Flashback 1. flashbackup query

과거 특정 시점의 COMMIT 된 데이터를 검색할 수 있다. SELECT 문의 AS OF 절을 사용하여 과거 시점을 지정한다.

테이블 생성 후 scn_to_timestamp 조회

```
SYS@PROD1>create table hr.emp_30
  2  as
  3  select *
  4  from hr.employees
  5  where department_id = 30;

Table created.

SYS@PROD1>select employee_id, salary
  2  from hr.emp_30;

EMPLOYEE_ID  SALARY
----------- ----------
      114     11000
      115      3100
      116      2900
      117      2800
      118      2600
      119      2500

6 rows selected.

SYS@PROD1> select current_scn, scn_to_timestamp(current_scn)
  2  from v$database;

CURRENT_SCN SCN_TO_TIMESTAMP(CURRENT_SCN)
--------------------------------------------
2558629 06-JUL-20 02.05.53.000000000 PM
```

테이블 업데이트 후 커밋하고 다시 한번 scn_to_timestamp를 조회한다.

```
SYS@PROD1>update hr.emp_30
  2  set salary = 3000
  3  where employee_id = 114;
1 row updated.

SYS@PROD1>commit;
Commit complete.

SYS@PROD1>select current_scn, scn_to_timestamp(current_scn)
  2  from v$database;

CURRENT_SCN SCN_TO_TIMESTAMP(CURRENT_SCN)
-------------------------------------------
2558818 06-JUL-20 02.06.38.000000000 PM

1 row selected.
```

업데이트 전으로 돌아가기 위한 flashback query

* 현재상태 조회

```
SYS@PROD1>select employee_id, salary
  2  from hr.emp_30
  3  where employee_id = 114;

EMPLOYEE_ID  SALARY
----------- ----------
      114      3000

1 row selected.
```

* timestamp로 조회

```
SYS@PROD1>select employee_id, salary
  2  from hr.emp_30
  3  as of timestamp to_timestamp('2020/07/06 14:05:53', 'yyyy/mm/dd hh24:mi:ss')
  4  where employee_id = 114;

EMPLOYEE_ID  SALARY
----------- ----------
      114     11000

1 row selected.
```

* interval 로 조회

```
SYS@PROD1>select employee_id, salary
  2  from hr.emp_30
  3  as of timestamp(systimestamp - interval '5' minute)
  4  where employee_id = 144;

EMPLOYEE_ID  SALARY
----------- ----------
      114     11000

1 row selected.
```

* 테이블 만들기 이전이기 때문에 오류 발생

```
SYS@PROD1>select employee_id, salary
  2  from hr.emp_30
  3  as of timestamp(systimestamp - interval '10' minute)
  4  where employee_id = 144;
from hr.emp_30
        *
ERROR at line 2:
ORA-01466: unable to read data - table definition has changed
```

Flashback 2. flashback versions query

과거 두 개의 지점의 Point-in-time 또는 두 개의 지점의 SCN 사이에 존재하는 행의 모든 버전 확인

VERSIONS PSEUDO-COLUMNS :

- VERSIONS_STARTTIME (start timestamp of version)
- VERSIONS_STARTSCN (start SCN of version)
- VERSIONS_ENDTIME (end timestamp of version)
- VERSIONS_ENDSCN (end SCN of version)
- VERSIONS_XID (transaction ID of version)
- VERSIONS_OPERATION (DML operation of version)

현재 SCN과 테이블을 생성한 후 SCN을 조회한다.

```
SYS@PROD1>select current_scn, scn_to_timestamp(current_scn)
  2  , to_char(systimestamp, 'yyyy/mm/dd hh24:mi:ss.sssss')
  3  from v$database;

CURRENT_SCN SCN_TO_TIMESTAMP(CURRENT_SCN)         TO_CHAR(SYSTIMESTAMP,'YYY
----------- ----------------------------------  --------------------------
    2560353 06-JUL-20 02.41.20.000000000 PM2020/07/06 14:41:23.52883

1 row selected.

SYS@PROD1>create table scott.emp_20
  2  as
  3  select employee_id, last_name, salary
  4  from hr.employees
  5  where department_id = 20;

Table created.

SYS@PROD1>select *
  2  from scott.emp_20;

EMPLOYEE_ID LAST_NAME                  SALARY
----------- ------------------------- ----------
      201 Hartstein                    13000
      202 Fay                     6000

2 rows selected.

SYS@PROD1>select current_scn, scn_to_timestamp(current_scn)
  2  , to_char(systimestamp, 'yyyy/mm/dd hh24:mi:ss.sssss')
  3  from v$database;

CURRENT_SCN SCN_TO_TIMESTAMP(CURRENT_SCN)         TO_CHAR(SYSTIMESTAMP,'YYY
----------- ----------------------------------  --------------------------
    2560669 06-JUL-20 02.51.54.000000000 PM2020/07/06 14:51:54.53514

1 row selected.
```

테이블을 업데이트 및 삭제 후 커밋하고 SCN을 조회한다.

```
SYS@PROD1>update scott.emp_20
  2   set salary = salary * 1.3
  3   where employee_id = 201;

1 row updated.

SYS@PROD1>delete scott.emp_20
  2   where employee_id = 202;

1 row deleted.

SYS@PROD1>commit;
Commit complete.

SYS@PROD1>select current_scn, scn_to_timestamp(current_scn)
  2   , to_char(systimestamp, 'yyyy/mm/dd hh24:mi:ss.sssss')
  3   from v$database;

CURRENT_SCN SCN_TO_TIMESTAMP(CURRENT_SCN)        TO_CHAR(SYSTIMESTAMP,'YYY
----------- ----------------------------------  --------------------------
    2560699 06-JUL-20 02.53.03.000000000 PM2020/07/06 14:53:05.53585

1 row selected.

SYS@PROD1>select *
  2   from scott.emp_20;

EMPLOYEE_ID LAST_NAME                          SALARY
----------- ------------------------- ----------
      201 Hartstein                         16900

1 row selected.
```

업데이트 전으로 돌아가기 위한 flashback versions query 방법

```
SYS@PROD1>select versions_xid, employee_id, last_name, salary
  2   from scott.emp_20
  3   versions between scn minvalue and maxvalue;

VERSIONS_XID  EMPLOYEE_ID LAST_NAME       SALARY
---------------- ----------- ---------------  ----------
05000D00A3060000    202 Fay                    6000
05000D00A3060000    201 Hartstein              16900
                    201 Hartstein              13000
                    202 Fay                    6000
4 rows selected.
```

```
SYS@PROD1>select versions_xid, employee_id, last_name, salary
  2  from sscott.emp_20
  3  versions between scn 2560669 and 2560699

VERSIONS_XID  EMPLOYEE_ID LAST_NAME        SALARY
---------------- ----------- ---------------  ----------
05000D00A3060000    202 Fay                   6000
05000D00A3060000    201 Hartstein             16900
                    201 Hartstein             13000
                    202 Fay                   6000
4 rows selected.


SYS@PROD1>select versions_xid, employee_id, last_name, salary
  2  from scott.emp_20
  3  versions between timestamp systimestamp - interval '5' minute and
systimestamp;

VERSIONS_XID  EMPLOYEE_ID LAST_NAME        SALARY
---------------- ----------- ---------------  ----------
05000D00A3060000    202 Fay                   6000
05000D00A3060000    201 Hartstein             16900
                    201 Hartstein             13000
                    202 Fay                   6000
4 rows selected.
```

Flashback 3. flashback versions query

Flashbackup Transction Query 수행되었던 Transaction의 Operation 검색

잠깐!! Supplemental logging 설정을 해보자

* disable : redo log에 변경된 칼럼 정보만 기록
* enable : 하나의 칼럼이 변경되더라도 전체 row의 정보를 모두 redo log에 저장

ORACLE 9i R2 버전부터 supplemental logging 기능의 기본값이 disable 로 바뀌었다.
enable의 경우, redo log의 양이 커지기 때문에 성능저하 우려가 있으나 실제로는 크게 차이가 나지 않는다.
주의할 점은 활성화 시키고 난 후부터 생성된 redo log만 분석이 되고 활성화 이전의 redo log는 분석에 제한이 있다.

1. 조회

```
SQL> select supplemental_log_data_min from v$database;

SUPPLEMENTAL_LOG
----------------
NO
```

2. 활성화(Enable)

```
SQL> alter database add supplemental log data;
Database altered.
```

3. 비활성화(Disable)

```
SQL> alter database drop supplemental log data;
Database altered.
```

테이블 emp_60, emp_90 생성 후 SCN 조회

```
SYS@PROD1>alter database add supplemental log data;
Database altered.

SYS@PROD1>conn hr/hr
Connected.

HR@PROD1>create table emp_60
  2  as
  3  select employee_id, last_name, salary
  4  from employees
  5  where department_id = 60;

Table created.

HR@PROD1>create table emp_90
  2  as
  3  select employee_id, last_name, salary
  4  from employees
  5  where department_id = 90;

Table created.

HR@PROD1>select *
  2  from emp_60;

EMPLOYEE_ID LAST_NAME                     SALARY
----------- ------------------------- ----------
        103 Hunold                       9000
        104 Ernst                        6000
        105 Austin                       4800
        106 Pataballa                    4800
        107 Lorentz                      4200

5 rows selected.

HR@PROD1>select *
  2  from emp_90;

EMPLOYEE_ID LAST_NAME                     SALARY
----------- ------------------------- ----------
        100 King                        24000
        101 Kochhar                     17000
        102 De Haan                     17000

3 rows selected.

HR@PROD1>select current_scn, scn_to_timestamp(current_scn)
  2  , to_char(systimestamp, 'yyyy/mm/dd hh24:mi:ss.sssss')
  3  from v$database;

CURRENT_SCN SCN_TO_TIMESTAMP(CURRENT_SCN)        TO_CHAR(SYSTIMESTAMP,'YYY
----------- ---------------------------------   ---------------------------
    2561961 06-JUL-20 03.10.15.000000000 PM2020/07/06 15:10:17.54617

1 row selected.
```

* 1차 업데이트 및 삭제 후 커밋

```
HR@PROD1>update emp_60
  2  set salary = salary * 1.1
  3  where employee_id = 107;

HR@PROD1>delete emp_60
  2  where employee_id = 103;

HR@PROD1>select *
  2  from emp_60;

EMPLOYEE_ID LAST_NAME                 SALARY
----------- -------------------- ----------
        104 Ernst                      6000
        105 Austin                     4800
        106 Pataballa                  4800
        107 Lorentz                    4620

4 rows selected.

HR@PROD1>update emp_90
  2  set salary = 24
  3  where employee_id = 100;

1 row updated.

HR@PROD1>select *
  2  from emp_90;

EMPLOYEE_ID LAST_NAME                         SALARY
----------- ------------------------- ----------
        100 King                             24
        101 Kochhar                       17000
        102 De Haan                       17000

3 rows selected.

HR@PROD1>commit;
Commit complete.
```

* 2차 업데이트 후 커밋

```
HR@PROD1>update emp_60
  2  set last_name = 'ENKIM'
  3  where employee_id = 106;

HR@PROD1>commit;
Commit complete.
```

* SCN 조회

```
HR@PROD1>select current_scn, scn_to_timestamp(current_scn)
  2  , to_char(systimestamp, 'yyyy/mm/dd hh24:mi:ss.sssss')
  3  from v$database;

CURRENT_SCN SCN_TO_TIMESTAMP(CURRENT_SCN)               TO_CHAR(SYSTIMESTAMP,'YYY
----------- ------------------------------------------ -------------------------
    2562011 06-JUL-20 03.12.23.000000000 PM        2020/07/06 15:12:24.54744

1 row selected.
```

versions_xid 로 변경사항들을 조회해본다.

```
HR@PROD1>select versions_xid, employee_id, last_name, salary
  2  from emp_60
  3  versions between timestamp to_timestamp('2020/07/06 15:10:17.54617',
'yyyy/mm/dd hh24:mi:ss.sssss') and to_timestamp('2020/07/06 15:12:24.54744',
'yyyy/mm/dd hh24:mi:ss.sssss');

VERSIONS_XID  EMPLOYEE_ID LAST_NAME        SALARY
---------------- ----------- ----------------- ----------
07000D00B1050000    106 ENKIM                    4800
090011006D060000    103 Hunold                   9000
090011006D060000    107 Lorentz            4620
                    103 Hunold                   9000
                    104 Ernst                    6000
                    105 Austin                   4800
                    106 Pataballa                4800
                    107 Lorentz            4200

8 rows selected.


HR@PROD1>select table_name, operation
  2  from flashback_transaction_query
  3  where xid = '090011006D060000';

TABLE_NAME                 OPERATION
------------------------------ ------------------------------
EMP_90                     UPDATE
EMP_60                     DELETE
EMP_60                     UPDATE
                                 BEGIN

4 rows selected.


HR@PROD1>select table_name, operation, undo_sql
  2  from flashback_transaction_query
  3  where xid = '090011006D060000';

TABLE_NAME        OPERATION             UNDO_SQL
-------------------- -------------------- ---------------------------------------------------
EMP_90               UPDATE               update "HR"."EMP_90" set "SALARY" = '24000' where
                                          ROWID = 'AAAW1UAAGAAAAErAAA';

EMP_60               DELETE               insert into "HR"."EMP_60"("EMPLOYEE_ID","LAST_NAME
                                          ","SALARY") values ('103','Hunold','9000');

EMP_60               UPDATE               update "HR"."EMP_60" set "SALARY" = '4200' where R
                                          OWID = 'AAAW1TAAGAAAAEbAAE';

                     BEGIN

4 rows selected.
```

잠깐!! flashback_transaction_query 조회 시 권한문제가 발생할 때에는?

```
ORA-01031: insufficient privileges
SYS@PROD1>grant select any transaction to hr;
```

Flashback 4. flashbackup table

테이블 생성 후 SCN 조회

```
SYS@PROD1>create table scott.emp_flash
  2  as
  3  select employee_id, last_name, salary
  4  from hr.employees
  5  where department_id = 30;

Table created.

SYS@PROD1>select current_scn, scn_to_timestamp(current_scn)
  2  from v$database;

CURRENT_SCN  SCN_TO_TIMESTAMP(CURRENT_SCN)
-----------  ---------------------------------------------------------------
    2562974  06-JUL-20 03.41.18.000000000 PM

SYS@PROD1>select * from scott.emp_flash;

EMPLOYEE_ID LAST_NAME                     SALARY
----------- ------------------------- ----------
      114 Raphaely                11000
      115 Khoo                          3100
      116 Baida                         2900
      117 Tobias                        2800
      118 Himuro                        2600
      119 Colmenares                    2500

6 rows selected.
```

업데이트 후 SCN 조회

```
SYS@PROD1>update scott.emp_flash
  2  set salary = 1000;

SYS@PROD1>commit;
Commit complete.

SYS@PROD1>select current_scn, scn_to_timestamp(current_scn)
  2  from v$database;

CURRENT_SCN SCN_TO_TIMESTAMP(CURRENT_SCN)
---------------------------------------------------------------------------
    2562996 06-JUL-20 03.42.12.000000000 PM

SYS@PROD1>select * from scott.emp_flash;

EMPLOYEE_ID LAST_NAME                     SALARY
----------- ------------------------- ----------
      114 Raphaely          1000
      115 Khoo                    1000
      116 Baida                   1000
      117 Tobias                  1000
      118 Himuro                  1000
      119 Colmenares              1000
```

```
6 rows selected.
```

테이블 레벨로 flashback 복구

* 로우레벨 플래쉬백 복구방법과 다르게, 테이블 레벨 플래쉬백 복구방법은 테이블에 있는 전체 로우에 대해서 어느 한 시점으로 복구하는 것이기 때문에 해당 테이블의 모든 로우의 데이터가 영향을 받게 된다.

```
SYS@PROD1>alter table scott.emp_flash enable row movement;
Table altered.

SYS@PROD1>flashback table scott.emp_flash
  2  to timestamp to_timestamp('2020/07/06 15:41:18', 'yyyy/mm/dd hh24:mi:ss');

Flashback complete.

SYS@PROD1>select *
  2  from scott.emp_flash;

EMPLOYEE_ID LAST_NAME                      SALARY
----------- ------------------------- ----------
        114 Raphaely                  11000
        115 Khoo                            3100
        116 Baida                           2900
        117 Tobias                          2800
        118 Himuro                          2600
        119 Colmenares                      2500

6 rows selected.
```

* row movement 상태 조회

```
SYS@PROD1>select row_movement
  2  from dba_tables
  3  where table_name = 'EMP_FLASH';

ROW_MOVE
--------
ENABLED

1 row selected.
```

* 다시 row movement를 disable 로 변경

```
SYS@PROD1>alter table scott.emp_flash disable row movement;
Table altered.

SYS@PROD1>select row_movement
  2  from dba_tables
  3  where table_name = 'EMP_FLASH';

ROW_MOVE
--------
DISABLED

1 row selected.


SYS@PROD1>drop table scott.emp_flash purge;
Table dropped.
```

Flashback 5. flashbackup drop and recycle bin

temp_emp 생성

```
SYS@PROD1>conn hr/hr
Connected.

HR@PROD1>create table temp_emp
  2  as
  3  select *
  4  from employees;
```

* 제약조건 확인

```
HR@PROD1>select constraint_name, constraint_type, search_condition
  2  from user_constraints
  3  where table_name = 'TEMP_EMP';

CONSTRAINT_NAME           C     SEARCH_CONDITION
------------------------ ---- ---------------------------
SYS_C0010401              C   "LAST_NAME" IS NOT NULL
SYS_C0010402              C   "EMAIL" IS NOT NULL
SYS_C0010403              C   "HIRE_DATE" IS NOT NULL
SYS_C0010404              C   "JOB_ID" IS NOT NULL
```

* 제약조건 추가

```
HR@PROD1>alter table temp_emp
  2  add constraint temp_emp_id_pk
  3  primary key(employee_id);
```

* 추가한 제약조건 조회

```
HR@PROD1>select constraint_name, constraint_type, search_condition, index_name
  2  from user_constraints
  3  where table_name = 'TEMP_EMP';

CONSTRAINT_NAME              CONST SEARCH_CONDITION              INDEX_NAME
------------------------------ ----- ------------------------------ ------------------------------
SYS_C0010404                 C     "JOB_ID" IS NOT NULL
SYS_C0010403                 C     "HIRE_DATE" IS NOT NULL
SYS_C0010402                 C     "EMAIL" IS NOT NULL
SYS_C0010401                 C     "LAST_NAME" IS NOT NULL
TEMP_EMP_ID_PK               P                                   TEMP_EMP_ID_PK
```

* 인덱스 컬럼 확인하기

```
HR@PROD1>select index_name, column_name from user_ind_columns
  3  where table_name = 'TEMP_EMP';

INDEX_NAME                     COLUMN_NAME
------------------------------ ------------------------------
TEMP_EMP_ID_PK                       EMPLOYEE_ID
```

temp_emp 삭제

```
HR@PROD1>drop table temp_emp;
Table dropped.

HR@PROD1>select *
  2  from temp_emp;
from temp_emp
     *
ERROR at line 2:
ORA-00942: table or view does not exist

HR@PROD1>select constraint_name, constraint_type, search_condition
  2  from user_constraints
  3  where table_name = 'TEMP_EMP';

no rows selected

HR@PROD1>select index_name, column_name
  2  from user_ind_columns
  3  where table_name = 'TEMP_EMP';

no rows selected
```

휴지통 보기

```
HR@PROD1>SHOW RECYCLEBIN
ORIGINAL NAME       RECYCLEBIN NAME          OBJECT TYPE  DROP TIME
---------------- ------------------------------ ------------ -------------------
TEMP_EMP     BIN$qcHFvwa8eaXgU284qMAOYQ==$0 TABLE          2020-07-06:16:25:00

HR@PROD1>select count(*)
  2  from "BIN$qcHFvwa8eaXgU284qMAOYQ==$0";

  COUNT(*)
----------
       107

1 row selected.
```

flashback 을 사용하여 drop 전 상태로 돌리기

```
HR@PROD1>flashback table temp_emp to before drop;
Flashback complete.

HR@PROD1>show recyclebin

HR@PROD1>select count(*)
  2  from temp_emp;

  COUNT(*)
----------
       107
```

제약조건 이름 변경 작업

```
HR@PROD1>select constraint_name, constraint_type, search_condition
  2  from user_constraints
  3  where table_name = 'TEMP_EMP';

CONSTRAINT_NAME           CONST SEARCH_CONDITION
------------------------------ ----- ------------------------------
BIN$qcHFvwa2eaXgU284qMAOYQ==$0 C     "LAST_NAME" IS NOT NULL
BIN$qcHFvwa3eaXgU284qMAOYQ==$0 C     "EMAIL" IS NOT NULL
BIN$qcHFvwa4eaXgU284qMAOYQ==$0 C     "HIRE_DATE" IS NOT NULL
BIN$qcHFvwa5eaXgU284qMAOYQ==$0 C     "JOB_ID" IS NOT NULL
BIN$qcHFvwa6eaXgU284qMAOYQ==$0 P


HR@PROD1>alter table temp_emp
  2  rename constraint "BIN$qcHFvwa6eaXgU284qMAOYQ==$0"
  3  to temp_emp_id_pk;


HR@PROD1>select constraint_name, constraint_type, search_condition
  2  from user_constraints
  3  where table_name = 'TEMP_EMP';

CONSTRAINT_NAME           CONST SEARCH_CONDITION
------------------------------ ----- ------------------------------
BIN$qcHFvwa2eaXgU284qMAOYQ==$0 C     "LAST_NAME" IS NOT NULL
BIN$qcHFvwa3eaXgU284qMAOYQ==$0 C     "EMAIL" IS NOT NULL
BIN$qcHFvwa4eaXgU284qMAOYQ==$0 C     "HIRE_DATE" IS NOT NULL
BIN$qcHFvwa5eaXgU284qMAOYQ==$0 C     "JOB_ID" IS NOT NULL
TEMP_EMP_ID_PK                 P
```

인덱스명 변경 작업

```
HR@PROD1>select index_name, column_name
  2  from user_ind_columns
  3  where table_name = 'TEMP_EMP';

INDEX_NAME                COLUMN_NAME
------------------------------ ------------------------------
BIN$qcHFvwa7eaXgU284qMAOYQ==$0 EMPLOYEE_ID

HR@PROD1>alter index "BIN$qcHFvwa7eaXgU284qMAOYQ==$0" rename to temp_emp_id_pk;
Index altered.

HR@PROD1>select index_name, column_name
  2  from user_ind_columns
  3  where table_name = 'TEMP_EMP';

INDEX_NAME                COLUMN_NAME
------------------------------ ------------------------------
TEMP_EMP_ID_PK                 EMPLOYEE_ID
```

다시 테이블을 지운다. 그리고 재생성 한 후 다시 지워본다.

```
HR@PROD1>drop table temp_emp;
Table dropped.

HR@PROD1>show recyclebin
ORIGINAL NAME       RECYCLEBIN NAME            OBJECT TYPE  DROP TIME
---------------- ------------------------------ ------------ -------------------
TEMP_EMP      BIN$qcHFvwbDeaXgU284qMAOYQ==$0 TABLE          2020-07-06:16:34:10

HR@PROD1>create table temp_emp
  2  as
  3  select *
  4  from employees;

Table created.

HR@PROD1>drop table temp_emp;
Table dropped.

HR@PROD1>show recyclebin
ORIGINAL NAME       RECYCLEBIN NAME            OBJECT TYPE  DROP TIME
---------------- ------------------------------ ------------ -------------------
TEMP_EMP      BIN$qcHFvwbIeaXgU284qMAOYQ==$0 TABLE          2020-07-06:16:34:30
TEMP_EMP      BIN$qcHFvwbDeaXgU284qMAOYQ==$0 TABLE          2020-07-06:16:34:10
```

다시 flashback table을 한다면 어느 테이블이 복구되는가?

```
HR@PROD1>flashback table temp_emp to before drop;
Flashback complete.

HR@PROD1>show recyclebin
ORIGINAL NAME       RECYCLEBIN NAME            OBJECT TYPE  DROP TIME
---------------- ------------------------------ ------------ -------------------
TEMP_EMP      BIN$qcHFvwbDeaXgU284qMAOYQ==$0 TABLE          2020-07-06:16:34:10
```

* 가장 최근에 지워진 테이블이 복구

또 다른 temp_emp를 지우려고 하면 이미 살아난 테이블의 이름이 temp_emp 로 존재하기 때문에 오류 발생

```
HR@PROD1>flashback table temp_emp to before drop;
flashback table temp_emp to before drop
*
ERROR at line 1:
ORA-38312: original name is used by an existing object

HR@PROD1>flashback table temp_emp to before drop
  2  rename to emp_temp;

Flashback complete.
```

초기화

```
HR@PROD1>purge recyclebin;
Recyclebin purged.

HR@PROD1>drop table temp_emp purge;
Table dropped.

HR@PROD1>drop table emp_temp purge;
Table dropped.
```