Advanced DBA Commands for Efficient DB Administration

Datagaurd, GoldenGate, RMAN and More

Asfaw Gedamu

2/03/2024

Table of Contents

Advanced DBA Utilities for Efficient DB Administration

I. Introduction

This guide outlines the optimal and most efficient use of various Oracle Database Administration (DBA) commands and scripts for managing critical database operations. Understanding these commands empowers DBAs to perform essential tasks like managing resources, monitoring performance, ensuring data integrity, and automating routine maintenance.

II. Privileges Required

[All sections] - DBA role or equivalent system privileges are required to execute most commands and scripts mentioned in this document. Granting specific privileges based on tasks can be implemented for granular control.

III. Script Execution

[All sections] Saved scripts can be executed through various methods, including:

- **SQL*Plus:** Launch SQL*Plus, navigate to the directory containing the script, and type `@script_name.sql`.

- **SQL Developer:** Open the script in SQL Developer and press F5 to execute.
- **Operating System Script:** Create a shell script that calls the SQL*Plus command to run the script. Schedule the script execution using cron jobs (Linux/Unix) or Task Scheduler (Windows).

ADRCI commands

ADRCI is the command line interface for diagnostic utility used for viewing diagnostics data like listener log , alert log ,incident and cor dump etc and creating incident packages. Below are the the list of useful commands.

- **`adrept`:** Launch Automatic Diagnostic Repository (ADR) report viewer.
- **`adrcat`:** View contents of the ADR.
- **`adrci`:** Command-line interface for managing and analyzing ADR data.

1. Get current base location:(Also known as ADR_BASE)

```
adrci> show base
ADR base is "/u01/app/oracle/"
```

2. Set new ORACLE_BASE(ADR_BASE)

```
adrci> set base /u01/app/grid
```

3. List current ORACLE_HOME

```
adrci> show home
```

4. Set new ORACLE_HOME

```
adrci> set homepath /u02/app/oracle
```

5. View alert log

```
adrci> show alert
adrci>  show alert -tail 100
```

6. Purge alerts and trace files

-- This will purge data older than 600 minutes.

```
adrci> purge -age 600 -type ALERT
adrci> purge -age 600 -type TRACE
adrci> purge -age 600 -type incident
adrci> purge -age 10080 -type cdump
```

7. Set control policy for auto purge of files

There are two types of policies,

LONGP_POLICY is used to purge below data . Default value is 365 days.

• ALERT
• INCIDENT
• SWEEP
• STAGE
• HM

SHORTP_POLICY is used to purge for below data Default value is 30 days.

• TRACE
• CDUMP
• UTSCDMP
• IPS

```
— Get existing control policy
adrci> show control
Change default value of control policy details.
-- Set in hours.

adrci> set control (SHORTP_POLICY = 240)
adrci> set control (LONGP_POLICY = 600)
```

8 . Create incident package:

```
adrci> show incident
adrci>IPS CREATE PACKAGE INCIDENT
(or)
adrci> ips pack incident  in /tmp
Generated package 9 in file
/tmp/ORA1578_20090602113045_COM_1.zip, mode complete

We can create empty package and add required incident or problem
or alert log files.
-- Create empty package
```

```
adrci>IPS CREATE PACKAGE

-- add the necessary incident files. ( package_number will be
displayed in the above command)

adrci>IPS ADD INCIDENT incident_number PACKAGE 2

adrci>IPS ADD FILE /u01/app/oracle/alert_db.log PACKAGE 2


-- Now generate the package file.

adrci>IPS GENERATE PACKAGE 2 IN /home/dbaclass/housekeeping
```

9. Unpack a ips file

```
adrci> ips unpack file ORA_98928.zip into /tmp/housekeeping
```

10. Pack all incident files within a particular time frame

```
--Generates the package with the incidents occurred between the
times '2019-05-01 12:00:00.00' and '2019-05-02 23:00:00.00'

ips pack time '2019-05-01 12:00:00.00' to '2019-05-02
23:00:00.00'
```

11. View package information

```
adrci> ips show package
adrci> ips show package 12 detail
```

12. Remove/delete package information:

```
-- Delete the complete package:
adrci> ips delete package 2

-- Remove incidents from the packages

adrci > ips remove incident 2 package 7


-- Remove the problem keys from packages
```

```
adrci > ips remove problem 4 package 8
```

ASMCMD commands

This article contains the list of useful asmcmd commands which will come handy in your day to day operations.

- **Manage Automatic Storage Management (ASM):** List disks, create/drop disk groups, manage ASM instances.
    - Example: `asmcmd list disk`

1. List all diskgroups:

```
ASMCMD> lsdg


-- Include dismounted diskgroups:

ASMCMD> lsdg --discovery

-- List diskgroups across all nodes of cluster:

ASMCMD> lsdg -g --discovery
```

2. List asm disks:

```
-- List all asm disks

ASMCMD> lsdsk -k

-- List disks of a diskgroup(CDATA) with free and total MB
ASMCMD> lsdsk -k -G CDATA

-- List disks of a diskgroup(CDATA) with group and disk number
ASMCMD> lsdsk -p -G CDATA

-- List disks with disk creation date
ASMCMD> lsdsk -t -G CDATA

-- List candidiate disks only

ASMCMD>  lsdsk --candidate -k
```

```
-- List member disks only
ASMCMD>  lsdsk --candidate -p
```

3. Get attributes of ASM diskgroups:

```
-- List attribute of all diskgroups:

ASMCMD> lsattr -lm

-- List attribute of specific diskgroup(DMARCH)

ASMCMD> lsattr -lm -G  DMARCH
Group_Name  Name                      Value       RO  Sys
DMARCH      access_control.enabled    FALSE       N   Y
DMARCH      access_control.umask      066         N   Y
DMARCH      au_size                   1048576     Y   Y
DMARCH      cell.smart_scan_capable   FALSE       N   N

-- List attributes with specific pattern

ASMCMD> lsattr -lm %au_size%
Group_Name  Name     Value    RO  Sys
CDATA       au_size  1048576  Y   Y
BDM         au_size  1048576  Y   Y
CRMG        au_size  1048576  Y   Y
PMARCH      au_size  1048576  Y   Y
BCMS        au_size  1048576  Y   Y
```

4. unmount diskgroup:

Unmount command works only on the local node. So if you want to unmount the diskgroup from all nodes of cluster, then run this command from all the nodes

```
-- unmount all diskgroups
ASMCMD> umount -a

--- unmount specific diskgroup(ARCH)
ASMCMD> umount ARCH
```

4. Mount diskgroup:

Mount command works only on the local node. So if you want to Mount the diskgroup from all nodes of cluster, then run this command from all the nodes.

```
-- mount all diskgroups on local node
ASMCMD> mount -a

--- mount a specific diskgroup on local node
ASMCMD> mount ARCH
```

5. Rebalance a diskgroup:

```
-- here asm_power_limit is 8 and diskgroup is ARCH

ASMCMD> rebal --power 8 ARCH
Rebal on progress.

-- Monitor progress

ASMCMD> lsop
Group_Name  Pass       State  Power  EST_WORK  EST_RATE
EST_TIME
ARCH        COMPACT    RUN    8      0         16831     0
ARCH        REBALANCE  DONE   8      0         0         0
```

6. Get password file of database

```
ASMCMD> pwget --dbuniquename DBACLASS
+CDATA/DBACLASS/PASSWORD/pwddbaclass.256.899912377
```

8 .Get password file of asm :

```
ASMCMD> pwget --asm
+MGMT/orapwASM
```

9. Get asm template info of a diskgroup:

```
ASMCMD> lstmpl -l -G ARCH
Group_Name  Group_Num  Name                          Stripe  Sys
Redund  PriReg  MirrReg
ARCH        1          ARCHIVELOG                    COARSE  Y
UNPROT  COLD    COLD
ARCH        1          ASMPARAMETERFILE              COARSE  Y
UNPROT  COLD    COLD
```

```
ARCH         1           AUDIT_SPILLFILES            COARSE  Y
UNPROT  COLD    COLD
ARCH         1           AUTOBACKUP                  COARSE  Y
UNPROT  COLD    COLD
ARCH         1           AUTOLOGIN_KEY_STORE         COARSE  Y
UNPROT  COLD    COLD
ARCH         1           BACKUPSET                   COARSE  Y
UNPROT  COLD    COLD
```

10. Check whether flex asm is enabled or not

```
ASMCMD> showclustermode
ASM cluster : Flex mode disabled
```

11.Check cluster state:

```
ASMCMD> showclusterstate
Normal
```

12. View asm version:

```
ASMCMD> showversion
ASM version          : 12.1.0.2.0
```

13. Get asm spfile location:

```
ASMCMD> spget
+MGMT/DBACLASS-cluster/ASMPARAMETERFILE/registry.253.899644763
```

14. Take backup of asm spfile:

```
-- copy backup of spfile to a specific location

ASMCMD> spbackup +MGMT/DBACLASS-
cluster/ASMPARAMETERFILE/registry.253.899644763
/home/oracle/asmspfile.ora
```

15. Find clients connected to a diskgroup:

```
ASMCMD> lsct DMARCH
DB_Name  Status     Software_Version  Compatible_version
Instance_Name  Disk_Group
```

```
DBACLASS  CONNECTED        12.1.0.2.0          12.1.0.2.0
DBACLASS1       DMARCH
```

16. Get asm diskstring

```
ASMCMD> dsget
parameter:ORCL:*
profile:ORCL:*
```

17. List asm users with password:

```
ASMCMD> lspwusr
Username sysdba sysoper sysasm
SYS    TRUE     TRUE    TRUE
ASMSNMP    TRUE    FALSE   FALSE
```

18. List open files of a diskgroup:

```
--Open files of a diskgroup ( ARCH)
ASMCMD>lsof -G ARCH
```

19 . List open files related to a database

```
--Open files of a database( DBACLASS)
ASMCMD>lsof --dbname DBACLASS
```

20. Check filter driver is enabled or not:

```
ASMCMD> afd_state
ASMCMD-9526: The AFD state is 'NOT INSTALLED' and filtering is
'DEFAULT' on host 'b20e4bay01'

filter driver disks:
```

21. List filter driver disks(if enabled)

```
ASMCMD> afd_lsdsk
```

22. Get filter driver asm diskstring

```
ASMCMD> afd_dsget
AFD discovery string:
```

Cluster Control (CRSCTL) commands

CRSCTL Utility is used to managed oracle clusterware resources and components.

- **Manage Cluster Resources:** Start/stop cluster resources, monitor cluster health.
  - Example: `crsctl start resource -r <resource_name>`
- **Manage Cluster Nodes:** Add/remove nodes from the cluster, check node status.
  - Example: `crsctl add node -n <node_name>`

1. STOP & START CRS: (run from root user)

```
$GRID_HOME/bin/crsctl stop crs

$GRID_HOME/bin/crsctl start crs
```

2. Enable/Disable auto restart of CRS.

```
$GRID_HOME/bin/crsctl disable crs

$GRID_HOME/bin/crsctl enable crs
```

3. Find the cluster name

```
$GRID_HOME/bin/cemutlo -n

or

$GRID_HOME/bin/olsnodes -c
```

4. Find grid version:

```
SYNATX - $GRID_HOME/bin/crsctl query crs softwareversion
<hostname)
$GRID_HOME/bin/crsctl query crs softwareversion host-
paragonacademy
```

5. 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
```

6. Find voting disk location

```
$GRID_HOME/bin/crsctl query css votedisk
```

7. Find OCR location:

```
$GRID_HOME/bin/ocrcheck
```

8. Find 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
```

9. Check CRS status of local node

```
crsctl check crs

CRS-4638: Oracle High Availability Services is online
CRS-4537: Cluster Ready Services is online
CRS-4529: Cluster Synchronization Services is online
CRS-4533: Event Manager is online
```

10. Check status of all crs resourcs

```
$GRID_HOME/bin/crsctl stat res -t
$GRID_HOME/bin/crsctl stat res -t -init
```

10. Check active version of cluster

```
crsctl query crs activeversion
Oracle Clusterware active version on the cluster is [12.1.0.2.0]
```

11. Stop and start high availability service (HAS)

```
crsctl stop has

crsctl start has
```

12. Check CRS status of remote nodes

```
crsctl check cluster

crsctl check cluster -all
```

13. Disk timeout from node to voting disk(disktimeout)

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

14. Network latency in the node interconnect (Misscount)

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

15. Move voting disk to another diskgroup:

```
crsctl replace votedisk +OCRVD
Successful addition of voting disk
2e4ded6cee504fc8bf078b080fb7be6f.
Successful addition of voting disk
8e87826024e24fffbf5add65c011fc66.
Successful addition of voting disk
e1ba56dedff84fa8bf5605e0302fc81e.
Successful deletion of voting disk
2b7ce864c44d4fecbf60885a188290af.
Successfully replaced voting disk group with +OCRVD.
CRS-4266: Voting file(s) successfully replaced
```

16. Add another voting disk:

```
crsctl add css votedisk
```

17. Delete voting disk:

```
crsctl delete css votedisk
```

18 . Get ocr disk backup details

```
 ocrconfig -showbackup
```

19 . Check whether standard or flex ASM

```
crsctl get cluster mode status

Cluster is running in "standard" mode
```

20 . Check CRS configuation

```
crsctl config crs
```

21 . Find cluster configuration information:

```
$ crsctl get cluster configuration

Name                 : dbaclass-cluster
Configuration        : Cluster
Class                : Standalone Cluster
Type                 : flex
The cluster is not extended.
------------------------------------------------------------------
----------------
        MEMBER CLUSTER INFORMATION
      Name       Version        GUID                          Deplo
yed Deconfigured,
```

21 . Find node roles in cluster

```
crsctl get node role status -all
Node 'dbhost1' active role is 'hub'
Node 'dbhost1' active role is 'hub'
```

22. crsctl has commands for standalone grid infrastrcuture

```
crsctl check has
crsctl config has
```

```
crsctl disable has
crsctl enable has
crsctl query has releaseversion
crsctl query has softwareversion
crsctl start has
crsctl stop has
```

DGMGRL commands

DGMGRL stands for **Data Guard Broker Command-Line Interface**. It's a command-line tool used to manage Oracle Data Guard configurations.This topic contains useful dgmgrl commands to manage the dataguard environments.

- **Manage Data Guard Configuration:** Create/drop standby databases, perform failover/switchover operations.
    - Example: dgmgrl create standby database configuration -d <standby_db_name>

1. Setup DG broker in the standby setup.(Run on both primary and standby)

```
- For standalone db  :

ALTER SYSTEM SET dg_broker_config_file1 =
'\U01\oradata\dr1node.dat' scope=both sid='*';
ALTER SYSTEM SET dg_broker_config_file2 =
'\U01\oradata\dr2node.dat' scope=both sid='*';

-- For oracle RAC/ASM file system;

ALTER SYSTEM SET dg_broker_config_file1 =
'+DATA/broker/dr1node.dat' scope=both sid='*';
ALTER SYSTEM SET dg_broker_config_file2 =
'+DATA/broker/dr2node.dat' scope=both sid='*';

ALTER SYSTEM SET DG_BROKER_START=TRUE scope=both sid='*';
```

2. Create configuration in dgbroker:

```
-- on primary
$dgmgl
```

```
DGMGRL> CONNECT sys/;
Connected.

-- create configuration with primary db_unique_name and its
service name .

DGMGRL> CREATE CONFIGURATION 'PROD_DG' AS PRIMARY DATABASE IS
'PRIMDB' CONNECT IDENTIFIER IS PRIMDB;

Configuration "PRIMDB" created with primary database "PRIMDB"

--- Add standby in the configuration:

DGMGRL> ADD DATABASE 'STYDB' AS CONNECT IDENTIFIER IS STYDB
MAINTAINED AS PHYSICAL;

Database "STYDB" added
```

3. Enable the configuration

```
DGMGRL> ENABLE CONFIGURATION;
Enabled.
```

At this stage our dg broker setup is completed.

4. View configuration of dgbroker:

```
DGMGRL> show configuration

DGMGRL> show configuration verbose
```

5. view database informations:

```
-- Here PRIMDB and STYDB are db_unique_name of primary and
standby db


DGMGRL> show database 'PRIMDB'
DGMGRL > show database 'STYDB'
DGMGRL> show database verbose 'PRIMDB'
```

6. View statusreport of databases

```
-- Here PRIMDB and STYDB are db_unique_name of primary and
standby db

show database PRIMDB statusreport
```

7. View database inconsistent properties

```
-- Here PRIMDB and STYDB are db_unique_name of primary and
standby db

show database PRIMDB InconsistentProperties
show database PRIMDB InconsistentLogXptProps

show database STYDB InconsistentProperties
show database STYDB InconsistentLogXptProps
```

8. Check whether all logfiles are archived or not(on primary)

```
show database PRIMDB sendQentries

PRIMARY_SEND_QUEUE
        STANDBY_NAME       STATUS     RESETLOGS_ID
THREAD              LOG_SEQ       TIME_GENERATED
TIME_COMPLETED    FIRST_CHANGE#     NEXT_CHANGE#       SIZE
(KBs)
                         CURRENT       1022762318
1                  294  10/30/2019 11:09:26
12298130044308                           274219
```

9. Check information of received log sequence(not applied) (Run for standby)

```
DGMGRL>show database STYDB recvqentries

STANDBY_RECEIVE_QUEUE
              STATUS     RESETLOGS_ID           THREAD
LOG_SEQ       TIME_GENERATED       TIME_COMPLETED
FIRST_CHANGE#     NEXT_CHANGE#       SIZE (KBs)
         NOT_APPLIED       1022762318               1
293  10/30/2019 10:03:06  10/30/2019 11:09:26   12298109948824
12298130044308          3487164
```

10. Check database wait events:

```
DGMGRL>show database PRIMDB topwaitevents
```

11.Validate database information:

```
dgmgrl> validate database verbose 'PRIMDB'

dgmgrl> validate database 'PRIMDB'

dgmgrl> validate database 'STYDB'
```

12. Enable tracing for troubleshooting:

```
-- For standalone:

DGMGRL> edit configuration set property tracelevel=support;
DGMGRL> edit database PRIMDB set property LogArchiveTrace=8191;
DGMGRL> edit database STYDB set property LogArchiveTrace=8191;


-- For RAC:

DGMGRL> EDIT INSTANCE * ON DATABASE 'PRIMDB' SET PROPERTY
LogArchiveTrace=8191;
```

13. Disable tracing:

```
DGMGRL> edit configuration reset property tracelevel ;
DGMGRL> edit database PRIMDB reset property logarchivetrace;
DGMGRL> edit database STYDB reset property logarchivetrace;
```

14. Switchover using dgmgrl:

```
DGMGRL> connect sys/oracle
Password:
Connected as sys.

DGMGRL> switchover to STYDB
Performing switchover NOW, please wait...
Operation requires a connection to instance "STYDB1" on database
"STYDB"
Connecting to instance "STYDB1"...
Connected as SYSDBA.
New primary database "STYDB" is opening...
Oracle Clusterware is restarting database "PRIMDB" ...
Switchover succeeded, new primary is "STYDB"
```

15. Convert physical standby to snapshot standby

```
DGMGRL> convert database 'STYDB' to snapshot standby;
```

16. Convert snapshot to physical standby db

```
DGMGRL> CONVERT DATABASE 'STYDB' to PHYSICAL STANDBY;
```

Flashback related commands

Below are the collection of useful flashback related commands.

- **`flashback_table`:** Recover data to a specific point in time.
- **`flashback_transaction`:** Revert a transaction.
- **`flashback_query`:** Analyze historical data based on past queries.

1. How to check whether flashback is enabled or not:

```
select flashback_on from v$database;
```

2. Enable flashback in 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/B2PMT3'
scope=both;
alter database flashback on;
```

3. Disable flashback in database:

```
alter database flashback off;
```

4. Create flashback restore point :

```
create restore point FLASHBACK_PREP guarantee flashback
database;
```

5. Find the list of restore points:

```
-- From SQL prompt:
SQL>Select * from v$restore_points:
```

```
-- From RMAN prompt:
RMAN>LIST RESTORE POINT ALL;
```

6. Drop restore point:

```
drop restore point FLASHBACK_PREP;
```

7. Flashback database to restore point:

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

-- Get the restore point name:

SQL> select NAME,time from v$restore_point;

NAME                           TIME
------------------------------ ---------------------------------------
FLASHBACK_PREP                 21-MAR-17 03.41.33.000000000 PM

--Shutdown database and start db in Mount stage:

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

-- Get the restore point name:

SQL> select NAME,time from v$restore_point;

NAME
TIME
--------------------------------          ----------------------
-------------------------
FLASHBACK_PREP                                 21-MAR-17
03.41.33.000000000 PM

--Shutdown database and start db in Mount stage:

shutdown immediate;
startup mount;

--flashback db to restore point:

flashback database to restore point FLASHBACK_PREP;
```

```
--Open with resetlog:

alter database open resetlogs;
```

8. 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;
```

9. Flashback database to particular SCN or timestamp:

```
shutdown immediate;
startup mount;
--FLASHBACK DATABASE TO SCN 202381; -- Use this for particular
scn
--FLASHBACK DATABASE TO TIMESTAMP (SYSDATE-1/24); - Use for
flashback to last one hour
--FLASHBACK DATABASE TO TIMESTAMP to_timestamp('2018-03-11
16:00:00', 'YYYY-MM-DD HH24:MI:SS');- to specific timestamp:

alter database open resetlogs;
```

10. Flashback a table from recyclebin:

```
-- First get whether the table name exists in recyclebin or not:

SELECT object_name, original_name, createtime FROM recyclebin
where original_name='EMP';

 -- restore the table as same name:
FLASHBACK TABLE int_admin_emp TO BEFORE DROP;

-- Restore that table to a new name:

FLASHBACK TABLE int_admin_emp TO BEFORE DROP
   RENAME TO int2_admin_emp;
```

11. Get flashback are usage info:

```
SELECT * FROM V$FLASH_RECOVERY_AREA_USAGE;
```

12. How far can we 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;
```

Jobs Scheduler Commands in oracle

To schedule a job at a particular time in the database, first we need to create a schedule, then a program and then job.

- **`dbms_scheduler.create_job`:** Create scheduled jobs to automate tasks.
- **`dbms_scheduler.drop_job`:** Delete a scheduled job.
- **`dbms_scheduler.run_job`:** Manually execute a scheduled job.

1. Create a schedule

A schedule defines the start date, end time and repeat interval details

```
BEGIN
DBMS_SCHEDULER.CREATE_SCHEDULE (
Schedule_name => 'DAILYBILLINGJOB',
Start_date => SYSTIMESTAMP,
Repeat_interval =>'FREQ=DAILY;BYHOUR=11; BYMINUTE=30',
Comments => 'DAILY BILLING JOB'
);
END;
```

2. Create a program:

A program defines the name and type of the procedure, executed .package or script which executed.

```
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;
```

3. Create job:

A job defines the schedule name and the program name.

```
DBMS_SCHEDULER.CREATE_JOB (
job_name => 'DAILYBILLINGJOB_RUN',
program_name => 'DAILYBILLINGJOB',
schedule_name => 'DAILYBILLINGJOB_SCHED',
enabled => FLASE,
comments => 'daily billing job'
);
END;
```

Instead of creating scheduler,job and program separately, we can create the scheduler job with below commad directly.

Simple command to create scheduler job:

```
BEGIN
   DBMS_SCHEDULER.CREATE_JOB (
          job_name => '"HWS"."MV_REF_DBA_DATA"',
          job_type => 'PLSQL_BLOCK',
          job_action => 'dbms_refresh.refresh(""HWS"."STC_NEXT_DBA_MV_DATA"");',
          number_of_arguments => 0,
          start_date => NULL,
          repeat_interval => 'FREQ=DAILY;BYHOUR=8;BYMINUTE=00;BYSECOND=00',
          end_date => NULL,
          enabled => FALSE,
          auto_drop => FALSE,
          comments => 'Converted_dba_jobs');

   DBMS_SCHEDULER.enable(
```

```
        name => '"HWS"."MV_REF_FTTH_DATA"');
END;
```

Note: if the job_type is procedure, then use job_type='STORED_PROCEDURE'

4. View schedule details of all schedulers:

```
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;
```

5.Enable a job

EXECUTE DBMS_SCHEDULER.ENABLE('SCOTT.MONTHLYBILLING');

6.Disable a job

EXECUTE DBMS_SCHEDULER.DISABLE('SCOTT.MONTHLYBILLING');

7.Stop a running job

EXECUTE DBMS_SCHEDULER.STOP_JOB('SCOTT.MONTHLYBILLING');

8.Drop a running job

EXECUTE DBMS_SCHEDULER.DROP_JOB('SCOTT.MONTHLYBILLING');

9. Run a job immediately

EXECUTE DBMS_SCHEDULER.RUN_JOB('SCOTT.MONTHLYBILLING');

10. Drop a schedule:

```
BEGIN
DBMS_SCHEDULER.DROP_SCHEDULE(
schedule_name => 'DAILYBILLINGJOB_SCHED',
force => TRUE
);
END;
```

11. Drop a scheduler job:

DBMS_SCHEDULER.drop_job (job_name => 'SCOTT.MONTHLYBILLING');

12. Scheduler shell script in dbms_scheduler:

— This feature in available from oracle 12c onward

Create a 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;
/
```

Then 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');
```

13. 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;
```

14. Get DDL of a scheduler job:

```
select dbms_metadata.get_ddl('PROCOBJ','DUP_ACC','SCOTT') from
dual;
```

15. Copy scheduler job from one user to another :

```
Exec
dbms_scheduler.copy_job('SCOTT.MY_JOB_2','DBACLASS.MY_JOB_2');
```

16. Get log information of 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;
```

17. 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;
```

18. Managing scheduler credentials:

— Create a credential:

```
BEGIN
dbms_credential.create_credential (
CREDENTIAL_NAME => 'ORACLEOSUSER',
USERNAME => 'oracle',
PASSWORD => 'oracle@123',
DATABASE_ROLE => NULL,
WINDOWS_DOMAIN => NULL,
COMMENTS => 'Oracle OS User',
ENABLED => true
```

```
);
END;
/


--Drop a credential
exec dbms_scheduler.drop_credential('ORACLEOSUSER');
--View credential details
 select owner,CREDENTIAL_NAME,USERNAME,ENABLED from
DBA_CREDENTIALS;
--Change username and password in a credentials :
exec
DBMS_SCHEDULER.SET_ATTRIBUTE(name=>'ORACLEOSUSER',attribute=>'pa
ssword',value=>'oracle');
```

19. View and manage auto task jobs in database:

```
set lines 180 pages 1000
col client_name for a40
col attributes for a60
select client_name, status,attributes,service_name from dba_autotask_client
/
BEGIN
  DBMS_AUTO_TASK_ADMIN.disable(
    client_name => 'auto space advisor',
    operation   => NULL,
    window_name => NULL);
END;
/

BEGIN
  DBMS_AUTO_TASK_ADMIN.disable(
    client_name => 'sql tuning advisor',
    operation   => NULL,
    window_name => NULL);
END;
/

BEGIN
  DBMS_AUTO_TASK_ADMIN.disable(
    client_name => 'auto optimizer stats collection',
    operation   => NULL,
    window_name => NULL);
```

```
END;
/
```

Server Control (srvctl) commands

SRVCTL is known as server control utility, which is used to add, remove,relocate and manage different crs services or components in RAC database.

- **Manage Instances:** Startup/shutdown instances, monitor instance status.
  - Example: `srvctl start instance -d <instance_name>`
- **Manage Services:** Start/stop database services, manage listener configuration.
  - Example: `srvctl modify service <service_name> set SERVICE_STATUS=STARTED`

1. STOP DATABASE :

SYNTAX – *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
```

2. START DATABASE
SYNTAX – *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
```

3. STOP AN INSTANCE

SYNTAX – *srvctl stop instance -d db_unique_name [-i "instance_name_list"]} [-o stop_options] [-f]*
e.g

```
srvctl stop instance -d PRODB -i PRODB1
```

4. START AN INSTANCE

SYNTAX – *srvctl start instance -d db_unique_name [-i "instance_name_list"} [-o start_options]*
e.g

```
srvctl start instance -d PRODB -i PRODB1
```

5. REMOVING DB FROM CRS:
SYNTAX – *srvctl remove database -d db_unique_name [-f] [-y] [-v]*
e.g

```
srvctl remove database -d PRODB -f -y
```

6. ADDING DB IN CRS :
SYNTAX – *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
```

7. REMOVING AN INSTANCE FROM CRS:
SYNTAX – *srvctl remove instance -d DB_UNIQUE_NAME -i INSTANCE_NAME*
e.g

```
srvctl remove instance -d PRODB - I PRODB1
```

8.ADDING AN INSTANCE TO CRS:
SYNTAX – *srvctl add instance –d db_unique_name –i inst_name -n node_name*
e.g

```
srvctl add instance -d PRODB - i PRODB1 -n rachost1
```

9. Enable/disable auto restart of the instance

```
srvctl enable instance -d DB_UNIQUE_NAME-i INSTANCE_NAME
srvctl disable instance -d DB_UNIQUE_NAME-i INSTANCE_NAME
```

10. Enable/disable auto restart of the database

```
srvctl enable database -d DB_UNIQUE_NAME
srvctl disable database -d DB_UNIQUE_NAME
```

11. ADDING A SERVICE:

SYNTAX – *srvctl add servicec -d {DB_NAME} -s {SERVICE_NAME} -r {"preferred_list"} -a {"available_list"} [-P {BASIC | NONE | PRECONNECT}]*

e.g

```
srvctl add service -d PREDB -s PRDB_SRV -r "PREDB1,PREDB2" -a
"PREDB2" -P BASIC
```

12.REMOVING A SERVICE:

SYNTAX – *srvctl remove service -d {DB_NAME} -s {SERVICE_NAME}*
e.g

srvctl remove service -d PREDB -s PRDB_SRV

13. START A SERVICE
SYNTAX– *srvctl start servicec -d {DB_NAME} -s {SERVICE_NAME}*
e.g

srvctl start service -d PREDB -s PRDB_SRV

14. STOP A SERVICE

SYNTAX– *srvctl stop servicec -d {DB_NAME} -s {SERVICE_NAME}*
e.g

srvctl stop service -d PREDB -s PRDB_SRV

15. 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

16. Check the status of service
SYNTAX – *srvctl status service -d {database_name} -s {service_name}*

srvctl status service -d PREDB -s PRDB_SVC

17. Check the configuration of service
SYNTAX – *srvctl config service -d {database_name} -s {service_name}*

srvctl config service -d PREDB -s PRDB_SVC

18. Check scan listener configuration

```
srvctl config scan_listener

SCAN Listener LISTENER_SCAN1 exists. Port: TCP:1522
Registration invited nodes:
Registration invited subnets:
SCAN Listener is enabled.
SCAN Listener is individually enabled on nodes:
SCAN Listener is individually disabled on nodes:
SCAN Listener LISTENER_SCAN2 exists. Port: TCP:1522
Registration invited nodes:
Registration invited subnets:
SCAN Listener is enabled.
SCAN Listener is individually enabled on nodes:
SCAN Listener is individually disabled on nodes:
SCAN Listener LISTENER_SCAN3 exists. Port: TCP:1522
Registration invited nodes:
Registration invited subnets:
SCAN Listener is enabled.
SCAN Listener is individually enabled on nodes:
SCAN Listener is individually disabled on nodes:
```

19. Modify scan_listener port:

```
srvctl modify scan_listener -p {new-SCAN-port}

srvctl modify scan_listener -p 1523

$GRID_HOME/bin/srvctl stop scan_listener
$GRID_HOME/bin/srvctl start scan_listener

Alter system set remote_listener='orcl-scan.stc.com.sa:1523'
scope=both sid='*';
```

20. Manage MGMTDB in oracle 12c:

```
srvctl status mgmtdb
Database is enabled
Instance -MGMTDB is running on node node12-1
```

```
-- stop and start MGMT db.

srvctl stop mgmtdb
srvctl start mgmtdb
```

21. 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
```

22. Set environment variables through srvctl.

```
-- setenv to set env variables.(ORCL is the db_unique_name)

srvctl setenv database -db ORCL -env
"ORACLE_HOME=/oracle/app/oracle/product/12.1.0.2/dbhome_1"
srvctl setenv database -db ORCL -env
"TNS_ADMIN=/oracle/app/oracle/product/12.1.0.2/dbhome_1/network/
admin"

--getenv to view the env setting:

srvctl getenv database -db ORCL

ORCL:
ORACLE_HOME=/oracle/app/oracle/product/12.1.0.2/dbhome_1
TNS_ADMIN=/oracle/app/oracle/product/12.1.0.2/dbhome_1/network/a
dmin
```

23. Check status and config of ASM instance:

```
srvctl config asm
ASM home:
Password file: +MGMT/orapwASM
ASM listener: LISTENER


srvctl status asm
ASM is running on ses11-4,ses11-5
```

24. Stop and start services running from ORACLE_HOME

```
srvctl stop home -oraclehome /oracle/product/12.1.0.2/dbhome_1 -
statefile /home/oracle/state.txt  -node dbhost-1

srvctl start home -oraclehome /oracle/product/12.1.0.2/dbhome_1
-statefile /home/oracle/state.txt  -node dbhost-1
```

25. Create a TAF policy

```
srvctl add service -db ORCLDB -service TAF_ORCL -preferred
ORCLDB1 -available ORCLDB2 -tafpolicy BASIC -failovertype SELECT
srvctl start service -db OMPREDB -service TAF_ORCL
```

Obey command in goldengate

Like we execute .sql file against SQL, we can write multiple goldengate commands inside a .oby file and execute it against GGSCI prompt using **OBEY** command.

EXAMPLE:

In the below example, we will add supplemental login(add trandata) for multiple tables using obey command.

1. Create a text file and put goldengate commands.

vi addtran.oby

dblogin USERID ggate, PASSWORD ggate123
add trandata dbaclass.TEST3
add trandata dbaclass.TEST4

2. execute obey command from GGSCI

SYNTAX – obey < text_file_name.oby>

```
ggsci> obey addtran.oby
 ./ggsci

GGSCI > obey addtran.oby

GGSCI > dblogin USERID ggate, PASSWORD ggate123

Successfully logged into database.

GGSCI > add trandata dbaclass.TEST3


2017-07-17 11:24:29  WARNING OGG-06439  No unique key is defined
for table TEST3.

Logging of supplemental redo data enabled for table
DBACLASS.TEST3.
TRANDATA for scheduling columns has been added on table
'DBACLASS.TEST3'.
TRANDATA for instantiation CSN has been added on table
'DBACLASS.TEST3'.

GGSCI  4> add trandata dbaclass.TEST4


2017-07-17 11:24:33  WARNING OGG-06439  No unique key is defined
for table TEST4.

Logging of supplemental redo data enabled for table
DBACLASS.TEST4.
TRANDATA for scheduling columns has been added on table
'DBACLASS.TEST4'.
TRANDATA for instantiation CSN has been added on table
'DBACLASS.TEST4'.
```

We can see, it executed all the commands mentioned inside the text file.

NESTED OBEY:

Can we put one obey file inside another obey file.?? Well this nested obey is controlled by the keyword ALLOWNESTED.

NOALLOWNESTED:

This is the default setting. Any attempt to run nested obey file will throw below error.

ERROR: Nested OBEY scripts not allowed. Use ALLOWNESTED to allow nested scripts.

ALLOWNESTED:

This parameter Enables the use of nested OBEY files. i.e we can use one obey file inside another obey file.

Let's create two obey files:

```
--- obey file 1

cat infotran1.oby

dblogin USERID ggate, PASSWORD ggate123
info trandata dbaclass.TEST3

--- obey file 2
cat infotran2.oby

dblogin USERID ggate, PASSWORD ggate123
info trandata dbaclass.TEST4
```

Let's try with the default one(NOALLOWNESTED)

Here I have created two obey files and put them inside another obey file

```
cat infotran.oby

obey infotran1.oby
obey infotran2.oby
```

Execute the obey file

```
GGSCI > obey infotran1.oby

ERROR: Nested OBEY scripts not allowed. Use ALLOWNESTED to allow
nested scripts.
```

As expected it is throwing error,

Now let's use ALLOWNESTED parameter:

```
cat infotran.oby

ALLOWNESTED
obey infotran1.oby
obey infotran2.oby
Execute the obey file:
GGSCI > obey infotran.oby

GGSCI > ALLOWNESTED

Nested OBEY scripts allowed.

GGSCI > obey infotran1.oby

**** Halting script [infotran.oby], starting script
[infotran1.oby]...

GGSCI > dblogin USERID ggate, PASSWORD ggate123

Successfully logged into database.

GGSCI > info trandata dbaclass.TEST3
```

Logging of supplemental redo log data is enabled for table PARAGON.TEST3.

```
Columns supplementally logged for table PARAGON.TEST3: CREATED,
DATA_OBJECT_ID, EDITION_NAME, GENERATED, LAST_DDL_TIME,
NAMESPACE, OBJECT_ID, OBJECT_NAME, OBJECT_TYPE, OWNER,
SECONDARY, STATUS, SU
BOBJECT_NAME, TEMPORARY, TIMESTAMP.

Prepared CSN for table DBACLASS.TEST3: 11733401025760
GGSCI  6>

GGSCI  6> **** Terminating script [infotran1.oby], resuming
script [infotran.oby]...

GGSCI  6> obey infotran2.oby

**** Halting script [infotran.oby], starting script
[infotran2.oby]...

GGSCI > dblogin USERID ggate, PASSWORD ggate123
```

```
Successfully logged into database.

GGSCI > info trandata dbaclass.TEST4


Logging of supplemental redo log data is enabled for table
DBACLASS.TEST4.

Columns supplementally logged for table DBACLASS.TEST4: CREATED,
DATA_OBJECT_ID, EDITION_NAME, GENERATED, LAST_DDL_TIME,
NAMESPACE, OBJECT_ID, OBJECT_NAME, OBJECT_TYPE, OWNER,
SECONDARY, STATUS, SU
BOBJECT_NAME, TEMPORARY, TIMESTAMP.

Prepared CSN for table DBACLASS.TEST4: 11733401025806
GGSCI >

GGSCI > **** Terminating script [infotran2.oby], resuming script
[infotran.oby]...
```

With ALLOWNESTED, obey file executed successfully. The maximum number of nested levels is 16.

OPATCH commands

- **Patch Management:** Apply Oracle patches to update software.
  - Example: `opatch
- **Verify Patch Installation:** Check the status of applied patches.
  - Example: `opatch lsinventory`

1. list inventory details of patch.

```
$ORACLE_HOME/OPatch/opatch lsinventory
```

2. list patchsets applied :

$ORACLE_HOME/OPatch/opatch lspatches

3. Find opatch version:

$ORACLE_HOME/OPatch/opatch version

4. Find details of a particular patch(before applying):

```
$ORACLE_HOME/OPatch/opatch query -all {PATCH_PATH}

$ORACLE_HOME/OPatch/opatch query -all
/software/PSUPATCH/30089984
```

5. Apply a patch to RDBMS HOME:

```
-- You may need to shutdown the database and listener services:

cd /SOFTWARE/PSUPATCH/30089984 -- Go to the patch path:
$ORACLE_HOME/OPatch/opatch apply
```

6. Rollback an patch from RDBMS HOME:

```
$ORACLE_HOME/OPatch/opatch rollback -id [patch_id]
$ORACLE_HOME/OPatch/opatch rollback -id 30089984
```

7. Apply one off patch in grid_home:

root # $GI_HOME/crs/install/rootcrs.sh -prepatch.

oracle$ cd /SOFTWARE/PSUPATCH/30089984 -- Go to the patch path:
oracle$ $GRID_HOME/OPatch/opatch apply

root# $GI_HOME/crs/install/rootcrs.sh -postpatch

8. Check conflict against ORACLE_HOME

```
- go to patch folder.
cd 27734982
```

```
[27734982]$ $ORACLE_HOME/OPatch/opatch prereq
CheckConflictAgainstOHWithDetail -ph ./
```

9.Check whether active executables are running:

- go to patch folder.
cd 27734982

[27734982]$ $ORACLE_HOME/OPatch/opatch prereq CheckActiveFilesAndExecutables -ph ./

10.Opatch command using different inventory location:

$ORACLE_HOME/OPatch/opatch lsinventory -invPtrLoc /etc/orainv/orainventory

Oracle auditing related commands

This articles contains useful command for both traditional and unified auditing .

1. How to enable auditing:(traditional)

```
-- 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
```

2. statement level auditing:

```
-- Shows the list of statements that can be audited

select * from STMT_AUDIT_OPTION_MAP;

-- Enable  statement level auditing:

audit table by DBACLASS.

audit table by DBACLASS whenever successful;

audit role by DBACLASS;

-- To disable auditing:

noaudit table by DBACLASS;

-- find statements audited in the database:
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
/
```

3. object level auditing:

```
audit insert,update on DBACLASS.EMP by MANAGER;
AUDIT delete on DBACLASS.EMP;
```

-- disable auditing:

```
noaudit insert,update on DBACLASS.EMP by MANAGER;
noAUDIT delete on DBACLASS.EMP by MANAGER;
```

-- Audit SELECT/DML activites done by an user(DBACLASS):

```
audit select table,insert table,update table,delete table by
DBACLASS by access;
audit execute procedure by dbaclass by access;
audit all by dbaclass by access;
```

4. Privilege level auditing:

```
-- Enable privilege auditing:

audit drop any table ;
audit create table;
audit drop user;

-- Find privileges audited in the 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
/
```

5. Find 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
/
```

6. Enable auditing for sys user:

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

7. Enable pure unified auditing:

```
--For more info on unified auditing refer - > Unified auditing
in oracle 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
```

8. View unified audit policies present in db:

```
-- 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
```

9. View unified audit records:

```
- 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
```

10. Create unified audit policy:

```
-- Create audit policy with audit options:

create audit policy test_case2
ACTIONS CREATE TABLE,
INSERT ON classdba.EMP_TAB,
TRUNCATE TABLE,
select on classdba.PROD_TAB;

select
```

```
POLICY_NAME,audit_option,AUDIT_CONDITION,OBJECT_SCHEMA,OBJECT_NA
ME 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';
```

11 . Exclude particular user from audit policy:

```
SQL> noaudit policy TEST_CASE2;

Noaudit succeeded.

SQL> audit policy TEST_CASE2 except stcdba;

Audit succeeded.

SQL> SQL> select USER_NAME,POLICY_NAME,ENABLED_OPT from
AUDIT_UNIFIED_ENABLED_POLICIES where POLICY_NAME='TEST_CASE2';

USER_NAME     POLICY_NAME  ENABLED_OPT
------------- ------------ ----------------------
CLASSDBA        TEST_CASE2   EXCEPT
```

12. Purge audit table using dbms package:

```
-- Move aud$ table to new tablespace if present under SYSTEM tablespace:

select owner,segment_name,segment_type,tablespace_name,bytes/1024/1024 from
dba_segments where segment_name='AUD$';

OWNER       SEGMENT_NAME SEGMENT_TYPE      TABLESPACE_NAME
BYTES/1024/1024
----------- ----------- ---------------- ------------------------------ --------------
SYS         AUD$        TABLE            SYSTEM                                    176


BEGIN
```

```
DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_LOCATION(audit_trail_type =>
DBMS_AUDIT_MGMT.AUDIT_TRAIL_AUD_STD,
audit_trail_location_value => 'TS_AUDIT');
END;
-- Move aud$ table to new tablespace if present under SYSTEM
tablespace:

select
owner,segment_name,segment_type,tablespace_name,bytes/1024/1024
from dba_segments where segment_name='AUD$';

OWNER        SEGMENT_NAME SEGMENT_TYPE       TABLESPACE_NAME
BYTES/1024/1024
------------ ------------ ------------------ ------------------
----------- ---------------
SYS          AUD$         TABLE              SYSTEM
176


BEGIN
DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_LOCATION(audit_trail_type =>
DBMS_AUDIT_MGMT.AUDIT_TRAIL_AUD_STD,
audit_trail_location_value => 'TS_AUDIT');
END;
/
PL/SQL procedure successfully completed.
```

```
-- Initialise cleanup:

BEGIN
  DBMS_AUDIT_MGMT.init_cleanup(
    audit_trail_type         => DBMS_AUDIT_MGMT.AUDIT_TRAIL_ALL,
    default_cleanup_interval => 12 /* hours */);
END;
/
```

```
PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

-- set archive duration:
```

```
BEGIN
  DBMS_AUDIT_MGMT.set_last_archive_timestamp(
    audit_trail_type  => DBMS_AUDIT_MGMT.AUDIT_TRAIL_AUD_STD,
    last_archive_time => SYSTIMESTAMP-30);
END;
/

-- set archive duration:

BEGIN
  DBMS_AUDIT_MGMT.set_last_archive_timestamp(
    audit_trail_type  => DBMS_AUDIT_MGMT.AUDIT_TRAIL_AUD_STD,
    last_archive_time => SYSTIMESTAMP-30);
END;
/
```

```
-- Run the purge job:

BEGIN
  DBMS_AUDIT_MGMT.clean_audit_trail(
   audit_trail_type        =>
DBMS_AUDIT_MGMT.AUDIT_TRAIL_AUD_STD,
   use_last_arch_timestamp => TRUE);
END;
/
```

RMAN commands

In this article we have listed the most useful RMAN commands .

- **Backup and Recovery:** Perform full/incremental backups, restore databases from backups.
  - Example: `run { allocate channel ch1 type disk ...; backup database; }`

1. Take full backup(incremental level 0 of the database)

```
rman target /
configure backup optimization on;
configure controlfile autobackup on;
configure controlfile autobackup format for device type disk to
'/backup/fullbackup/%F';
```

```
configure maxsetsize to unlimited;
configure device type disk parallelism 4;


run
{
allocate channel c1 type disk format '/backup/fullbackup/%I-
%Y%M%D-%U' maxpiecesize 3G;
allocate channel c2 type disk format '/backup/fullbackup/%I-
%Y%M%D-%U' maxpiecesize 3G;
allocate channel c3 type disk format '/backup/fullbackup/%I-
%Y%M%D-%U' maxpiecesize 3G;
allocate channel c4 type disk format '/backup/fullbackup/%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 ;
}
```

2. Take incremental backup:

```
rman target /
set echo on;
configure backup optimization on;
configure controlfile autobackup on;
configure controlfile autobackup format for device type disk to
'/backup/fullbackup/%F';
configure maxsetsize to unlimited;
configure device type disk parallelism 4;

run
{
allocate channel c1 type disk format '/backup/fullbackup/%I-
%Y%M%D-%U' maxpiecesize 3G;
allocate channel c2 type disk format '/backup/fullbackup/%I-
%Y%M%D-%U' maxpiecesize 3G;
allocate channel c3 type disk format '/backup/fullbackup/%I-
%Y%M%D-%U' maxpiecesize 3G;
allocate channel c4 type disk format '/backup/fullbackup/%I-
%Y%M%D-%U' maxpiecesize 3G
backup  as compressed backupset incremental level 1  check
```

```
logical database plus archivelog;
release channel c1 ;
release channel c2 ;
release channel c3 ;
release channel c4 ;
}
```

3. Take archive log backup:

```
rman target /
set echo on;
configure backup optimization on;
configure controlfile autobackup on;
configure controlfile autobackup format for device type disk to
'/backup/fullbackup/%F';
configure maxsetsize to unlimited;
configure device type disk parallelism 4;


run
{
allocate channel c1 type disk format '/backup/fullbackup/%I-
%Y%M%D-%U' maxpiecesize 3G;
allocate channel c2 type disk format '/backup/fullbackup/%I-
%Y%M%D-%U' maxpiecesize 3G;
allocate channel c3 type disk format '/backup/fullbackup/%I-
%Y%M%D-%U' maxpiecesize 3G;
allocate channel c4 type disk format '/backup/fullbackup/%I-
%Y%M%D-%U' maxpiecesize 3G
backup archivelog all FILESPERSET=4;
delete noprompt archivelog all completed before 'SYSDATE-3';
release channel c1 ;
release channel c2 ;
release channel c3 ;
release channel c4 ;
}
```

4. Delete old archivelogs(without taking backup)

```
rman target /
DELETE noprompt force  ARCHIVELOG ALL COMPLETED BEFORE 'sysdate-
1';
CROSSCHECK ARCHIVELOG ALL;
DELETE EXPIRED ARCHIVELOG ALL;
```

5. Apply archivelog policy on standby database:

```
RMAN> configure archivelog deletion policy to applied on
standby;
```

6. backup/restore archivelogs between specific sequence:

```
-- In standalone:
RMAN>backup format '/export/dump/%d_%s_%p_%c_%t.arc.rman'
archivelog from sequence 100 until sequence 105;
```

```
-- In rac for particular thread:

RMAN> backup format '/export/dump/%d_%s_%p_%c_%t.arc.rman'
archivelog from sequence 100 until sequence 102 thread 2;
```

```
-- List and restore:

RMAN> list archivelog sequence between 10 and 20 ;

RMAN> restore archivelog sequence between 10 and 20 ;
```

7 . backup tablespace or datafile backup:

```
RMAN> backup tablespace users to destination
'/filedb/backp/database';

RMAN> BACKUP DATAFILE '/u01/apporacle/users01.dbf' to
destination '/filedb/backp/database';

-- backup to default location:

RMAN> BACKUP DATAFILE '/u01/apporacle/users01.dbf';
```

8. Enable block change tracking:

```
SQL> select name from v$database;

NAME
---------
TESTDB
```

```
SQL> select filename,status from v$block_change_tracking;

FILENAME
--------------------------------------------------------------------------------
STATUS
----------

DISABLED


SQL> alter database enable block change tracking using file
'/export/home/oracle/RMAN/TESTDB/TRACKING_FILE/block_change_TEST
DB.log';

Database altered.
```

9. View and modify rman configuration details:

```
RMAN>show configuration
RMAN configuration parameters are:
CONFIGURE RETENTION POLICY TO RECOVERY WINDOW OF 3 DAYS;
CONFIGURE BACKUP OPTIMIZATION ON;
CONFIGURE DEFAULT DEVICE TYPE TO DISK; # default
CONFIGURE CONTROLFILE AUTOBACKUP ON;
CONFIGURE CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE SBT_TAPE
TO '%F'; # default
CONFIGURE CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE DISK TO
'%F'; # default
CONFIGURE DEVICE TYPE 'SBT_TAPE' PARALLELISM 2 BACKUP TYPE TO
COMPRESSED BACKUPSET;
CONFIGURE DEVICE TYPE DISK PARALLELISM 1 BACKUP TYPE TO
BACKUPSET; # default
CONFIGURE DATAFILE BACKUP COPIES FOR DEVICE TYPE SBT_TAPE TO 1;
# default
CONFIGURE DATAFILE BACKUP COPIES FOR DEVICE TYPE DISK TO 1; #
default
CONFIGURE ARCHIVELOG BACKUP COPIES FOR DEVICE TYPE SBT_TAPE TO
1; # default
CONFIGURE ARCHIVELOG BACKUP COPIES FOR DEVICE TYPE DISK TO 1; #
default
CONFIGURE CHANNEL DEVICE TYPE 'SBT_TAPE' PARMS
'SBT_LIBRARY=mylibrary.disksbt,ENV=(BACKUP_PARAM=value)';
CONFIGURE MAXSETSIZE TO UNLIMITED; # default
```

```
CONFIGURE ENCRYPTION FOR DATABASE OFF; # default
CONFIGURE ENCRYPTION ALGORITHM 'AES128'; # default
CONFIGURE ARCHIVELOG DELETION POLICY TO NONE; # default
CONFIGURE SNAPSHOT CONTROLFILE NAME TO
'/disk1/oracle/dbs/snapcf_ev.f'; # default
```

```
-- Modify configuration:

RMAN>CONFIGURE DEVICE TYPE sbt PARALLELISM 2;

RMAN>CONFIGURE RETENTION POLICY TO RECOVERY WINDOW OF 2 DAYS;

RMAN>CONFIGURE ENCRYPTION FOR DATABASE ON;
```

11. Monitor rman backup:

```
alter session set nls_date_format='dd-mon-yyyy hh24:mi:ss';

set lines 180
col TIME_TAKEN_DISPLAY format a30
col OUTPUT_BYTES_DISPLAY format a30
col status format a10
col OUTPUT_DEVICE_TYPE format a15
select
start_time,end_time,output_device_type,status,input_type,output_
bytes_display,time_taken_display
from V$RMAN_BACKUP_JOB_DETAILS
order by start_time asc;
```

```
SELECT START_TIME, SOFAR, TOTALWORK,round(TIME_REMAINING/60,2)
"TIME_REMAINING(mins)",ROUND(SOFAR/TOTALWORK*100,2) "%COMPLETE"
FROM GV$SESSION_LONGOPS
WHERE OPNAME LIKE 'RMAN%' AND TOTALWORK != 0 AND OPNAME LIKE
'%aggregate%' AND SOFAR <> TOTALWORK;
```

12.Monitor rman restore throughput:

```
set linesize 126
column Pct_Complete format 99.99
column client_info format a25
column sid format 999
column MB_PER_S format 999.99
select s.client_info,
l.sid,
l.serial#,
l.sofar,
l.totalwork,
round (l.sofar / l.totalwork*100,2) "Pct_Complete",
aio.MB_PER_S,
aio.LONG_WAIT_PCT
from v$session_longops l,
v$session s,
(select sid,
serial,
100* sum (long_waits) / sum (io_count) as "LONG_WAIT_PCT",
sum (effective_bytes_per_second)/1024/1024 as "MB_PER_S"
from v$backup_async_io
group by sid, serial) aio
where aio.sid = s.sid
and aio.serial = s.serial#
and l.opname like 'RMAN%'
and l.opname not like '%aggregate%'
and l.totalwork != 0
and l.sofar <> l.totalwork
and s.sid = l.sid
and s.serial# = l.serial#
order by 1;
```

13. RMAN CHECKSYNTAX command:

RMAN CHECKSYNTAX:
check the syntax of RMAN commands interactively without actually executing the commands.Use the command-line argument CHECKSYNTAX to start the RMAN client in a mode in which it only parses the commands

```
$ rman checksyntax

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

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

RMAN> backup database;

The command has no syntax errors
```

14. List command in rman:

```
RMAN> list backup of database summary;
RMAN> LIST BACKUP OF DATAFILE 1;
RMAN>LIST BACKUP OF TABLESPACE DATA;
RMAN>LIST BACKUP OF ARCHIVELOG ALL;
RMAN>LIST BACKUP OF CONTROLFILE;
RMAN>LIST INCARNATION;
```

Statistics gathering commands

This article contains all the useful gather statistics related commands.

- **dbms_stats.gather_schema_stats:** Collect statistics for a specific schema.
- **dbms_stats.gather_table_stats:** Gather statistics for individual tables.
- **dbms_stats.gather_index_stats:** Collect statistics for indexes.

1. Gather dictionary stats:

```
-- It gathers statistics for dictionary schemas 'SYS', 'SYSTEM'
and other internal schemas.

EXEC DBMS_STATS.gather_dictionary_stats;
```

2. Gather fixed object stats:

```
-- Fixed object means gv$ or v$views

EXEC DBMS_STATS.GATHER_FIXED_OBJECTS_STATS;
```

3. Gather full database stats:

```
EXEC DBMS_STATS.gather_database_stats;
```

```
-- With estimate_percent to 15 percent or any other value , if
the db size very huge.

EXEC DBMS_STATS.gather_database_stats(estimate_percent => 15);
EXEC DBMS_STATS.gather_database_stats(estimate_percent => 15,
cascade => TRUE);

-- With auto sample size and parallel degree

EXEC DBMS_STATS.gather_database_stats(estimate_percent =>
DBMS_STATS.AUTO_SAMPLE_SIZE,degree => 8);
```

4. Gather schema statistics:

```
EXEC DBMS_STATS.gather_schema_stats('DBACLSS');

EXEC DBMS_STATS.gather_schema_stats('DBACLASS', estimate_percent
=> 25);

EXEC DBMS_STATS.gather_schema_stats('DBACLASS', estimate_percent
=> 100, cascade => TRUE);
```

```
-- STATS WITH AUTO ESTIMATION and degree 8

exec dbms_stats.gather_schema_stats( ownname =>
'DBACLASS',method_opt => 'FOR ALL COLUMNS SIZE 1',
granularity => 'ALL', degree => 8, cascade => TRUE,
estimate_percent=>dbms_stats.auto_sample_size);
```

5. Gather table statistics:

```
EXEC DBMS_STATS.gather_table_stats('DBACLASS', 'EMP');
EXEC DBMS_STATS.gather_table_stats('DBACLASS', 'EMP',
estimate_percent => 15);
EXEC DBMS_STATS.gather_table_stats('DBACLASS', 'EMP',
estimate_percent => 15, cascade => TRUE);

exec DBMS_STATS.GATHER_TABLE_STATS  (ownname => 'DBACLASS' ,
tabname => 'EMP',cascade => true,
method_opt=>'for all indexed columns size 1', granularity =>
'ALL', degree => 8);
```

```
exec DBMS_STATS.GATHER_TABLE_STATS (ownname => 'DBACLASS' ,
tabname => 'EMP',
cascade => true, method_opt=>'FOR ALL COLUMNS SIZE 1',
granularity => 'ALL', degree => 8);
```

6. Gather stats for single partition of a table:

```
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;
/
```

7. Lock/unlock statistics:

```
-- Lock stats of a schema:
EXEC DBMS_STATS.lock_schema_stats('DBACLASS');

-- Lock stats of a table:
EXEC DBMS_STATS.lock_table_stats('DBACLASS', 'EMP');

-- Lock stats of a partition:
EXEC DBMS_STATS.lock_partition_stats('DBACLASS', 'EMP', 'EMP');

-- unlock stats of a schema:

EXEC DBMS_STATS.unlock_schema_stats('DBACLASS');
-- unlock stats of a table:

EXEC DBMS_STATS.unlock_table_stats('DBACLASS', 'DBACLASS');
--unlock stats of a partition:

EXEC DBMS_STATS.unlock_partition_stats('DBACLASS', 'EMP',
'TEST_JAN2016');

--- check stats status:
```

```
SELECT stattype_locked FROM dba_tab_statistics WHERE table_name
= 'TEST' and owner = 'SCOTT';
```

8 . Delete statistics:

```
-- Delete complete db statistics:
EXEC DBMS_STATS.delete_database_stats;

-- Delete schema statistics:
EXEC DBMS_STATS.delete_schema_stats('DBACLASS');

-- Delete table statistics:
EXEC DBMS_STATS.delete_table_stats('DBACLASS', 'EMP');

-- Delete column statistics:
EXEC DBMS_STATS.delete_column_stats('DBACLASS', 'EMP', 'EMPNO');

-- Delete index statistics:

EXEC DBMS_STATS.delete_index_stats('DBACLASS', 'EMP_PK');

-- Delete dictionary statistics:
EXEC DBMS_STATS.delete_dictionary_stats;


-- Delete fixed object statistics:

exec dbms_stats.delete_fixed_objects_stats;

-- Delete system statistics:

exec dbms_stats.delete_system_stats('STAT_TAB');
```

8. Setting statistics preference:

```
-- View preference details for the database:

SELECT dbms_stats.get_prefs('PUBLISH') EST_PCT FROM dual;

-- View Publish preference for table


-- View Publish preference for schema:
```

```
select dbms_stats.get_prefs('PUBLISH', 'SCOTT') from dual

-- View preference details for table

select
dbms_stats.get_prefs(ownname=>'DBACLASS',tabname=>'EMP',pname=>'
PUBLISH') FROM DUAL;
select
DBMS_STATS.get_prefs(ownname=>'DBACLASS',tabname=>'EMP',pname=>'
INCREMENTAL') FROM DUAL;
select
DBMS_STATS.get_prefs(ownname=>'DBACLASS',tabname=>'EMP',pname=>'
GRANULARITY') FROM DUAL;
select
DBMS_STATS.get_prefs(ownname=>'DBACLASS',tabname=>'EMP',pname=>'
STALE_PERCENT')  FROM DUAL;
select
DBMS_STATS.get_prefs(ownname=>'DBACLASS',tabname=>'EMP',pname=>'
ESTIMATE_PERCENT')  FROM DUAL;
select
DBMS_STATS.get_prefs(ownname=>'DBACLASS',tabname=>'EMP',pname=>'
DEGREE')  FROM DUAL;

-- Set table  preferences

exec
dbms_stats.set_table_prefs('DBACLASS','EMP','PUBLISH','FALSE');
exec
dbms_stats.set_table_prefs('DBACLASS','EMP','ESTIMATE_PERCENT','
20');
exec dbms_stats.set_table_prefs('DBACLASS','EMP','DEGREE','8');


-- Set schema preferences:

exec dbms_stats.SET_SCHEMA_PREFS('DBATEST','PUBLISH','FALSE');
exec
dbms_stats.SET_SCHEMA_PREFS('DBATEST','ESTIMATE_PERCENT','20');
exec dbms_stats.SET_SCHEMA_PREFS('DBATEST','CASCADE','TRUE');


-- Set database preference:
```

```
exec dbms_stats.set_database_prefs('PUBLISH', 'TRUE');
exec dbms_stats.set_database_prefs('DEGREE', '16');


-- Set global preference:

exec dbms_stats.set_global_prefs('PUBLISH', 'TRUE');
exec dbms_stats.set_global_prefs('DEGREE', '16');
```

9 . Deleting preferences :

```
-- Deleting schema preference:

exec dbms_stats.delete_schema_prefs('DBACLASS', 'DEGREE');
exec dbms_stats.delete_schema_prefs('DBACLASS', 'CASCADE');


-- Delete database preference:
exec dbms_stats.delete_database_prefs('ESTIMATE_PERCENT',
FALSE);
exec dbms_stats.delete_database_prefs('DEGREE', FALSE);
```

10 . Publish pending statistics:

```
-- For schema DBACLASS
exec dbms_stats.publish_pending_stats('DBACLASS',null);

-- For table DBACLASS.EMP
EXEC DBMS_STATS.PUBLISH_PENDING_STATS ('DBACLASS','EMP');
11. Delete pending statistics:
-- for table DBACLASS.EMP
exec dbms_stats.delete_pending_stats('DBACLASS', 'EMP');

-- For schema DBACLASS
exec dbms_stats.delete_pending_stats('DBACLASS', null);
```

12. Upgrade stats table:

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');
```

13. View/modify statistics retention period:

```
-- 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);
```

14. create stats table:

```
--- Create staging table to store the statistics data

exec dbms_stats.create_stat_table(ownname => 'SCOTT', stattab =>
'STAT_BACKUP',tblspace=>'USERS');
```

15. Export stats data:

```
-- Export full database stats to a table SCOTT.STAT_BACKUP

exec dbms_stats.export_database_stats(statown => 'SCOTT'
,stattab=>'STAT_BACKUP');

-- Export stats for table DBACLASS.EMP to a stats table
SCOTT.STAT_BACKUP

exec dbms_stats.export_table_stats(ownname=>'DBACLASS',
tabname=>'EMP', statown =>'SCOTT',stattab=>'STAT_BACKUP',
cascade=>true);

-- Export stats for schema DBACLASS to a stats table
SCOTT.STAT_BACKUP

exec dbms_stats.export_schema_stats(ownname=>'DBACLASS', statown
=>'SCOTT' , stattab=>'STAT_BACKUP');

-- Export fixed object stats to table SCOTT.STAT_BACKUP

exec dbms_stats.export_fixed_objects_stats(statown => 'SCOTT'
,stattab=>'STAT_BACKUP');
```

```
-- Export dictionary stats to table SCOTT.STAT_BACKUP

exec dbms_stats.export_dictionary_stats(statown => 'SCOTT'
,stattab=>'STAT_BACKUP');

-- Export stats for index DBACLAS.EMP_UK1 to SCOTT.STAT_BACKUP
table

exec dbms_stats.export_index_stats(ownname=>'DBACLASS',
indname=>'EMP_UK1', statown =>'SCOTT',stattab=>'STAT_BACKUP');
```

16. Import stats table data:

```
-- Import full database stats from stats table SCOTT.STAT_BACKUP

exec dbms_stats.import_database_stats(statown => 'SCOTT'
,stattab=>'STAT_BACKUP');

-- Import stats for table DBACLASS.EMP from stats table
SCOTT.STAT_BACKUP

exec dbms_stats.import_table_stats(ownname=>'DBACLASS',
tabname=>'EMP', statown =>'SCOTT',stattab=>'STAT_BACKUP',
cascade=>true);

-- Import stats for schema DBACLASS from  stats table
SCOTT.STAT_BACKUP

exec dbms_stats.import_schema_stats(ownname=>'DBACLASS', statown
=>'SCOTT' , stattab=>'STAT_BACKUP');

-- Import fixed object stats from stats table SCOTT.STAT_BACKUP

exec dbms_stats.import_fixed_objects_stats(statown => 'SCOTT'
,stattab=>'STAT_BACKUP');

-- Import dictionary stats from  table SCOTT.STAT_BACKUP

exec dbms_stats.import_dictionary_stats(statown => 'SCOTT'
,stattab=>'STAT_BACKUP');
```

```
-- Import stats for index DBACLAS.EMP_UK1 from
SCOTT.STAT_BACKUP table

exec dbms_stats.import_index_stats(ownname=>'DBACLASS',
indname=>'EMP_UK1', statown =>'SCOTT',stattab=>'STAT_BACKUP');
```

17 . Few stats related sql queries:

```
-- Check stale stats for table:

select owner,table_name,STALE_STATS from dba_tab_statistics
where owner='&SCHEMA_NAME' and table_name='&TABLE_NAME';

--Check stale stats for index:

select owner,INDEX_NAME,TABLE_NAME from DBA_IND_STATISTICS where
owner='&SCHEMA_NAME' and index_name='&INDEX_NAME';

-- 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';

-- Space used to store statistic data in SYSAUX tablespace:


SQL> select occupant_desc, space_usage_kbytes from
v$sysaux_occupants where OCCUPANT_DESC like '%Statistics%';

-- Check whether table stats locked or not:

select owner, table_name, stattype_locked from
dba_tab_statistics where stattype_locked is not null and owner
not in ('SYS','SYSTEM');
```

TFACTL commands

- **Manage Trace Files:** Enable/disable tracing, view trace file contents.

- Example: `tfaectl trace <session_id> set ON`

1. Check tfactl status with version:

```
tfactl status
```

2. Check tfactl tool status:

```
tfactl toolstatus
```

3. Get config details:

```
tfactl print config
```

4. List of user having access to tfactl:

```
 tfactl access lsusers
```

6. changing property of a user:

```
tfactl access promote -user oracle
```

5.Adding or removing users from access list of tfactl:

```
 tfactl access add -user rpdtro
 tfactl access remove -user rdptro
```

6. change port number for tfactl:

```
tfactl set port=5001
```

NOTE – make sure to restart the tfactl after port change.

7. Stop/ start tfactl:

```
tfactl stop

tfactl start
```

8. Enable /disable autostart of tfactl upon reboot:

```
 tfactl disable

 tfactl enable
```

9. Find tfactl version with simple command:

```
tfactl version
AHF VERSION: 20.2.0
```

10. Collect diagnostic report pass the time of incident in YYYY-MM-DD HH24:MI:SS)

```
 tfactl diagcollect -all
```

11. Get notificationaddress email:

```
 tfactl get notificationAddress
```

12. Change notification address email:

```
tfactl set notificationAddress=oracle:admin@paragonacademy.com
```

13. Generate summary report :

```
 tfactl summary

-- Genearate complete summary overview in html
tfactl summary -html

-- Generate patching summary:

tfactl summary -patch -html

-- Generate asm summary
tfactl summary -asm -html
```

14.view smtp details :

```
 tfactl print smtp
```

15. Manage logs using tfactl managelogs:

```
tfactl  managelogs -show usage
```

16. Purge old logs:

```
-- This is just a dry run:

tfactl managelogs -purge -older 5d -dryrun

-- This will actually delete the logs older than 5 days

tfactl managelogs -purge -older 5d

-- Delete only GI logs:

tfactl managelogs -purge -gi 5d

tfactl run managelogs -purge -older 5d -gi

-- Delete only database logs:
tfactl run managelogs -purge -older -5d -database
```

17. Get repository location and usage:

```
tfactl print repository
```

18. Get component details:

```
tfactl print components
```

19. Find diag collection details of tfactl:

```
tfactl print collections
```

V. Conclusion

This guide provides a foundational understanding of essential DBA commands and scripts for various Oracle database management tasks. Effective DBAs will leverage these tools along with their knowledge of database architecture, optimization techniques, and security best practices to ensure optimal database performance, availability, and data integrity.

Additional Notes

- This document serves as a general guideline, and specific commands and scripts may vary depending on the Oracle version and database configuration.
- It's crucial to consult Oracle documentation and refer to metalink notes (now My Oracle Support Knowledge Base) for detailed information and best practices for each command.
- Regularly review and update the document to reflect changes in the database environment and incorporate newly learned best practices.

By following these guidelines and continuously expanding their knowledge, DBAs can effectively utilize the power of Oracle utilities to manage their databases efficiently and ensure their optimal health.