16

Performing Point-in-Time Recovery

Objectives

After completing this lesson, you should be able to:

- Distinguish and describe point-in-time recovery (PITR) of the database, tablespace, and table
- Identify the circumstances where PITR is a good solution and where it cannot be used
- List what operations occur when you perform a point-in-time recovery
- Determine the correct target time for the point-in-time recovery
- Perform automated TSPITR
- Perform table recovery from backups

Point-in-Time Recovery

Point-in-time recovery benefits:

- Quick recovery of one or more objects to an earlier time
- No effect on other objects

Recovery scope:

- Database point-in-time recovery (DBPITR), also referred to as incomplete recovery
- Tablespace point-in-time recovery (TSPITR)
- Table point-in-time recovery (TPITR)

Performing Point-in-Time Recovery

Perform database point-in-time (incomplete) recovery by doing the following:

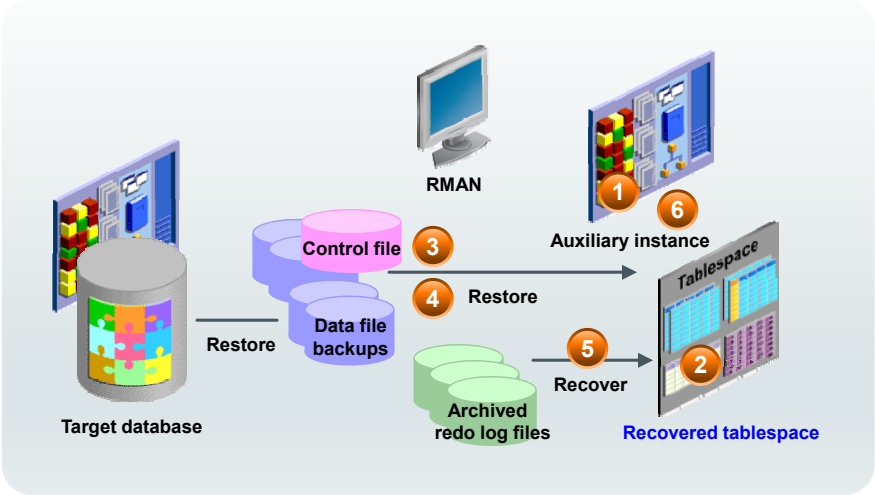
1. Determine the target point of the restore: SCN, time, restore point, or log sequence number.
2. Set the NLS environment variables appropriately.
3. Mount the database.
4. Prepare and execute a `RUN` block, using the `SET UNTIL`, `RESTORE`, and `RECOVER` commands.
5. Open the database in `READ ONLY` mode and verify that the recovery point is correct.
6. Open the database by using `RESETLOGS`.

When to Use TSPITR

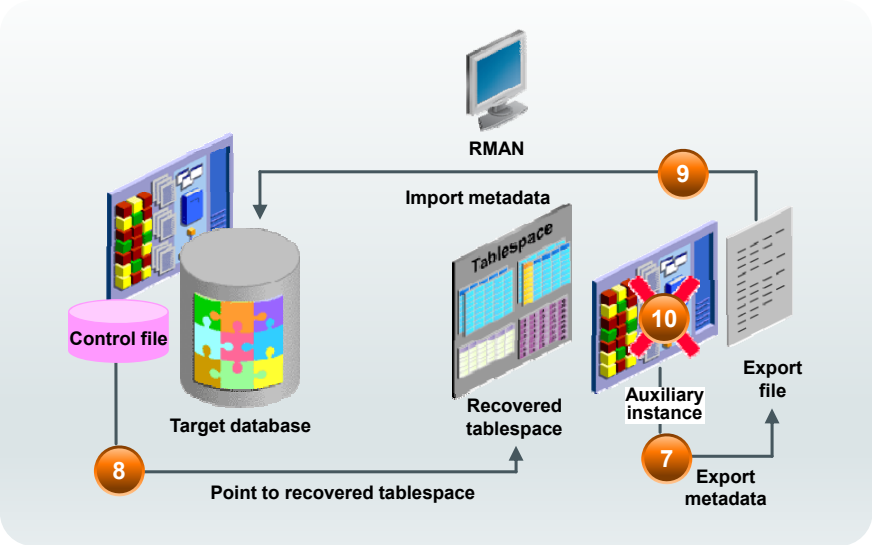
TSPITR can be used in the following situations:

- To recover data lost after an erroneous `TRUNCATE TABLE` statement
- To recover from logical corruption of a table
- To undo the effects of a batch job or DML statements that have affected only a part of the database
- To recover a dropped tablespace

Tablespace Point-in-Time Recovery: Architecture



Tablespace Point-in-Time Recovery: Architecture



Preparing for TSPITR

To prepare for TSPITR, perform the following steps:

- Determine the correct target time.
- Determine what is needed in the recovery set.
- Identify and preserve objects that will be lost after PITR.

Determining the Correct Target Time

- Use the following methods to determine the correct target time:
  - Flashback Query
  - Flashback Transaction Query
  - Flashback Version Query
- Simple alternative to TSPITR: Flashback data (if still available as undo)

Determining the Tablespaces for the Recovery Set

- If objects in the tablespace that you are recovering have relationships with objects in other tablespaces, you can:
  - Add the tablespace that contains the related objects to the recovery set
  - Suspend the relationship for the duration of TSPITR
  - Remove the relationship
- Use the `DBMS_TTS.TRANSPORT_SET_CHECK` procedure to determine whether the tablespaces in the recovery set are self-contained.

```
DBMS_TTS.TRANSPORT_SET_CHECK ('USERS,EXAMPLE');
SELECT *  FROM TRANSPORT_SET_VIOLATIONS;
```

Identifying Objects That Will Be Lost

- Objects created in the tablespace after the target recovery time are lost.
- Query `TS_PITR_OBJECTS_TO_BE_DROPPED` to determine which objects will be lost after TSPITR.
- Use Export before TSPITR and Import after TSPTIR to preserve and re-create the lost objects.

Performing Fully Automated TSPITR

1. Configure channels required for TSPITR on the target instance.
2. Specify the auxiliary destination by using the `AUXILIARY DESTINATION` option.

```
RMAN> CONNECT TARGET
RMAN> RECOVER TABLESPACE users, example
    > UNTIL TIME '2018-06-29:08:00:00'
    > AUXILIARY DESTINATION
    > '/u01/app/oracle/oradata/aux';
```

3. Back up the recovered tablespaces and bring them online.

Alternative Location

- `CONFIGURE AUXNAME` for a persistent alternative location for an auxiliary set data file image copy
- `SET NEWNAME` for an alternative location for the duration of a `RUN` command

```
RUN
{
SET NEWNAME FOR DATAFILE '$ORACLE_BASE/oradata/orcl/users01.dbf'
TO '/u01/backup/users01.dbf';

RECOVER TABLESPACE users UNTIL SEQUENCE 1300 THREAD 1;
}
```


PITR of PDBs

- PDB PITR

```
RMAN> ALTER PLUGGABLE DATABASE pdb1 CLOSE;
RMAN> RUN {
       SET UNTIL SCN = 1851648 ;
       RESTORE pluggable DATABASE pdb1;
       RECOVER pluggable DATABASE pdb1
        AUXILIARY DESTINATION='/u01/app/oracle/oradata';
       ALTER PLUGGABLE DATABASE pdb1 OPEN RESETLOGS;
          }
```

- PDB TSPITR

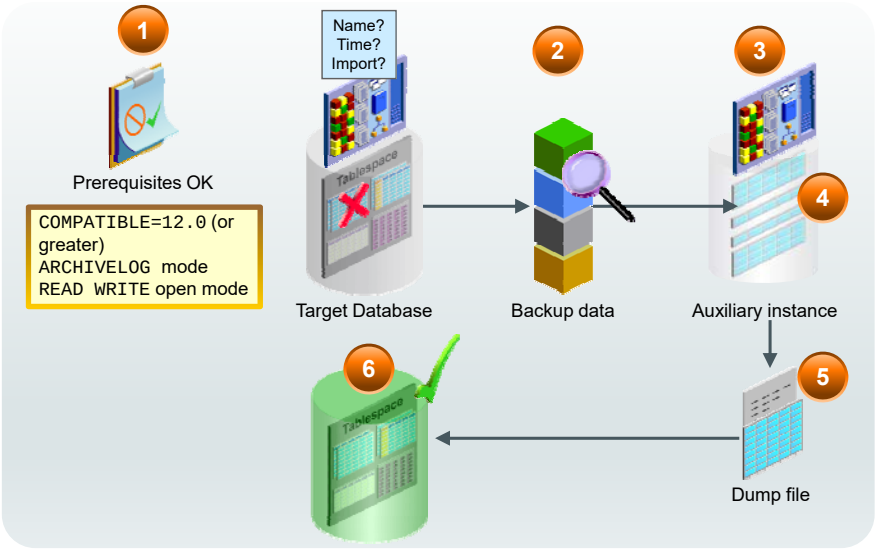
```
RMAN> RECOVER TABLESPACE pdb1:test_tbs
              UNTIL SCN 832972
              AUXILIARY DESTINATION '/tmp/CDB1/reco';
RMAN> ALTER TABLESPACE pdb1:test_tbs ONLINE;
```

Recovering Tables from Backups

When to recover tables and table partitions from RMAN backups:

- Small number of tables (no TSPITR)
- Not in a self-contained tablespace (no TSPITR)
- Purged tables (no Flashback drop)
- Beyond the available undo (no Flashback Table)
- After a structural DDL change (no Flashback Table)

Table Recovery: Graphical Overview



Prerequisites and Limitations

To recover tables and table partitions from RMAN backups, the target database must be:

- In read/write mode
- In ARCHIVELOG mode

Limitations of recovery: No tables and table partitions from:

- The SYS schema
- The SYSTEM and SYSAUX tablespaces
- Standby databases

Specifying the Recovery Point in Time

Recover table and table partitions to the state they were in by specifying:

- `UNTIL SCN integer`: The system change number (SCN)
- `UNTIL TIME 'date_string'`: The time in the date format:
  - Of the NLS_LANG and NLS_DATE_FORMAT environment variables, or
  - Date constants, for example, `SYSDATE - 5`
- `UNTIL SEQUENCE integer (THREAD integer)`: The log sequence number and thread number

Process Steps of Table Recovery

1. Perform the planning tasks and start an RMAN session with the `CONNECT TARGET` command.
2. Enter the `RECOVER TABLE` command.
3. RMAN determines the backup based on your specification.
4. RMAN creates an auxiliary instance by using the `AUXILIARY DESTINATION` clause, if specified.
5. RMAN recovers your tables or table partitions, up to the specified point in time, into this auxiliary instance.
6. RMAN creates a Data Pump export dump file that contains the recovered objects with the `DUMP FILE=`*name* and `DATAPUMP DESTINATION=<`*OS path*`>`.

Note: If a file with the name specified by `DUMP FILE` exists in the location in which the dump file must be created, then the export fails.

Process Steps of Table Recovery

7. RMAN imports the recovered objects into the target database unless you specified `NOTABLEIMPORT`.
8. RMAN optionally renames the recovered tables or table partitions with the `REMAP TABLE` and the `REMAP TABLESPACE` clauses. (Existing objects are not changed.)

   Note: If you remap a table, the dependent objects are excluded from the import. You must re-create indexes and constraints.

Summary

In this lesson, you should have learned how to:

- Distinguish and describe point-in-time recovery (PITR) of table, tablespace, and database
- Identify the circumstances where PITR is a good solution and where it cannot be used
- List what operations occur when you perform a PITR
- Determine the correct target time for the PITR
- Perform automated TSPITR
- Perform table recovery from backups

Practice Overview

- Recovering from Media Failure: Incomplete Recovery
- Recovering a Table from a Backup