# Oracle Database Backups

**Backups are two types.**

1. **logical backup**
2. **physical backup**

**logical backup:**

* We can take logical elements backup like **Schema**, **Table**, **Tablespace** and other objects.

**Tools used for these backup’s:**

* Traditional (exp/imp)

or

* datapump (expdp/impdp)

**Physical backup:**

* **C R D** files backup
* For hot backup db must enable with archive log.

**Tools used for physical backup’s:**

* We can take by using **CONVENTIONAL** and **RMAN** backups.

**Conventional backup’s are 2 types:**

* Cold is consistent offline backup
* hot is inconsistent online backup

**Rman backup’s are 3 types:**

* Cold is consistant offline backup
* hot is inconsistant online backup
* Incremental backup
* Incremental level ‘0’
* Incremental level ‘1’

\* Incremental level ‘1’ **Differential backup**

\* Incremental level ‘1’ **Cumulative backup**

**LOGICAL BACKUP’s**:

* Full db and tablespace backup is recomended to take from previliged user. (**SYS/SYSTEM**)
* If we are taking backup of Palikila tablespace from user u1. This will backup u1 user data on that tablespace. it will not backup other users data on that tablespace.

**Traditional export / import**:

1. exp help=y shows list of available options to use in exp.
2. It is a client side tool. Dump files are stored in the same location where you have triggered backup.
3. There is no data security.
4. It is a single stream of execution. There is no parallism.this will create only one dumpfile.

**Table backup: exp**

* exp username/password file=(give backup file name).dmp log=(give backup file name).log table=username.tablename
* exp system/manager file=table\_t1.dmp log=table\_t1.log tables=u1.t1

**Schema backup: exp**

* exp username/password file=(schema name).dmp log=(schema name).log OWNER=(schema name)
* exp system/system file=u2.dmp log=u2.log OWNER=u2

**Tablespace backup: exp**

* exp username/password file=tablespace\_tablespace name.dmp log=tablespace\_tablespacename.log TABLESPACES=tablespace name
* exp system/system file=tablespace\_palikila.dmp log=tablespace\_palikila.log TABLESPACES=PALIKILA

**Full database backup: exp**

* **nohup** is to run backup in background. When we use nohup, In the end we should use **&**
* This will create **nohup.out** file we can monitor that.
* nohup exp system/teameis file=full.dmp log=full.log full=y &
* To see backup rinning or not use **Jobs**
* tail -f nohup.out

**Consistent backup in traditional export:**

* **Consistent** full db backup will take only the data available up to the particular time when your executing the job.if any data update happen. This will not backup that data.
* consistent=y
* nohup exp system/teameis file=full.dmp log=full.log full=y consistent=y &

**Datapump or expdp:** expdp help=y

1. Datapump was interdicted from oracle 10g.
2. It is a server side tool. (**directory**)
3. There is a data protection. Dump file is created in the db server itself.
4. We can use parallel.
5. There is a control over the job. (**attach**)
6. **nohup** is to run backup in background. When we use nohup, In the end we should use **&**
7. This will create **nohup.out** file. we can monitor that.
8. If we didn’t pass directory parameter in expdp. Dump is created in default directory **DATA\_PUMP\_DIR**.

**Datapump architecture:**

**Client Process:**

* The process which started over the client terminal that initiates **DATAPUMP** export or import job. This could be running from anywhere either a terminal prompt, OEM or from some API. Once the job is started client process itself is not needed.

**Shadow Process:**

* Shadow process starts when the client process initiates a session with the oracle database. This process will create master table that stores the information of the datapump job. Once the job is done table will be dropped from the database.

**Master Control Process:**

* This is the process which controls the entire datapump jobs. The job state and its progress are maintained by this process.

**Worker Process:**

* When the job request is received from the **MCP** based on parallel clause number of worker process will create and this will take the actual backup.

**Table backup: expdp**

* nohup expdp \"/as sysdba\" dumpfile=table\_t1.dmp logfile=table\_t1.log directory=EXPDP tables=u1.t1 &

**Schema backup: expdp**

nohup expdp system/teameis dumpfile=schema\_u1\_%U.dmp logfile=schema\_u1.log schemas=u1 directory=EXPDP parallel=2 &

**Full db backup: expdp**

* If we start this backup at 7 am and it’s going till 9am
* In between 7 to 9 if any data updated .it will take that data also into backup.
* nohup expdp \"/as sysdba\" directory=expdp\_mouli dumpfile=full\_db\_%U.dmp logfile=full\_db.log full=y filesize=50m parallel=2 &
* Based on the filesize option it will create number of dumpfiles.

**Consistent backup: expdp**

* **Consistent** full db backup will take only the data available up to the particular time when your executing the job.if any data update happen. This will not backup that data.
* Select CURRENT\_SCN from v$database;
* FLASHBACK\_SCN=
* FLASHBACK\_TIME=systimestamp
* expdp \"/as sysdba\" directory=EXPDP dumpfile=full\_db\_%U.dmp logfile=full\_db.log full=y size=50m FLASHBACK\_TIME=systimestamp parallel=5

**To create directory in os and sql level:**

* Mkdir /SSD/oracle/backup
* Create directory EXPDP as ‘/SSD/oracle/backup’;
* Grant read,write on directory EXPDP to sys;
* Select \* from dba\_directories;

**To check jobs status of datapump**

* select \* from dba\_datapump\_jobs;

**To control the datapump backup:**

* expdp system/system attach=job name(**SYS\_EXPORT\_SCHEMA\_01**)
* stop\_job
* start\_job
* continue\_client (shows the job log file)
* kill \_job Or kill from os level
* Ps -ef|grep expdp
* Kill -9 id number

**PHYSICAL BACKUP’S:**

* **C R D** files backup

**Conventional cold backup:**

1. Check **C R D** files location.
2. Shut down the database.
3. Create backup directory in **OS** level.
4. Copy **C R D** files to created backup directory
5. Copy datafiles to created directory.

**Conventional Hot backup**:

**Begin backup mode**: The datafiles headers are freezed and no more data transaction with these datafiles. Transactions during this period are written into physical archives.

* select status from v$backup;
* alter database begin backup;
* alter database end backup;

1. db must be enable with archivelog mode
2. Check the max count of archives.
3. Create directory in **OS** level.
4. Put the database in begin backup mode.
5. Take backup of **only datafiles**.
6. End the begin backup mode.
7. Check the max count of archives.
8. Copy archives between step 2 and step 7 to backup location.

**RMAN Backups:**

* Rman takes only used blocks backup.
* By default rman backups information is stored in controlfile for 7 day.
* To store Rman backups information permanently we have to create recovery catalog.

**To specify Rman backup location:**

* Create wanted backup location in OS LEVEL.
* show all;

RMAN>configure channel device type disk format ‘/prod/hyd/backup/%d\_%s.rman’;

* %d= database name
* %s= sequence name
* %c= multiple backup copies

**To check backups in RMAN:**

* list backup;

**To check deleted backups in RMAN:**

* crosscheck backup;

**To delete expired backup in RMAN:**

* delete expired backup; yes/ no

OR

* delete noprompt expired backup;

**To delete normal backup in RMAN:**

* delete backup ; yes/No

OR

* delete noprompt backup;

**RMAN cold backup:**

1. Shut down the database.
2. Put the db in mount state.
3. Connect to Rman.

* RMAN > backup database tag=coldbkp;
* RMAN > backup current controlfile;(tag)
* Tag is for identification.

1. open the database (startup)

**RMAN Hot Backup:**

1. db must be enabled with archive log mode.
2. When db is in online mode we can take Rman hot backup.
3. Connect to RMAN

* RMAN>backup database tag= hotbkp;
* RMAN>backup archivelog all tag=archivebkp;
* RMAN>backup current controlfile tag=ctlbkp;

**RMAN Incremental Backup:**

**Level ‘0’**

RMAN> backup incremental level 0 database;

* it’s a full db backup.
* before taking level ‘1’ backup we have to take level ‘0’ backup.
* Data updating during level 0 will backed up in level 1 backup.

**Level ‘1’ / Differential Backup:**

RMAN> backup incremental level 1 database;

* **A differential incremental backup, backs up all the blocks that have changed after the most recent incremental backup, which can be either level 1 or 0.**
* **RMAN determines the most recent level 1 backup and then backs up all the blocks changed after that backup, if no level 1 then all the blocks changed after the last level 0 backup are backed up. If no level 0 backup exist, then a level 0 backup is taken.**
* Takes less disk space and more recovery time.
* Only **updated** blocks data backup in this.

**Level ‘1’ / Cumulative Backup**

RMAN> backup incremental level 1 cumulative database;

* **A cumulative incremental backup, backs up all the blocks that have changed after the most recent incremental backup at level 0 only.**
* More disk space and takes less recovery time.

**Block Change Tracking:**

* **CTWR** will continusly monitor all the datafiles for the data blocks which are modified.
* If any block is modified that information is written into **XML** file.
* When the incremental backup is triggered **RMAN** reads this file and takes backup.
* If this is enabled there is a slite poor performance of the database.
* desc v$block\_change\_tracking
* Set DB\_CREATE\_FILE\_DEST parameter.
* alter system set db\_change\_file\_dest='/SSD/ABC/block\_c\_t';
* alter database enable block change tracking;

**To Take image copy backup:**

* By default RMAN will take only used data blocks. but if we take backup as image copy it will take full size backup.
* backup as copy datafile 1;
* backup as copy database;

## **RMAN Recovery Catalog**

* By default **RMAN** backup information is stored in control file for 7 days.
* To store RMAN backup information permanentaly it is recommended to have catalog database.
* The recovery catalog serves as a secondary metadata repository. If the target control file and all backups are lost, then the RMAN metadata still exists in the recovery catalog.
* A recovery catalog centralizes metadata for all your target databases. Storing the metadata in a single place makes reporting and administration tasks easier to perform.
* Catalog database is created on separated server.

**To Create Catalog Database:**

**Catalog Database server:**

1. Create tablespace.
2. Create user
3. Give qouta to user on tablespace.
4. Assign the below PRIVILEGE to user.

* Grant create session to user;
* Grant RECOVERY\_CATALOG\_OWNER to user;

1. Create listener

**Source database:**

* On the source database add TNS of catalog database.

**To connect in source db:**

* sqlplus ract/ract@10.100.24.51:1521/abc
* rman target / catalog=rcat/rcat@target ip address:port number/target db name
* rman target / catalog=rcat/rcat@abc
* This will connect with source rman and catalog database.

RMAN> create catalog;

RMAN> register database;

* In catalog db tables are created under user to store Rman backups information.
* Database registration should be done for every database which needs recovery catalog.

**To store before Rman backup information in catalog:**

* connect to rman target / catalog=rcat/rcat@abc
* resync catalog

**RMAN CONFIGURATION:**

1. **RETENTION POLICY is of 2 types:**

* You can configure retention policy to either number of days or number of copies. By default the RETENTION POLICY is set to redundancy 1.
* **REDUNDANCY 1**
* RMAN will keep 1 backup copies of each file. Backup copies which are older than 1 copies can be marked as obsolete and hence can be deleted.
* **CONFIGURE RETENTION POLICY TO REDUNDANCY 1;**
* **RECOVERY WINDOW 2**
* RMAN will mark those backups which are older than 2 days as obsolete and hence can be deleted.
* **CONFIGURE RETENTION POLICY TO RECOVERY WINDOW OF 2 DAYS;**

1. **OPTIMIZATION**:

* By default it is in **OFF** state.
* If this is **ON** rman will backup only the datafiles which are data modified.
* If not modified it will skip the datafile backup.
* **CONFIGURE BACKUP OPTIMIZATION ON;**
* **CONFIGURE BACKUP OPTIMIZATION OFF;**

1. **DEVICE TYPE TO DISK or TAPE:**

* By default it is disk.
* If we want TAPE we should configure tape.
* **CONFIGURE DEFAULT DEVICE TYPE TO 'DISK';**
* **CONFIGURE DEFAULT DEVICE TYPE TO 'TAPE';**

1. **CONTROLFILE AUTOBACKUP:**

* By default it is **OFF**.
* If this is **ON** whenever some changes are done in the database level control file backup.
* Anyways controlfile will backedup along with **RMAN** backup.

1. **CONTROLFILE AUTOBACKUP FORMAT:**

* This will take the control file in specified format.
* **CONFIGURE CONTROLFILE AUTOBACKUP ON;**
* **CONFIGURE CONTROLFILE AUTOBACKUP OFF;**

1. **DEVICE TYPE DISK PARALLELISM:**

* **PARALLELISM** is specified depends on server cpu core’s.
* For 1 cpu core we can specify parallelism to 2.
* We can specify number of channels that is equal to parallelism.

1. **DATAFILE BACKUP COPIES / ARCHIVELOG BACKUP COPIES**

* This will specify backup copies of datafile and archivelog.
* If this is specified to more than 1, Then %c need to add in **DEVICE TYPE DISK FORMAT**
* This is not required more then 1 copy.

1. **DEVICE TYPE DISK FORMAT**

* This will specify the backup stored location.

1. **MAXSETSIZE TO**

* This will specify the max size of the backup file.
* If the specified size is less then the backup file. then backup will nothappen.

1. **ARCHIVELOG DELETION POLICY**

* This is about archive log deletion policy.
* For ex: CONFIGURE ARCHIVELOG DELETION POLICY TO backed up 2 times to disk;
* Archives which are backed uo 2 times to disk will be deleted.

1. **SNAPSHOT CONTROLFILE**

* Whenever rman backup is triggered controlfile will be locked.untill the backup is done there will not be any update in the controlfile.
* So instead of locking actual controlfile rman will create snapshot controlfile which is copy of existing controlfile in ORACLE\_HOME/dbs location and uses this file.

**BACKUPS**:

* backup archivelog all;
* backup database;
* backup datafile 1;
* backup datafile 2;
* Backup current controlfile;

**For standby controlfile backup**:

* backup current controlfile for standby;