Practice

Performing Incremental Backups

Practice Target

In this practice you will use RMAN to create incremental backups and incrementally updated backups.

Practice Overview

In high level, in this practice, you will perform the following tasks:

- Take incremental database backups, differential and cumulative
- Enable Block Change Tracking (BCT) in the database
- Create database incrementally updated backups

Assumptions

- This practice assumes that you have `srv1` up and running from the **non-CDB** snapshot.

A. Taking Incremental Database Backup

In the following steps you will create database incremental backups.

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. Take level zero incremental backup of the database.

   Although level zero incremental backup has the same contents as the full database backup, technically they are not the same. Only zero incremental backup can be used for incremental backups.

```
BACKUP INCREMENTAL LEVEL 0 DATABASE TAG 'DBLVL0';
```

4. Write down the time period taken for making level zero incremental backup.

5. List backupsets of the database. Examine the output and see how you can tell that the listed backupset is a level zero incremental backup.

```
LIST BACKUP OF DATABASE;
```

6. Write down the size of level zero incremental backupset file.

7. Open another Putty session and login to the database as `HR` user. You will use this session to generate some user activities. This session will be referred to as the **client session**.

```
sqlplus hr/ABcd##1234
```

8. In the **client** session, run the following code to simulate changes made by the application on the data. The code makes a copy of the EMPLOYEES table, keeps updating randomly selected employees from the table, and it issues a COMMIT every 100 updates. After 3 minutes, the code finishes. The target of this execution is to generate redo entries in the database.

```
CREATE TABLE EMP AS SELECT * FROM EMPLOYEES;

DECLARE
 N NUMBER;
 B DATE;
BEGIN
 B := SYSDATE;
 N :=1;
-- endless loop
WHILE TRUE LOOP
 -- update the salary of a randomly selected employee
 UPDATE EMP SET SALARY = DBMS_RANDOM.VALUE(1000,10000) WHERE EMPLOYEE_ID=
ROUND(DBMS_RANDOM.VALUE(100,206));
 -- commit every 100 UPDATE statement
 IF MOD(N,100)=0 THEN
  COMMIT;
 END IF;
 -- exit from the loop after 3 minutes
 IF ((SYSDATE-B)* 24 * 60) >=3 THEN
  COMMIT;
  EXIT;
 END IF;
END LOOP;
END;
/

DROP TABLE EMP PURGE;
```

9. Exit from the client session after it finishes and close its Putty session.

10. Switch the redo log file.

```
ALTER SYSTEM SWITCH LOGFILE;
```

11. In the Putty session, take level one incremental backup of the database.

```
BACKUP INCREMENTAL LEVEL 1 DATABASE TAG 'DBLVL1';
```

- Is this a cumulative or a differential incremental backup?

12. Write down the period of time taken for making level one incremental backup.

13. List backupsets of database. Compare the sizes of level zero and level one incremental backups.

```
LIST BACKUP OF DATABASE;
```

14. Write down the size of the level one incremental backupset file.

15. Compare between the periods and sizes of the level zero and level one incremental backups.

16. Take level one incremental backup of the database for the second time.

```
BACKUP INCREMENTAL LEVEL 1 DATABASE TAG 'DBLVL1';
```

17. Write down the period of time taken for making level one incremental backup in the second attempt.

    The second level 1 incremental backup must be much faster than the first level 1 incremental backup because there was hardly any changes on the database during the time between the two backups.

18. List backupsets of database. Compare between the sizes of the last two level one incremental backups.

    Observe that the last incremental backup is much less in its size than the size of the previous level one incremental backup.

```
LIST BACKUP OF DATABASE;
```

19. Take a cumulative level one incremental backup of the database.

```
BACKUP INCREMENTAL LEVEL 1 CUMULATIVE DATABASE TAG 'DBLVL1';
```

20. List backupsets of database. Compare between the last two taken level one incremental backups.

    The cumulative backup takes space nearly the same as the first attempt level 1 incremental backupset.

```
LIST BACKUP OF DATABASE;
```

B. Enabling Block Change Tracking

In the following steps you will enable the block change tracking feature in the database and study its influence on the incremental backups.

21. Determine if block change tracking is enabled

```
SELECT STATUS, FILENAME FROM V$BLOCK_CHANGE_TRACKING;
```

22. Enable block change tracking:

Note: RMAN does not support taking backup of BCT file, even if we save it in FRA.

```
ALTER SESSION SET DB_CREATE_FILE_DEST='/u01/app/oracle/fast_recovery_area';
ALTER DATABASE ENABLE BLOCK CHANGE TRACKING;

# verify:
SELECT status, filename FROM V$BLOCK_CHANGE_TRACKING;
```

23. Take level one incremental backup of the database.

```
BACKUP INCREMENTAL LEVEL 1 DATABASE TAG 'DBLVL1';
```

24. Run the following query to check if the BCT has been used for the incremental backup.

```
SELECT USED_CHANGE_TRACKING, FILE#, AVG(DATAFILE_BLOCKS), AVG(BLOCKS_READ)
FROM   V$BACKUP_DATAFILE
WHERE  INCREMENTAL_LEVEL > 0
GROUP  BY USED_CHANGE_TRACKING, FILE# ORDER BY 1;
```

25. Take level one incremental backup of the database.

```
BACKUP INCREMENTAL LEVEL 1 DATABASE TAG 'DBLVL1';
```

26. Check if the BCT was used for the incremental backup.

```
SELECT USED_CHANGE_TRACKING, FILE#, AVG(DATAFILE_BLOCKS), AVG(BLOCKS_READ)
FROM   V$BACKUP_DATAFILE
WHERE  INCREMENTAL_LEVEL > 0
GROUP  BY USED_CHANGE_TRACKING, FILE# ORDER BY 1;
```

Clean up

27. Delete all the backupsets.

```
DELETE BACKUPSET;
```

C. Creating Database Incrementally Updated Backup

In the following steps, you will create incrementally updated database backup.

Note: "**incrementally updated database backup**" is not the same as "**incremental backups**".

28. Execute the following run block:

```
Run
{
 RECOVER COPY OF DATABASE WITH TAG 'incr_update';
 BACKUP INCREMENTAL LEVEL 1 FOR RECOVER OF COPY WITH TAG 'incr_update' DATABASE;
}
```

29. Examine the output of the executing run block in the previous step and try figuring out what RMAN has performed.

Observe that the output of executing the code above is a copy of the database data files.

30. Issue the following command to list the image copies of the database. Take a copy of the command output.

Observe that the only way to distinguish between the image copies that are generated by the `BACKUP AS COPY` command and the image copies that are generated by the "`BACKUP INCREMENTAL LEVEL 1 FOR RECOVER OF COPY`" is the `TAG` value.

```
LIST COPY OF DATABASE;
```

31. (optional) Issue the following query.

The view `V$DATAFILE_COPY` can inform about incremental level of image files.

```
SELECT FILE#, STATUS, CREATION_TIME, COMPLETION_TIME,
TAG, BLOCKS*BLOCK_SIZE/1024/1024 SIZE_MB, INCREMENTAL_LEVEL
FROM V$DATAFILE_COPY
WHERE DELETED ='NO'
ORDER BY FILE#, RECID;
```

32. Execute the following command twice, just to generate some redo archive logs.

```
ALTER SYSTEM SWITCH LOGFILE;
```

33. Execute the run block again.

34. Examine its output again and see what RMAN has done.

Observe the following message from the run block execution output. This indicates that RMAN has created incremental level 1 backup.

```
channel ORA_DISK_1: starting incremental level 1 datafile backup set
```

35. List the backup sets of the database. Observe their type.

Incremental backup level 1 is taken.

```
LIST BACKUPSET OF DATABASE;
```

36. Execute the following command twice, just to generate some redo archive logs.

```
ALTER SYSTEM SWITCH LOGFILE;
```

37. Execute the run block again (this is the third time).

38. Examine its output again and see what RMAN performed.

From the output, we can see that RMAN is performing two actions:

- "starting incremental datafile backup set restore": applying the already taken level 1 changes on the data files
- "starting incremental level 1 datafile backup set": taking level 1 incremental backup

39. Issue the following command and take a copy of the command output. Compare it with the output of the same command taken earlier.

You will notice that the "`Ckp SCN`" has been incremented in the datafile copies, which is an indication that the datafiles copies have been updated.

```
LIST COPY OF DATABASE;
```

40. Delete the database image copies and the backupsets.

```
DELETE COPY OF DATABASE;
DELETE BACKUPSET;
```

Cleanup

41. Shutdown `srv1` and restore it from its **non-CDB** snapshot.

Summary

In this practice, you gained practical experience on performing the following tasks:

- Take incremental database backups: differential and cumulative. Differential incremental backups takes less space than the cumulative incremental backups.
- Enable Block Change Tracking (BCT) in the database. BCT increases the speed of making incremental backups. It does not take a lot of disk space and, therefore, it is recommended to enable it in a production database.
- Create database incrementally updated backup. Incrementally updated backup is a fast method of keeping an updated image copy of the database.