

Re-instantiating Schema Replication with Oracle Streams Downstream Capture

Have you ever had Streams fail to replicate data to a target database (for whatever reason) during heavy load on the source database and Streams has fallen behind by several hours? You are not concerned about the data on the target database as this is a test system; you just want Streams to “catch-up” as quickly as possible to a point where testing can continue.

The following steps illustrate how to fast-track the “catch-up process”, to get Streams back to “WAITING FOR REDO” state.

Note. This procedure will not apply any changes made on the source database to the target database from time of failure to when you re-instantiate the schema.

1. Logon to target database as Streams Admin user and stop the downstream capture and apply processes.

```
sqlplus streams_admin/streams_admin

SQL> exec dbms_apply_adm.stop_apply(apply_name=>'SRC_SCHEMA_APPLY')

SQL> exec
dbms_capture_adm.stop_capture(capture_name=>'SRC_SCHEMA_CAPTURE')
```

2. Instantiate source schema so we know from what point to start replication

```
SQL> set serverout on
SQL>
DECLARE
  iscn NUMBER;
BEGIN
  iscn := DBMS_FLASHBACK.GET_SYSTEM_CHANGE_NUMBER@&src_db_name();
  DBMS_APPLY_ADM.SET_SCHEMA_INSTANTIATION_SCN(
    source_schema_name => 'SRC',
    source_database_name => '&src_db_name',
    instantiation_scn => iscn,
    recursive => TRUE);

  dbms_output.put_line('Instantiation SCN = '||iscn);

END;
/

Instantiation SCN = 1256797484

PL/SQL procedure successfully completed.
```

3. Make a note of the Instantiation System Change Number and alter the downstream capture process on the target database to change the start SCN to

this number.

```
SQL> exec
dbms_capture_adm.alter_capture(capture_name=>'SRC_SCHEMA_CAPTURE', start_scn=>1256797484)
```

4. The following messages can be seen in the alert log of the database instance where Streams is running:

```
Thu Nov 26 12:01:12 2009
knlciAlterCapture: start scn changed.
scn: 0x0000.4ae9352c
Thu Nov 26 12:01:46 2009
knlciAlterCapture: start scn changed.
scn: 0x0000.4ae9352c
```

(Convert the SCN Hex number 4ae9352c to Decimal to get 1256797484)

5. Start the apply and capture processes on the target database.

```
SQL> exec dbms_apply_adm.start_apply(apply_name=>'SRC_SCHEMA_APPLY')
SQL> exec
dbms_capture_adm.start_capture(capture_name=>'SRC_SCHEMA_CAPTURE')
```

6. The following messages can be seen in the alert log of the database instance where Streams is running:

```
Thu Nov 26 12:01:12 2009
knlciAlterCapture: start scn changed.
scn: 0x0000.4ae9352c
Thu Nov 26 12:01:46 2009
knlciAlterCapture: start scn changed.
scn: 0x0000.4ae9352c
Thu Nov 26 12:02:30 2009
Streams APPLY AP01 for SRC_SCHEMA_APPLY started with pid=73, OS
id=20087
Thu Nov 26 12:02:30 2009
Streams CAPTURE CP01 for SRC_SCHEMA_CAPTURE started with pid=74, OS
id=20089

CP02: Warning: capture (SRC_SCHEMA_CAPTURE) start_scn is higher than
last_acked scn. LCRs not seen at apply (SRC_SCHEMA_APPLY) might be
filtered out!
Streams CAPTURE CP02 for SRC_SCHEMA_CAPTURE with pid=132, OS id=26534
is in combined capture and apply mode.
Streams CAPTURE CP01 for SRC_SCHEMA_CAPTURE with pid=97, OS id=26438
is in combined capture and apply mode.
Streams downstream capture SRC_SCHEMA_CAPTURE uses
downstream_real_time_mine: TRUE
Starting persistent Logminer Session with sid = 8 for Streams Capture
SRC_SCHEMA_CAPTURE
LOGMINER: Parameters summary for session# = 8
LOGMINER: Number of processes = 4, Transaction Chunk Size = 1
LOGMINER: Memory Size = 100M, Checkpoint interval = 1000M
LOGMINER: SpillScn 0, ResetLogScn 1
LOGMINER: krxpsr summary for session# = 8
LOGMINER: StartScn: 1337626267 (0x0000.4fba8e9b)
```

```

LOGMINER: EndScn: 0
LOGMINER: HighConsumedScn: 1394932052 (0x0000.5324f954)
LOGMINER: session_flag 0x1
LOGMINER: LowCkptScn: 1337320672 (0x0000.4fb5e4e0)
LOGMINER: HighCkptScn: 1337545817 (0x0000.4fb95459)
LOGMINER: SkipScn: 1337320672 (0x0000.4fb5e4e0)
Thu Nov 26 12:02:39 2009
LOGMINER: session#=8, reader MS00 pid=143 OS id=20298 sid=974 started
Thu Nov 26 12:02:39 2009
LOGMINER: session#=8, builder MS01 pid=144 OS id=20300 sid=979
started
Thu Nov 26 12:02:39 2009
LOGMINER: session#=8, preparer MS02 pid=145 OS id=20302 sid=969
started
..
LOGMINER: Begin mining logfile for session 8 thread 1 sequence 51427,
+FLASH/tgt_db/archivelog/2009_11_25/thread_1_seq_51427.5942.703879123
LOGMINER: Begin mining logfile for session 8 thread 2 sequence 53349,
+FLASH/tgt_db/archivelog/2009_11_25/thread_2_seq_53349.5900.703879313
LOGMINER: Begin mining logfile for session 13 thread 1 sequence
51417,
+FLASH/tgt_db/archivelog/2009_11_25/thread_1_seq_51417.4816.703877451
LOGMINER: Begin mining logfile for session 13 thread 2 sequence
53346,
+FLASH/tgt_db/archivelog/2009_11_25/thread_2_seq_53346.4910.703877989
LOGMINER: End mining logfile for session 13 thread 1 sequence 51417,
+FLASH/tgt_db/archivelog/2009_11_25/thread_1_seq_51417.4816.703877451

```

Conclusion

Streams combined capture and apply processes will start and LogMiner will mine each of the Foreign Archived Log files sent from the source database, searching for the new Instantiation SCN. This process provides a considerably faster “catch-up” method as only the capture process is executing, the apply process remains idle as seen in the query output below.

CAPTURE PROCESS STATUS

```
SQL> select CAPTURE_NAME, STATE from v$streams_capture;
```

CAPTURE_NAME	STATE
-----	-----
SRC_SCHEMA_CAPTURE	CAPTURING CHANGES

or

CAPTURE_NAME	STATE
-----	-----
SRC_SCHEMA_CAPTURE	CREATING LCR

APPLY PROCESS STATUS

```
SQL> select APPLY_NAME, STATE from v$streams_apply_reader;
```

APPLY_NAME	STATE
-----	-----

```
SRC_SCHEMA_APPLY          DEQUEUE MESSAGES
```

```
SQL> select APPLY_NAME, STATE from v$streams_apply_server;
```

APPLY_NAME	STATE
-----	-----
SRC_SCHEMA_APPLY	IDLE
SRC_SCHEMA_APPLY	IDLE
SRC_SCHEMA_APPLY	IDLE
SRC_SCHEMA_APPLY	IDLE

```
SQL> select APPLY_NAME, STATE from v$streams_apply_coordinator;
```

APPLY_NAME	STATE
-----	-----
SRC_SCHEMA_APPLY	IDLE

Follow-up

- All steps are conducted on the target database which is running archive log downstream capture.
- A database link needs to exist from the target to the source database in order for re-instantiation commands to succeed.
- Please note that following schema re-instantiation, you will have to synchronise your target database with the source.
-

To re-instantiate the schema

1. Identify the current SCN of the source database (iscn)
2. Run the following procedure on target database

```
DECLARE
    iscn NUMBER; -- Variable to hold instantiation SCN value
BEGIN
    iscn := DBMS_FLASHBACK.GET_SYSTEM_CHANGE_NUMBER();
    DBMS_APPLY_ADM.SET_SCHEMA_INSTANTIATION_SCN(
        SOURCE_SCHEMA_NAME => ' ', source_database_name =>
        ', RECURSIVE=>TRUE,
        instantiation_scn => iscn );
END;
/
```

To re-synchronise the data

1. Stop the capture process on target database
2. Stop the apply process on target database
3. Export the source schema (using exp with object_consistent=y)
4. Delete all apply errors on target database
5. Import it in target database (using imp with streams_instantiation=y)
6. Start the apply process
7. start the capture process