Oracle Flashback Recovery

When restoring the whole thing, rolling it forward, opening with resetlogs, exporting the data, importing the data, and then deleting the restore is not desirable.

Travis Parker

October 22, 2013

Oops – dropped a table

```
SQL> create table oops as select * from all_objects where rownum < 100;
Table created.

SQL> create unique index ind_oops_objectid on oops (object_id);
Index created.

SQL> create index ind_oops_objname on oops (object_name);
Index created.

SQL> drop table oops;   --oops
Table dropped
```

Flashback Drop

```
--This is your chance to get the index names, they're gone after you flash the table
   back
SQL> SELECT OBJECT_NAME, ORIGINAL_NAME, TYPE FROM RECYCLEBIN; --get index names

OBJECT_NAME                              ORIGINAL_NAME                    TYPE
BIN$ozHVKCxhnyTgQKjABAFXPg$0             IND_OOPS_OBJECTID                INDEX
BIN$ozHVKCxinyTgQKjABAFXPg==$0           IND_OOPS_OBJNAME                 INDEX
BIN$ozHVKCxjnyTgQKjABAFXPg==$0           OOPS                             TABLE

SQL> flashback table oops to before drop;
Flashback complete.

SQL> alter index "BIN$ozHVKCxhnyTgQKjABAFXPg==$0" rename to IND_OOPS_OBJECTID;
Index altered.

SQL> alter index "BIN$ozHVKCxinyTgQKjABAFXPg==$0" rename to IND_OOPS_OBJNAME;
Index altered.
```

What if it has been re-created but you need the original back?

```
SQL> create table oops as select * from all_objects where rownum < 100;
Table created.

SQL> drop table oops;
Table dropped.

SQL> create table oops as select * from all_users;
Table created.

SQL> drop table oops;
Table dropped.

SQL> SELECT object_name, original_name, createtime FROM recyclebin;

OBJECT_NAME                              ORIGINAL_NAME       CREATETIME
BIN$ozHVKCyGnyTgQKjABAFXPg==$0           OOPS                2011-05-14:16:39:00
BIN$ozHVKCyKnyTgQKjABAFXPg==$0           OOPS                2011-05-14:16:39:31

SQL> flashback table "BIN$ozHVKCyGnyTgQKjABAFXPg==$0" to before drop;
Flashback complete.
```

Oops – hosed a table with DML

```
SQL> select count(*) from oops;

COUNT(*)
----------
        99

SQL> delete from oops;     --oops
99 rows deleted.

SQL> commit;               ---bigger oops
Commit complete.

SQL> select systimestamp from dual;

SYSTIMESTAMP
-----------------------------------
14-MAY-11 04.52.20.335143 PM -04:00

SQL> select count(*) from oops;

COUNT(*)
----------
         0
```

Flashback Table

```
SQL> alter table oops enable row
   movement;

Table altered.


SQL> flashback table oops to timestamp
   to_timestamp('14-MAY-11 04.52.00 PM',
                'DD-MON-YY HH.MI.SS AM');
Flashback complete.

SQL> select count(*) from oops;

COUNT(*)
----------
        99
```

Another Option - Flashback Query

```
SQL> create table xx (id number PRIMARY
   KEY, val varchar2(10));
Table created.

SQL> insert into xx values (1, 'One');
1 row created.

SQL> insert into xx values (2, 'Two');
1 row created.

SQL> commit;
Commit complete.

SQL> select systimestamp from dual;
14-MAY-11 06.49.37.617953 PM -04:00

SQL> delete from xx;
2 rows deleted.

SQL> commit;
Commit complete.
```

Another Option - Flashback Query

```
SQL> insert into xx select * from xx as of
timestamp to_timestamp('14-MAY-11 06.49.37
PM', 'DD-MON-YY HH.MI.SS AM');
2 rows created.

SQL> commit;
Commit complete.

SQL> select * from xx;

        ID VAL
---------- ----------
         1 One
         2 Two
```

Another option – Flashback Transaction

```
SQL> create table xx (id NUMBER PRIMARY KEY, val
varchar2(10));
Table created.

SQL> insert into xx values (1, 'One');
1 row created.
SQL> insert into xx values (2, 'Two');
1 row created.

SQL> commit;
Commit complete.

SQL> update xx set val = 'BOGUS';
2 rows updated.

SQL> commit;
Commit complete.

SQL> insert into xx values (3, 'Three');
1 row created.
SQL> commit;

        ID VAL
---------- ----------
         1 BOGUS
         2 BOGUS
         3 Three
```

Flashback Versions Query – Listing Versions Of Rows

```
 1  select versions_xid XID, versions_startscn START_SCN, versions_endscn END_SCN,
versions_operation OPERATION,
id, val
FROM xx
VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE


XID               START_SCN    END_SCN O         ID VAL
---------------- ---------- ---------- - ---------- ----------
01001200520A0000    5206343            I          3 Three
06002100D50B0000    5206322            U          2 BOGUS
06002100D50B0000    5206322            U          1 BOGUS
040006006F0A0000    5206289    5206322 I          2 Two
040006006F0A0000    5206289    5206322 I          1 One
```

Flashback Transaction Query – Listing Transactions

```
--This should work, but gives incorrect results.
SQL> select operation, logon_user, undo_sql
     from flashback_transaction_query
     where xid = HEXTORAW('06002100D50B0000');

--Another option - DBMS_FLASHBACK.TRANSACTION_BACKOUT procedure.
PROCEDURE TRANSACTION_BACKOUT
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 NUMTXNS                        NUMBER                  IN
 XIDS                           XID_ARRAY               IN
 OPTIONS                        BINARY_INTEGER          IN     DEFAULT
 TIMEHINT                       TIMESTAMP               IN

PROCEDURE TRANSACTION_BACKOUT
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 NUMTXNS                        NUMBER                  IN
 NAMES                          TXNAME_ARRAY            IN
 OPTIONS                        BINARY_INTEGER          IN     DEFAULT
 SCNHINT                        NUMBER                  IN     DEFAULT

PROCEDURE TRANSACTION_BACKOUT
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 NUMTXNS                        NUMBER                  IN
 NAMES                          TXNAME_ARRAY            IN
 OPTIONS                        BINARY_INTEGER          IN     DEFAULT
 TIMEHINT                       TIMESTAMP               IN
```

Oops - Truncated a table

PRIMARY	STANDBY
SQL> Create table xx as select * from all objects where rownum < 10; Table created	
SQL> select dbms_flashback.get_system_change_number from dual; 5198049	
SQL> truncate table xx; Table truncated.	
SQL> select dbms_flashback.get_system_change_number from dual; 5199356	
	SQL> select * from xx; No rows selected.

Flashback Database on the Standby

PRIMARY	STANDBY
	DGMGRL> edit database 'ORCLSB1' set state = 'APPLY-OFF'; Succeeded.
	SQL> alter database close; Database altered.
	SQL> flashback database to scn 5198049; Flashback complete.
	SQL> alter database open read only; Database altered.
	SQL> select count(*) from parktr01.xx; COUNT(*) ---------- 9
Create public database link ORCLSB1 using 'ORCLSB1'; Database link created.	

Repair the Primary

PRIMARY	STANDBY
SQL> connect parktr01 Enter password: Connected.	
SQL> insert into parktr01.xx select * from parktr01.xx@ORCLSB1; 9 rows created.	
SQL> commit; Commit complete.	

Recover the standby and resume apply

PRIMARY	STANDBY
	SQL> alter database close; Database altered.
	SQL> recover standby database; ORA-00279: change 5198050 generated at 05/17/2011 19:46:28 needed for thread 1 ORA-00289: suggestion : +FLASH ORA-00280: change 5198050 for thread 1 is in sequence #817 Specify log: {<RET>=suggested | filename | AUTO | CANCEL} auto ORA-16145: archival for thread# 1 sequence# 817 in progress
	Welcome to DGMGRL, type "help" for information. DGMGRL> connect / Connected. DGMGRL> edit database 'ORCLSB1' set state='APPLY-ON'; Succeeded.

Standby Database Testing

PRIMARY	STANDBY
	DGMGRL> edit database 'ORCLSB1' set state='APPLY-OFF'; Succeeded.
	SQL> create restore point before_activation guarantee flashback database; Restore point created.
SQL> alter system archive log current; System altered.	
DGMGRL> edit database 'ORCL' set state='TRANSPORT-OFF'; Succeeded.	
	SQL> alter database activate standby database; Database altered.
	SQL> select controlfile_type from v$database; CURRENT

Standby Database Testing – Test It

PRIMARY	STANDBY
	SQL> alter database open; Database altered.
	SQL> create table parktr01.standby_testing (id NUMBER, val varchar2(10)); Table created.
	SQL> insert into parktr01.standby_testing values (1, 'One'); 1 row created. SQL> commit; Commit complete.

Standby Database Testing – Flash back to before activation

PRIMARY	STANDBY
	SQL> startup mount force; ORACLE instance started.
	SQL> flashback database to restore point before_activation; Flashback complete.
	SQL> select controlfile_type from v$database; BACKUP
	SQL> alter database convert to physical standby; Database altered.
	SQL> startup mount force; Database mounted.
DGMGRL> edit database 'ORCL' set state='TRANSPORT-ON'; Succeeded.	
	DGMGRL> edit database 'ORCLSB1' set state='APPLY-ON'; Succeeded.

Standby Database Testing – Verify

PRIMARY	STANDBY
	SQL> select controlfile_type from v$database; STANDBY
	SQL> alter database open read only; Database altered.
	SQL> desc parktr01.standby_testing; ERROR: ORA-04043: object parktr01.standby_testing does not exist
	DGMGRL> show database verbose 'ORCLSB1'; Database - ORCLSB1 Role: PHYSICAL STANDBY Intended State: APPLY-ON Transport Lag: 0 seconds Apply Lag: 0 seconds
	SQL> startup mount force; Database mounted.

Standby Database Testing – Final Cleanup

PRIMARY	STANDBY
	`SQL> drop restore point before_activation;` `Restore point dropped.`
	`DGMGRL> edit database 'ORCLSB1' set state='APPLY-ON';` `Succeeded.`

Flashback Requirements

FB Type	Requirement/Mechanism
Flashback Drop	Recyclebin = ON Free space in tablespace
Flashback Query/Versions Query	Automated Undo Management (UNDO not RBS) Sufficient UNDO_RETENTION Large Enough UNDO Tablespace FLASHBACK and SELECT on objects
Flashback Transaction Query	Supplemental Logging • Alter database add supplemental log data; • SELECT ANY TRANSACTION
Flashback Transaction	• Archivelog Mode • Alter database Add supplemental log data; • Alter database add supplemental log data (PRIMARY KEY) columns; • Additional requirements for LOBs, foreign key dependencies • SELECT ANY TRANSACTION • S,I,U,D on affected tables
Flashback Database	Alter database flashback on; Db_flashback_retention_target (minutes)

References

How To Open Physical Standby For Read Write Testing and Flashback [ID 805438.1]

How To Recover From A Drop/Truncate/Delete Table Done On Primary Using Flashback On A Standby Database [ID 958557.1]

Oracle Database Advanced Application Developer's Guide 11g Release 2 (11.2) – Chapter 12, "Using Oracle Flashback Technology"

Step by Step Guide on How To Reinstate Failed Primary Database into Physical Standby [ID 738642.1]