Practice 8

Improving Backups

Practice Target

In this practice you will gain experience on using three `BACKUP` command options.

Practice Overview

In this practice, you will perform the following tasks:

- Enable compression when making backupsets
- Take backups with multisection option
- Create an archival backup

Assumptions

This practice assumes the `srv1` appliance is up and running and its database `ORADB` is running in `OPEN` state.

Note

I recommend taking a snapshot of the appliances before you start implementing the practice.

A. Enabling Block Compression

In the following steps, you will enable block compression and study its influence.

1. Open Putty and login to `srv1` as `oracle`.

2. Invoke RMAN and connect to the local database as target.

```
rman target "'/ as SYSBACKUP'"
```

3. Run the following command to display the compression algorithm that is currently configured in RMAN.

`BASIC` is the default compression algorithm in RMAN. Using it does not require a separate license.

```
SHOW COMPRESSION ALGORITHM;
```

4. Issue the following command to take backup of the entire database.

```
run {
 ALLOCATE CHANNEL c1 DEVICE TYPE DISK;
 BACKUP DATABASE TAG 'NO_COMP';
}
```

5. List the backupset files produced by the command executed in the previous step. Take a note of the "`Elapsed Time`" value, "`Compressed`" value and the size.

```
LIST BACKUPSET TAG 'NO_COMP';
```

6. Delete the produced backupset.

```
DELETE BACKUPSET TAG 'NO_COMP';
```

7. Issue the following commands to take backup of the entire database with compression algorithm set to `MEDIUM`.

You can set the compression algorithm in the RMAN persistent settings level using `CONFIGURE` command.

Setting compression algorithm requires "Advanced Compression Options" licenses.

```
SET COMPRESSION ALGORITHM 'MEDIUM';
BACKUP AS COMPRESSED BACKUPSET DATABASE TAG 'COMP_MEDIUM';
```

8. List the backupset files produced and take a note of the "`Elapsed Time`" value, "`Compressed`" value and the backupset size.

```
LIST BACKUPSET TAG 'COMP_MEDIUM';
```

9. Delete the produced backupset.

```
DELETE BACKUPSET TAG 'COMP_MEDIUM';
```

10. Set the compression algorithm to `HIGH` and produce the backupset again.

```
SET COMPRESSION ALGORITHM 'HIGH';
BACKUP AS COMPRESSED BACKUPSET DATABASE TAG 'COMP_HIGH';
```

11. List the backupset files produced and take a note of the "`Compressed`" value and the backupset size. Compare between the sizes obtained from the previous tests.

```
LIST BACKUPSET TAG 'COMP_HIGH';
```

12. Delete the produced backupset.

```
DELETE BACKUPSET TAG 'COMP_HIGH';
```

Conclusion

The higher you go with compression algorithm, the better compression ratio you obtain but it will be at the expense of CPU processing and backup time. In real life scenario, a similar benchmark testing should be established to judge compression influence. The CPU consumption and backup elapsed time could be substantial.

B. Backup with Multisection Option

In the following steps, you perform backup with multisection option.

13. Set the compression algorithm back to its default value `BASIC`. Alternatively, exit from RMAN and invoke it again.

```
SET COMPRESSION ALGORITHM 'BASIC';
```

14. Issue the following command and observe the size of the `SOETBS` tablespace. Round the size to the highest MB.

```
REPORT SCHEMA;
```

15. Issue the following command after replacing the <*section size value*> with half of the value obtained from the previous step.

```
RUN {
 ALLOCATE CHANNEL c1 DEVICE TYPE DISK;
 ALLOCATE CHANNEL c2 DEVICE TYPE DISK;
 BACKUP TABLESPACE SOETBS TAG 'SOETBS_MULTIS' SECTION SIZE <section size value>M;
}
```

16. Study the output of the previous command. You will see two channels running in parallel. The input file of each channel is the same datafile.

17. Issue the following command and observe how many backup pieces belongs to the backupset.

```
LIST BACKUPSET TAG 'SOETBS_MULTIS';
```

18. Obtain the "BS Key" value from the output of the previous step and substitute it in the following query. Run the query afterwards.

```
SELECT PIECES,MULTI_SECTION
FROM V$BACKUP_SET
WHERE RECID=<BS Key> ;
```

19. Obtain the physical size of each backup piece. Use the following command format.

```
HOST 'ls -alh <backup piece full name>';
```

- Why do you think they are not equal in sizes, although each channel backed up half of the input file?

Clean up

20. Delete the backupset.

```
DELETE BACKUPSET TAG 'SOETBS_MULTIS';
```

C. Creating an Archival Backup

In the following steps, you create an archival backup. An archival backup is a backup that does not obey the normal retention policy and is not saved in the FRA.

21. Shutdown the database and start it up in `MOUNT` state.

Archival backups can be taken online or offline. However, usually archival backups are taken for very long term recovery, like yearly or quarterly. If you take it offline, you do not need archived redo log files or incremental backups to restore and recover the database. And you do not have to open the recovered database using `RESETLOGS` option.

However, it is not required to have the database in mount state. If archival backups are taken online, they will be automatically self-contained; i.e. all the needed archive logs and incremental backups will be included.

```
SHUTDOWN IMMEDIATE
STARTUP MOUNT
```

22. Try creating an archival backup by executing the following command.

The command fails (`ORA-19811`) because you cannot save the archival backup in the FRA. Archival backups are usually kept for long time and saving it in FRA may cause it to run out of free disk space.

```
BACKUP DATABASE KEEP UNTIL TIME 'SYSDATE+365';
```

23. Create an archival backup by executing the following command.

```
BACKUP DATABASE FORMAT '/media/sf_extdisk/%U' TAG DB_KEEP KEEP UNTIL TIME 'SYSDATE+365'
RESTORE POINT DB_KEEP_18Q3;
```

24. Open the database for read/write operations.

```
ALTER DATABASE OPEN;
```

25. Issue the following command and observe the value of the "Keep".

```
LIST BACKUP tag 'DB_KEEP';
```

26. Issue the following query to obtain information about the archival backupsets.

```
SELECT S.RECID, S.SET_COUNT, S.BACKUP_TYPE,  S.PIECES, P.PIECE#, S.KEEP_OPTIONS,
S.KEEP_UNTIL
FROM V$BACKUP_SET S, V$BACKUP_PIECE P
WHERE S.RECID = P.RECID
  AND S.KEEP='YES';
```

27. List the restore points registered in the control file.

```
LIST RESTORE POINT ALL;
```

Clean up

28. Delete the backupset and drop the restore point.

```
DELETE BACKUPSET TAG 'DB_KEEP';
DROP RESTORE POINT DB_KEEP_18Q3;
```

Summary

In this practice, you performed the following tasks:

- Enable compression when making backupsets
- Take backups with multisection option
- Create an archival backup