Flashback Technologies Step by Step

Alejandro Vargas
Principal Support Consultant
Oracle Advanced Support Services

Flashback Database

Flashback Database was introduced in 10g, it enables the possibility to revert the actual database to a previous point in time; and to do this faster than with other available methods.

A new background process was introduced, the RVWR. It is responsible for writing flashback logs which stores pre-images of data blocks.

Flashback Database is useful to restore the database in the event of logical data corruptions or human errors that invalidate large amounts of data.

Flashback Database cannot be used in the case of media failure, in this case a database backup must be used.

To be able to use Flashback Database, archive log mode must be enabled and the last shutdown must have been a clean shutdown.

Enabling Flashback Database

To enable flashback database the following requirements need to be met:

- Set flashback database on
- Set the following parameters, they are dynamic and do not require shutdown
    - DB_RECOVERY_FILE_DEST ➔ Flashback log location
    - DB_RECOVERY_FILE_DEST_SIZE ➔ Max Flashback available space
    - DB_FLASHBACK_RETENTION_TARGET ➔ Max time available to Flashback
- Clean shutdown – startup of the database is necessary in order to use Flashback Database.

```
SQL> shutdown immediate;
Database closed.
Database dismounted.
ORACLE instance shut down.

SQL> startup mount;
ORACLE instance started.

Total System Global Area  167772160 bytes
Fixed Size                  1218316 bytes
Variable Size              67111156 bytes
Database Buffers           96468992 bytes
Redo Buffers                2973696 bytes
Database mounted.

SQL> Alter database flashback on;

Database altered.

SQL> show parameters flash

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
db_flashback_retention_target        integer     1440
SQL> show parameters recover

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
db_recovery_file_dest                string      /vmasmtest/flashrecarea
db_recovery_file_dest_size           big integer 2G
recovery_parallelism                 integer     0

SQL> alter database open;
```

```
Database altered.

SQL> select flashback_on from v$database;

FLASHBACK_ON
------------------
YES
```

Check the Flashback database processes on the Operating System prompt

```
In this server there are two databases, both of the have Flashback enabled.
In this example I'm working with database redfox

{oracle} /home/oracle [vmractest1] > ps -efa | grep ora_ | grep rvwr
oracle   21984     1  0 Sep25 ?        00:05:42 ora_rvwr_whiteowl
oracle   12431     1  0 16:02 ?        00:00:01 ora_rvwr_redfox
```

Flashback Database Options

Flashback Database can be implemented based on one of these three options:

- SCN
- Timestamp
- Log Sequence Number

In all cases the database must be shutdown cleanly and then mounted.

Flashback Database Examples

In these examples we will check flashing back a database after running a small batch.

Check if automatical archival is enabled and if flashback logs are being generated

```
SQL> archive log list
Database log mode              Archive Mode
Automatic archival             Enabled
Archive destination            USE_DB_RECOVERY_FILE_DEST
Oldest online log sequence     1
Next log sequence to archive   2
Current log sequence           2

SQL> show parameters db_recovery_file_dest

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
db_recovery_file_dest                string      /vmasmtest/flashrecarea
db_recovery_file_dest_size           big integer 2G


SQL> !ls -l /vmasmtest/flashrecarea/REDFOX/flashback
total 8020
-rw-r-----    1 oracle   dba       8200192 Oct  8 16:56 o1_mf_3jng3yd4_.flb
```

Check SCN and Timestamp and Log Sequences before running the batch

SQL> select dbms_flashback.get_system_change_number from dual;

```
GET_SYSTEM_CHANGE_NUMBER
------------------------
          479899

SQL> select sysdate from dual;

SYSDATE
--------------
08/10/07 16:54

SQL> select SEQUENCE#,BYTES,ARCHIVED,STATUS, FIRST_TIME from  v$log
  2  /

 SEQUENCE#      BYTES ARC STATUS           FIRST_TIME
---------- ---------- --- ---------------- --------------
         2   52428800 NO  CURRENT          08/10/07 15:58
         0   52428800 YES UNUSED
         1   52428800 YES INACTIVE         08/10/07 15:54


Execute the batch

...

292428 rows created.

Commit complete.

584856 rows created.   <<<<<< Control records to flashback

Commit complete.
```

Check SCN and Timestamp and Log Sequences while running the batch

Initial SCN	***Next SCN***
479899	481960

Initial Date	***Next Date***
08/10/07 16:54	08/10/07 17:15

Initial Sequences

SEQUENCE#	BYTES	ARC	STATUS	FIRST_TIME
2	52428800	NO	CURRENT	08/10/07 15:58
0	52428800	YES	UNUSED	
1	52428800	YES	INACTIVE	08/10/07 15:54

Next Sequences

SEQUENCE#	BYTES	ARC	STATUS	FIRST_TIME
5	52428800	NO	CURRENT	08/10/07 17:14
3	52428800	YES	INACTIVE	08/10/07 17:13
4	52428800	YES	ACTIVE	08/10/07 17:14

Continue the batch

```
create table my2ndsource: Table created.

99 rows created.

Commit complete.
```

```
1169712 rows created.

Commit complete.
```

Check SCN and Timestamp and Log Sequences while running the batch

Initial SCN	***Next SCN***	***Next SCN***
479899	481960	482934

Initial Date	***Next Date***	***Next Date***
08/10/07 16:54	08/10/07 17:15	08/10/07 17:19

Initial Sequences

SEQUENCE#	BYTES	ARC	STATUS	FIRST_TIME
2	52428800	NO	CURRENT	08/10/07 15:58
0	52428800	YES	UNUSED	
1	52428800	YES	INACTIVE	08/10/07 15:54

Next Sequences

SEQUENCE#	BYTES	ARC	STATUS	FIRST_TIME
5	52428800	NO	CURRENT	08/10/07 17:14
3	52428800	YES	INACTIVE	08/10/07 17:13
4	52428800	YES	ACTIVE	08/10/07 17:14

Next Sequences

SEQUENCE#	BYTES	ARC	STATUS	FIRST_TIME
8	52428800	YES	ACTIVE	08/10/07 17:17

```
         9   52428800 NO  CURRENT          08/10/07 17:17
         7   52428800 YES ACTIVE           08/10/07 17:17
```

Shutdown and Startup Mount the Database

```
SQL> conn / as sysdba
Connected.

SQL> shutdown immediate;
Database closed.
Database dismounted.
ORACLE instance shut down.

SQL> startup mount;
ORACLE instance started.

Total System Global Area  167772160 bytes
Fixed Size                  1218316 bytes
Variable Size              67111156 bytes
Database Buffers           96468992 bytes
Redo Buffers                2973696 bytes
Database mounted.
```

Flashback Database Options

At this point we can use any of these commands to flashback the database to a previous point in time
In this example we will flashback prior to the create table my2ndsource, so we can use any of:

- SCN 481960
- TIMESTAMP 08/10/07 17:15
- SEQUENCE 5

The commands can be:

- flashback database to scn 481960;
- flashback database to TIMESTAMP to_date('08/10/07 17:15','dd/mm/yy hh24:mi');
- flashback database to sequence=5 thread=1;

After the flashback command open resetlogs must be used.

Flashback command

```
SQL> flashback database to TIMESTAMP to_date('08/10/07 17:15','dd/mm/yy hh24:mi');

Flashback complete.
```

On the alert log we see

```
Mon Oct  8 17:54:14 2007
flashback database to TIMESTAMP to_date('08/10/07 17:15','dd/mm/yy hh24:mi')
Mon Oct  8 17:54:15 2007
Flashback Restore Start
Flashback Restore Complete
Flashback Media Recovery Start
Flashback Media Recovery Log
/vmasmtest/flashrecarea/REDFOX/archivelog/2007_10_08/o1_mf_1_2_3jnl8wd7_.arc
Flashback Media Recovery Log
/vmasmtest/flashrecarea/REDFOX/archivelog/2007_10_08/o1_mf_1_3_3jnl9tgz_.arc
Mon Oct  8 17:54:34 2007
Flashback Media Recovery Log
/vmasmtest/flashrecarea/REDFOX/archivelog/2007_10_08/o1_mf_1_4_3jnlbkpz_.arc
Flashback Media Recovery Log
/vmasmtest/flashrecarea/REDFOX/archivelog/2007_10_08/o1_mf_1_5_3jnlj2mb_.arc
Mon Oct  8 17:54:47 2007
```

```
Incomplete Recovery applied until change 481950
Flashback Media Recovery Complete
Completed: flashback database to TIMESTAMP to_date('08/10/07 17:15','dd/mm/yy hh24:mi')
```

Open the database with resetlogs option

```
SQL> alter database open resetlogs;

Database altered.
```

Check the control table (584856 rows created. <<<<<< Control records to flashback**)**

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

  COUNT(*)
----------
    584856
```

Flashback Table

Flashback Table enables recovery of tables to a point in time in the past.

Enabling Flashback Table

A table can be flashbacked to an earlier SCN or timestamp, the pre-requisites to be able to perform this operation are:

1. ***Flashback any table***, or ***Flashback*** privilege on the table to be flashed back.
2. ***Select, Insert, Delete, and Alter*** object privileges on the table.
3. ***Row movement*** must be enabled for all tables in the Flashback list

Flashback Table Examples

Flashback to SCN

```
In this example a table is inserted a couple of times checking the system change number (SCN)
each time.
The table is flashbacked to it's original version, and then flashbacked again to a later
version. Flashback table depends on undo records and if you have the undo available you can go
up and down in time without problem.

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

```
Table altered.

SQL> select count(*) from users;

  COUNT(*)
----------
        25

SQL> select dbms_flashback.get_system_change_number from dual;

GET_SYSTEM_CHANGE_NUMBER
------------------------
                 2294900

SQL> insert into users select * from users;

25 rows created.

SQL> commit;

Commit complete.

SQL> select dbms_flashback.get_system_change_number from dual;

GET_SYSTEM_CHANGE_NUMBER
------------------------
                 2294911

SQL> insert into users select * from users;

50 rows created.

SQL> commit;
```

```
Commit complete.

SQL> select dbms_flashback.get_system_change_number from dual;

GET_SYSTEM_CHANGE_NUMBER
------------------------
                 2294917

SQL> select count(*) from users;

  COUNT(*)
----------
       100

SQL>  flashback table users to scn 2294900;

Flashback complete.

SQL> select count(*) from users;

  COUNT(*)
----------
        25

SQL> flashback table users to scn 2294917;

Flashback complete.

SQL>  select count(*) from users;

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

Flashback Query Example

Flashback Query is based on undo. In this example a set of records is updated and then restored from a previous version based on the timestamp.

Two different syntaxes are shown to get to a timestamp:

- As of TIMESTAMP (SYSTIMESTAMP - INTERVAL '5' MINUTE)
- As of TIMESTAMP to_date('08/10/07 11:58','dd/mm/yy hh24:mi')

```
SQL> select USERNAME,ACCOUNT_STATUS from users
  2  /

USERNAME                       ACCOUNT_STATUS
------------------------------ --------------------------------
SYS                            OPEN
SYSTEM                         OPEN
DBSNMP                         OPEN
SIEBEL                         OPEN
CX_NUMBER_MGMT                 OPEN
SIEBEL_RO                      OPEN
AVARGAS                        OPEN
OUTLN                          EXPIRED & LOCKED
MGMT_VIEW                      EXPIRED & LOCKED
MDSYS                          EXPIRED & LOCKED
ORDSYS                         EXPIRED & LOCKED
EXFSYS                         EXPIRED & LOCKED
```

```
DMSYS                          EXPIRED & LOCKED
WMSYS                          EXPIRED & LOCKED
CTXSYS                         EXPIRED & LOCKED
ANONYMOUS                      EXPIRED & LOCKED
SYSMAN                         EXPIRED & LOCKED
XDB                            EXPIRED & LOCKED
ORDPLUGINS                     EXPIRED & LOCKED
SI_INFORMTN_SCHEMA             EXPIRED & LOCKED
OLAPSYS                        EXPIRED & LOCKED
SCOTT                          EXPIRED & LOCKED
TSMSYS                         EXPIRED & LOCKED
MDDATA                         EXPIRED & LOCKED
DIP                            EXPIRED & LOCKED

25 rows selected.

SQL>  SELECT SYSDATE FROM DUAL;

SYSDATE
--------------
08/10/07 11:58

SQL> update users set ACCOUNT_STATUS='UNKNOWN' where ACCOUNT_STATUS like 'EXPIRED%';

18 rows updated.

SQL> commit;

Commit complete.

SQL> select USERNAME,ACCOUNT_STATUS from users where ACCOUNT_STATUS='UNKNOWN';

USERNAME                       ACCOUNT_STATUS
```

```
------------------------------ --------------------------------
OUTLN                          UNKNOWN
MGMT_VIEW                      UNKNOWN
MDSYS                          UNKNOWN
ORDSYS                         UNKNOWN
EXFSYS                         UNKNOWN
DMSYS                          UNKNOWN
WMSYS                          UNKNOWN
CTXSYS                         UNKNOWN
ANONYMOUS                      UNKNOWN
SYSMAN                         UNKNOWN
XDB                            UNKNOWN
ORDPLUGINS                     UNKNOWN
SI_INFORMTN_SCHEMA             UNKNOWN
OLAPSYS                        UNKNOWN
SCOTT                          UNKNOWN
TSMSYS                         UNKNOWN
MDDATA                         UNKNOWN
DIP                            UNKNOWN

18 rows selected.
```

Get back to the original values

```
SQL> SELECT USERNAME,ACCOUNT_STATUS from users
  2  AS OF TIMESTAMP (SYSTIMESTAMP - INTERVAL '5' MINUTE)
  3   where ACCOUNT_STATUS like 'EXPIRED %';

USERNAME                       ACCOUNT_STATUS
------------------------------ --------------------------------
OUTLN                          EXPIRED & LOCKED
MGMT_VIEW                      EXPIRED & LOCKED
MDSYS                          EXPIRED & LOCKED
```

```
ORDSYS                         EXPIRED & LOCKED
EXFSYS                         EXPIRED & LOCKED
DMSYS                          EXPIRED & LOCKED
WMSYS                          EXPIRED & LOCKED
CTXSYS                         EXPIRED & LOCKED
ANONYMOUS                      EXPIRED & LOCKED
SYSMAN                         EXPIRED & LOCKED
XDB                            EXPIRED & LOCKED
ORDPLUGINS                     EXPIRED & LOCKED
SI_INFORMTN_SCHEMA             EXPIRED & LOCKED
OLAPSYS                        EXPIRED & LOCKED
SCOTT                          EXPIRED & LOCKED
TSMSYS                         EXPIRED & LOCKED
MDDATA                         EXPIRED & LOCKED
DIP                            EXPIRED & LOCKED

18 rows selected.
```

Compare old and new values

```
SQL> l
  1  select a.username,a.ACCOUNT_STATUS, b.ACCOUNT_STATUS ACCOUNT_STATUS_OLD
  2  from users a,
  3  users  AS OF TIMESTAMP to_date('08/10/07 11:58','dd/mm/yy hh24:mi') b
  4*  where a.username=b.username
SQL> /

USERNAME             ACCOUNT_STATUS       ACCOUNT_STATUS_OLD
-------------------- -------------------- --------------------------------
SYS                  OPEN                 OPEN
SYSTEM               OPEN                 OPEN
DBSNMP               OPEN                 OPEN
SIEBEL               OPEN                 OPEN
```

```
CX_NUMBER_MGMT       OPEN                 OPEN
SIEBEL_RO            OPEN                 OPEN
AVARGAS              OPEN                 OPEN
OUTLN                UNKNOWN              EXPIRED & LOCKED
MGMT_VIEW            UNKNOWN              EXPIRED & LOCKED
MDSYS                UNKNOWN              EXPIRED & LOCKED
ORDSYS               UNKNOWN              EXPIRED & LOCKED
EXFSYS               UNKNOWN              EXPIRED & LOCKED
DMSYS                UNKNOWN              EXPIRED & LOCKED
WMSYS                UNKNOWN              EXPIRED & LOCKED
CTXSYS               UNKNOWN              EXPIRED & LOCKED
ANONYMOUS            UNKNOWN              EXPIRED & LOCKED
SYSMAN               UNKNOWN              EXPIRED & LOCKED
XDB                  UNKNOWN              EXPIRED & LOCKED
ORDPLUGINS           UNKNOWN              EXPIRED & LOCKED
SI_INFORMTN_SCHEMA   UNKNOWN              EXPIRED & LOCKED
OLAPSYS              UNKNOWN              EXPIRED & LOCKED
SCOTT                UNKNOWN              EXPIRED & LOCKED
TSMSYS               UNKNOWN              EXPIRED & LOCKED
MDDATA               UNKNOWN              EXPIRED & LOCKED
DIP                  UNKNOWN              EXPIRED & LOCKED

25 rows selected.
```

Create a view on the old version

```
create view tmpv
as select * from users
AS OF TIMESTAMP to_date('08/10/07 11:58','dd/mm/yy hh24:mi')
/
```

Update the table to restore old values

```
UPDATE users
  SET account_status = ( SELECT account_status
                         FROM tmpv
                         WHERE users.username = tmpv.username)
  WHERE EXISTS
                        ( SELECT account_status
                         FROM tmpv
                         WHERE users.username = tmpv.username)
/


SQL> l
  1* select username, account_status from users
SQL> /

USERNAME                       ACCOUNT_STATUS
------------------------------ --------------------------------
SYS                            OPEN
SYSTEM                         OPEN
DBSNMP                         OPEN
SIEBEL                         OPEN
CX_NUMBER_MGMT                 OPEN
SIEBEL_RO                      OPEN
AVARGAS                        OPEN
OUTLN                          EXPIRED & LOCKED
MGMT_VIEW                      EXPIRED & LOCKED
MDSYS                          EXPIRED & LOCKED
ORDSYS                         EXPIRED & LOCKED
EXFSYS                         EXPIRED & LOCKED
DMSYS                          EXPIRED & LOCKED
WMSYS                          EXPIRED & LOCKED
CTXSYS                         EXPIRED & LOCKED
ANONYMOUS                      EXPIRED & LOCKED
```

```
SYSMAN                         EXPIRED & LOCKED
XDB                            EXPIRED & LOCKED
ORDPLUGINS                     EXPIRED & LOCKED
SI_INFORMTN_SCHEMA             EXPIRED & LOCKED
OLAPSYS                        EXPIRED & LOCKED
SCOTT                          EXPIRED & LOCKED
TSMSYS                         EXPIRED & LOCKED
MDDATA                         EXPIRED & LOCKED
DIP                            EXPIRED & LOCKED

25 rows selected.
```

Flashback Transaction Query Example

Oracle Flashback Transaction Query permits to view changes made to a table by a single transaction, or by all the transactions during a period of time. You can query the same row at several point in time to review their different versions.

It makes possible to perform recovery at the Row Level.

For example it may help to restore a row of data that was deleted by error.

In this example a row will be deleted and restored using the previous image still available from the undo records.

Get the SCN and actual time as references for the exercise

```
SQL> select dbms_flashback.get_system_change_number from dual;

GET_SYSTEM_CHANGE_NUMBER
```

```
-------------------------
                  2302638

SQL> select sysdate from dual;

SYSDATE
--------------
08/10/07 14:54

SQL> select username,account_status,created from users
  2  where username='AVARGAS';

USERNAME                       ACCOUNT_STATUS                   CREATED
------------------------------ -------------------------------- --------------
AVARGAS                        OPEN                             02/04/07 09:09

SQL> delete from users where username = 'AVARGAS';

1 row deleted.

SQL> COMMIT;

Commit complete.
```

Review the flashback records for the deleted row, two alternative syntaxes are shown.

```
select username,account_status,created
from users
where username='AVARGAS';

no rows selected

select username,account_status,created
```

```
from users
as of SCN 2302638
where username='AVARGAS';

USERNAME                       ACCOUNT_STATUS                    CREATED
------------------------------ --------------------------------- --------------
AVARGAS                        OPEN                              02/04/07 09:09

select username,account_status,created
from users
as of TIMESTAMP to_date('08/10/07 14:54','dd/mm/yy hh24:mi')
where username='AVARGAS';

USERNAME                       ACCOUNT_STATUS                    CREATED
------------------------------ --------------------------------- --------------
AVARGAS                        OPEN                              02/04/07 09:09
```

Restore the deleted row.

```
insert into users
select * from users
as of TIMESTAMP to_date('08/10/07 14:54','dd/mm/yy hh24:mi')
where username='AVARGAS'
/

1 row created.

commit;

Commit complete.

select username,account_status,created
from users
```

```
where username='AVARGAS'
/

USERNAME                       ACCOUNT_STATUS                   CREATED
------------------------------ -------------------------------- --------------
AVARGAS                        OPEN                             02/04/07 09:09
```

Flashback Drop

Flashback Drop is different from Flashback table. It uses the recycle bin instead of undo records. It restores the table to its state before the drop. Row movement must be enabled in order to be able to restore a table from the recycle bin.

Enabling the Recycle Bin

```
ALTER SYSTEM SET recyclebin = ON;
```

The Recycle bin can also be enabled at the session level.

```
ALTER SESSION SET recyclebin = ON;
```

Flashback to Before Drop

A dropped table will remain on the recycle bin until purged by the user or overwritten by other recycle bin entries, dropped tables recovered from the recycle bin do not preserve referential integrity constraints, if they existed they will need to be recreated manually.

On this exercise we drop the table, then we restore it by flashing back to before drop, and after that we flash back to certain SCN, using the available undo records.

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

Table altered.
```

```
SQL> drop table users;

Table dropped.

SQL> select count(*) from users;
select count(*) from users
                     *
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> flashback table users to before drop;

Flashback complete.

SQL> select count(*) from users;

  COUNT(*)
----------
       100

SQL> flashback table users to scn 2294900;

Flashback complete.

SQL> select count(*) from users;

  COUNT(*)
----------
        25
```

Flashback to Before Drop with rename

On this exercise we drop the table, then we restore it by flashing back to before drop, and we rename it, after that we flash back to certain SCN, note that despite the new name the records belonging to the table users are applied.

```
SQL> drop table users;

Table dropped.

SQL> select object_name,original_name from user_recyclebin;

OBJECT_NAME                    ORIGINAL_NAME
------------------------------ --------------------------------
BIN$O/estCISH+7gQAQKTx5nsA==$0 USERS

SQL> flashback table users to before drop rename to oldusers;

Flashback complete.

SQL> select object_name,original_name from user_recyclebin;

no rows selected

SQL> select count(*) from oldusers;

  COUNT(*)
----------
        25

SQL> flashback table oldusers to scn 2294917;
```

```
Flashback complete.

SQL> select count(*) from oldusers;

  COUNT(*)
----------
       100
```

References

Metalink Notes:

Note 238674.1 - How to restore the old data using flashback queries?
Note 249319.1 - Configure flashback database
Note 270060.1 - Use Flashback Table Feature and Resolve errors
Note 270535.1 - Restrictions on Flashback Table Feature
Note 317499.1 - 10G Oracle Flashback Transaction Query - Introduction and usage
Note 369755.1 - Flashback Logs-Space management