DATABASE INFO

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)
/
```

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');
```

Resize datafile without ORA-03297

```
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);
```

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

```
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';
```

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
```

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;
```

Get installed SQL patches 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 DataPump 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$%' and state='NOT RUNNING'
ORDER BY 1,2;
-- Drop the tables
SELECT 'drop table ' || owner_name || '.' || job_name || ';'
FROM dba_datapump_jobs WHERE state='NOT RUNNING' and job_name NOT LIKE 'BIN$%'
```

Get Alert log location in db

```
set pagesize 299
set lines 299
col value for a65
select * from v$diag_info where NAME='Diag Trace';
```

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;
```

Character Set 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 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
/
```

Generate multiple AWR report

```
where trunc(BEGIN_INTERVAL_TIME)=trunc(sysdate-&no)
order by 1;
CREATE OR REPLACE DIRECTORY awr_reports_dir AS '/tmp/awrreports';
DECLARE
-- Adjust before use.
l_snap_start NUMBER := 4884; --Specify Initial Snap ID
l_snap_end NUMBER := 4892; --Specify End Snap ID
l_dir VARCHAR2(50) := 'AWR_REPORTS_DIR';
l_last_snap NUMBER := NULL;
l_dbid v$database.dbid%TYPE;
l_instance_number v$instance.instance_number%TYPE;
l_file UTL_FILE.file_type;
l_file_name VARCHAR(50);
BEGIN
SELECT dbid
INTO l_dbid
FROM v$database;
SELECT instance_number
INTO l_instance_number
FROM v$instance;
FOR cur_snap IN (SELECT snap_id
FROM dba_hist_snapshot
WHERE instance_number = l_instance_number
AND snap_id BETWEEN l_snap_start AND l_snap_end
ORDER BY snap_id)
LOOP
IF l_last_snap IS NOT NULL THEN
l_file := UTL_FILE.fopen(l_dir, 'awr_' || l_last_snap || '_' || cur_snap.snap_id || '.html', 'w',
32767);
FOR cur_rep IN (SELECT output
FROM TABLE(DBMS_WORKLOAD_REPOSITORY.awr_report_html(l_dbid, l_instance_number, l_last_snap,
cur_snap.snap_id)))
LOOP
UTL_FILE.put_line(l_file, cur_rep.output);
END LOOP;
UTL_FILE.fclose(l_file);
END IF;
l_last_snap := cur_snap.snap_id;
END LOOP;
EXCEPTION
```

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;
```

DATAGUARD MONITORING

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;
```

DB MONITORING

Explain 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));
```

Explain plan of sql_id from AWR

```
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
```

Explain 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);
```

Explain 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 .
```

Enable trace for a sql_id

```
alter system set events 'sql_trace [sql:8krc88r46raff]';
```

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 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;
```

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');
```

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

```
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 v$access where owner='&OWNER' andobject='&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.
```

Op SQL queries using literal values

```
select * from (
select plan_hash_value, count(distinct(hash_value)), sum(executions),
sum(parse_calls)
from gv$sql
group by plan_hash_value
having count(distinct(hash_value)) > 10
order by 2 desc
) where rownum<21;
```

Objects causing flushing of shared pool

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

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;
```

Mutex sleep in database

```
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;
```

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
```

File 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;
```

File 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",status
from gv$session where status='ACTIVE'
order by 1
/
```

Find waitevent in database

```
select a.sid,substr(b.username,1,10) username,substr(b.osuser,1,10) osuser,
substr(b.program||b.module,1,15) program,substr(b.machine,1,22) machine,
a.event,a.p1,b.sql_hash_value
from v$session_wait a,V$session b
where b.sid=a.sid
and a.event not in('SQL*Net message from client','SQL*Net message to client',
'smon timer','pmon timer')
and username is not null
order by 6
/
```

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 the sessions

```
ELECT 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;
```

Script to monitor tablespaces 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;
```

Scripts to monitor undo tablespaces 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;
```

Script to monitor TEMP tablespaces 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 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'
```

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;
/
```

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');
```

Purge recyclebin in database

```
SQL> SQL> select count(*) from DBA_RECYCLEBIN ;
COUNT(*)
----------
2132
SQL> purge recyclebin;
Recyclebin purged.
SQL> select count(*) from DBA_RECYCLEBIN ;
COUNT(*)
----------
0
```

EXPDP/IMPDP

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"
```

SQL file 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

TABLE_EXISTS_ACTION option in IMPDP:
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

EXCLUDE/INCLUDE option in EXPDP

```
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

FLASHBACK TECH

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

```
Flashback table DBACLASS.EMP to before drop;
-- Restore the dropped table with a new name
Flashback table DBACLASS.EMP to before drop rename to EMP_BACKUP;
SQL> select NAME,time from v$restore_point;
NAME 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;
create restore point BEFORE_UPG guarantee flashback database;
-- Check the restore_points present in database
select * from v$restore_point;
-- Drop restore point;
drop restore point BEFORE_UPG;
```

Flashback db using 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 area usage info
```
SELECT * FROM V$FLASH_RECOVERY_AREA_USAGE;
```
Enable archivelog mode in standalone DM
```
-- Set log archive dest
alter system set log_archive_dest_1='LOCATION=/uv1249/arch/PROD' scope=spfile;
-- Enable archive mode in mount stage
shutdown immediate;
startup mount;
alter database archivelog;
-- Open db
alter database open;
```
List flashback restore points
```
-- From SQL prompt:
SQL>Select * from v$restore_points:
-- From RMAN prompt:
RMAN>LIST RESTORE POINT ALL;
```
MULTITENANT (CB PBD)
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

```
elect 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

```
-- 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
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;
```

NETWORK MANAGEMENT

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';
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
```

Create DM 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 DM

```
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 DM

```
--- 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 DM

```
-- 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 DM

```
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;
```

OBJECT MANAGEMENT

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');
```

Space usage 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
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

```
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
```

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.
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 required 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

```
@$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;
```

Enable/disable triggers of a schema

```
- For disabling triggers of a schema
select 'ALTER TRIGGER '||OWNER||'.'||TRIGGER_NAME||' DISABLE '||';' from dba_triggers where
owner='&SCHEMA_NAME';
- For disableing triggers for a table
select 'ALTER TRIGGER '||OWNER||'.'||TRIGGER_NAME||' DISABLE '||';' from dba_triggers where table_name =
('&TABLE_NAME') and owner='&SCHEMA_NAME';
-- Similarly for enabling
select 'ALTER TRIGGER '||OWNER||'.'||TRIGGER_NAME||' ENABLE '||';' from dba_triggers where
owner='&SCHEMA_NAME';
select 'ALTER TRIGGER '||OWNER||'.'||TRIGGER_NAME||' ENABLE '||';' from dba_triggers where table_name =
('&TABLE_NAME') and owner='&SCHEMA_NAME';
```

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;
```

OEM/CLOUD CONTROL

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 OME 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
-- agnet 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);
```

PARTITIONING

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;
```

Merge partition

```
-- MERGE PARTITION - FOR COMBINING MULTIPLE PARTITIONS TO A NEW ONE ( 12C ONWARS)
-- SYNTAX : ALTER TABLE <SCHEMA_NAME>.<TABLE_NAME> MERGE PARTITIONS <
PARTITION1,PARTITION2,...> < UPDATE GLOBAL INDEXES(optional)>;
--- NOTE: UPDATE GLOBAL INDEXES is required if GLOBAL INDEX is present
ALTER TABLE CMADMIN.DBACLASS MERGE PARTITIONS DBACLASS_JAN, DBACLASS_FEB,
DBACLASS_MAR INTO partition DBACLASS_Q1;
```

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-partition to partition

```
-- 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';
```

RMAN SCRIPTS

RMAN full db backup run block 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

```
ckup as compressed backupset incremental level 1 check logical database plus archivelog;
release channel c1 ;
release channel c2 ;
releconfigure 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;
baase 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 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 File system

```
--- 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 b/w 2 sequence

```
--- 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;
pool 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.
```

Monitor RMAN an backup progress

```
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

```
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
-- Now put the command for checking syntax
RMAN> backup database;
The command has no syntax errors
```

SCHEDULER & JOBS

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:
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
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;
```

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

```
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

```
exec dbms_scheduler.copy_job(',');
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/dop 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) ;
```

SRVCTL Commands

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
```

Add/remove a service

```
ADDING A SERVICE:
-------------------------------------------
SYNTAX:
------------
srvctl add servicec -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
```

Stop/Start a service

```
SYNTAX
srvctl start servicec -d {DB_NAME} -s {SERVICE_NAME}
srvctl stop servicec -d {DB_NAME} -s {SERVICE_NAME}
EXAMPLE:
---------------
srvctl start service -d PREDB -s PRDB_SRV
srvctl stop service -d PREDB -s PRDB_SRV
```

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
```

Set env variables using srvctl3

```
-- 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
```

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
```

STATISTICS

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
);
/
```

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;
/
```

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/modify stats retention in DB

```
-- 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 statistics 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');
```

TABLESPACE & DATA FILE

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 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_NAMETABLESPACE_NAME
---------- --------------------------------------------------
37 cdb1/testin1.dbf PRODUCING
```

Drop tablespace in oracle DB

```
-- 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/alter data file

```
-- 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/alter Temp file

```
- 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 data file

```
----- 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 data files

```
-- 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;
```

USER MANAGEMENT

Create user in oracle

```
SYNTAX :
create user <USER_NAME> identified by <PASSWORD>
default tablespace <TABLESPACE_NAME>
temporary tablespace <TEMP_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 a 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 "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';
```

ASM

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 disk group 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 disk groups

```
-- For 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
-- For umount 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 NSMREDOA 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
```

Create ASM disk in Linux using oracleasm

```
Create ASM disk in Linux using oracleasm
-- 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
```

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;
```

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/pwdPRDPREoracle
--- For all version.
orapwd file='+DATA/orapwPRODPRE' ENTRIES=10 DBUNIQUENAME='PRODPRE'
```

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
```

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: ********
```

Monitor ASM diskgroup i/o

```
--- Run from toad,sql devl
select * from V$ASM_DISK_IOSTAT;
```

Enable tracing for ASMCMD

```
Enable tracing for 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
```

AUDITING

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 a 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;
```

CRSCTL & RAC

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

```
$GRID_HOME/bin/cemutlo -n
or$
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 host-dbaclass1
```

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 disk timeout 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
```