**  
**set head on**  
**COLUMN Tablespace format a25 heading 'Tablespace Name'**  
**COLUMN autoextensible format a11 heading 'AutoExtend'**  
**COLUMN files\_in\_tablespace format 999 heading 'Files'**  
**COLUMN total\_tablespace\_space format 99999999 heading 'TotalSpace'**  
**COLUMN total\_used\_space format 99999999 heading 'UsedSpace'**  
**COLUMN total\_tablespace\_free\_space format 99999999 heading 'FreeSpace'**  
**COLUMN total\_used\_pct format 9999 heading '%Used'**  
**COLUMN total\_free\_pct format 9999 heading '%Free'**  
**COLUMN max\_size\_of\_tablespace format 99999999 heading 'ExtendUpto'**  
**COLUM total\_auto\_used\_pct format 999.99 heading 'Max%Used'**  
**COLUMN total\_auto\_free\_pct format 999.99 heading 'Max%Free'**  
**WITH tbs\_auto AS**  
**(SELECT DISTINCT tablespace\_name, autoextensible**  
**FROM dba\_data\_files**  
**WHERE autoextensible = 'YES'),**  
**files AS**  
**(SELECT tablespace\_name, COUNT (\*) tbs\_files,**  
**SUM (BYTES/1024/1024) total\_tbs\_bytes**  
**FROM dba\_data\_files**  
**GROUP BY tablespace\_name),**  
**fragments AS**  
**(SELECT tablespace\_name, COUNT (\*) tbs\_fragments,**  
**SUM (BYTES)/1024/1024 total\_tbs\_free\_bytes,**  
**MAX (BYTES)/1024/1024 max\_free\_chunk\_bytes**  
**FROM dba\_free\_space**  
**GROUP BY tablespace\_name),**  
**AUTOEXTEND AS**  
**(SELECT tablespace\_name, SUM (size\_to\_grow) total\_growth\_tbs**  
**FROM (SELECT tablespace\_name, SUM (maxbytes)/1024/1024 size\_to\_grow**  
**FROM dba\_data\_files**  
**WHERE autoextensible = 'YES'**  
**GROUP BY tablespace\_name**  
**UNION**  
**SELECT tablespace\_name, SUM (BYTES)/1024/1024 size\_to\_grow**  
**FROM dba\_data\_files**  
**WHERE autoextensible = 'NO'**  
**GROUP BY tablespace\_name)**  
**GROUP BY tablespace\_name)**  
**SELECT c.instance\_name,a.tablespace\_name Tablespace,**  
**CASE tbs\_auto.autoextensible**  
**WHEN 'YES'**  
**THEN 'YES'**  
**ELSE 'NO'**  
**END AS autoextensible,**  
**files.tbs\_files files\_in\_tablespace,**  
**files.total\_tbs\_bytes total\_tablespace\_space,**  
**(files.total\_tbs\_bytes - fragments.total\_tbs\_free\_bytes**  
**) total\_used\_space,**  
**fragments.total\_tbs\_free\_bytes total\_tablespace\_free\_space,**  
**round(( ( (files.total\_tbs\_bytes - fragments.total\_tbs\_free\_bytes)**  
**/ files.total\_tbs\_bytes**  
**)**  
**\* 100**  
**)) total\_used\_pct,**  
**round(((fragments.total\_tbs\_free\_bytes / files.total\_tbs\_bytes) \* 100**  
**)) total\_free\_pct**  
**FROM dba\_tablespaces a,v$instance c , files, fragments, AUTOEXTEND, tbs\_auto**  
**WHERE a.tablespace\_name = files.tablespace\_name**  
**AND a.tablespace\_name = fragments.tablespace\_name**  
**AND a.tablespace\_name = AUTOEXTEND.tablespace\_name**  
**AND a.tablespace\_name = tbs\_auto.tablespace\_name(+)**  
**order by total\_free\_pct;**

### Monitor undo tablespace usage

**select a.tablespace\_name, SIZEMB, USAGEMB, (SIZEMB - USAGEMB) FREEMB**  
**from (select sum(bytes) / 1024 / 1024 SIZEMB, b.tablespace\_name**  
**from dba\_data\_files a, dba\_tablespaces b**  
**where a.tablespace\_name = b.tablespace\_name**  
**and b.contents = 'UNDO'**  
**group by b.tablespace\_name) a,**  
**(select c.tablespace\_name, sum(bytes) / 1024 / 1024 USAGEMB**  
**from DBA\_UNDO\_EXTENTS c**  
**where status <> 'EXPIRED'**  
**group by c.tablespace\_name) b**  
**where a.tablespace\_name = b.tablespace\_name;**

### Monitor TEMP tablespace usage

**select a.tablespace\_name tablespace,**  
**d.TEMP\_TOTAL\_MB,**  
**sum (a.used\_blocks \* d.block\_size) / 1024 / 1024 TEMP\_USED\_MB,**  
**d.TEMP\_TOTAL\_MB - sum (a.used\_blocks \* d.block\_size) / 1024 / 1024 TEMP\_FREE\_MB**  
**from v$sort\_segment a,**  
**(**  
**select b.name, c.block\_size, sum (c.bytes) / 1024 / 1024 TEMP\_TOTAL\_MB**  
**from v$tablespace b, v$tempfile c**  
**where b.ts#= c.ts#**  
**group by b.name, c.block\_size**  
**) d**  
**where a.tablespace\_name = d.name**  
**group by a.tablespace\_name, d.TEMP\_TOTAL\_MB;**

### Find blocking sessions

**SELECT**  
**s.inst\_id,**  
**s.blocking\_session,**  
**s.sid,**  
**s.serial#,**  
**s.seconds\_in\_wait**  
**FROM**  
**gv$session s**  
**WHERE**  
**blocking\_session IS NOT NULL;**

### Find long running operations

**select sid,inst\_id,opname,totalwork,sofar,start\_time,time\_remaining**  
**from gv$session\_longops**  
**where totalwork<>sofar**  
**/**

### Find locks present in database

col session\_id head 'Sid' form 9999  
col object\_name head "Table|Locked" form a30  
col oracle\_username head "Oracle|Username" form a10 truncate   
col os\_user\_name head "OS|Username" form a10 truncate   
col process head "Client|Process|ID" form 99999999  
col mode\_held form a15  
select lo.session\_id,lo.oracle\_username,lo.os\_user\_name,  
lo.process,do.object\_name,  
decode(lo.locked\_mode,0, 'None',1, 'Null',2, 'Row Share (SS)',  
3, 'Row Excl (SX)',4, 'Share',5, 'Share Row Excl (SSX)',6, 'Exclusive',  
to\_char(lo.locked\_mode)) mode\_held  
from v$locked\_object lo, dba\_objects do  
where lo.object\_id = do.object\_id  
order by 1,5  
/

### Find queries triggered from a procedure

-- Below script will provide the dependent queries getting triggered from a procedure.

**SELECT s.sql\_id, s.sql\_text**   
**FROM gv$sqlarea s JOIN dba\_objects o ON s.program\_id = o.object\_id**   
**and o.object\_name = '&procedure\_name';**

### Get sid from os pid

**- Get sid from os pid ( server process)**

**col sid format 999999**  
**col username format a20**  
**col osuser format a15**  
**select b.spid,a.sid, a.serial#,a.username, a.osuser**  
**from v$session a, v$process b**  
**where a.paddr= b.addr**  
**and b.spid='&spid'**  
**order by b.spid;**

### kill all session of a user

**BEGIN**  
**FOR r IN (select sid,serial# from v$session where username = 'TEST\_ANB')**  
**LOOP**  
**EXECUTE IMMEDIATE 'alter system kill session ''' || r.sid**   
**|| ',' || r.serial# || '''';**  
**END LOOP;**  
**END;**  
**/**

### Kill all sessions of a sql\_id

**select 'alter system kill session ' ||''''||SID||','||SERIAL#||' immediate ;' from v$session**   
**where sql\_id='&sql\_id';**

**FOR RAC**

**select 'alter system kill session ' ||''''||SID||','||SERIAL#||',@'||inst\_id||''''||' immediate ;'**   
**from gv$session where sql\_id='&sql\_id'**

### get parallel query detail

**col username for a9**  
**col sid for a8**  
**set lines 299**  
**select**  
**s.inst\_id,**  
**decode(px.qcinst\_id,NULL,s.username,**  
**' - '||lower(substr(s.program,length(s.program)-4,4) ) ) "Username",**  
**decode(px.qcinst\_id,NULL, 'QC', '(Slave)') "QC/Slave" ,**  
**to\_char( px.server\_set) "Slave Set",**  
**to\_char(s.sid) "SID",**  
**decode(px.qcinst\_id, NULL ,to\_char(s.sid) ,px.qcsid) "QC SID",**  
**px.req\_degree "Requested DOP",**  
**px.degree "Actual DOP", p.spid**  
**from**  
**gv$px\_session px,**  
**gv$session s, gv$process p**  
**where**  
**px.sid=s.sid (+) and**  
**px.serial#=s.serial# and**  
**px.inst\_id = s.inst\_id**  
**and p.inst\_id = s.inst\_id**  
**and p.addr=s.paddr**  
**order by 5 , 1 desc**  
**/**

### Kill snipped session in db

**It will generate kill session statements for all snipped sessions:**

**select 'alter system kill session '''||sid||','||serial#||''' immediate;' from v$session where status='SNIPED' ;**

### Top Query with high elapsed time

**Queries in last 1 hour ( Run from Toad, for proper view)**

**Select module,parsing\_schema\_name,inst\_id,sql\_id,CHILD\_NUMBER,sql\_plan\_baseline,sql\_profile,plan\_hash\_value,sql\_fulltext,**

**to\_char(last\_active\_time,'DD/MM/YY HH24:MI:SS' ),executions, elapsed\_time/executions/1000/1000,**

**rows\_processed,sql\_plan\_baseline from gv$sql where last\_active\_time>sysdate-1/24**

**and executions <> 0 order by elapsed\_time/executions desc**

### Monitor parallel queries

**select**  
**s.inst\_id,**  
**decode(px.qcinst\_id,NULL,s.username,**  
**' - '||lower(substr(s.program,length(s.program)-4,4) ) ) "Username",**  
**decode(px.qcinst\_id,NULL, 'QC', '(Slave)') "QC/Slave" ,**  
**to\_char( px.server\_set) "Slave Set",**  
**to\_char(s.sid) "SID",**  
**decode(px.qcinst\_id, NULL ,to\_char(s.sid) ,px.qcsid) "QC SID",**  
**px.req\_degree "Requested DOP",**  
**px.degree "Actual DOP", p.spid**  
**from**  
**gv$px\_session px,**  
**gv$session s, gv$process p**  
**where**  
**px.sid=s.sid (+) and**  
**px.serial#=s.serial# and**  
**px.inst\_id = s.inst\_id**  
**and p.inst\_id = s.inst\_id**  
**and p.addr=s.paddr**  
**order by 5 , 1 desc**

### Find the locked objects

**SET PAGESIZE 1000**  
**SET VERIFY OFF**

**COLUMN owner FORMAT A20**  
**COLUMN username FORMAT A20**  
**COLUMN object\_owner FORMAT A20**  
**COLUMN object\_name FORMAT A30**  
**COLUMN locked\_mode FORMAT A15**

**SELECT b.inst\_id,**  
**b.session\_id AS sid,**  
**NVL(b.oracle\_username, '(oracle)') AS username,**  
**a.owner AS object\_owner,**  
**a.object\_name,**  
**Decode(b.locked\_mode, 0, 'None',**  
**1, 'Null (NULL)',**  
**2, 'Row-S (SS)',**  
**3, 'Row-X (SX)',**  
**4, 'Share (S)',**  
**5, 'S/Row-X (SSX)',**  
**6, 'Exclusive (X)',**  
**b.locked\_mode) locked\_mode,**  
**b.os\_user\_name**  
**FROM dba\_objects a,**  
**gv$locked\_object b**  
**WHERE a.object\_id = b.object\_id**  
**ORDER BY 1, 2, 3, 4;**

**SET PAGESIZE 14**  
**SET VERIFY ON**

### Check open cursors

**Current open cursor**

**select a.value, s.username, s.sid, s.serial#**  
**from v$sesstat a, v$statname b, v$session s**  
**where a.statistic# = b.statistic# and s.sid=a.sid**  
**and b.name = 'opened cursors current';**

**Max allowed open cursor and total open cursor**

**select max(a.value) as highest\_open\_cur, p.value as max\_open\_cur**  
**from v$sesstat a, v$statname b, v$parameter p**  
**where a.statistic# = b.statistic# and b.name = 'opened cursors current'**  
**and p.name= 'open\_cursors'**  
**group by p.value;**

### Session login history from ASH

**select c.username,a.SAMPLE\_TIME, a.SQL\_OPNAME, a.SQL\_EXEC\_START, a.program, a.module, a.machine, b.SQL\_TEXT**  
**from DBA\_HIST\_ACTIVE\_SESS\_HISTORY a, dba\_hist\_sqltext b, dba\_users c**  
**where a.SQL\_ID = b.SQL\_ID(+)**  
**and a.user\_id=c.user\_id**  
**and c.username='&username'**  
**order by a.SQL\_EXEC\_START asc;**

### Buffer Cache hit ratio

**SELECT ROUND((1-(phy.value / (cur.value + con.value)))\*100,2) "Cache Hit Ratio"**  
**FROM v$sysstat cur, v$sysstat con, v$sysstat phy**  
**WHERE cur.name = 'db block gets'**  
**AND con.name = 'consistent gets'**  
**AND phy.name = 'physical reads'**  
**/**

### Find top disk\_reads by an user

**select username users, round(DISK\_READS/Executions) DReadsExec,Executions Exec, DISK\_READS DReads,sql\_text**  
**from gv$sqlarea a, dba\_users b**  
**where a.parsing\_user\_id = b.user\_id**  
**and Executions > 0**  
**and DISK\_READS > 100000**  
**order by 2 desc;**

### Get os pid from sid

**set lines 123**  
**col USERNAME for a15**  
**col OSUSER for a8**  
**col MACHINE for a15**  
**col PROGRAM for a20**  
**select b.spid, a.username, a.program , a.osuser ,a.machine, a.sid, a.serial#, a.status from gv$session a, gv$process b**  
**where addr=paddr(+) and sid=&sid;**

### Get active sid of a pl/sql object

**select sid, sql\_id,serial#, status, username, program**  
**from v$session**  
**where PLSQL\_ENTRY\_OBJECT\_ID in (select object\_id**  
**from dba\_objects**  
**where object\_name in ('&PROCEDURE\_NAME'));**

### Find buffer cache usage

**col object\_name format a30**  
**col to\_total format 999.99**

**SELECT owner, object\_name, object\_type, count, (count / value) \* 100 to\_total**  
**FROM (**  
**SELECT a.owner, a.object\_name, a.object\_type,**  
**count(\*) count**  
**FROM dba\_objects a,**  
**x$bh b**  
**WHERE a.object\_id = b.obj**  
**and a.owner not in ('SYS', 'SYSTEM')**  
**GROUP BY a.owner, a.object\_name, a.object\_type**  
**ORDER BY 4),**  
**v$parameter**  
**WHERE name = 'db\_cache\_size'**  
**AND (count / value) \* 100 > .005**  
**ORDER BY to\_total desc**  
**/**

### Monitor rollback transactions

**select state,UNDOBLOCKSDONE,UNDOBLOCKSTOTAL,**  
**UNDOBLOCKSDONE/UNDOBLOCKSTOTAL\*100**  
**from gv$fast\_start\_transactions;**

**alter session set nls\_date\_format='dd-mon-yyyy hh24:mi:ss';**  
**select usn, state, undoblockstotal "Total", undoblocksdone "Done", undoblockstotal-undoblocksdone "ToDo",**  
**decode(cputime,0,'unknown',**  
**sysdate+(((undoblockstotal-undoblocksdone) / (undoblocksdone / cputime)) / 86400)) "Estimated time to complete"**  
**from v$fast\_start\_transactions;**

**select a.sid, a.username, b.xidusn, b.used\_urec, b.used\_ublk**  
**from v$session a, v$transaction b**  
**where a.saddr=b.ses\_addr**  
**order by 5 desc;**

### Find column usage statistics

**set lines 150**  
**set pages 500**  
**col table\_name for a20**  
**col column\_name for a20**  
**select a.object\_name table\_name, c.column\_name,equality\_preds, equijoin\_preds, range\_preds, like\_preds**  
**from dba\_objects a, col\_usage$ b, dba\_tab\_columns c**  
**where a.object\_id=b.OBJ#**  
**and c.COLUMN\_ID=b.INTCOL#**  
**and a.object\_name=c.table\_name**  
**and b.obj#=a.object\_id**  
**and a.object\_name='&table\_name'**  
**and a.object\_type='TABLE'**  
**and a.owner='&owner'**  
**order by 3 desc,4 desc, 5 desc;**

### Get background process details

**col ksbddidn for a15**  
**col ksmfsnam for a20**  
**col ksbdddsc for a60**  
**set lines 150 pages 5000**  
**SELECT ksbdd.ksbddidn, ksmfsv.ksmfsnam, ksbdd.ksbdddsc**  
**FROM x$ksbdd ksbdd, x$ksbdp ksbdp, x$ksmfsv ksmfsv**  
**WHERE ksbdd.indx = ksbdp.indx**  
**AND ksbdp.addr = ksmfsv.ksmfsadr**  
**ORDER BY ksbdd.ksbddidn;**

### oracle db is 32bit or 64 bit?

**select**  
**length(addr)\*4 || '-bits' word\_length**  
**from**  
**v$process**  
**where**  
**ROWNUM =1;**

### oracle license usage info

**select**  
**samp.dbid,**  
**fu.name,**  
**samp.version,**  
**detected\_usages,**  
**total\_samples,**  
**decode(to\_char(last\_usage\_date, 'MM/DD/YYYY, HH:MI:SS'),**  
**NULL, 'FALSE',**  
**to\_char(last\_sample\_date, 'MM/DD/YYYY, HH:MI:SS'), 'TRUE',**  
**'FALSE')**  
**currently\_used,**  
**first\_usage\_date,**  
**last\_usage\_date,**  
**aux\_count,**  
**feature\_info,**  
**last\_sample\_date,**  
**last\_sample\_period,**  
**sample\_interval,**  
**mt.description**  
**from**  
**wri$\_dbu\_usage\_sample samp,**  
**wri$\_dbu\_feature\_usage fu,**  
**wri$\_dbu\_feature\_metadata mt**  
**where**  
**samp.dbid = fu.dbid and**  
**samp.version = fu.version and**  
**fu.name = mt.name and**  
**fu.name not like '\_DBFUS\_TEST%' and /\* filter out test features \*/**  
**bitand(mt.usg\_det\_method, 4) != 4 /\* filter out disabled features \*/;**

### db optimizer processing rate

**select OPERATION\_NAME, DEFAULT\_VALUE from  
V$OPTIMIZER\_PROCESSING\_RATE where OPERATION\_NAME  
in ('IO\_BYTES\_PER\_SEC','CPU\_BYTES\_PER\_SEC', 'CPU\_ROWS\_PER\_SEC');**

### expdp to asm diskgroup

**Create a directory pointing to asm diskgroup( for dumpfiles)**

*SQL> create directory SOURCE\_DUMP as '+NEWTST/TESTDB2/TEMPFILE';*  
*Directory created*

**Create a directory pointing to a normal filesystem ( required for logfiles)**

*SQL> create directory EXPLOG as '/export/home/oracle';*  
*Directory created.*  
**export parfile**

*dumpfile=test.dmp*   
*logfile=EXPLOG:test.log*   
*directory=SOURCE\_DUMP*   
*tables=dbatest.EMPTAB*   
*exclude=statistics*

### xplain plan of sql\_id from cursor

**First get the child number of the sql\_id .One sql\_id can have multiple child number( one for each plan\_hash\_value)**

**SQL> select sql\_id,child\_number,plan\_hash\_value from gv$sql where sql\_id='9n2a2c8pvu6bm'; SQL\_ID CHILD\_NUMBER PLAN\_HASH\_VALUE ------------- ------------ --------------- 9n2a2c8pvu6bm 1 13761463 Now get the explain plan for cursor: *SELECT \* from TABLE(DBMS\_XPLAN.DISPLAY\_CURSOR('&sqlid',&child\_number));***

**set lines 200**

**SELECT \* FROM table(DBMS\_XPLAN.DISPLAY\_AWR('&sql\_id'));**

### Get sql\_text from sid

**col sql\_text form a80**  
**set lines 120**  
**select sql\_text from gv$sqltext where hash\_value=**  
**(select sql\_hash\_value from gv$session where sid=&1 and inst\_id=&inst\_id)**  
**order by piece**  
**/**

### xplain plan of a sql statement

Generate explain plan  
-- Syntax EXPLAIN PLAN FOR < SQL STATEMENT> ;

**explain plan for**  
**select count(\*) from dbaclass;**  
View explain plan

**select \* from table(dbms\_xplan.display);**

### xplain plan of a sql baseline

--- SYNTAX

-- SELECT \* FROM TABLE(DBMS\_XPLAN.display\_sql\_plan\_baseline(plan\_name=>'<SQL BASELINE NAME>'));

**SELECT \* FROM TABLE(DBMS\_XPLAN.display\_sql\_plan\_baseline(plan\_name=>'SQL\_PLAN\_gbhrw1v44209a5b2f7514'));**

### Get bind values of a sql\_id

**SELECT**   
**sql\_id,**  
**b. LAST\_CAPTURED,**  
**t.sql\_text sql\_text,**   
**b.HASH\_VALUE,**  
**b.name bind\_name,**  
**b.value\_string bind\_value**   
**FROM**  
**gv$sql t**   
**JOIN**  
**gv$sql\_bind\_capture b using (sql\_id)**  
**WHERE**  
**b.value\_string is not null**   
**AND**  
**sql\_id='&sqlid'**  
**/**

### Flush a sql query from cursor

**First get the address, hash\_value of the sql\_id**

**select ADDRESS, HASH\_VALUE from V$SQLAREA where SQL\_ID like '5qd8a442c328k';**

ADDRESS HASH\_VALUE   
--------------- ------------  
C000007067F39FF0 4000666812

**Now flush the query**

**SQL> exec DBMS\_SHARED\_POOL.PURGE ('C000007067F39FF0, 4000666812', 'C');**

***Note : For RAC, same need to be executed on all the nodes .***

**--- CREATE THE BELOW TRIGGER TO ENABLE TRACE ALL SESSION OF USER ( SCOTT)**  
**CREATE OR REPLACE TRIGGER USER\_TRACE\_TRG**  
**AFTER LOGON ON DATABASE**  
**BEGIN**  
**IF USER = 'SCOTT'**  
**THEN**  
**execute immediate 'alter session set events ''10046 trace name context forever, level 12''';**  
**END IF;**  
**EXCEPTION**  
**WHEN OTHERS THEN**  
**NULL;**  
**END;**  
**/**

### Enable trace for a session

**EXEC DBMS\_SYSTEM.set\_sql\_trace\_in\_session(sid=>321, serial#=>1234, sql\_trace=>FALSE);**

**Get the trace file name**

**SELECT p.tracefile FROM v$session s JOIN v$process p ON s.paddr = p.addr WHERE s.sid = 321;**

### 10053 OPTIMIZER TRACE

**begin**  
**dbms\_sqldiag.dump\_trace(p\_sql\_id=>'dmx08r6ayx800',**  
**p\_child\_number=>0,**  
**p\_component=>'Compiler',**  
**p\_file\_id=>'TEST\_OBJ3\_TRC');**  
**END;**  
**/**

### Enable trace for a sql\_id

**alter system set events 'sql\_trace [sql:8krc88r46raff]';**

### execution detail of a sql\_id in cursor

**select module,parsing\_schema\_name,inst\_id,sql\_id,plan\_hash\_value,child\_number,sql\_fulltext,**  
**to\_char(last\_active\_time,'DD/MM/YY HH24:MI:SS' ),sql\_plan\_baseline,executions,**  
**elapsed\_time/executions/1000/1000,rows\_processed from gv$sql**  
**where sql\_id in ('&sql\_id');**

### Enable tracing for a listener

- Set to the listener you want to trace

LSNRCTL> **set cur LISTENER\_TEST**

-- Enable Trace:

LSNRCTL> **set trc\_level ADMIN**  
Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=IPC)(KEY=LISTENER\_TEST)))  
LISTENER\_TEST parameter "trc\_level" set to admin  
The command completed successfully

### Pga usage by sessions

**set lines 2000**  
**SELECT SID, b.NAME, ROUND(a.VALUE/(1024\*1024),2) MB FROM**   
**v$sesstat a, v$statname b**  
**WHERE (NAME LIKE '%session uga memory%' OR NAME LIKE '%session pga memory%')**  
**AND a.statistic# = b.statistic# order by ROUND(a.VALUE/(1024\*1024),2) desc**

### segments with high physical read

**set pagesize 200**  
**setlinesize 120**  
**col segment\_name format a20**  
**col owner format a10**  
**select segment\_name,object\_type,total\_physical\_reads**  
**from ( select owner||'.'||object\_name as segment\_name,object\_type,**  
**value as total\_physical\_reads**  
**from v$segment\_statistics**  
**where statistic\_name in ('physical reads')**  
**order by total\_physical\_reads desc)**  
**where rownum <=10;**

### I/O usage of 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;**

### Current SGA usage

**select round(used.bytes /1024/1024 ,2) used\_mb**  
**, round(free.bytes /1024/1024 ,2) free\_mb**  
**, round(tot.bytes /1024/1024 ,2) total\_mb**  
**from (select sum(bytes) bytes**  
**from v$sgastat**  
**where name != 'free memory') used**  
**, (select sum(bytes) bytes**  
**from v$sgastat**  
**where name = 'free memory') free**  
**, (select sum(bytes) bytes**  
**from v$sgastat) tot**  
**/**

### Top running queries from ASH

**Query to get list of top running sqls in PAST between sysdate-1 to sysdate-23/34 . You can change accordingly**

**SELECT active\_session\_history.user\_id,**  
**dba\_users.username,**  
**sqlarea.sql\_text,**  
**SUM(active\_session\_history.wait\_time +**  
**active\_session\_history.time\_waited)/1000000 ttl\_wait\_time\_in\_seconds**  
**FROM v$active\_session\_history active\_session\_history,**  
**v$sqlarea sqlarea,**  
**dba\_users**  
**WHERE active\_session\_history.sample\_time BETWEEN SYSDATE - 1 AND SYSDATE-23/24**  
**AND active\_session\_history.sql\_id = sqlarea.sql\_id**  
**AND active\_session\_history.user\_id = dba\_users.user\_id**  
**and dba\_users.username not in ('SYS','DBSNMP')**  
**GROUP BY active\_session\_history.user\_id,sqlarea.sql\_text, dba\_users.username**  
**ORDER BY 4 DESC**  
**/**

### Find blocking sessions from ASH

**Query will list the blocking session details between SYSDATE - 1 AND SYSDATE-23/24 ( PAST)**

**set pagesize 50**  
**set linesize 120**  
**col sql\_id format a15**  
**col inst\_id format '9'**  
**col sql\_text format a50**  
**col module format a10**  
**col blocker\_ses format '999999'**  
**col blocker\_ser format '999999'**  
**SELECT distinct**  
**a.sql\_id ,**  
**a.inst\_id,**  
**a.blocking\_session blocker\_ses,**  
**a.blocking\_session\_serial# blocker\_ser,**  
**a.user\_id,**  
**s.sql\_text,**  
**a.module,a.sample\_time**  
**FROM GV$ACTIVE\_SESSION\_HISTORY a,**  
**gv$sql s**  
**where a.sql\_id=s.sql\_id**  
**and blocking\_session is not null**  
**and a.user\_id <> 0 -- exclude SYS user**  
**and a.sample\_time BETWEEN SYSDATE - 1 AND SYSDATE-23/24**  
**/**

### Top cpu consuming sessions

**col program form a30 heading "Program"**  
**col CPUMins form 99990 heading "CPU in Mins"**  
**select rownum as rank, a.\***  
**from (**  
**SELECT v.sid, program, v.value / (100 \* 60) CPUMins**  
**FROM v$statname s , v$sesstat v, v$session sess**  
**WHERE s.name = 'CPU used by this session'**  
**and sess.sid = v.sid**  
**and v.statistic#=s.statistic#**  
**and v.value>0**  
**ORDER BY v.value DESC) a**  
**where rownum < 11;**

### Sessions holding library cache lock

**For standalone db:**

**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%';**

**For RAC DB:**

**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%';**

or

**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%';**

### Objects locked by library cache

**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\_requestedmode\_held**  
**and session\_id in ( select sid**  
**from v$session\_wait where wait\_time=0**  
**and event like '%library cache%') ;**

### Sessions accessing an object

**set lines 299**  
**column object format a30**  
**column owner format a10**  
**select \* from gv$access where owner='&OWNER' and object='&object\_name' and**  
**/**

### Sqls doing full table scan

**select sql\_id,object\_owner,object\_name from V$SQL\_PLAN where**  
**operation='TABLE ACCESS' and**  
**options='FULL' and**  
**object\_owner not in ('SYS','SYSTEM','DBSNMP');**

### Dictionary cache hit ratio

**select sum(gets) as "Gets", sum(getmisses) as "Misses",**  
**(1-(sum(getmisses)/sum(gets)))\*100 as "CACHE HIT RATIO"**  
**from gv$rowcache;**

**NOTE - CACHE HIT RATIO SHOULD BE MORE THAN 95 PERCENT.**

**column mux format a18 heading 'Mutex Type' trunc;**  
**column loc format a32 heading 'Location' trunc;**  
**column sleeps format 9,999,999,990 heading 'Sleeps';**  
**column wt format 9,999,990.9 heading 'Wait |Time (s)';**  
**select e.mutex\_type mux**  
**, e.location loc**  
**, e.sleeps - nvl(b.sleeps, 0) sleeps**  
**, (e.wait\_time - nvl(b.wait\_time, 0))/1000000 wt**  
**from DBA\_HIST\_MUTEX\_SLEEP b**  
**, DBA\_HIST\_MUTEX\_SLEEP e**  
**where b.snap\_id(+) = &bid**  
**and e.snap\_id = &eid**  
**and b.dbid(+) = e.dbid**  
**and b.instance\_number(+) = e.instance\_number**  
**and b.mutex\_type(+) = e.mutex\_type**  
**and b.location(+) = e.location**  
**and e.sleeps - nvl(b.sleeps, 0) > 0**  
**order by e.wait\_time - nvl(b.wait\_time, 0) desc;**

### Queries causing high physical read

**SELECT schema, sql\_text, disk\_reads, round(cpu,2) FROM**  
**(SELECT s.parsing\_schema\_name schema, t.sql\_id, t.sql\_text, t.disk\_reads,**  
**t.sorts, t.cpu\_time/1000000 cpu, t.rows\_processed, t.elapsed\_time**  
**FROM v$sqlstats t join v$sql s on(t.sql\_id = s.sql\_id)**  
**WHERE parsing\_schema\_name = 'SCOTT'**  
**ORDER BY disk\_reads DESC)**  
**WHERE rownum <= 5;**

### Objects causing latch contention

**col OBJECT\_NAME for a30**  
**col owner for a12**  
**with bh\_lc as**  
**(select**  
**lc.addr, lc.child#, lc.gets, lc.misses, lc.immediate\_gets, lc.immediate\_misses,**  
**lc.spin\_gets, lc.sleeps,**  
**bh.hladdr, bh.tch tch, bh.file#, bh.dbablk, bh.class, bh.state, bh.obj**  
**from**  
**v$session\_wait sw,**  
**v$latchname ld,**  
**v$latch\_children lc,**  
**x$bh bh**  
**where lc.addr =sw.p1raw**  
**and sw.p2= ld.latch#**  
**and ld.name='cache buffers chains'**  
**and lower(sw.event) like '%latch%'**  
**and bh.hladdr=lc.addr**  
**)**  
**select bh\_lc.hladdr, bh\_lc.tch, o.owner, o.object\_name, o.object\_type,**  
**bh\_lc.child#,**  
**bh\_lc.gets, bh\_lc.misses, bh\_lc.immediate\_gets,**  
**bh\_lc.immediate\_misses, spin\_gets, sleeps**  
**from**  
**bh\_lc, dba\_objects o**  
**where bh\_lc.obj = o.data\_object\_id(+)**  
**order by 1,2 desc;**

### Latch type and sql hash value

**Set lines 160 pages 100**  
**Column event format A35**  
**Column name format A35**  
**select x.event, x.sql\_hash\_value,**  
**case when x.event like 'latch%' then**  
**l.name**  
**else ' '**  
**end name,**  
**x.cnt from (**  
**select substr(w.event, 1, 28) event, s.sql\_hash\_value, w.p2,count(\*) cnt**  
**from v$session\_wait w, v$session s, v$process p**  
**where s.sid=w.sid**  
**and p.addr = s.paddr**  
**and s.username is not null**  
**and w.event not like '%pipe%'**  
**and w.event not like 'SQL\*%'**  
**group by substr(w.event, 1, 28), sql\_hash\_value,w.p2**  
**) x,**  
**v$latch l**  
**where**  
**x.p2 = l.latch#(+)**  
**order by cnt;**

### Objects causing flushing of shared pool

**Set lines 160 pages 100**  
**Select \* from x$ksmlru order by ksmlrnum;**

### Sql tuning advisor for sql\_id from cursor

**Create tuning task**

**set long 1000000000**  
**DECLARE**  
**l\_sql\_tune\_task\_id VARCHAR2(100);**  
**BEGIN**  
**l\_sql\_tune\_task\_id := DBMS\_SQLTUNE.create\_tuning\_task (**  
**sql\_id => 'apwfwjhgc9sk8',**  
**scope => DBMS\_SQLTUNE.scope\_comprehensive,**  
**time\_limit => 500,**  
**task\_name => 'apwfwjhgc9sk8\_tuning\_task\_1',**  
**description => 'Tuning task for statement apwfwjhgc9sk8');**  
**DBMS\_OUTPUT.put\_line('l\_sql\_tune\_task\_id: ' || l\_sql\_tune\_task\_id);**  
**END;**  
**/**

**Execute tuning task**

**EXEC DBMS\_SQLTUNE.execute\_tuning\_task(task\_name => 'apwfwjhgc9sk8\_tuning\_task\_1');**

**Generate report**

**SET LONG 10000000;**  
**SET PAGESIZE 100000000**  
**SET LINESIZE 200**  
**SELECT DBMS\_SQLTUNE.report\_tuning\_task('apwfwjhgc9sk8\_tuning\_task\_1') AS recommendations FROM dual;**  
**SET PAGESIZE 24**

### run sga target advisory

**- STATISTICS\_LEVEL should be TYPICAL/ALL.**

SQL> show parameter statistics\_level

NAME TYPE VALUE  
------------------------------------ -------------------------------- --------------------------  
statistics\_level string **TYPICAL**

select \* from v$sga\_target\_advice order by sga\_size;

### Run shared pool advisory

**SELECT shared\_pool\_size\_for\_estimate "Size of Shared Pool in MB",**  
**shared\_pool\_size\_factor "Size Factor",**  
**estd\_lc\_time\_saved "Time Saved in sec" FROM v$shared\_pool\_advice;**

### Generate addm report

**cd $ORACLE\_HOME/rdbms/admin**

SQL> **@addmrpt.sql**

Specify the Begin and End Snapshot Ids  
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  
Enter value for begin\_snap: 1058  
Begin Snapshot Id specified: 1058

Enter value for end\_snap: 1059  
End Snapshot Id specified: 1059

### Get redo log member info

**col member for a56**  
**set pagesize 299**  
**set lines 299**  
**select l.group#, l.thread#,**  
**f.member,**  
**l.archived,**  
**l.status,**  
**(bytes/1024/1024) "Size (MB)"**  
**from**  
**v$log l, v$logfile f**  
**where f.group# = l.group#**  
**order by 1,2**  
**/**

### Get DDL of all tablespaces

**set heading off;**  
**set echo off;**  
**Set pages 999;**  
**set long 90000;**  
**spool ddl\_tablespace.sql**  
**select dbms\_metadata.get\_ddl('TABLESPACE',tb.tablespace\_name) from dba\_tablespaces tb;**  
**spool off**

### Get DDL of all privileges granted to user

**set feedback off pages 0 long 900000 lines 20000 pagesize 20000 serveroutput on**  
**accept USERNAME prompt "Enter username :"**  
**--This line add a semicolon at the end of each statement**  
**execute dbms\_METADATA.SET\_TRANSFORM\_PARAM(DBMS\_METADATA.SESSION\_TRANSFORM,'SQLTERMINATOR',true);**  
**-- This will generate the DDL for the user and add his objects,system and role grants**  
**SELECT DBMS\_METADATA.GET\_DDL('USER',username) as script from DBA\_USERS where username='&username'**  
**UNION ALL**  
**SELECT DBMS\_METADATA.GET\_GRANTED\_DDL('SYSTEM\_GRANT',grantee)as script from DBA\_SYS\_PRIVS where grantee='&username' and rownum=1**  
**UNION ALL**  
**SELECT DBMS\_METADATA.GET\_GRANTED\_DDL('ROLE\_GRANT',grantee)as script from DBA\_ROLE\_PRIVS where grantee='&username' and rownum=1**  
**UNION ALL**  
**SELECT DBMS\_METADATA.GET\_GRANTED\_DDL('OBJECT\_GRANT',grantee)as script from DBA\_TAB\_PRIVS where grantee='&username' and rownum=1;**

### Get size of the database

**col "Database Size" format a20**  
**col "Free space" format a20**  
**col "Used space" format a20**  
**select round(sum(used.bytes) / 1024 / 1024 / 1024 ) || ' GB' "Database Size"**  
**, round(sum(used.bytes) / 1024 / 1024 / 1024 ) -**  
**round(free.p / 1024 / 1024 / 1024) || ' GB' "Used space"**  
**, round(free.p / 1024 / 1024 / 1024) || ' GB' "Free space"**  
**from (select bytes**  
**from v$datafile**  
**union all**  
**select bytes**  
**from v$tempfile**  
**union all**  
**select bytes**  
**from v$log) used**  
**, (select sum(bytes) as p**  
**from dba\_free\_space) free**  
**group by free.p**  
**/**

### View hidden parameter setting

**Set lines 2000**  
**col NAME for a45**  
**col DESCRIPTION for a100**  
**SELECT name,description from SYS.V$PARAMETER WHERE name LIKE '\\_%' ESCAPE '\'**

**/**

### Get ACL details in database

**set lines 200**  
**COL ACL\_OWNER FOR A12**  
**COL ACL FOR A67**  
**COL HOST FOR A34**  
**col PRINCIPAL for a20**  
**col PRIVILEGE for a13**  
**select ACL\_OWNER,ACL,HOST,LOWER\_PORT,UPPER\_PORT FROM dba\_network\_acls;**

**select ACL\_OWNER,ACL,PRINCIPAL,PRIVILEGE from dba\_network\_acl\_privileges;**

### Archive generation per hour

**set lines 299**  
**SELECT TO\_CHAR(TRUNC(FIRST\_TIME),'Mon DD') "DG Date",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'00',1,0)),'9999') "12AM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'01',1,0)),'9999') "01AM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'02',1,0)),'9999') "02AM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'03',1,0)),'9999') "03AM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'04',1,0)),'9999') "04AM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'05',1,0)),'9999') "05AM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'06',1,0)),'9999') "06AM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'07',1,0)),'9999') "07AM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'08',1,0)),'9999') "08AM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'09',1,0)),'9999') "09AM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'10',1,0)),'9999') "10AM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'11',1,0)),'9999') "11AM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'12',1,0)),'9999') "12PM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'13',1,0)),'9999') "1PM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'14',1,0)),'9999') "2PM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'15',1,0)),'9999') "3PM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'16',1,0)),'9999') "4PM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'17',1,0)),'9999') "5PM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'18',1,0)),'9999') "6PM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'19',1,0)),'9999') "7PM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'20',1,0)),'9999') "8PM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'21',1,0)),'9999') "9PM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'22',1,0)),'9999') "10PM",**  
**TO\_CHAR(SUM(DECODE(TO\_CHAR(FIRST\_TIME,'HH24'),'23',1,0)),'9999') "11PM"**  
**FROM V$LOG\_HISTORY**  
**GROUP BY TRUNC(FIRST\_TIME)**  
**ORDER BY TRUNC(FIRST\_TIME) DESC**  
**/**

### Find active transactions in db

**col name format a10**  
**col username format a8**  
**col osuser format a8**  
**col start\_time format a17**  
**col status format a12**  
**tti 'Active transactions'**

**select s.sid,username,t.start\_time, r.name, t.used\_ublk "USED BLKS",**  
**decode(t.space, 'YES', 'SPACE TX',**  
**decode(t.recursive, 'YES', 'RECURSIVE TX',**  
**decode(t.noundo, 'YES', 'NO UNDO TX', t.status)**  
**)) status**  
**from sys.v\_$transaction t, sys.v\_$rollname r, sys.v\_$session s**  
**where t.xidusn = r.usn**  
**and t.ses\_addr = s.saddr**  
**/**

### Find who locked your account

**-- Return code 1017 ( INVALID LOGIN ATTEMPT)**  
**-- Return code 28000 ( ACCOUNT LOCKED)**

**set pagesize 1299**  
**set lines 299**  
**col username for a15**  
**col userhost for a13**  
**col timestamp for a39**  
**col terminal for a23**  
**SELECT username,userhost,terminal,timestamp,returncode**  
**FROM dba\_audit\_session**  
**WHERE username='&USER\_NAME' and returncode in (1017,28000);**

### Find duplicate rows in table

**--- Reference metalink id - 332494.1**  
**-- Save as duplicate.sql and run as @duplicate.sql**

**REM This is an example SQL\*Plus Script to detect duplicate rows from**  
**REM a table.**  
**REM**  
**set echo off**  
**set verify off heading off**  
**undefine t**  
**undefine c**  
**prompt**  
**prompt**  
**prompt Enter name of table with duplicate rows**  
**prompt**  
**accept t prompt 'Table: '**  
**prompt**  
**select 'Table '||upper('&&t') from dual;**  
**describe &&t**  
**prompt**  
**prompt Enter name(s) of column(s) which should be unique. If more than**  
**prompt one column is specified, you MUST separate with commas.**  
**prompt**  
**accept c prompt 'Column(s): '**  
**prompt**  
**select &&c from &&t**  
**where rowid not in (select min(rowid) from &&t group by &&c)**  
**/**

### generate resize datafile script

**generate resize datafile script without ORA-03297 error**

**select 'alter database datafile'||' '''||file\_name||''''||' resize '||round(highwater+2)||' '||'m'||';' from (**  
**select /\*+ rule \*/**  
**a.tablespace\_name,**  
**a.file\_name,**  
**a.bytes/1024/1024 file\_size\_MB,**  
**(b.maximum+c.blocks-1)\*d.db\_block\_size/1024/1024 highwater**  
**from dba\_data\_files a ,**  
**(select file\_id,max(block\_id) maximum**  
**from dba\_extents**  
**group by file\_id) b,**  
**dba\_extents c,**  
**(select value db\_block\_size**  
**from v$parameter**  
**where name='db\_block\_size') d**  
**where a.file\_id= b.file\_id**  
**and c.file\_id = b.file\_id**  
**and c.block\_id = b.maximum**  
**order by a.tablespace\_name,a.file\_name);**

### Database growth per month

**select to\_char(creation\_time, 'MM-RRRR') "Month", sum(bytes)/1024/1024/1024 "Growth in GB**  
**from sys.v\_$datafile**  
**where to\_char(creation\_time,'RRRR')='&YEAR\_IN\_YYYY\_FORMAT'**  
**group by to\_char(creation\_time, 'MM-RRRR')**  
**order by to\_char(creation\_time, 'MM-RRRR');**

### Get database uptime

**select to\_char(startup\_time, 'DD-MM-YYYY HH24:MI:SS'),floor(sysdate-startup\_time) DAYS from v$Instance;**

### Scn to timestamp and viceversa

-- Get current scn value:

**select current\_scn from v$database;**

-- Get scn value at particular time:

**select timestamp\_to\_scn('19-JAN-08:22:00:10') from dual;**

-- Get timestamp from scn:

**select scn\_to\_timestamp(224292)from dual;**

### Disable/enable all triggers of schema

**----- Connect to the user and run this.**

**BEGIN**  
**FOR i IN (SELECT trigger\_name**  
**FROM user\_triggers) LOOP**

**EXECUTE IMMEDIATE 'ALTER TRIGGER ' || i.trigger\_name || ' DISABLE';**  
**END LOOP;**  
**END;**  
**/**

### Ger row\_count of all the tables of a schema

**select table\_name,**  
**to\_number(extractvalue(dbms\_xmlgen.getXMLtype('select /\*+ PARALLEL(8) \*/ count(\*) cnt from "&&SCHEMA\_NAME".'||table\_name),'/ROWSET/ROW/CNT'))**  
**rows\_in\_table from dba\_TABLES**  
**where owner='&&SCHEMA\_NAME';**

### Monitor index usage

***---Index monitoring is required, to find whether indexes are really in use or not. Unused can be dropped to avoid overhead.***  
***--* First *enable monitoring usage for the indexes.***

**alter index siebel.S\_ASSET\_TEST monitoring usage;**

***--Below query to find the index usage:***

**select \* from v$object\_usage;**

### Spool sql query output to HTML

**We can spool output of an sql query to html format:**

**set pages 5000**  
**SET MARKUP HTML ON SPOOL ON PREFORMAT OFF ENTMAP ON -**  
**HEAD "<TITLE>EMPLOYEE REPORT</TITLE> -**  
**<STYLE type='text/css'> -**  
**<!-- BODY {background: #FFFFC6} --> -**  
**</STYLE>" -**  
**BODY "TEXT='#FF00Ff'" -**  
**TABLE "WIDTH='90%' BORDER='5'"**  
**spool report.html**  
**Select \* from scott.emp;**  
**spool off**  
**exit**

### Get installed sqlpatches in db

*--- From 12c onward*

set lines 2000  
**select patch\_id,status,description from dba\_registry\_sqlpatch;**

*--- For 11g and below:*

set lines 2000  
**select \* from dba\_registry\_history;**

### Cleanup orphaned datapump jobs

**-- Find the orphaned Data Pump jobs:**

SELECT owner\_name, job\_name, rtrim(operation) "OPERATION",  
rtrim(job\_mode) "JOB\_MODE", state, attached\_sessions  
FROM dba\_datapump\_jobs  
WHERE job\_name NOT LIKE 'BIN$%' and state='NOT RUNNING'  
ORDER BY 1,2;

**-- Drop the tables**

SELECT 'drop table ' || owner\_name || '.' || job\_name || ';'  
FROM dba\_datapump\_jobs WHERE state='NOT RUNNING' and job\_name NOT LIKE 'BIN$%'

### Installed RDBMS components

**col comp\_id for a10**  
**col comp\_name for a56**  
**col version for a12**  
**col status for a10**  
**set pagesize 200**  
**set lines 200**  
**set long 999**  
**select comp\_id,comp\_name,version,status from dba\_registry;**

### Characterset info of database

**set pagesize 200**  
**set lines 200**  
**select parameter,value from v$nls\_parameters where parameter like 'NLS\_%CHAR%';**

### View/modify AWR retention

**-- View current AWR retention period**

select retention from dba\_hist\_wr\_control;

**-- Modify retention period to 7 days and interval to 30 min**

select dbms\_workload\_repository.modify\_snapshot\_settings (interval => 30, retention => 10080);

*NOTE - 7 DAYS = 7\*24\*3600= 10080 minutes*

### Find optimal undo retention size

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 gv$datafile a, gv$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,  
gv$parameter e,  
gv$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';

### Purge old awr snapshots

**-- Find the AWR snapshot details.**

select snap\_id,begin\_interval\_time,end\_interval\_time from sys.wrm$\_snapshot order by snap\_id

**-- Purge snapshot between snapid 612 to 700**

execute dbms\_workload\_repository.drop\_snapshot\_range(low\_snap\_id =>612 , high\_snap\_id =>700);

**-- Verify again**

select snap\_id,begin\_interval\_time,end\_interval\_time from sys.wrm$\_snapshot order by snap\_id

### Modify moving window baseline size

**-- Check the current moving window baseline size:**

select BASELINE\_TYPE,MOVING\_WINDOW\_SIZE from dba\_hist\_baseline;

**-- Modify window\_size to (7 days):**

execute dbms\_workload\_repository.modify\_baseline\_window\_size(window\_size=> 7);

### Open database link information

set pagesize 200  
set lines 200  
col db\_link for a19  
set long 999  
SELECT db\_link,  
owner\_id,  
logged\_on,  
heterogeneous,  
open\_cursors,  
in\_transaction,  
update\_sent  
FROM gv$dblink  
ORDER BY db\_link;

### utilization of current redo log ( in % )

**SELECT le.leseq "Current log sequence No",**  
**100\*cp.cpodr\_bno/le.lesiz "Percent Full",**  
**cp.cpodr\_bno "Current Block No",**  
**le.lesiz "Size of Log in Blocks"**  
**FROM x$kcccp cp, x$kccle le**  
**WHERE le.leseq =CP.cpodr\_seq**  
**AND bitand(le.leflg,24) = 8**  
**/**

### Table not having index on fk column

**select \* from (**  
**select c.table\_name, co.column\_name, co.position column\_position**  
**from user\_constraints c, user\_cons\_columns co**  
**where c.constraint\_name = co.constraint\_name**  
**and c.constraint\_type = 'R'**  
**minus**  
**select ui.table\_name, uic.column\_name, uic.column\_position**  
**from user\_indexes ui, user\_ind\_columns uic**  
**where ui.index\_name = uic.index\_name**  
**)**  
**order by table\_name, column\_position;**

**select**  
**a.constraint\_name cons\_name**  
**,a.table\_name tab\_name**  
**,b.column\_name cons\_column**  
**,nvl(c.column\_name,'\*\*\*No Index\*\*\*') ind\_column**  
**from user\_constraints a**  
**join**  
**user\_cons\_columns b on a.constraint\_name = b.constraint\_name**  
**left outer join**  
**user\_ind\_columns c on b.column\_name = c.column\_name**  
**and b.table\_name = c.table\_name**  
**where constraint\_type = 'R'**  
**order by 2,1;**

### Get cpu memory info of db server

**set pagesize 200**  
**set lines 200**  
**col name for a21**  
**col stat\_name for a25**  
**col value for a13**  
**col comments for a56**  
**select STAT\_NAME,to\_char(VALUE) as VALUE ,COMMENTS from v$osstat where  
stat\_name IN ('NUM\_CPUS','NUM\_CPU\_CORES','NUM\_CPU\_SOCKETS')**  
**union**  
**select STAT\_NAME,VALUE/1024/1024/1024 || ' GB' ,COMMENTS from  
v$osstat where stat\_name IN ('PHYSICAL\_MEMORY\_BYTES');**

### Get database incarnation info

**set heading off**  
**set feedback off**  
**select 'Incarnation Destination Configuration' from dual;**  
**select '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*' from dual;**  
**set heading on**  
**set feedback on**

**select INCARNATION# INC#, RESETLOGS\_CHANGE# RS\_CHANGE#, RESETLOGS\_TIME,**  
**PRIOR\_RESETLOGS\_CHANGE# PRIOR\_RS\_CHANGE#, STATUS,**  
**FLASHBACK\_DATABASE\_ALLOWED FB\_OK from v$database\_incarnation;**

### View timezone info in db

SELECT version FROM v$timezone\_file;

SELECT PROPERTY\_NAME, SUBSTR(property\_value, 1, 30) value  
FROM DATABASE\_PROPERTIES  
WHERE PROPERTY\_NAME LIKE 'DST\_%'  
ORDER BY PROPERTY\_NAME;

### Get asm disk info

**set pagesize 2000**  
**set lines 2000**  
**set long 999**  
**col path for a54**  
**select name, path, header\_status, total\_mb free\_mb, trunc(bytes\_read/1024/1024) read\_mb, trunc(bytes\_written/1024/1024) write\_mb from v$asm\_disk;**

### Get ASM diskgroup details

**SELECT name, free\_mb, total\_mb, free\_mb/total\_mb\*100 as percentage**  
**FROM v$asm\_diskgroup;**

### drop an asm disk

**-----Dropping one disk:**

alter diskgroup data drop disk DATA\_ASM0001;

**-----Dropping multiple disk:**

alter diskgroup data drop disk DATA\_ASM0001,DATA\_ASM00002, DATA\_ASM0003 rebalance power 100;

**---- Monitoring the rebalance operation:**

select \* from v$asm\_operation;

### Monitor ASM disk rebalance

**set pagesize 299**  
**set lines 2999**  
**select GROUP\_NUMBER,OPERATION,STATE,POWER,**  
**ACTUAL,ACTUAL,EST\_MINUTES from gv$asm\_operation;**

### execute runcluvfy.sh for RAC precheck

**Runcluvfy.sh script is available after unzipping the grid software.**

**syntax - ./runcluvfy.sh stage -pre crsinst -n host1,host2,host3 -verbose**

**./runcluvfy.sh stage -pre crsinst -n classpredb1,classpredb2 -verbose**

### copy asm file to remote asm instance

**ASM file can be copied to remote asm instance(diskgroup) using asmcmd command.**

**SYNTAX - asmcmd> cp - -port asm\_port file\_name remote\_asm\_user/remote\_asm\_pwd@remote\_host:Instancce\_name:TARGET\_ASM\_PATH**

**ASMCMD> cp --port 1521 s\_srv\_new21.dbf sys/oracle@172.20.17.69.+ASM1:+ARCL/s\_srv\_new21.dbf**

### Mount/dismount ASM diskgroups

**To mount a diskgroup,(This is instance specific, for mounting on all nodes, run the same on all nodes)**

**SQL>alter diskgroup DATA mount;**

or

**asmcmd>mount DATA**

**To unmount a diskgroup,(This is instance specific, for unmounting on all nodes, run the same on all nodes)**

**SQL>alter diskgroup DATA dismount;**

or

**asmcmd>umount DATA**

**To mount/Dismount all the diskgroups**

**SQL>alter diskgroup ALL mount;**

**SQL>alter diskgroup ALL dismount;**

### Drop ASM diskgroup

**-- To drop a diskgroup, make sure the diskgroup has been dismounted from all the remote nodes, It should be mounted only on the local nodes, where we will run the drop command.**

**drop diskgroup DSMREDOA including contents;**

### Clock Synchronization status in RAC

**Clock Synchronization across the cluster nodes**

**cd $GRID\_HOME/bin**  
**cluvfy comp clocksync -n all**

**Check whether ctss or ntp is running**

**crsctl check ctss**  
CRS-4700: The Cluster Time Synchronization Service is in Observer mode.

***Observer means - Time sync between nodes are taken care by NTP***  
***Active means - Time sync between nodes are taken care by CTSS***

**Check the asm disk labelling**  
#**/etc/init.d/oracleasm querydisk /dev/sdn1**  
Device "/dev/sdn" is not marked as an ASM disk

**Create asm disk**  
**# /etc/init.d/oracleasm createdisk ARCDATA /dev/sdn1**  
Marking disk "ARCDATA" as an ASM disk: [ OK ]

**Check the asm disk labelling**  
# **/etc/init.d/oracleasm querydisk /dev/sdn1**  
Device "/dev/sdn1" is marked an ASM disk with the label "ARCDATA"

**List the asm disks present**  
# **/etc/init.d/oracleasm listdisks**  
ARCDATA

### Stop/start cluster in rac standalone

-- Oracle RAC in standalone is known as oracle restart, where only HAS(high availability service) component is available.

**crsctl stop has**

**crsctl start has**

### Create password file in ASM DG

**For oracle 12c only**

**ASMCMD> pwcreate –dbuniquename {db\_unique\_name} {file\_path} {sys\_password}**

**ASMCMD> pwcreate --dbuniquename PRDPRE +DATA/PWDFILE/pwdPRDPRE oracle**

**For all version.**

**orapwd file='+DATA/orapwPRODPRE' ENTRIES=10 DBUNIQUENAME='PRODPRE'**

### Change asm rebalance power

***-- Default value of asm\_power\_limit.***

SQL> **show parameter asm\_power\_limit**  
NAME TYPE VALUE  
------------------------------------ ----------- -----------------------------  
asm\_power\_limit integer 1

***-- Check for ongoing rebalance operations and their power.***

**select INST\_ID,GROUP\_NUMBER, OPERATION, STATE, POWER, EST\_RATE, EST\_MINUTES from GV$ASM\_OPERATION;**

***- Alter the asm rebalance.***

**alter diskgroup SALDATA rebalance power 4;**

### Modify asm user password

***-- list asm users***

ASMCMD> **lspwusr**  
Username sysdba sysoper sysasm  
SYS TRUE TRUE TRUE  
**ASMSNMP TRUE FALSE FALSE -- >**

***-- Modify user password***

ASMCMD> **orapwusr --modify asmsnmp**  
Enter password: \*\*\*\*\*\*\*\*

**--- Run from toad,sql devl**

select \* from V$ASM\_DISK\_IOSTAT;

### Enable tracing for asmcmd

$**export DBI\_TRACE=1**

$ **asmcmd**

### How to Change ASM sys password

**$ export ORACLE\_SID=+ASM**  
**$ asmcmd**

**ASMCMD> orapwusr --modify --password sys**  
**Enter password: \*\*\*\*\*\***  
**ASMCMD> exit**

**Alternatively, we can use orapwd to recreate pwd file.**

### Find and update asm spfile

**-- To find current asm spfile:**

$asmcmd  
ASMCMD> spget

**--- To update asm spfile to a new pfile:**

$asmcmd  
ASMCMD> spset +OCR\_VOTING/CLUSTER/ASMPARAMETERFILE/spfileASM.ora

### Copy spfile from one diskgroup to another

ASMCMD> spcopy +DATA/ASM/ASMPARAMETERFILE/registry.263.3728290 +MGMT/spfileCopyASM.ora

### Stop and start db using srvctl

**-- SYNTAX FOR STOP DB**   
--- ***srvctl stop database -d db\_name [-o stop\_options]*** where stop\_options is normal/immediate(default)/transactional/abort  
**e.g**   
srvctl stop database -d PRODB -o normal  
srvctl stop database -d PRODB -o immediate  
srvctl stop database -d PRODB -o transactional  
srvctl stop database -d PRODB -o abort

**-- SYNTAX FOR START DB**   
-- ***srvctl start database -d db\_name [-o start\_options]*** where start\_option is nomount/mount/open(default)

**e.g**   
srvctl start database -d PRODB -o nomount  
srvctl start database -d PRODB -o mount  
srvctl start database -d PRODB -o open

### add/remove db using srvctl

SYNTAX FOR REMOVING DB SERVICE:   
---***srvctl remove database -d db\_unique\_name [-f] [-y] [-v]***  
e.g:  
srvctl remove database -d PRODB -f -y

SYNTAX FOR ADDING DB SERVICE :  
-- ***srvctl add database -d db\_unique\_name -o ORACLE\_HOME [-p spfile]***  
e.g:  
srvctl add database -d PRODB -o /u01/app/oracle/product/12.1.0.2/dbhome\_1 -p +DATA/PRODDB/parameterfile/spfilePRODB.ora

### Add/remove instance using srvctl

SYNTAX FOR REMOVING INSTANCE  
---***srvctl remove instance -d DB\_UNIQUE\_NAME -i INSTANCE\_NAME***  
e.g  
srvctl remove instance -d PRODB - I PRODB1

SYNTAX FOR ADDING INSTANCE   
--- ***srvctl add instance –d db\_unique\_name –i inst\_name -n node\_name***  
e.g  
srvctl add instance -d PRODB - i PRODB1 -n rachost1

### Stop and start instance using srvctl

SYNTAX FOR STOPPING INSTANCE  
***-- srvctl stop instance -d db\_unique\_name [-i "instance\_name\_list"]} [-o stop\_options] [-f]***  
e.g   
srvctl stop instance -d PRODB -i PRODB1

SYNTAX FOR STARTING INSTANCE  
***-- srvctl start instance -d db\_unique\_name [-i "instance\_name\_list"} [-o start\_options]***  
e.g  
srvctl start instance -d PRODB -i PRODB1

### Enable/disable db/instance using srvctl

-- ENABLE - Reenables management by Oracle Restart for a component.  
-- DISABLE - Disables management by Oracle Restart for a component.

***srvctl enable instance -d DB\_UNIQUE\_NAME-i INSTANCE\_NAME***  
***srvctl disable instance -d DB\_UNIQUE\_NAME-i INSTANCE\_NAME***  
***srvctl enable database -d DB\_UNIQUE\_NAME***  
***srvctl disable database -d DB\_UNIQUE\_NAME***

### Relocate a service

**SYNTAX -**

***srvctl relocate service -d {database\_name} -s {service\_name} -i {old\_inst\_name} -r {new\_inst\_name}***

EXAMPLE:(Relocating service PRDB\_SRV from PREDB2 to PREDB1)

**srvctl relocate service -d PREDB -s PRDB\_SVC -i PREDB2 -t PREDB1**

Check the status of service

**srvctl status service -d PREDB -s PRDB\_SVC**

### stop/start a service

SYNTAX:  
---------  
**srvctl start service -d {DB\_NAME} -s {SERVICE\_NAME}**  
**srvctl stop service -d {DB\_NAME} -s {SERVICE\_NAME}**

EXAMPLE:  
---------------  
**srvctl start service -d PREDB -s PRDB\_SRV**  
**srvctl stop service -d PREDB -s PRDB\_SRV**

### Add/remove a service

**ADDING A SERVICE:**  
**--------------------**  
SYNTAX:  
------------  
***srvctl add service -d {DB\_NAME} -s {SERVICE\_NAME} -r {"preferred\_list"} -a {"available\_list"} [-P {BASIC | NONE | PRECONNECT}]***

EXAMPLE:  
---------------  
**srvctl add service -d PREDB -s PRDB\_SRV -r "PREDB1,PREDB2" -a "PREDB2" -P BASIC**

**REMOVING A SERVICE:**  
**------------------------------------------**

SYNTAX:  
-------------  
***srvctl remove service -d {DB\_NAME} -s {SERVICE\_NAME}***

EXAMPLE:  
--------  
**srvctl remove service -d PREDB -s PRDB\_SRV**

### Enable trace for srvctl commands

**-- set this to enable trace at os**

SRVM\_TRACE=true  
export SRVM\_TRACE

**-- run any srvctl command**  
srvctl status database -d ORACL

### Set env variables using srvctl

**-- setenv to set env variables.(ORCL is the db\_unique\_name)**

srvctl setenv database -db ORCL -env "ORACLE\_HOME=/oracle/app/oracle/product/12.1.0.2/dbhome\_1"  
srvctl setenv database -db ORCL -env "TNS\_ADMIN=/oracle/app/oracle/product/12.1.0.2/dbhome\_1/network/admin"

**--getenv to view the env setting:**

srvctl getenv database -db ORCL

### Manage MGMTDB in 12c RAC

**-- check status of mgmtdb in orcle 12c RAC**

srvctl status mgmtdb

**-- stop and start MGMT db.**

srvctl stop mgmtdb  
srvctl start mgmtdb

### asm config using srvctl

srvctl config asm

srvctl config asm -detail

### Enable/Disable autorestart of crs

Run as root user

**$GRID\_HOME/bin/crsctl enable crs**  
CRS-4622: Oracle High Availability Services autostart is enabled.

**$GRID\_HOME/bin/crsctl disable crs**  
CRS-4621: Oracle High Availability Services autostart is disabled.

### Find the cluster name in RAC

**$GRID\_HOME/bin/cemutlo -n**

or

**$GRID\_HOME/bin/olsnodes -c**

### Stop and start CRS

-- stop crs ( run from root)

**$GRID\_HOME/bin/crsctl stop crs**

-- start crs( run from root)

**$GRID\_HOME/bin/crsctl start crs**

### Find OCR and VD location

Find voting disk location

**$GRID\_HOME/bin/crsctl query css votedisk**

Find OCR location.

**$GRID\_HOME/bin/ocrcheck**

### Find the grid version

***SYNTAX*** *- $GRID\_HOME/bin/crsctl query crs softwareversion*

**$GRID\_HOME/bin/crsctl query crs softwareversion**

### check cluster component status

**$GRID\_HOME/bin/crsctl stat res -t**

**$GRID\_HOME/bin/crsctl check crs**

**$GRID\_HOME/bin/crsctl check cssd**

**$GRID\_HOME/bin/crsctl check crsd**

**$GRID\_HOME/bin/crsctl check evmd**

### Get cluster\_interconnect details

**$GRID\_HOME/bin/oifcfg getif**

app-ipmp0 172.21.39.128 global public  
loypredbib0 172.16.3.192 global cluster\_interconnect  
loypredbib1 172.16.4.0 global cluster\_interconnect

**select NAME,IP\_ADDRESS from v$cluster\_interconnects;**

NAME IP\_ADDRESS  
--------------- ----------------  
loypredbib0 172.16.3.193  
loypredbib1 172.16.4.1

### Manual backup of ocr and list backups

List down the backups of OCR

**$GRID\_HOME/bin/ocrconfig -showbackup**

Take manual OCR backup

**$GRID\_HOME/bin/ocrconfig -manualbackup**

### Move voting disk to new diskgroup

**$GRID\_HOME/bin/crsctl replace votedisk +NEW\_DG**

Check the status using below command

**$GRID\_HOME/bin/crsctl query css votedisk**

### get disktimeout values

Disk timeout from node to voting disk(disktimeout)

**crsctl get css disktimeout**

CRS-4678: Successful get disktimeout 200 for Cluster Synchronization Services.

Network latency in the node interconnect (Misscount)

**crsctl get css misscount**

CRS-4678: Successful get misscount 30 for Cluster Synchronization Services.

### get node info using olsnodes

**-- List of nodes in the cluster**  
olsnodes

**-- Nodes with node number**  
olsnodes -n

**-- Node with vip**  
olsnodes -i  
olsnodes -s -t

**-- Leaf or Hub**  
olsnodes -a

**-- Getting private ip details of the local node**  
olsnodes -l -p

**-- Get cluster name**  
olsnodes -c

### Get interface info in RAC

**oifcfg iflist -p -n**

backup0 172.21.56.0 PRIVATE 255.255.254.0  
cdnet0 162.168.1.0 PRIVATE 255.255.255.0  
cdnet0 169.254.0.0 PUBLIC 255.255.128.0  
cdnet1 162.168.2.0 PRIVATE 255.255.255.0  
cdnet1 169.254.128.0 PUBLIC 255.255.128.0  
pap-ipmp0 172.20.179.128 PUBLIC 255.255.255.128  
tan-ipmp0 172.20.128.0 PRIVATE 255.255.252.0  
dppp0 162.168.224.0 PRIVATE 255.255.255.0

### Get OLR info in RAC

-- OLR(ORACLE LOCAL REGISTRY)

Get current OLR location:(run from root only)

**$GRID\_HOME/bin/ocrcheck -local**

List the OLR backups:

**$GRID\_HOME/bin/ocrconfig -local -showbackup**

Take manual OLR backup:

**$GRID\_HOME/bin/ocrconfig -local -manualbackup**

### Get cluster configuration information

$ **crsctl get cluster configuration**  
Name : dbaclass-cluster  
Configuration : Cluster  
Class : Standalone Cluster  
Type : flex  
The cluster is not extended.  
--------------------------------------------------------------------------------  
MEMBER CLUSTER INFORMATION

Name Version GUID Deployed Deconfigured  
================================================================================  
================================================================================

$ **crsctl get node role status -all**  
Node 'hostnode1' active role is 'hub'  
Node 'hostnode2' active role is 'hub'

### Create sql baseline from cursor cache

DECLARE

l\_plans\_loaded PLS\_INTEGER;  
BEGIN  
l\_plans\_loaded := DBMS\_SPM.load\_plans\_from\_cursor\_cache(  
sql\_id => '&sql\_id');  
END;  
/

**-- Create baseline with a particular hash value**

DECLARE  
l\_plans\_loaded PLS\_INTEGER;  
BEGIN  
l\_plans\_loaded := DBMS\_SPM.load\_plans\_from\_cursor\_cache(  
sql\_id => '&sql\_id', plan\_hash\_value => '&plan\_hash\_value');  
END;  
/

### drop a sql baseline

**declare**   
**drop\_result pls\_integer;**  
**begin**   
**drop\_result := DBMS\_SPM.DROP\_SQL\_PLAN\_BASELINE(**   
**plan\_name => '&sql\_plan\_baseline\_name');**   
**dbms\_output.put\_line(drop\_result);**   
**end;**   
**/**

*You can get the sql baseline from a sql\_id from below command:*

*SELECT sql\_handle, plan\_name FROM dba\_sql\_plan\_baselines WHERE signature IN*   
*( SELECT exact\_matching\_signature FROM gv$sql WHERE sql\_id='&SQL\_ID');*

### create baselines for all sqls of a schema

**DECLARE**   
**nRet NUMBER;**  
**BEGIN**  
**nRet := dbms\_spm.load\_plans\_from\_cursor\_cache(**  
**attribute\_name => 'PARSING\_SCHEMA\_NAME',**  
**attribute\_value => '&schema\_name'**  
**);**  
**END;**

### drop a sql profile

**BEGIN**  
**DBMS\_SQLTUNE.drop\_sql\_profile (**  
**name => '&sql\_profile',**  
**ignore => TRUE);**  
**END;**  
**/**

*You can get the respective sql\_profile of a sql\_id from below:*

***select distinct***   
***p.name sql\_profile\_name,***  
***s.sql\_id***  
***from***   
***dba\_sql\_profiles p,***  
***DBA\_HIST\_SQLSTAT s***  
***where***  
***p.name=s.sql\_profile and s.sql\_id='&sql\_id';***

### Get sql\_profile of a sql\_id

-- Script for getting sql\_profile created for a sql\_id

**select distinct**   
**p.name sql\_profile\_name,**  
**s.sql\_id**  
**from**   
**dba\_sql\_profiles p,**  
**DBA\_HIST\_SQLSTAT s**  
**where**  
**p.name=s.sql\_profile and s.sql\_id='&sql\_id';**

### sql tuning advisor for sql\_id

Create tuning task:

**DECLARE**  
**l\_sql\_tune\_task\_id VARCHAR2(100);**  
**BEGIN**  
**l\_sql\_tune\_task\_id := DBMS\_SQLTUNE.create\_tuning\_task (**  
**sql\_id => '12xca9smf3hfy',**  
**scope => DBMS\_SQLTUNE.scope\_comprehensive,**  
**time\_limit => 500,**  
**task\_name => '12xca9smf3hfy\_tuning\_task',**  
**description => 'Tuning task1 for statement 12xca9smf3hfy');**  
**DBMS\_OUTPUT.put\_line('l\_sql\_tune\_task\_id: ' || l\_sql\_tune\_task\_id);**  
**END;**  
**/**

Execute tuning task:

**EXEC DBMS\_SQLTUNE.execute\_tuning\_task(task\_name => '12xca9smf3hfy\_tuning\_task');**

Get the tuning advisory report

**set long 65536**  
**set longchunksize 65536**  
**set linesize 100**  
**select dbms\_sqltune.report\_tuning\_task('12xca9smf3hfy\_tuning\_task') from dual;**

### Disable/enable sql profile

**EXEC DBMS\_SQLTUNE.ALTER\_SQL\_PROFILE('&sql\_profile\_name','STATUS','DISABLED');**

### Find sql baseline info from sql\_id

Pass the sql\_id to get the respective sql baseline

**SELECT sql\_handle, plan\_name FROM dba\_sql\_plan\_baselines WHERE**  
**signature IN ( SELECT exact\_matching\_signature FROM gv$sql WHERE sql\_id='&SQL\_ID')**

### Alter/disable a sql plan baseline

--- To disable a baseline:

**Begin**  
**dbms\_spm.alter\_sql\_plan\_baseline(sql\_handle =>'SQL\_SQL\_5818768f40d7be2a',**  
**plan\_name => 'SQL\_PLAN\_aaxsg8yktm4h100404251',**  
**attribute\_name=> 'enabled',**  
**attribute\_value=>'NO');**  
**END;**  
**/**  
**Begin**  
**dbms\_spm.alter\_sql\_plan\_baseline(sql\_handle =>'SQL\_SQL\_5818768f40d7be2a',**  
**plan\_name => 'SQL\_PLAN\_aaxsg8yktm4h100404251',**  
**attribute\_name=> 'fixed',**  
**attribute\_value=>'NO');**  
**END;**  
**/**

-- To enable again, just modify the attribute\_value to YES,

### Adding partitions 11g/12c

-- SYNTAX : ***ALTER TABLE <SCHEMA\_NAME>.<TABLE\_NAME> ADD PARTITION < PARTITION\_NAME> VALUES LESS THAN < HIGH\_VALUE> TABLESPACE <TABLESPACE\_NAME > < UPDATE GLOBAL INDEXES(optional)>;***  
-- NOTE: UPDATE GLOBAL INDEXES is required if GLOBAL INDEX is present

**ALTER TABLE CMADMIN.DBACLASS ADD PARTITION DBACLASS\_JAN VALUES**  
**LESS THAN (TO\_DATE('01-FEB-2016','DD-MON-YYYY')) TABLESPACE USERS UPDATE GLOBAL INDEXES;**

-- In oracle 12c(new feature), we can add multiple partition in one command:

**ALTER TABLE CMADMIN.DBACLASS ADD**   
**PARTITION DBACLASS\_JAN VALUES LESS THAN (TO\_DATE('01-FEB-2016','DD-MON-YYYY')) TABLESPACE USERS,**  
**PARTITION DBACLASS\_FEB VALUES LESS THAN (TO\_DATE('01-MAR-2016','DD-MON-YYYY')) TABLESPACE USERS,**  
**PARTITION DBACLASS\_MAR VALUES LESS THAN (TO\_DATE('01-APR-2016','DD-MON-YYYY')) TABLESPACE USERS,**  
**UPDATE GLOBAL INDEXES;**

### Dropping partition 11g/12c

-- SYNTAX : ***ALTER TABLE <SCHEMA\_NAME>.<TABLE\_NAME> DROP PARTITION < PARTITION\_NAME> < UPDATE GLOBAL INDEXES(optional)>;***  
--- NOTE: UPDATE GLOBAL INDEXES is required if GLOBAL INDEX is present

**ALTER TABLE CMADMIN.DBACLASS DROP PARTITION DBACLASS\_JAN UPDATE GLOBAL INDEXES;**

--- In oracle 12c, we can drop multiple partitions in one command

**ALTER TABLE CMADMIN.DBACLASS DROP PARTITIONS DBACLASS\_JAN, DBACLASS\_FEB, DBACLASS\_MAR UPDATE GLOBAL INDEXES;**

### Truncate partitions

- SYNTAX : ***ALTER TABLE <SCHEMA\_NAME>.<TABLE\_NAME> TRUNCATE PARTITION < PARTITION\_NAME> < UPDATE GLOBAL INDEXES(optional)>;***  
--- NOTE: UPDATE GLOBAL INDEXES is required if GLOBAL INDEX is present

**ALTER TABLE CMADMIN.DBACLASS TRUNCATE PARTITION DBACLASS\_JAN UPDATE GLOBAL INDEXES;**

--- In oracle 12c, we can truncate multiple partitions in one command

**ALTER TABLE CMADMIN.DBACLASS TRUNCATE PARTITIONS DBACLASS\_JAN, DBACLASS\_FEB, DBACLASS\_MAR UPDATE GLOBAL INDEXES;**

### Make a partition ready only(12CR2)

-- From oracle 12.2.0.1 Relase, we can make few partitions of a table read only.

SQL> **alter table dbatest.ORDER\_TAB modify partition CREATED\_2105\_P10 read only;**

Table altered.

SQL> **select partition\_name,read\_only from dba\_tab\_partitions where table\_name='ORDER\_TAB';**

PARTITION\_NAME READ  
-------------------------------- ----  
CREATED\_2105\_P10 YES  
CREATED\_2105\_P11 NO  
CREATED\_2105\_P12 NO  
CREATED\_2105\_P8 NO  
CREATED\_2105\_P9 NO  
CREATED\_MX NO

6 rows selected.

### Split partition online(12cR2 only)

SQL> **alter table order\_tab split partition CREATED\_MX into**  
**(partition CREATED\_2106\_P2 VALUES LESS THAN (TO\_DATE('01/03/2016', 'DD/MM/YYYY')),PARTITION CREATED\_MX) ONLINE;**

Table altered.

SQL> **select partition\_name,read\_only,high\_value from dba\_tab\_partitions where table\_name='ORDER\_TAB';**

### Non-partitioned to partitioned online(12CR2 only)

-- In Oracle 12cR2, we can convert non partitioned table to partitioned online using alter table command.

**alter table BSSTDBA.ORDER\_TAB modify**  
**PARTITION BY RANGE (CREATED)**  
**(partition created\_2105\_p8 VALUES LESS THAN (TO\_DATE('01/09/2015', 'DD/MM/YYYY')),**  
**partition created\_2105\_p9 VALUES LESS THAN (TO\_DATE('01/10/2015', 'DD/MM/YYYY')),**  
**partition created\_2105\_p10 VALUES LESS THAN (TO\_DATE('01/11/2015', 'DD/MM/YYYY')),**  
**partition created\_2105\_p11 VALUES LESS THAN (TO\_DATE('01/12/2015', 'DD/MM/YYYY')),**  
**partition created\_2105\_p12 VALUES LESS THAN (TO\_DATE('01/01/2016', 'DD/MM/YYYY')),**  
**PARTITION Created\_MX VALUES LESS THAN (MAXVALUE)) ONLINE;**

### Rename a partition

**ALTER TABLE employee RENAME PARTITION TAB3 TO TAB4;**

### Get row\_count of partitions of a table

**set serverout on size 1000000**  
**set verify off**  
**declare**  
**sql\_stmt varchar2(1024);**  
**row\_count number;**  
**cursor get\_tab is**  
**select table\_name,partition\_name**  
**from dba\_tab\_partitions**  
**where table\_owner=upper('&&TABLE\_OWNER') and table\_name='&&TABLE\_NAME';**  
**begin**  
**dbms\_output.put\_line('Checking Record Counts for table\_name');**  
**dbms\_output.put\_line('Log file to numrows\_part\_&&TABLE\_OWNER.lst ....');**  
**dbms\_output.put\_line('....');**  
**for get\_tab\_rec in get\_tab loop**  
**BEGIN**  
**sql\_stmt := 'select count(\*) from &&TABLE\_OWNER..'||get\_tab\_rec.table\_name**  
**||' partition ( '||get\_tab\_rec.partition\_name||' )';**

**EXECUTE IMMEDIATE sql\_stmt INTO row\_count;**  
**dbms\_output.put\_line('Table '||rpad(get\_tab\_rec.table\_name**  
**||'('||get\_tab\_rec.partition\_name||')',50)**  
**||' '||TO\_CHAR(row\_count)||' rows.');**  
**exception when others then**  
**dbms\_output.put\_line**  
**('Error counting rows for table '||get\_tab\_rec.table\_name);**  
**END;**  
**end loop;**  
**end;**  
**/**  
**set verify on**

### Find the table partition keys

--- describes the partitioning key columns for all partitioned objects of a schema

**set pagesize 200**  
**set lines 200**  
**set long 999**  
**col owner for a12**  
**col name for a20**  
**col object\_type for a20**  
**col column\_name for a32**  
**SELECT owner, NAME, OBJECT\_TYPE,column\_name**  
**FROM dba\_part\_key\_columns where owner='&OWNER'**  
**ORDER BY owner, NAME;**

### Move partition to new tablespace

- Move a single partition to a new tablespace

**ALTER TABLE SCOTT.EMP MOVE PARTITION EMP\_Q1 TABLESPACE TS\_USERS;**

--- Move a single partition to a new tablespace WITH PARALLEL

**ALTER TABLE SCOTT.EMP MOVE PARTITION EMP\_Q1 TABLESPACE TS\_USERS PARALLEL(DEGREE 4) NOLOGGING;**

- Dynamic script to move all partitions of a table

**select**  
**'ALTER TABLE '||TABLE\_OWNER ||'.'||table\_name||' MOVE PARTITION '||partition\_name||' TABLESPACE TS\_USERS PARALLEL(DEGREE 4) NOLOGGING;'**  
**from dba\_tab\_partitions where table\_name='&TABLE\_NAME' and table\_owner='&SCHEMA\_NAME';**

### Gather stats for a table

**BEGIN**  
**DBMS\_STATS.GATHER\_TABLE\_STATS (**  
**ownname => 'SCOTT',**  
**tabname => 'TEST',**  
**cascade => true, ---- For collecting stats for respective indexes**  
**method\_opt=>'for all indexed columns size 1',**  
**granularity => 'ALL',**  
**estimate\_percent =>dbms\_stats.auto\_sample\_size,**  
**degree => 8);**  
**END;**  
**/**

-- For a single table partition

**BEGIN**  
**DBMS\_STATS.GATHER\_TABLE\_STATS (**  
**ownname => 'SCOTT',**  
**tabname => 'TEST', --- TABLE NAME**  
**partname => 'TEST\_JAN2016' --- PARTITOIN NAME**  
**method\_opt=>'for all indexed columns size 1',**  
**GRANULARITY => 'APPROX\_GLOBAL AND PARTITION',**  
**degree => 8);**  
**END;**  
**/**

### Gather stats for schema

**Begin**  
**dbms\_stats.gather\_schema\_stats(**  
**ownname => 'SCOTT', --- schema name**  
**options => 'GATHER AUTO',**  
**estimate\_percent => dbms\_stats.auto\_sample\_size,**  
**method\_opt => 'for all columns size repeat',**  
**degree => 24**  
**);**  
**END;**  
**/**

### Lock/unlock statistics

--- Lock statistics

**EXEC DBMS\_STATS.lock\_schema\_stats('SCOTT');**  
**EXEC DBMS\_STATS.lock\_table\_stats('SCOTT', 'TEST');**  
**EXEC DBMS\_STATS.lock\_partition\_stats('SCOTT', 'TEST', 'TEST\_JAN2016');**

-- Unlock statistics

**EXEC DBMS\_STATS.unlock\_schema\_stats('SCOTT');**  
**EXEC DBMS\_STATS.unlock\_table\_stats('SCOTT', 'TEST');**  
**EXEC DBMS\_STATS.unlock\_partition\_stats('SCOTT', 'TEST', 'TEST\_JAN2016');**

--- check stats status:

**SELECT stattype\_locked FROM dba\_tab\_statistics WHERE table\_name = 'TEST' and owner = 'SCOTT';**

### Export import statistics

--- Create staging table to store the statistics data

**exec dbms\_stats.create\_stat\_table(ownname => 'SCOTT', stattab => 'STAT\_BACKUP',tblspace=>'USERS');**

-- Export stats

**exec dbms\_stats.export\_table\_stats(ownname=>'SCOTT', tabname=>'EMP', stattab=>'STAT\_BACKUP', cascade=>true);**

-- Import stats

**exec dbms\_stats.import\_table\_stats(ownname=>'SCOTT', tabname=>'EMP', stattab=>'STAT\_BACKUP', cascade=>true);**

### Check stale stats

STALE STATS FOR TABLE

**select owner,table\_name,STALE\_STATS from dba\_tab\_statistics where owner='&SCHEMA\_NAME' and table\_name='&TABLE\_NAME';**

FOR INDEX

**select owner,INDEX\_NAME,TABLE\_NAME from DBA\_IND\_STATISTICS where owner='&SCHEMA\_NAME' and index\_name='&INDEX\_NAME';**

### Table statistics history

For getting history of TABLE statistics

**setlines 200**  
**col owner for a12**  
**col table\_name for a21**  
**select owner,TABLE\_NAME,STATS\_UPDATE\_TIME from dba\_tab\_stats\_history where table\_name='&TABLE\_NAME';**

### Publish Pending stats

Publish Pending stats for table

**EXEC DBMS\_STATS.PUBLISH\_PENDING\_STATS ('SCHEMA\_NAME,'TABLE\_NAME');**

Publish pending stats for a schema

**exec dbms\_stats.publish\_pending\_stats('SCHEMA\_NAME',null);**

### Get statistics preference setting

Setting Publish preference

**exec dbms\_stats.set\_table\_prefs('SCOTT','EMP','PUBLISH','FALSE');**

Check the publish preference status

**select dbms\_stats.get\_prefs('PUBLISH', 'SCOTT','EMP') FROM DUAL;**

Similarly for schema also use as below:

**select dbms\_stats.get\_prefs('PUBLISH', 'SCOTT') from dual**

**exec dbms\_stats.SET\_SCHEMA\_PREFS('DBATEST','PUBLISH','FALSE');**

--- FOR INDEX

**SET\_INDEX\_STATS**  
**GET\_INDEX\_STATS**

-- FOR DATABASE

**SET\_DATABASE\_PREFS**

-- View current stats retention

**select dbms\_stats.get\_stats\_history\_retention from dual;**

-- Modify the stats retention

**exec DBMS\_STATS.ALTER\_STATS\_HISTORY\_RETENTION(60);**

### Space used to store stats

--- Space currently used to store statistics in SYSAUX in KBytes,

**select occupant\_desc, space\_usage\_kbytes from v$sysaux\_occupants**  
**where OCCUPANT\_DESC like '%Statistics%';**

### Enable incremental stats collection

-- Check the status of incremental pref

**select dbms\_stats.get\_prefs('INCREMENTAL', tabname=>'EMPLOYEE',ownname=>'SCOTT') from dual;**

FALSE

-- Enable incremental stats collection

**SQL> exec DBMS\_STATS.SET\_TABLE\_PREFS('SCOTT','EMPLOYEE','INCREMENTAL','TRUE');**

PL/SQL procedure successfully completed.

-- Check the pref again:

**select dbms\_stats.get\_prefs('INCREMENTAL', tabname=>'EMPLOYEE',ownname=>'SCOTT') from dual;**

TRUE

### Delete statistics

Delete statistics of the complete database

**EXEC DBMS\_STATS.delete\_database\_stats;**

-- Delete statistics of a single schema

**EXEC DBMS\_STATS.delete\_schema\_stats('DBACLASS');**

-- Delete statistics of single tabale  
**EXEC DBMS\_STATS.delete\_table\_stats('DBACLASS', 'DEPT');**

-- Delete statistics of a column  
**EXEC DBMS\_STATS.delete\_column\_stats('DBACLASS', 'DEPT', 'CLASS');**

--Delete statistics of an index

**EXEC DBMS\_STATS.delete\_index\_stats('DBACLASS', 'CLASS\_IDX');**

--Delete dictionary a in db

**EXEC DBMS\_STATS.delete\_dictionary\_stats;**

### Upgrade statistics in db

***-- If we are importing stats table from higher version to lower version,***  
***then before importing in the database, we need to upgrade the stats table.***

**EXECUTE DBMS\_STATS.UPGRADE\_STAT\_TABLE(OWNNAME =>'RAJ',STATTAB =>'STAT\_TEST');**

### Flashback a table to point in time

**ALTER TABLE DBACLASS.EMP ENABLE ROW MOVEMENT;**  
**FLASHBACK TABLE DBACLASS.EMP TO TIMESTAMP**  
**TO\_TIMESTAMP('2017-01-10 09:00:00', `YYYY-MM-DD HH24:MI:SS');**

### Recover a dropped table

Restore the dropped table with same name:

**SQL>flashback table DBACLASS.EMP to before drop;**

Restore the dropped table with a new name

**SQL>Flashback table DBACLASS.EMP to before drop rename to EMP\_BACKUP;**

Note - To recover the table, table should be present in recyclebin:

**select \* from dba\_recyclebin;**

### Flashback query as of timestamp

**SELECT \* FROM DBACLASS.EMP AS OF TIMESTAMP**   
**TO\_TIMESTAMP('2017-01-07 10:00:00', 'YYYY-MM-DD HH:MI:SS');**

**SELECT \* FROM DBACLASS.EMP AS OF TIMESTAMP SYSDATE -1/24;**

### Enable flashback for database

Make sure database is in archivelog mode

**alter system set db\_recovery\_file\_dest\_size=10G scope=both;**  
**alter system set db\_recovery\_file\_dest='/dumparea/FRA/B2BRBMT3' scope=both;**  
**alter database flashback on;**

### Create/drop flashback restore point

-- To create a guarantee flashback restore point;

**SQL>create restore point BEFORE\_UPG guarantee flashback database;**

-- Check the restore\_points present in database

**SQL>select \* from v$restore\_point;**

-- Drop restore point;

**SQL> drop restore point BEFORE\_UPG;**

### Flashback db using restore point

--- Below are the steps for flashback database to a guaranteed restore point;

1. Get the restore point name:

**SQL> select NAME,time from v$restore\_point;**

NAME TIME  
-------------------------------- -----------------------------------------------  
GRP\_1490100093811 21-MAR-17 03.41.33.000000000 PM

2. Shutdown database and start db in Mount stage:

**shutdown immediate;**  
**startup mount;**

3. flashback db to restore point:

**flashback database to restore point GRP\_1490100093811;**

4. Open with resetlog:

**alter database open resetlogs:**

### Flashback a procedure/package

--- Like, tables ,If you have dropped or recreated a package/procedure, by using flashback ,we can get the proc code, before drop.

get the object\_id

SQL> **select object\_id from dba\_objects where owner='DBACLASS' and object\_name='VOL\_DISCOUNT\_INSERT';**

OBJECT\_ID  
----------  
2201943

Now get the flashback code using timestamp

**select SOURCE from sys.source$ as of timestamp**  
**to\_timestamp('23-Apr-2017 10:00:20','DD-Mon-YYYY hh24:MI:SS')**  
**where obj#=2201943 ;**

### How far we can flashback

--How Far Back Can We Flashback To (Time)

**select to\_char(oldest\_flashback\_time,’dd-mon-yyyy hh24:mi:ss’) “Oldest Flashback Time”**  
**from v$flashback\_database\_log;**

--How Far Back Can We Flashback To (SCN)

**col oldest\_flashback\_scn format 99999999999999999999999999**  
**select oldest\_flashback\_scn from v$flashback\_database\_log;**

### Flashback a table to point in time

**ALTER TABLE DBACLASS.EMP ENABLE ROW MOVEMENT;**  
**FLASHBACK TABLE DBACLASS.EMP TO TIMESTAMP**  
**TO\_TIMESTAMP('2017-01-10 09:00:00', `YYYY-MM-DD HH24:MI:SS');**

### Flashback a table to point in time

**ALTER TABLE DBACLASS.EMP ENABLE ROW MOVEMENT;**  
**FLASHBACK TABLE DBACLASS.EMP TO TIMESTAMP**  
**TO\_TIMESTAMP('2017-01-10 09:00:00', `YYYY-MM-DD HH24:MI:SS');**

### Flashback a table to point in time

**ALTER TABLE DBACLASS.EMP ENABLE ROW MOVEMENT;**  
**FLASHBACK TABLE DBACLASS.EMP TO TIMESTAMP**  
**TO\_TIMESTAMP('2017-01-10 09:00:00', `YYYY-MM-DD HH24:MI:SS');**

### rman full database backup script

**configure backup optimization on;**  
**configure controlfile autobackup on;**  
**configure controlfile autobackup format for device type disk to '/archiva/backup/%F';**  
**configure maxsetsize to unlimited;**  
**configure device type disk parallelism 4;**  
**run**  
**{**  
**allocate channel c1 type disk format '/archiva/backup/%I-%Y%M%D-%U' maxpiecesize 3G;**  
**allocate channel c2 type disk format '/archiva/backup/%I-%Y%M%D-%U' maxpiecesize 3G;**  
**allocate channel c3 type disk format '/archiva/backup/%I-%Y%M%D-%U' maxpiecesize 3G;**  
**allocate channel c4 type disk format '/archiva/backup/%I-%Y%M%D-%U' maxpiecesize 3G;**  
**backup as compressed backupset incremental level 0 check logical database plus archivelog;**  
**release channel c1 ;**  
**release channel c2 ;**  
**release channel c3 ;**  
**release channel c4 ;**  
**}**

### RMAN INCR db backup run block

**configure backup optimization on;**  
**configure controlfile autobackup on;**  
**configure controlfile autobackup format for device type disk to '/archiva/backup/%F';**  
**configure maxsetsize to unlimited;**  
**configure device type disk parallelism 4;**  
**run**  
**{**  
**allocate channel c1 type disk format '/archiva/backup/%I-%Y%M%D-%U' maxpiecesize 3G;**  
**allocate channel c2 type disk format '/archiva/backup/%I-%Y%M%D-%U' maxpiecesize 3G;**  
**allocate channel c3 type disk format '/archiva/backup/%I-%Y%M%D-%U' maxpiecesize 3G;**  
**allocate channel c4 type disk format '/archiva/backup/%I-%Y%M%D-%U' maxpiecesize 3G;**  
**backup as compressed backupset incremental level 1 check logical database plus archivelog;**  
**release channel c1 ;**  
**release channel c2 ;**  
**release channel c3 ;**  
**release channel c4 ;**  
**}**

### rman tablespace backup run block

**configure controlfile autobackup on;**  
**configure controlfile autobackup format for device type disk to '/archiva/backup/%F';**  
**configure maxsetsize to unlimited;**  
**configure device type disk parallelism 4;**  
**run**  
**{**  
**allocate channel c1 type disk format '/archiva/backup/%I-%Y%M%D-%U' maxpiecesize 3G;**  
**allocate channel c2 type disk format '/archiva/backup/%I-%Y%M%D-%U' maxpiecesize 3G;**  
**backup tablespace USERS,TOOLS;**  
**release channel c1 ;**  
**release channel c2 ;**  
**}**

### RMAN datafile(s) backup run block

**configure controlfile autobackup on;**  
**configure controlfile autobackup format for device type disk to '/archiva/backup/%F';**  
**configure maxsetsize to unlimited;**  
**configure device type disk parallelism 4;**  
**run**  
**{**  
**allocate channel c1 type disk format '/archiva/backup/%I-%Y%M%D-%U' maxpiecesize 3G;**  
**allocate channel c2 type disk format '/archiva/backup/%I-%Y%M%D-%U' maxpiecesize 3g;**  
**backup datafile 3,4;**  
**release channel c1 ;**  
**release channel c2 ;**  
**}**

### delete archive older than 1 day

**DELETE ARCHIVELOG ALL COMPLETED BEFORE 'sysdate-1';**  
**CROSSCHECK ARCHIVELOG ALL;**  
**DELETE EXPIRED ARCHIVELOG ALL;**

### backup archivelogs using RMAN

Backup all archivelogs known to controlfile

**backup archivelog all;**

Backup all archivelogs known to controlfile and delete them once backed up

**backup archivelog all delete input ;**

Backup archivlogs known to controlfile and the logs which haven't backed up once also

**backup archivelog all not backed up 1 times;**

### Copy archive from ASM to Mount point

--- Copy archive log from ASM to regular mount point using RMAN:  
--- Connect to RMAN in RAC db

RMAN> **copy archivelog '+B2BSTARC/thread\_2\_seq\_34.933' to '/data/thread\_2\_seq\_34.933';**

### backup archive between 2 sequence number

For taking backup of archivelog between seq number 1000 to 1050

RMAN> **backup format '/archive/%d\_%s\_%p\_%c\_%t.arc.bkp'**  
**archivelog from sequence 1000 until sequence 1050;**

For RAC ,need to mention the thread number also

RMAN> **backup format '/archive/%d\_%s\_%p\_%c\_%t.arc.bkp'**  
**archivelog from sequence 1000 until sequence 1050 thread 2;**

### Enable trace for RMAN

-- To diagnose rman script, use trace as below.

**spool trace to '/**tmp**/rman\_trace.out';**  
report schema;  
list backup summary;  
list backup of datafile 1;  
list copy of datafile 1;  
**spool trace off;**

### Recover dropped table with RMAN 12c

RMAN>**recover table SCOTT.SALGRADE until time “to\_date(’08/09/2016 18:49:40',’mm/dd/yyyy hh24:mi:ss’)”**  
**auxiliary destination ‘/u03/arch/TEST/BACKUP’**  
**datapump destination ‘/u03/arch/TEST/BACKUP';**

*auxiliary destination* – Location where all the related files for auxiliary instance will be placed

*datapump destination* – Location where the export dump of the table will be placed

NOTE - This feature is available only in oracle 12c and later.

**SELECT SID, SERIAL#, CONTEXT, SOFAR, TOTALWORK,**  
**ROUND(SOFAR/TOTALWORK\*100,2) "%\_COMPLETE"**  
**FROM V$SESSION\_LONGOPS**  
**WHERE OPNAME LIKE 'RMAN%'**  
**AND OPNAME NOT LIKE '%aggregate%'**  
**AND TOTALWORK != 0**  
**AND SOFAR <> TOTALWORK;**

### Restore archivelog from rman tape

-----Below script will restore the archive sequences from 7630 to 7640 to /dumparea location

**connect target sys/\*\*\*\*\*\*@CRM\_DB**  
**connect catalog RMAN\_tst/\*\*\*\*\*@catdb**

**run**  
**{**  
**allocate channel t1 type SBT\_TAPE parms ‘ENV=(NSR\_SERVER=nwwerpw,NSR\_CLIENT=tsc\_test01,NSR\_DATA\_VOLUME\_POOL=DD086A1)’connect sys/\*\*\*\*@CRM\_DB;**  
**set archivelog destination to ‘/dumparea/';**  
**restore archivelog from sequence 7630 until sequence 7640;**  
**release channel t1;**  
**}**

### Enable block change tracking

-- Enable block change tracking

**alter database enable block change tracking using file**  
**'/export/home/oracle/RMAN/TESTDB/TRACKING\_FILE/block\_change\_TESTDB.log';**

-- Check status:

**select filename,status from v$block\_change\_tracking;**

### check the syntax of RMAN commands

--- check the syntax of RMAN commands interactively without actually executing the commands

$ **rman checksyntax**

Recovery Manager: Release 12.1.0.2.0 - Production on Sun Jan 29 12:04:24 2017

Copyright (c) 1982, 2014, Oracle and/or its affiliates. All rights reserved.

-- Now put the command for checking syntax

RMAN> **backup database;**

The command has no syntax errors

### Create user in oracle

***SYNTAX :***

***create user identified by default tablespace temporary tablespace ;***

***Eg:***

**create user SCOTT identified by oracle#41234**  
**default tablespace users**  
**temporary tablespace TEMP;**

-To create an user, which will prompt for new password upon login:

**create user SCOTT identified by oracle#41234**  
**default tablespace users**  
**temporary tablespace TEMP**  
**password expire;**

### Alter an user

-- Change password of an user

**ALTER USER SCOTT identified by NEW\_PWD;**

-- Change user profile;

**ALTER USER SCOTT PROFILE SIEBEL\_PROFILE;**

-- Unlock/lock a user

**ALTER USER SCOTT account unlock;**  
**ALTER USER SCOTT account lock;**

-- Make sure account expiry, so upon login, it will ask for new one

**ALTER USER SCOTT password expire;**

### Change default tablespace of user

-- Get default tablespace of a user:

**set lines 200**  
**col username for a23**  
**select username,DEFAULT\_TABLESPACE from dba\_users where username='SCOTT';**

USERNAME DEFAULT\_TABLESPACE  
----------------------- ------------------------------  
SCOTT USERS

-- Change default tablespace of a user:

**ALTER USER SCOTT DEFAULT TABLESPACE DATATS;**

**select username,DEFAULT\_TABLESPACE from dba\_users where username='SCOTT';**

USERNAME DEFAULT\_TABLESPACE  
----------------------- ------------------------------  
SCOTT DATATS

### Tablespace quota for a user

-- Get the current tablespace quota information of an user  
**set lines 299**  
**select TABLESPACE\_NAME,BYTES/1024/1024 "UTILIZIED\_SPACE" ,MAX\_BYTES/1024/1024 "QUOTA\_ALLOCATED" from dba\_ts\_quotas where username='&USER\_NAME';**

TABLESPACE\_NAME UTILIZIED\_SPACE QUOTA\_ALLOCATED  
------------------------------ --------------------------- --------------------------  
USERS .0625 1024

--- Change the tablespace quota for the user to 5G

**ALTER USER SCOTT QUOTA 5G ON USERS;**

--- Grant unlimited tablespace quota:

**ALTER USER SCOTT QUOTA UNLIMITED ON USERS;**

### View Privileges granted to an user

-- System privileges granted to an user ( scott)

**SELECT \* FROM DBA\_SYS\_PRIVS where grantee='SCOTT';**

-- Roles granted to an user ( scott)

**SELECT \* FROM DBA\_ROLE\_PRIVS where grantee='SCOTT';**

-- Object privileges granted to an user ( SCOTT)

**SELECT \* FROM DBA\_TAB\_PRIVS WHERE GRANTEE='SCOTT';**

-- Column specific privileges granted

**SELECT \* FROM DBA\_COL\_PRIVS WHERE WHERE GRANTEE='SCOTT';**

### grant table/column privilege to user

-- Table privileges

**GRANT READ ANY TABLE TO SCOTT;**

**GRANT SELECT ANY TABLE TO SCOTT;**

**GRANT INSERT, UPDATE, DELETE ON TESTUSER1.EMPTABL on SCOTT;**  
**GRANT ALL ON TESTUSER1.EMPTABL on SCOTT;**

-- Grant privilege on few columns of a table  
--Only INSERT,UPDATE can be granted at COLUMN level.

**GRANT insert (emp\_id) ON TESTUSER1.EMPTABL TO SCOTT;**  
**GRANT UPDATE(emp\_id) ON TESTUSER1.EMPTABL TO SCOTT;**

### Connect to user without knowing password

*--- You can connect to another user without knowing the password, with grant connect through privilege*  
*--- Suppose a user TEST1 wants to connect to TEST2 user and create a table and we don’t know the password of TEST2.*

**Conn / as sysdba**  
**SQL >alter user TEST2 grant connect through TEST1;**

**User altered.**

**SQL >conn TEST1[TEST2]**  
**Enter password:< Give password for TEST1>**

**SQL >show user**  
**USER is "TEST2"**  
**SQL >create table emp\_test as select \* from emp;**

**Table created.**

**SQL > conn / as sysdba**  
**connected**  
**SQL > select owner from dba\_tables where table\_name='EMP\_TEST';**

**OWNER**  
**------**  
**TEST2**

### Common user/role in CDB

---A user that is present in both root container and PDB is known as common user. User need to be created after connecting to CDB root.

**create user c##dbaclass identified by dbaclass container=all;**

-- Similar to user, common role we can create in CDB root.

**Create role C##DBAROLE;**

### User creation details in user$ table

**SELECT NAME, type#, ctime, ptime, exptime, ltime**  
**FROM sys.user$**  
**WHERE NAME IN ('SYS', 'SYSTEM')**  
**ORDER BY NAME;**

NOTE:

CTIME -> USER CREATION TIME  
PTIME -> LAST PASSWORD CHANGE TIME  
EXPTIME -> PASSWORD EXPIRY DATE  
LTIME - > ACCOUNT LOCK TIME

### Create /alter profile in database

**-- CREATE PROFILE**

**CREATE PROFILE "APP\_PROFILE"**  
**LIMIT**  
**COMPOSITE\_LIMIT UNLIMITED**  
**SESSIONS\_PER\_USER UNLIMITED**  
**CPU\_PER\_SESSION UNLIMITED**  
**CPU\_PER\_CALL UNLIMITED**  
**LOGICAL\_READS\_PER\_SESSION UNLIMITED**  
**LOGICAL\_READS\_PER\_CALL UNLIMITED**  
**IDLE\_TIME 90**  
**CONNECT\_TIME UNLIMITED**  
**PRIVATE\_SGA UNLIMITED**  
**FAILED\_LOGIN\_ATTEMPTS 10**  
**PASSWORD\_LIFE\_TIME 180**  
**PASSWORD\_REUSE\_TIME UNLIMITED**  
**PASSWORD\_REUSE\_MAX UNLIMITED**  
**PASSWORD\_VERIFY\_FUNCTION NULL**  
**PASSWORD\_LOCK\_TIME UNLIMITED**  
**PASSWORD\_GRACE\_TIME UNLIMITED;**

**-- ALTER PROFILE:**

**ALTER PROFILE APP\_PROFILE LIMIT PASSWORD\_LIFE\_TIME UNLIMITED;**

\*SESSION\_PER\_USER – No. of allowed concurrent sessions for a user  
\*CPU\_PER\_SESSION – CPU time limit for a session, expressed in hundredth of seconds.  
\*CPU\_PER\_CALL – Specify the CPU time limit for a call (a parse, execute, or fetch), expressed in hundredths of seconds.  
\*CONNECT\_TIME – Specify the total elapsed time limit for a session, expressed in minutes.  
\*IDLE\_TIME – Specify the permitted periods of continuous inactive time during a session, expressed in minutes.  
\*LOGICAL\_READS\_PER\_SESSION – Specify the permitted number of data blocks read in a session, including blocks read from memory and disk  
\*LOGICAL\_READS\_PER\_CALL –permitted number of data blocks read for a call to process a SQL statement (a parse, execute, or fetch).  
\*PRIVATE\_SGA – SGA a session can allocate in the shared pool of the system global area (SGA), expressed in bytes.

\*FAILED\_LOGIN\_ATTEMPTS – No. of failed attempts to log in to the user account before the account is locked  
\*PASSWORD\_LIFE\_TIME : No. of days the account will be open. after that it will expiry.  
\*PASSWORD\_REUSE\_TIME : number of days before which a password cannot be reused  
\*PASSWORD\_REUSE\_MAX : number of days before which a password can be reused  
\*PASSWORD\_LOCK\_TIME :Number of days the user account remains locked after failed login  
\*PASSWORD\_GRACE\_TIME :Number of grace days for user to change password  
\*PASSWORD\_VERIFY\_FUNCTION :PL/SQL that can be used for password verification

### Default users in oracle 12c

-- List default users ( valid from 12c onwards)  
**select username from dba\_users where ORACLE\_MAINTAINED='Y';**

### Create tablespace in oracle db

-- Create New tablespace

**Create tablespace DATA datafile '/u01/dbaclass/oradata/data01.dbf' size 5G autoextend on next 500M;**

-- Create tablespace on ASM diskgroup

**Create tablespace DATA datafile '+DATAG' size 5G autoextend on next 500M;**

-- Create big tablespace:

**CREATE BIGFILE TABLESPACE BIGTS datafile '/u01/dbaclass/oradata/bigts01.dbf' size 100G autoextend on NEXT 1G;**

### Rename tablespace in oracle db

SQL? **select file\_id,file\_name,tablespace\_name from dba\_data\_files where file\_id=37;** FILE\_ID FILE\_NAME TABLESPACE\_NAME ---------- -------------------------------------------------------- ------------------------------ 37 /home/oracle/app/oracle/oradata/cdb1/testin1.dbf TESTING --- Rename the tablespace\_name from TESTING to PRODUCING; SQL? **alter tablespace TESTING rename to PRODUCING;**  Tablespace altered. SQL? **select file\_id,file\_name,tablespace\_name from dba\_data\_files where file\_id=37;** FILE\_ID FILE\_NAME TABLESPACE\_NAME ---------- -------------------------------------------------------- ------------------------------ 37 /home/oracle/app/oracle/oradata/cdb1/testin1.dbf PRODUCING

*-- Drop a tablespace without removing the physical database files.*  SQL? **drop tablespace TESTING;** Tablespace dropped. SQL? **select file\_name from dba\_data\_files where tablespace\_name='TESTING';**  no rows selected *-- Drop tablespace including the physical datafiles.* SQL? **drop tablespace TESTING including contents and datafiles;** Tablespace dropped.

### Add/Drop/Alter datafile

-- Add a datafile to a tablespace

**Alter tablespace USERS add datafile '/u01/data/users02.dbf' size 5G;**

-- Enable autoextend on for a datafile;

**Alter database datafile '/u01/data/users02.dbf' autoextend on;**

-- Resize a datafile

**alter database datafile '/u01/data/users02.dbf' resize 10G;**

-- Make a datafile offline/online

**Alter database datafile '/u01/data/users02.dbf' offline;**

**Alter database datafile '/u01/data/users02.dbf' online;**

-- Drop a datafile:

**Alter tablespace USERS drop datafile '/u01/data/users02.dbf';**

### Add/drop Tempfile

-- Add tempfile to temp tablespace:

**alter tablespace TEMP1 add tempfile '/u01/dbaclass/tempfile/temp02.dbf' size 1G autoextend on next 200M;**

-- Resize temp file:

**alter database tempfile '/u01/dbaclass/tempfile/temp02.dbf' resize 2G;**

-- Drop tempfile:

**ALTER DATABASE TEMPFILE '/u01/dbaclass/tempfile/temp02.dbf' DROP INCLUDING DATAFILES;**

### Rename/move a datafile

For oracle 12c, move or rename of datafile can be done online with one line:

SQL> **alter database move datafile '/home/oracle/producing1.dbf' to '/home/oracle/app/oracle/oradata/cdb1/testin1.dbf';**

For 11g, u have to follow below steps:( It needs downtime for the datafile)

--Make the tablespace offline:

**alter database datafile '/home/oracle/app/oracle/oradata/cdb1/testin1.dbf' offline;**

-- Move the file physically to a new location.

**mv /home/oracle/app/oracle/oradata/cdb1/testin1.dbf /home/oracle/producing1.dbf**

-- Rename at db level

**alter database rename file '/home/oracle/app/oracle/oradata/cdb1/testin1.dbf' to '/home/oracle/producing1.dbf';**

-- Recover the datafile:

**recover datafile 37;**

-- Make the datafile online:

**alter database datafile '/home/oracle/producing1.dbf' online;**

### Checkpoint time of datafiles

-- REFERENCE - ORAFAQ

**set feed off**  
**set pagesize 10000**  
**set linesize 500**  
**break on grantee skip 1**  
**column datum new\_value datum noprint**  
**column file\_nr format 999999 heading 'File#'**  
**column checkpoint\_time format A20 heading 'Checkpoint|Time'**  
**column file\_name format A59 heading 'Filename'**

**select FILE# file\_nr,**  
**to\_char(CHECKPOINT\_TIME,'DD.MM.YYYY:HH24:MI:SS') checkpoint\_time,**  
**name file\_name**  
**from v$datafile\_header;**

### Occupants usage in sysaux tablespace

**select occupant\_name,occupant\_desc,space\_usage\_kbytes**  
**from v$sysaux\_occupants;**

### Status of PDBS in multitenant

SQL> **select dbid,name,open\_mode,TOTAL\_SIZE/1024/1024 from v$pdbs;**

DBID NAME OPEN\_MODE TOTAL\_SIZE/1024/1024  
---------- ------------------------------ ---------- --------------------  
3987628790 PDB$SEED READ ONLY 830  
1360187792 PDB1 READ WRITE 905  
3819422575 PDB2 MOUNTED 0

SQL> **show pdbs**

CON\_ID CON\_NAME OPEN MODE RESTRICTED  
---------- ------------------------------ ---------- ----------  
2 PDB$SEED READ ONLY NO  
3 PDB1 READ WRITE NO  
4 PDB2 MOUNTED

### Tablespace info in Multitenant

**SET LINES 132 PAGES 100**  
**COL con\_name FORM A15 HEAD "Container|Name"**  
**COL tablespace\_name FORM A15**  
**COL fsm FORM 999,999,999,999 HEAD "Free|Space Meg."**  
**COL apm FORM 999,999,999,999 HEAD "Alloc|Space Meg."**  
**--**  
**COMPUTE SUM OF fsm apm ON REPORT**  
**BREAK ON REPORT ON con\_id ON con\_name ON tablespace\_name**  
**--**  
**WITH x AS (SELECT c1.con\_id, cf1.tablespace\_name, SUM(cf1.bytes)/1024/1024 fsm**  
**FROM cdb\_free\_space cf1**  
**,v$containers c1**  
**WHERE cf1.con\_id = c1.con\_id**  
**GROUP BY c1.con\_id, cf1.tablespace\_name),**  
**y AS (SELECT c2.con\_id, cd.tablespace\_name, SUM(cd.bytes)/1024/1024 apm**  
**FROM cdb\_data\_files cd**  
**,v$containers c2**  
**WHERE cd.con\_id = c2.con\_id**  
**GROUP BY c2.con\_id**  
**,cd.tablespace\_name)**  
**SELECT x.con\_id, v.name con\_name, x.tablespace\_name, x.fsm, y.apm**  
**FROM x, y, v$containers v**  
**WHERE x.con\_id = y.con\_id**  
**AND x.tablespace\_name = y.tablespace\_name**  
**AND v.con\_id = y.con\_id**  
**UNION**  
**SELECT vc2.con\_id, vc2.name, tf.tablespace\_name, null, SUM(tf.bytes)/1024/1024**  
**FROM v$containers vc2, cdb\_temp\_files tf**  
**WHERE vc2.con\_id = tf.con\_id**  
**GROUP BY vc2.con\_id, vc2.name, tf.tablespace\_name**  
**ORDER BY 1, 2;**

### Temp tablespace details in Multitenant

**select a.name,b.FILE\_ID,b.tablespace\_name,b.file\_name from V$CONTAINERS a , CDB\_TEMP\_FILES b where a.con\_id=b.con\_id;**

### show History of PDBS

**set lines 299**  
**set pagesize 299**  
**col db\_name for a10**  
**col CLONED\_FROM\_PDB\_NAME for a12**  
**col pdb\_name for a18**  
**SELECT DB\_NAME, CON\_ID, PDB\_NAME, OPERATION, OP\_TIMESTAMP, CLONED\_FROM\_PDB\_NAME FROM CDB\_PDB\_HISTORY;**

### currently connected PDB name

SQL> **show con\_name**

CON\_NAME  
------------------------------  
PDB1

SQL> **select sys\_context('USERENV','CON\_NAME') FROM DUAL;**

SYS\_CONTEXT('USERENV','CON\_NAME')  
-----------------------------------  
PDB1

### stop and start pluggable db

-- Open/close all the pluggable db:  
-- Connect to root container:

**alter pluggable database all open;**

**alter pluggable database all close immediate;**

-- Stop/start a pluggable db:

SQL> **alter session set container=PDB1;**

Session altered.

SQL> **startup**  
Pluggable Database opened.  
SQL> **shutdown**  
Pluggable Database closed.

### Drop a pluggable database

-- Need to run from root container;

SQL> **show con\_name**

CON\_NAME  
------------------------  
CDB$ROOT

**ALTER PLUGGABLE DATABASE PDB1 CLOSE IMMEDIATE;**

**DROP PLUGGABLE DATABASE PDB1 INCLUDING DATAFILE;**

### Check undo mode in Multitenant db (oracle 12.2)

-- Local undo mode means that each container has its own undo tablespace for every instance in which it is open.  
-- Shared undo mode means that there is one active undo tablespace for a single-instance CDB

**select \* from database\_properties where property\_name='LOCAL\_UNDO\_ENABLED';**

### Is the Database is a Multitenant or not

-- If the output is YES mean it is a multitenant database, else normal db

SQL> **SELECT CDB FROM V$DATABASE;**  
CDB  
---  
YES

### Services associated with PDBs

**COLUMN NETWORK\_NAME FOR A34**  
**COLUMN PDB FOR A15**  
**COLUMN CON\_ID FOR 999**  
**SELECT PDB, NETWORK\_NAME, CON\_ID FROM CDB\_SERVICES WHERE PDB IS NOT NULL AND CON\_ID > 2 ORDER BY PDB;**

### View container DB information

**COLUMN NAME FORM A8**  
**SELECT NAME, CON\_ID, DBID, CON\_UID, GUID FROM V$CONTAINERS;**

### Manage dbms\_schedulerjobs

Enable a job

**EXECUTE DBMS\_SCHEDULER.ENABLE('SCOTT.MONTHLYBILLING');**

Disable a job

**EXECUTE DBMS\_SCHEDULER.DISABLE('SCOTT.MONTHLYBILLING');**

Stop a running job

**EXECUTE DBMS\_SCHEDULER.STOP\_JOB('SCOTT.MONTHLYBILLING');**

Drop a running job

**EXECUTE DBMS\_SCHEDULER.DROP\_JOB('SCOTT.MONTHLYBILLING');**

Run a job immediately

**EXECUTE DBMS\_SCHEDULER.RUN\_JOB('SCOTT.MONTHLYBILLING');**

### Create and scheduler a scheduler job

-- TO schedule a job, first create a schedule, then a program and then a job  
--Create a schedule

**BEGIN**  
**DBMS\_SCHEDULER.CREATE\_SCHEDULE (**  
**Schedule\_name => 'DAILYBILLINGJOB',**  
**Start\_date => SYSTIMESTAMP,**  
**Repeat\_interval =>'FREQ=DAILY;BYHOUR=11; BYMINUTE=30',**  
**Comments => 'DAILY BILLING JOB'**  
**);**  
**END;**

-- Create a program

**BEGIN**  
**DBMS\_SCHEDULER.CREATE\_PROGRAM (**  
**program\_name => 'DAILYBILLINGJOB',**  
**program\_type => 'STORED\_PROCEDURE',**  
**program\_action => 'DAILYJOB.BILLINGPROC'**  
**number\_of\_arguments =>0,**  
**enabled => TRUE,**  
**comments => 'DAILY BILLING JOB'**  
**);**  
**END;**

-- Now create the job:

**BEGIN DBMS\_SCHEDULER.CREATE\_JOB (**  
**job\_name => 'DAILYBILLINGJOB\_RUN',**  
**program\_name => 'DAILYBILLINGJOB',**  
**schedule\_name => 'DAILYBILLINGJOB\_SCHED',**  
**enabled => FLASE,**  
**comments => 'daily billing job'**  
**);**  
**END;**

-- ENABLE THE JOB

**EXECUTE DBMS\_SCHEDULER.ENABLE('DAILYBILLINGJOB\_RUN');**

### Drop a schedule

**BEGIN**  
**DBMS\_SCHEDULER.DROP\_SCHEDULE(**  
**schedule\_name => 'DAILYBILLINGJOB\_SCHED',**  
**force => TRUE**  
**);**  
**END;**

### scheduler shell script in dbms\_scheduler

-- This feature in available from oracle 12c onward

-- Create an credential store:

**BEGIN**  
**dbms\_credential.create\_credential (**  
**CREDENTIAL\_NAME => 'ORACLEOSUSER',**  
**USERNAME => 'oracle',**  
**PASSWORD => 'oracle@98765',**  
**DATABASE\_ROLE => NULL,**  
**WINDOWS\_DOMAIN => NULL,**  
**COMMENTS => 'Oracle OS User',**  
**ENABLED => true**  
**);**  
**END;**  
**/**

-- Create the job:

**exec dbms\_scheduler.create\_job(-**  
**job\_name=>'myscript4',-**  
**job\_type=>'external\_script',-**  
**job\_action=>'/export/home/oracle/ttest.2.sh',-**  
**enabled=>true,-**  
**START\_DATE=>sysdate,-**  
**REPEAT\_INTERVAL =>'FREQ=MINUTELY; byminute=1',-**  
**auto\_drop=>false,-**  
**credential\_name=>'ORACLEOSUSER');**

### Monitor scheduler jobs

-- Monitor currently running jobs

**SELECT job\_name, session\_id, running\_instance, elapsed\_time, FROM dba\_scheduler\_running\_jobs;**

-- View the job run details

**select \* from DBA\_SCHEDULER\_JOB\_RUN\_DETAILS;**

-- View the job related logs:

**select \* from DBA\_SCHEDULER\_JOB\_LOG;**

### All scheduler windows

--ALL SCHEDULER WINDOWS  
--Reference : Gwen Shapira

**set pagesize 300 linesize 200**  
**select \* from dba\_scheduler\_windows;A**

### View all scheduler schedules

**set pagesize 200**  
**set lines 299**  
**col START\_DATE for a45**  
**col REPEAT\_INTERVAL for a45**  
**col schedule\_name for a34**  
**select schedule\_name, schedule\_type, start\_date, repeat\_interval from dba\_scheduler\_schedules;**

### history of all scheduler job runs

**set pagesize 299**  
**set lines 299**  
**col JOB\_NAME for a24**  
**col actual\_start\_date for a56**  
**col RUN\_DURATION for a34**  
**select job\_name,status,actual\_start\_date,run\_duration from DBA\_SCHEDULER\_JOB\_RUN\_DETAILS order by ACTUAL\_START\_DATE desc;**

### log information for all Scheduler jobs

**set pagesize 299**  
**set lines 299**  
**col job\_name for a24**  
**col log\_date for a40**  
**col operation for a19**  
**col additional\_info a79**  
**select job\_name,log\_date,status,OPERATION,ADDITIONAL\_INFO from dba\_scheduler\_job\_log order by log\_date desc;**

### Get DDL of a scheduler job

***syntax*** - > select dbms\_metadata.get\_ddl('PROCOBJ','JOB\_NAME','JOB\_OWNER') from dual;

**select dbms\_metadata.get\_ddl('PROCOBJ','DUP\_ACC','SCOTT') from dual;**

### Scheduler job detail in CDB

**select CON\_ID, JOB\_NAME,JOB\_TYPE,ENABLED, STATE,NEXT\_RUN\_DATE, REPEAT\_INTERVAL from cdb\_scheduler\_jobs;**

### Copy scheduler job from one user to other

Below will copy job my\_job\_2 from user scott to dbaclass.

**exec dbms\_scheduler.copy\_job('SCOTT.MY\_JOB\_2','DBACLASS.MY\_JOB\_2');**

### Definition of job in dbms\_jobs

-- First get the job\_id and owner;  
**select job,log\_user,schema\_user from dba\_jobs;**

743,DBATEST  
--- connect to the owner , and get the definition of the job

**alter session set current\_schema=DBATEST;**

**set serveroutput on**  
**SQL> DECLARE**  
**callstr VARCHAR2(500);**  
**BEGIN**  
**dbms\_job.user\_export(743, callstr);**  
**dbms\_output.put\_line(callstr);**  
**END;**  
**/**

### Enable/disable/drop a dbms\_job

-- Get the job number from dba\_jobs.

**select job "jobno",schema\_user,what from dba\_jobs;**

-- Disable a job

**EXEC DBMS\_IJOB.BROKEN(jobno,TRUE);**

-- Enable a job

**EXEC DBMS\_IJOB.BROKEN(jobno,FALSE);**

--REMOVE A DBMS\_JOBS:

**EXEC DBMS\_IJOB.remove(jobno) ;**

### Check DB role(PRIMARY/STANDBY)

**SELECT DATABASE\_ROLE, DB\_UNIQUE\_NAME INSTANCE, OPEN\_MODE, PROTECTION\_MODE, PROTECTION\_LEVEL, SWITCHOVER\_STATUS FROM V$DATABASE;**

### Monitor standby background process

**SELECT PROCESS, STATUS, THREAD#, SEQUENCE#, BLOCK#, BLOCKS FROM V$MANAGED\_STANDBY ;**

### View dataguard message or errors

**SELECT MESSAGE FROM V$DATAGUARD\_STATUS;**

### Last log applied/Received in standby

**select 'Last Log applied : ' Logs, to\_char(next\_time,'DD-MON-YY:HH24:MI:SS') Time**  
**from v$archived\_log**  
**where sequence# = (select max(sequence#) from v$archived\_log where applied='YES')**  
**union**  
**select 'Last Log received : ' Logs, to\_char(next\_time,'DD-MON-YY:HH24:MI:SS') Time**  
**from v$archived\_log**  
**where sequence# = (select max(sequence#) from v$archived\_log);**

### Get standby redo log info

**set lines 100 pages 999**  
**col member format a70**  
**select st.group#**  
**, st.sequence#**  
**, ceil(st.bytes / 1048576) mb**  
**, lf.member**  
**from v$standby\_log st**  
**, v$logfile lf**  
**where st.group# = lf.group#**  
**/**

### Monitor lag in standby including RAC

**-- Applicable for 2 NODE RAC ALSO**

**column applied\_time for a30**  
**set linesize 140**  
**select to\_char(sysdate,'mm-dd-yyyy hh24:mi:ss') "Current Time" from dual;**  
**SELECT DB\_NAME, APPLIED\_TIME, LOG\_ARCHIVED-LOG\_APPLIED LOG\_GAP ,**  
**(case when ((APPLIED\_TIME is not null and (LOG\_ARCHIVED-LOG\_APPLIED) is null) or**  
**(APPLIED\_TIME is null and (LOG\_ARCHIVED-LOG\_APPLIED) is not null) or**  
**((LOG\_ARCHIVED-LOG\_APPLIED) > 5))**  
**then 'Error! Log Gap is '**  
**else 'OK!'**  
**end) Status**  
**FROM**  
**(**  
**SELECT INSTANCE\_NAME DB\_NAME**  
**FROM GV$INSTANCE**  
**where INST\_ID = 1**  
**),**  
**(**  
**SELECT MAX(SEQUENCE#) LOG\_ARCHIVED**  
**FROM V$ARCHIVED\_LOG WHERE DEST\_ID=1 AND ARCHIVED='YES' and THREAD#=1**  
**),**  
**(**  
**SELECT MAX(SEQUENCE#) LOG\_APPLIED**  
**FROM V$ARCHIVED\_LOG WHERE DEST\_ID=2 AND APPLIED='YES' and THREAD#=1**  
**),**  
**(**  
**SELECT TO\_CHAR(MAX(COMPLETION\_TIME),'DD-MON/HH24:MI') APPLIED\_TIME**  
**FROM V$ARCHIVED\_LOG WHERE DEST\_ID=2 AND APPLIED='YES' and THREAD#=1**  
**)**  
**UNION**  
**SELECT DB\_NAME, APPLIED\_TIME, LOG\_ARCHIVED-LOG\_APPLIED LOG\_GAP,**  
**(case when ((APPLIED\_TIME is not null and (LOG\_ARCHIVED-LOG\_APPLIED) is null) or**  
**(APPLIED\_TIME is null and (LOG\_ARCHIVED-LOG\_APPLIED) is not null) or**  
**((LOG\_ARCHIVED-LOG\_APPLIED) > 5))**  
**then 'Error! Log Gap is '**  
**else 'OK!'**  
**end) Status**  
**from (**  
**SELECT INSTANCE\_NAME DB\_NAME**  
**FROM GV$INSTANCE**  
**where INST\_ID = 2**  
**),**  
**(**  
**SELECT MAX(SEQUENCE#) LOG\_ARCHIVED**  
**FROM V$ARCHIVED\_LOG WHERE DEST\_ID=1 AND ARCHIVED='YES' and THREAD#=2**  
**),**  
**(**  
**SELECT MAX(SEQUENCE#) LOG\_APPLIED**  
**FROM V$ARCHIVED\_LOG WHERE DEST\_ID=2 AND APPLIED='YES' and THREAD#=2**  
**),**  
**(**  
**SELECT TO\_CHAR(MAX(COMPLETION\_TIME),'DD-MON/HH24:MI') APPLIED\_TIME**  
**FROM V$ARCHIVED\_LOG WHERE DEST\_ID=2 AND APPLIED='YES' and THREAD#=2**  
**)**  
**/**

### Monitor recovery progress in standby db

**select to\_char(START\_TIME,'DD-MON-YYYY HH24:MI:SS') "Recovery Start Time",to\_char(item)||' = '||to\_char(sofar)||' '||to\_char(units) "Progress"**  
**from v$recovery\_progress where start\_time=(select max(start\_time) from v$recovery\_progress);**

### Stop/start MRP process in standby

--Cancel MRP(media recovery) process in standby:

**alter database recover managed standby database cancel;**

--Start MRP(media recovery):

**alter database recover managed standby database disconnect from session;**

-- For real time media recovery

**alter database recover managed standby database using current logfile disconnect from session;**

### Move LOB segment to another tablespace

-- Find the lob segment details

**select table\_name,COLUMN\_NAME,SEGMENT\_NAME,TABLESPACE\_NAME from dba\_lobs where OWNER='DBACLASS'**

-- Move to new tablespace

**alter table DBACLASS.LOB\_SEG1 move lob (PAYLOAD) store as SYS\_LOB0000100201C00011$$ ( tablespace USERS);**

### Find tables with LOB seg in DB

**set pagesize 200**  
**set lines 200**  
**set long 999**  
**col owner for a15**  
**col table\_name for a20**  
**col column\_name for a21**  
**select a.owner,a.table\_name,a.column\_name, data\_type**  
**from dba\_lobs a, dba\_tab\_columns b**  
**where a.column\_name=b.column\_name**  
**and a.table\_name = b.table\_name**  
**and a.owner = b.owner**  
**and b.owner not in ('SYS','SYSTEM','DBSNMP','WMSYS');**

### managing columns of table

-- Add column

**alter table scott.emp add( empname varchar2(20));**  
**alter table scott.emp add( empid number,deptid number);**

-- Drop column

**alter table scott.emp drop (empname);**  
**alter table scott.emp drop (empid,deptid);**

-- Rename column

**alter table scott.emp rename column empname to asocname;**

-- set column unused

**alter table scott.emp set unused (empname);**

-- Drop unused columns from a table

**alter table scott.emp drop unused columns.**

### space usage by LOB column

SPACE USED BY LOB COLUMN:

**SELECT s.bytes FROM dba\_segments s JOIN dba\_lobs l USING (owner, segment\_name)**  
**WHERE l.table\_name = '&table\_name';**

ACTUAL SPACE USED BY LOB:

**SELECT nvl((sum(dbms\_lob.getlength( &lob\_column ))),0) AS bytes FROM &table\_name;**

### Find chained rows in table

-- First, analyze the table as below:

ANALYZE TABLE SCOTT.EMPTABLE LIST CHAINED ROWS;

-- Then check the row\_count in chained\_row table

select count(\*) from chained\_rows where table\_name='EMPTABLE';

The output of this query returns the number of chained rows in that table.

### object with mix or lowercase name

set lines 132 pages 1000  
col object\_name format a30 heading "Object Name";  
col object\_type format a10 heading "Object|Type";  
col created format a30 heading "Created";  
col status format a30 heading "Status";  
select OWNER,object\_name,object\_type,created,status from dba\_objects  
where (object\_name = lower(object\_name) or  
object\_name = initcap(lower(object\_name)))  
and object\_name != upper(object\_name);

### Find nested tables in db

--- Script to find nested tables of a schema:

set pagesize 200  
set lines 200  
set long 999  
col owner for a18  
col table\_name for a20  
col table\_type\_owner for a20  
col table\_type\_name for a20  
col parent\_table\_name for a20  
col parent\_table\_column for a20  
SELECT owner, table\_name, table\_type\_owner, table\_type\_name,  
parent\_table\_name, parent\_table\_column,  
LTRIM (storage\_spec) storage\_spec, LTRIM (return\_type) return\_type  
FROM dba\_nested\_tables  
WHERE owner='&SCHEMA\_NAME'  
And upper(table\_name) like '&&TABLE\_NAME'  
ORDER BY owner;

### Create/drop database link

-- Create public database link

Create public database link LINK\_PUB connect to system identified by oracle using 'PRODB';

where PRODB - > tnsname of the target db added in tnsnames.ora

-- Create private database link under Scott

connect scott/tiger

create database link LINK\_PRIV connect to system identified by oracle using 'PRODB';

-- Drop public database link

drop public database link TEST\_LINK ;

-- Drop private database link

connect scott/tiger

drop database link LINK\_PRIV;

NOTE - Private database link can be dropped only by the owner of the database link

### Top index sizes of table/schema

--- Find index name and their sizes of a table

SELECT idx.table\_name,bytes/1024/1024/1024  
FROM dba\_segments seg,  
dba\_indexes idx  
where idx.table\_name='&TABLE\_NAME'  
AND idx.index\_name = seg.segment\_name  
GROUP BY idx.table\_name order by 1;

-- Find total\_index\_size of respective tables in a schema

SELECT idx.table\_name, SUM(bytes/1024/1024/1024)  
FROM dba\_segments seg,  
dba\_indexes idx  
WHERE idx.table\_owner = 'SIEBEL'  
AND idx.owner = seg.owner  
AND idx.index\_name = seg.segment\_name  
GROUP BY idx.table\_name order by 1

### Create/drop synonyms

-- Create public synonym

CREATE PUBLIC SYNONYM emp\_view FOR scott.emp;

-- Create private synonym

CREATE SYNONYM priv\_view FOR scott.emp;

-- Drop synonym

DROP PUBLIC SYNONYM emp\_view;  
DROP SYNONYM priv\_view;

-- View synonym related info

SELECT \* FROM DBA\_SYNONYMS;

### Find column usage statistics

set lines 150  
set pages 500  
col table\_name for a20  
col column\_name for a20  
select a.object\_name table\_name, c.column\_name,equality\_preds, equijoin\_preds, range\_preds, like\_preds  
from dba\_objects a, col\_usage$ b, dba\_tab\_columns c  
where a.object\_id=b.OBJ#  
and c.COLUMN\_ID=b.INTCOL#  
and a.object\_name=c.table\_name  
and b.obj#=a.object\_id  
and a.object\_name='&table\_name'  
and a.object\_type='TABLE'  
and a.owner='&owner'  
order by 3 desc,4 desc, 5 desc;

### Estimate space require for index creation

--Below script is to get the required space for index creation, before actually it is being created.  
--- Lets check for create index DBACLASS.INDEX1 on DBACLASS.EMP(EMPNO)

SET SERVEROUTPUT ON  
DECLARE  
v\_used\_bytes NUMBER(10);  
v\_Allocated\_Bytes NUMBER(10);  
BEGIN

DBMS\_SPACE.CREATE\_INDEX\_COST  
(  
'create index DBACLASS.INDEX1 on DBACLASS.EMP(EMPNO)',  
v\_used\_Bytes,  
v\_Allocated\_Bytes  
);  
DBMS\_OUTPUT.PUT\_LINE('Used Bytes MB: ' || round(v\_used\_Bytes/1024/1024));  
DBMS\_OUTPUT.PUT\_LINE('Allocated Bytes MB: ' || round(v\_Allocated\_Bytes/1024/1024));  
END;  
/

### Compile invalid objects

-- To compile all objects in database

@$ORACLE\_HOME/rdbms/admin/utlrp.sql

-- Compile objects of a particular schema:

EXEC DBMS\_UTILITY.compile\_schema(schema => 'APPS');

-- Compiling a package;

ALTER PACKAGE APPS.DAIL\_REPORT COMPILE;

ALTER PACKAGE APPS.DAIL\_REPORT COMPILE BODY;

-- Compiling a procedure:

ALTER PROCEDURE APPS.REPORT\_PROC COMPILE;

-- Compiling a view:

ALTER VIEW APPS.USERSTATUS\_VW COMPILE;

-- Compiling a function:

ALTER FUNCTION APPS.SYNC\_FUN COMPILE;

### Find dependents of an object

select \* from dba\_dependencies where owner='&SCHEMA\_NAME' and name='&OBJECT\_NAME';  
select \* from dba\_dependencies where referenced\_owner = 'USER\_NAME' and referenced\_name = 'OBJECT\_NAME';

### Index rebuild in oracle

-- Index rebuild online

alter index TEST\_INDX rebuild online ;

--- Fast Index rebuild

alter index TEST\_INDX rebuild online parallel 8 nologging;  
alter index TEST\_INDX noparallel;  
alter index TEST\_INDX logging;

### Make index invisible

SQL> **select VISIBILITY from dba\_indexes where index\_name='IDX\_SESS' and owner='DBACLASS';**  
VISIBILIT  
---------  
VISIBLE

SQL> **ALTER INDEX DBACLASS.IDX\_SESS INVISIBLE;**  
Index altered.

SQL> **select VISIBILITY from dba\_indexes where index\_name='IDX\_SESS' and owner='DBACLASS';**

VISIBILIT  
---------  
INVISIBLE

-- Revert again to visible.

SQL> **ALTER INDEX DBACLASS.IDX\_SESS VISIBLE;**  
Index altered.

### Enable auditing in database

-- Auditing is disabled, when audit\_trail is set to NONE,

SQL> **show parameter audit\_trail**  
NAME TYPE VALUE  
------------------------------------ ----------- --------------------------  
audit\_trail string NONE

- Either set audit\_trail to DB or DB,EXTENDED.

alter system set audit\_trail='DB' scope=spfile;  
(or)  
alter system set audit\_trail='DB, EXTENDED' scope=spfile;

-- Restart the database.

shutdown immediate;  
startup;

SQL> **show parameter audit\_trail**  
NAME TYPE VALUE  
------------------------------------ ----------- --------------------------  
audit\_trail string DB

### Statements audited in oracle

col user\_name for a12 heading "User name"  
col audit\_option format a30 heading "Audit Option"  
set pages 1000  
prompt  
prompt System auditing options across the system and by user  
select user\_name,audit\_option,success,failure from sys.dba\_stmt\_audit\_opts  
order by user\_name, proxy\_name, audit\_option  
/

### Privileges Audited in database

col user\_name for a12 heading "User name"  
col privilege for a30 heading "Privilege"  
set pages 1000  
prompt  
prompt System Privileges audited across system  
select user\_name,privilege,success,failure from dba\_priv\_audit\_opts  
order by user\_name, proxy\_name, privilege  
/

### audit records of an user

col user\_name for a12 heading "User name"  
col timest format a13  
col userid format a8 trunc  
col obn format a10 trunc  
col name format a13 trunc  
col object\_name format a10  
col object\_type format a6  
col priv\_used format a15 trunc  
set verify off  
set pages 1000  
SET PAGESIZE 200  
SET LINES 299  
select username userid, to\_char(timestamp,'dd-mon hh24:mi') timest ,  
action\_name acname, priv\_used, obj\_name obn, ses\_actions  
from sys.dba\_audit\_trail  
where timestamp>sysdate-&HOURS\*(1/24) and username='&USER\_NAME'  
order by timestamp  
/

### Enable audit for sys operations

SQL>**ALTER SYSTEM SET audit\_sys\_operations=true SCOPE=spfile;**

SQL> **SHUTDOWN IMMEDIATE**  
SQL> **STARTUP**  
SQL> **show parameter audit\_sys\_operations**

NAME TYPE VALUE  
------------------------------------ ----------- ------------------------------  
audit\_sys\_operations boolean TRUE

### Enable pure unified auditing 12c

-- False means mixed auditing;

SELECT value FROM v$option WHERE parameter = 'Unified Auditing';  
VALUE  
-----------------  
FALSE

-- relink the library as mentioned

shutdown immediate;

cd $ORACLE\_HOME/rdbms/lib  
make -f ins\_rdbms.mk unaiaud\_on ioracle

startup

SELECT value FROM v$option WHERE parameter = 'Unified Auditing';

VALUE  
-----------------  
TRUE

### Unified audit policies present in db

-- Audit policies present in db:

select distinct POLICY\_NAME from AUDIT\_UNIFIED\_POLICIES;

-- Enabled audit policies in db:

select distinct policy\_name from AUDIT\_UNIFIED\_ENABLED\_POLICIES;

-- Get the audit options included in an policy

select AUDIT\_OPTION from AUDIT\_UNIFIED\_POLICIES where POLICY\_NAME='ORA\_SECURECONFIG';

### View unified audit report

- Unified report for last 1 hour:  
set lines 299  
col SQL\_TEXT for a23  
col action\_name for a18  
col UNIFIED\_AUDIT\_POLICIES for a23  
select action\_name,SQL\_TEXT,UNIFIED\_AUDIT\_POLICIES ,EVENT\_TIMESTAMP from unified\_AUDIT\_trail  
where EVENT\_TIMESTAMP > sysdate -1/24;

### Create unified audit policy

-- Create audit policy with audit options:

create audit policy test\_case2  
ACTIONS CREATE TABLE,  
INSERT ON bsstdba.EMP\_TAB,  
TRUNCATE TABLE,  
select on bsstdba.PROD\_TAB;

select POLICY\_NAME,audit\_option,AUDIT\_CONDITION,OBJECT\_SCHEMA,OBJECT\_NAME FROM  
AUDIT\_UNIFIED\_POLICIES where POLICY\_NAME='TEST\_CASE2';

-- Enable policy:

audit policy TEST\_CASE2;

select distinct policy\_name from AUDIT\_UNIFIED\_ENABLED\_POLICIES where policy\_name='TEST\_CASE2';

### Enable auditing for datapump jobs

-- Create policy

create audit policy expdp\_aduit actions component=datapump export;

-- Enable policy

audit policy expdp\_aduit;

-- View audit report:

select DBUSERNAME,DP\_TEXT\_PARAMETERS1 from UNIFIED\_AUDIT\_TRAIL where DP\_TEXT\_PARAMETERS1 is not null;

### Move aud$ table to new tablespace

-- Moving aud$ table to new tablespace AUDIT\_DATA

BEGIN  
DBMS\_AUDIT\_MGMT.SET\_AUDIT\_TRAIL\_LOCATION(audit\_trail\_type => DBMS\_AUDIT\_MGMT.AUDIT\_TRAIL\_AUD\_STD,  
audit\_trail\_location\_value => 'AUDIT\_DATA');  
END;  
/  
-- Query to view new tablespace

select owner,segment\_name,segment\_type,tablespace\_name,bytes/1024/1024 from  
dba\_segments where segment\_name='AUD$';

### Check encryption wallet status

-- Encryption wallet path and status:

SELECT \* FROM gv$encryption\_wallet;

### encrypt or decrypt a column

-- Before encrypting decrypting make sure TDE is enabled in database.-- Encrypt a column

alter table SCOTT.EMP modify ( emp\_name encrypt);

alter table SCOTT.EMP modify ( emp\_name encrypt using 'AES256');

-- Decrypt a column:

alter table SCOTT.EMP modify ( emp\_name decrypt);

NOTE - This activity will take time, according to the table size and it might block other session.Better to take downtime before doing this activity.

### Find redaction policy details

-- List down redaction policy details  
SQL> select \* from redaction\_policies;

-- List down redacted column details

SQL>select \* from REDACTION\_COLUMNS;

### Drop a redaction policy

-- Find the policy details

SELECT OBJECT\_OWNER,OBJECT\_NAME,POLICY\_NAME FROM REDACTION\_POLICIES ;

-- Drop the policy

BEGIN  
DBMS\_REDACT.drop\_policy (  
object\_schema => 'SCOTT',  
object\_name => 'CREDIT\_CARD\_TAB',  
policy\_name => 'CREDIT\_HIDE\_POLICY'  
);  
END;  
/

### Enable tracing for a listener

- Set to the listener you want to trace

LSNRCTL> **set cur LISTENER\_TEST**

-- Enable Trace:

LSNRCTL> **set trc\_level ADMIN**

Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=IPC)(KEY=LISTENER\_TEST)))  
LISTENER\_TEST parameter "trc\_level" set to admin  
The command completed successfully

### Create/drop database link

**-- Create public database link**

Create public database link LINK\_PUB connect to system identified by oracle using 'PRODB';

PRODBtnsnamedbtnsnames*where - > of the target added in .ora*

**-- Create private database link under Scott**

connect scott/tiger

create database link LINK\_PRIV connect to system identified by oracle using 'PRODB';

**-- Drop public database link**

drop public database link TEST\_LINK ;

**-- Drop private database link**

connect scott/tiger

drop database link LINK\_PRIV;

*NOTE - Private database link can be dropped only by the owner of the database link*

### create db link w/o modifying tnsnames.ora

create public database link IMFP connect to iwf identified by thr3iwf USING  
'(DESCRIPTION=(ADDRESS\_LIST=(  
ADDRESS=(PROTOCOL=TCP)(HOST=testoracle.com)(PORT=1522)))  
(CONNECT\_DATA=(SERVICE\_NAME=IMFP)))';

### Modify scan listener port

-- Modify the scan listener to use new port 1523:

srvctl modify scan\_listener -p 1523

-- Restart scan\_listenr

$GRID\_HOME/bin/srvctl stop scan\_listener  
$GRID\_HOME/bin/srvctl start scan\_listener

-- update remote\_listener in database

Alter system set remote\_listener='orcl-scan.stc.com.sa:1523' scope=both sid='\*';

### Create static listener for oracle db

LISTENER\_DBACLASS =  
(DESCRIPTION =  
(ADDRESS = (PROTOCOL = TCP)(HOST = 192.20.211.236)(PORT = 1527))  
)  
SID\_LIST\_LISTENER\_DBACLASS =  
(SID\_LIST =  
(SID\_DESC =  
(ORACLE\_HOME = /oracle/app/oracle/product/12.1.0/dbhome\_1)  
(SID\_NAME = DBACLASS)  
)  
)

lsnrctl start LISTENER\_DBACLASS

### Manage listener in oracle

-- stop/start listener

lsnrctl stop LISTENER\_DBACLASS  
lsnrctl start LISTENER\_DBACLASS

-- Reload listener

lsnrctl reload LISTENER\_DBACLASS

-- Check status of listener  
lsnrctl status LISTENER\_DBACLASS

--- view listener version  
lsnrctl version LISTENER\_DBACLASS

-- View listener services

lsnrctl services LISTENER\_DBACLASS

-- View listener service acl summary:

lsnrctl servacls LISTENER\_DBACLASS

### Manage ACLS in oracle

-- Create ACLS  
exec DBMS\_NETWORK\_ACL\_ADMIN.CREATE\_ACL('scott\_utl\_mail.xml','Allow mail to be send','SCOTT', TRUE, 'connect');

-- Assign ACL to nework

exec DBMS\_NETWORK\_ACL\_ADMIN.ASSIGN\_ACL('scott\_utl\_mail.xml','\*',25);

-- grant privilege to user:

exec DBMS\_NETWORK\_ACL\_ADMIN.ADD\_PRIVILEGE('scott\_utl\_mail.xml','SCOTT', TRUE, 'connect');  
exec DBMS\_NETWORK\_ACL\_ADMIN.ADD\_PRIVILEGE('scott\_utl\_mail.xml' ,'SCOTT', TRUE, 'resolve');

--Unassign network from ACL:

exec DBMS\_NETWORK\_ACL\_ADMIN.UNASSIGN\_ACL('scott\_utl\_mail.xml','\*',25);

-- remove privilege from an user:

exec DBMS\_NETWORK\_ACL\_ADMIN.DELETE\_PRIVILEGE('scott\_utl\_mail.xml','SCOTT', TRUE, 'connect');

-- Drop ACL:

exec DBMS\_NETWORK\_ACL\_ADMIN.DROP\_ACL ('scott\_utl\_mail.xml' );

### Find active services in db

--- It will show all the registered services for the database.

col NETWORK\_NAME for a25  
set pagesize 299  
set lines 299  
select NAME,INST\_ID,NETWORK\_NAME,CREATION\_DATE,GOAL,GLOBAL from GV$ACTIVE\_SERVICES where name not like 'SYS$%';

### Set local\_listener in db

-- Make the sure the a listener is already running with that port(i.e 1524 here)

alter system set LOCAL\_LISTENER='(ADDRESS = (PROTOCOL = TCP)(HOST = 162.20.217.15)(PORT = 1524))' scope=both;  
alter system register;

select type, value from v$listener\_network where TYPE='LOCAL LISTENER';

### View ACL information in db

set lines 200  
COL ACL\_OWNER FOR A12  
COL ACL FOR A67  
COL HOST FOR A34  
col PRINCIPAL for a20  
col PRIVILEGE for a13  
select ACL\_OWNER,ACL,HOST,LOWER\_PORT,UPPER\_PORT FROM dba\_network\_acls;  
select ACL\_OWNER,ACL,PRINCIPAL,PRIVILEGE from dba\_network\_acl\_privileges;

### Stop/start oms in cloud control

----stop/start oms in oem 12c/13c.  
cd $ORACLE\_HOME/bin

emctl stop oms  
emctl start oms

-- status of oms

emctl status oms

### stop/start agent in oem cloud control

--- stop/start agent in oem 12c//13c

cd $AGENT\_HOME/bin

./emctl start agent  
./emctl stop agent

---- status of agent

./emctl status agent

### Get oms repository details

--- Oem repository is a target db ,which contains all target details

cd $OMS\_HOME/bin  
./emctl config oms -list\_repos\_details

### Get oms/agent url details

-- OMS URL Details

cd $OMS\_HOME/bin  
./emctl status oms -details

-- agent url details

cd $AGENT\_HOME/bin

./emctl status agent -details

### target list monitored by OEM

-- Run from oms server($OMS\_HOME/bin)

-- List all the target

./emcli get\_targets

-- List the target types present:

./emcli get\_target\_types

-- List targets of particular target\_type(say oracle\_database)

./emcli get\_targets -targets="oracle\_database"

### Plugins installed on OMS server

-- Run from OMS server

-- List of plugins installed on OMS server.

./emcli list\_plugins\_on\_server

-- List of plugins installed on the target agents.

./emcli list\_plugins\_on\_agent

-- List plugins deployed on particular agent

./emcli list\_plugins\_on\_agent -agent\_names="172.15.36.93"

### change sysman pwd in oem cloud

-- Syntax to update sysman password in oms repository

./emctl config oms -change\_repos\_pwd -use\_sys\_pwd -sys\_pwd -new\_pwd < new sysman password>

-- Example (need only existing sys password)

./emctl config oms -change\_repos\_pwd -use\_sys\_pwd -sys\_pwd oracle1234 -new\_pwd oracle1234

-- Restart oms

./emctl stop oms  
./emctl start oms

### Enable/disable em express 12c

-- Check whether em is enabled or not.(if output 0 means, emexpress disabled)

select dbms\_xdb.getHttpPort() from dual;  
select dbms\_xdb\_config.getHttpsPort() from dual;

-- Enable emexpress with https:

SQL> exec dbms\_xdb\_config.sethttpsport(5500);

-- Enable emexpress with http:

SQL> exec dbms\_xdb\_config.sethttpport(8080);

-- Disable em express (set port to 0)

SQL> exec dbms\_xdb\_config.sethttpsport(0);

SQL> exec dbms\_xdb\_config.sethttpport(0);

### expdp with compression parameter

-- Create the directory if not present  
create directory EXPDIR as '/export/home/oracle/ORADUMP'

-- Below is the parfile for full db export

cat parfile=compressed.par

dumpfile=schema.dmp  
logfile=tables.log  
directory=EXPDIR  
FULL=Y  
compression=ALL

-- Run expdp command  
expdp parfile=compressed.par

### expdp/impdp with parallel option

-- Create the directory if not present  
create directory EXPDIR as '/export/home/oracle/ORADUMP'

-- Par file for export with parallel degree 4

cat parfile=parallel.par

dumpfile=parallel\_%U.dmp  
logfile=tables.log  
directory=EXPDIR  
schemas=PROD\_DATA  
parallel=4

NOTE - mention parallel value as per cpu core.

-- Run expdp command

expdp parfile=parallel.par

Same is the command for IMPDP.

### expdp/impdp for schemas

-- Create the directory if not present

create directory EXPDIR as '/export/home/oracle/ORADUMP'

-- Par file for export of SCHEMAS(PROD\_DATA,DEV\_DATA)

cat parfile=schema.par

dumpfile=schema.dmp  
logfile=tables.log  
directory=EXPDIR  
schemas=PROD\_DATA,  
DEV\_DATA

-- Run expdp

expdp parfile=schema.par

For impdp also use the similar command.

### expdp/impdp for TABLES

-- Create the directory if not present

create directory EXPDIR as '/export/home/oracle/ORADUMP'

--- Par file for export of multiple tables

cat parfile=tables.par

dumpfile=tables.dmp  
logfile=tables.log  
directory=EXPDIR  
tables=PROD\_DATA.EMPLOYEE,  
PROD\_DATA.DEPT,  
DEV\_DATA.STAGING

-- Run expdp command

expdp parfile=tables.par

### expdp with query clause

--- For exporting table data with query condition

----select \* from DBACLASS.EMP\_TAB WHERE created > sysdate -40;

-- Parfile

cat expdp\_query.par

dumpfile=test.dmp  
logfile=test1.log  
directory=TEST  
tables=dbaclass.EMP\_TAB  
QUERY=dbaclass.EMP\_TAB:"WHERE created > sysdate -40"

### sqlfile option with impdp

It can be used, only with impdp. This helps in generating the DDLs from a dumpfile.  
Suppose We have a dump file of table DBACLASS.DEP\_TAB . If you need the DDL of the table, then use sqlfile with impdp command as below.

PARFILE SAMPLE:

dumpfile=test.dmp  
logfile=test1.log  
directory=TEST  
tables=DBACLASS.DEP\_TAB  
sqlfile=emp\_tab.sql

note- DDL output will be logged in the emp\_tab.sql file

### TABLE\_EXISTS\_ACTION option with impdp

option in IMPDP:**TABLE\_EXISTS\_ACTION**

TABLE\_EXISTS\_ACTION  
Action to take if imported object already exists.  
Valid keywords are: APPEND, REPLACE, [SKIP] and TRUNCATE.

TABLE\_EXISTS\_ACTION=SKIP:

This is the default option with impdp. I.e if the the table exists, it will skip that table.

TABLE\_EXISTS\_ACTION=APPEND:

while importing the table, if the table exists in the database, then it will append the data on top the existing data in the table.

impdp dumpfile=emp\_tab.dmp logfile=emp\_tab.log directory=VEN table\_exists\_action=APPEND

TABLE\_EXISTS\_ACTION=TRUNCATE:

While importing the table, if the table exists in database, it will truncate the table and load the data.

impdp dumpfile=emp\_tab.dmp logfile=emp\_tab.log directory=VEN table\_exists\_action=TRUNCATE

TABLE\_EXISTS\_ACTION=REPLACE:

While importing, if the table exists in database, then it will drop it and recreate it from the dump

impdp dumpfile=emp\_tab.dmp logfile=emp\_tab.log directory=VEN table\_exists\_action=REPLACE

### EXCLUDE/INCLUDE option in expdp

EXCLUDE/INCLUDE option:

These two options can be used in both expdp or impdp to exclude or include, particular objects or object\_types:

Export a schemas DBACLASS, excluding TABLE EMP\_TAB and DEPT

dumpfile=test.dmp  
logfile=test1.log  
directory=TEST  
exclude=TABLE:"IN ('EMP\_TAB','DEPT')"  
schemas=DBACLASS

Exclude few schemas while import:

dumpfile=test.dmp  
logfile=test1.log  
directory=TEST  
EXCLUDE=SCHEMA:"IN ('WMSYS', 'OUTLN')"

export/Import only TABLE and INDEX ( OBJECT\_TYPE)

dumpfile=FULL.dmp  
logfile=full.log  
directory=exp\_dir  
directory=DBATEST  
INCLUDE=TABLE,INDEX

### expdp to multiple directories

Suppose you wish to take a expdp backup of a big table,  
but you don’t sufficient space in a single mount point to keep the dump.  
In this case, we take expdp dump to multiple directory.

Create directories to pointing to diff PATH

SQL> **create directory DIR1 as '/home/oracle/DIR1';**

Directory created.

SQL> **create directory DIR2 as '/home/oracle/DIR2';**

Directory created.

parfile content

dumpfile=DIR1:test\_%U.dmp,  
DIR2:test\_%U.dmp  
logfile=test.log  
directory=DIR1  
parallel=2  
tables=raj.test

### expdp to asm diskgroup

Create a directory pointing to asm diskgroup( for dumpfiles)

SQL> **create directory SOURCE\_DUMP as '+NEWTST/TESTDB2/TEMPFILE';**  
Directory created

Create a directory pointing to a normal filesystem ( required for logfiles)

SQL> **create directory EXPLOG as '/export/home/oracle';**  
Directory created.

export parfile

dumpfile=test.dmp  
logfile=EXPLOG:test.log  
directory=SOURCE\_DUMP  
tables=dbatest.EMPTAB  
exclude=statistics

### CLUSTER PARAMETER IN RAC

In a RAC database, if you are taking export with parallel option and the  
datapump directory is not shared between the nodes, then set CLUSTER=N in expdp/impdp

parfile content:

dumpfile=asset\_%U.dmp  
logfile=asset.log  
directory=VEN  
parallel=32  
cluster=N