프로젝트 보고서_5주차

1. 테이블 스페이스 생성

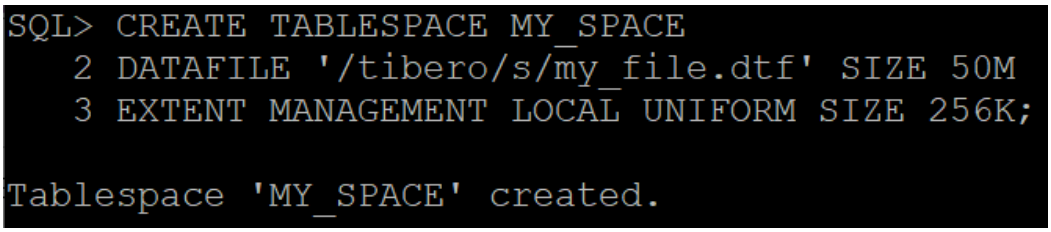
1.1. 수행

수행내역
TABLESPACE 생성
하나의 TABLESPACE에 두개 이상의 DATAFILE
DATAFILE 크기 자동 확장되도록 설정
TABLESPACE 제거
TIBERO 설치시 TABLESPACE 구성

1.2. 결과

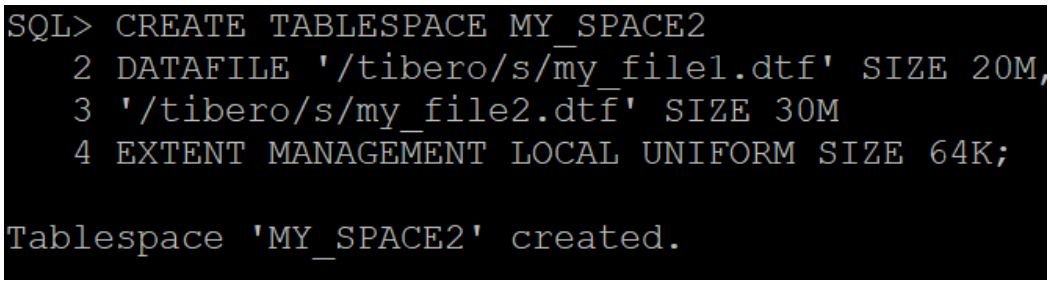
1. TABLESPACE 생성

```
CREATE TABLESPACE MY_SPACE
DATAFILE '/tibero/s/my_file.dtf' SIZE 50M
EXTENT MANAGEMENT LOCAL UNIFORM SIZE 256K;
```



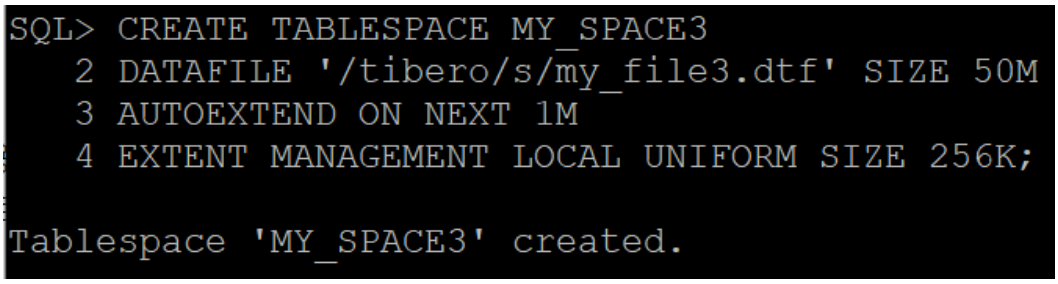
2. 하나의 TABLESPACE에 두개 이상의 DATAFILE

```
CREATE TABLESPACE MY_SPACE2
DATAFILE '/tibero/s/my_file1.dtf' SIZE 20M,
'/tibero/s/my_file2.dtf' SIZE 30M
EXTENT MANAGEMENT LOCAL UNIFORM SIZE 64K;
```



3. DATAFILE 크기 자동 확장되도록 설정

```
CREATE TABLESPACE MY_SPACE3
DATAFILE '/tibero/s/my_file3.dtf' SIZE 50M
AUTOEXTEND ON NEXT 1M
EXTENT MANAGEMENT LOCAL UNIFORM SIZE 256K;
```

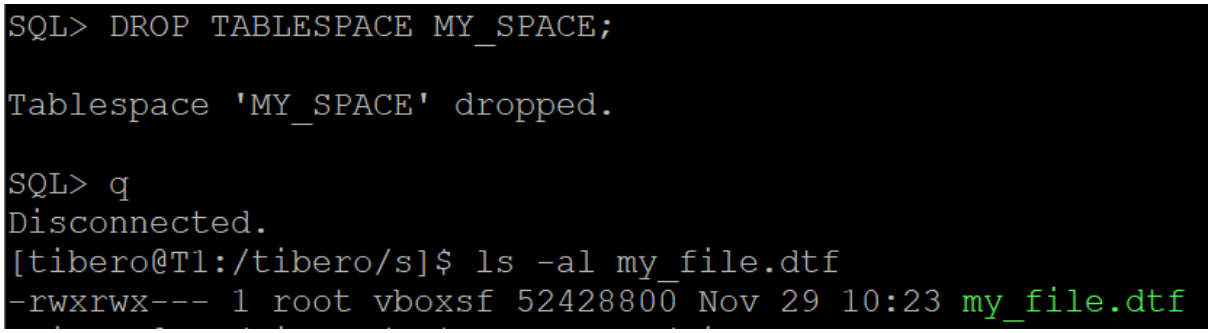


→ AUTOEXTEND로 크기가 모자라면 1MB씩 확장

4. TABLESPACE 제거

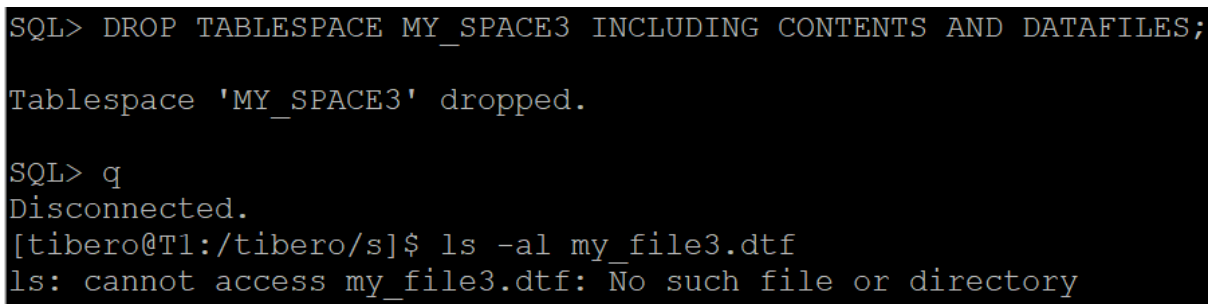
- tablespace만 삭제

```
DROP TABLESPACE MY_SPACE
```



- tablespace & datafile 도 같이 삭제

```
DROP TABLESPACE MY_SPACE3 INCLUDING CONTENTS AND DATAFILES;
```



5. TIBERO 설치시 TABLESPACE 구성

```
DEFAULT TEMPORARY TABLESPACE TEMP
  TEMPFILE '/tibero/tbdata/tibero/temp001.dtf' SIZE 380M
```

```
  AUTOEXTEND ON NEXT 64M MAXSIZE 3G
  EXTENT MANAGEMENT LOCAL AUTOALLOCATE
UNDO TABLESPACE UNDO
  DATAFILE '/tibero/tbdata/tibero/undo001.dtf' SIZE 380M
  AUTOEXTEND ON NEXT 64M MAXSIZE 3G
  EXTENT MANAGEMENT LOCAL UNIFORM SIZE 128k
DEFAULT TABLESPACE USR
  DATAFILE '/tibero/tbdata/tibero/usr001.dtf' SIZE 100m
  AUTOEXTEND ON NEXT 64m MAXSIZE 3G
SYSSUB
  DATAFILE '/tibero/tbdata/tibero/syssub001.dtf' SIZE 380m
  AUTOEXTEND ON NEXT 64M MAXSIZE 3G ;
```

- TEMPORARY TABLESPACE : 임시 파일 저장 공간
- UNDO TABLESPACE : 백업, 복구 정보 공간
- DEFAULT TABLESPACE USR : 디폴트 저장 공간

2. 테이블 스페이스 변경

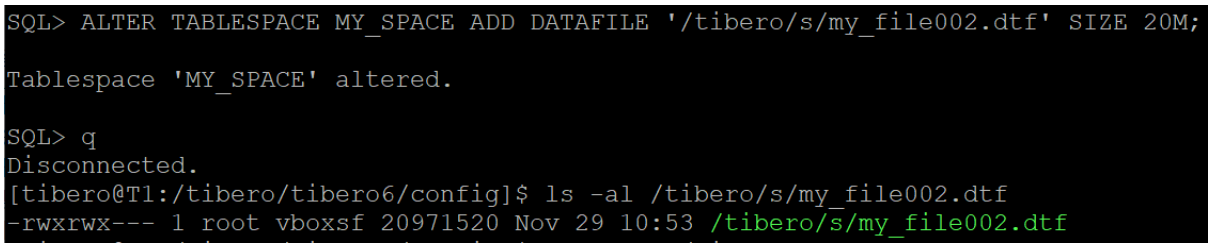
2.1. 수행

수행내역
새로운 데이터 파일 테이블 스페이스에 추가
절대경로 설정 O
절대경로 설정 X
데이터 파일 크기 변경
읽고 쓰는 모든 접근

2.2. 결과

1. 새로운 데이터 파일 테이블 스페이스에 추가 (절대경로 O)

```
ALTER TABLESPACE MY_SPACE ADD DATAFILE '/tibero/s/my_file002.dtf' SIZE 20M;
```



2. 새로운 데이터 파일 테이블 스페이스에 추가 (절대경로 X)

```
ALTER TABLESPACE MY_SPACE ADD DATAFILE 'my_file02.dtf' SIZE 20M;
```

```
SQL> ALTER TABLESPACE MY_SPACE ADD DATAFILE 'my_file02.dtf' SIZE 20M;

Tablespace 'MY_SPACE' altered.

SQL> q
Disconnected.
[tibero@T1:/tibero/s]$ cd $TB_HOME/config
[tibero@T1:/tibero/tibero6/config]$ cat $TB_SID.tip
# tip file generated from /tibero/tibero6/config/tip.template (Mon Nov 28 10:55:02 KS
T 2022)
#------------------------------------------------------------------------------
#
# RDBMS initialization parameter
#
#------------------------------------------------------------------------------

DB_NAME=tibero
LISTENER_PORT=8629
CONTROL_FILES="/tibero/tbdata/tibero/c1.ctl","/tibero/tbdata/tibero/c2.ctl"
DB_CREATE_FILE_DEST=/tibero/tbdata/tibero
LOG_ARCHIVE_DEST=/tibero/tbdata/tibero/arch
MAX_SESSION_COUNT=20
TOTAL_SHM_SIZE=600M
MEMORY_TARGET=1G
[tibero@T1:/tibero/tibero6/config]$ ls -al /tibero/tbdata/tibero/my*
-rw------- 1 tibero dba 20971520 Nov 29 10:51 /tibero/tbdata/tibero/my_file02.dtf
```

- 절대 경로 설정안하면 디폴트 경로($TB_HOME/config/$TB_SID.tip 에 설정된 DB_CREATE_FILE_DEST)에 DATAFILE 생성됨.
- 절대 경로 : /tibero/tbdata/tibero

3. 데이터 파일 크기 변경

- 초기 DATAFILE 크기 확인

```
SELECT TABLESPACE_NAME, BYTES/1024/1024 MB, FILE_NAME
FROM DBA_DATA_FILES
WHERE TABLESPACE_NAME='MY_SPACE';
```

#	TABLE...	MB	FILE_NAME
1	MY_SPACE	50	/tibero/s/my_file.dtf

- 크기 변경

```
ALTER DATABASE DATAFILE '/tibero/s/my_file.dtf' RESIZE 100M;
```

```
SQL> ALTER DATABASE DATAFILE '/tibero/s/my_file.dtf' RESIZE 100M;

Database altered.
```

4. 읽고 쓰는 모든 접근

- offline / online

```
ALTER TABLESPACE MY_SPACE OFFLINE;
ALTER TABLESPACE MY_SPACE ADD DATAFILE '/tibero/s/my_file01.dtf';
ALTER TABLESPACE MY_SPACE ONLINE;
ALTER TABLESPACE MY_SPACE ADD DATAFILE '/tibero/s/my_file01.dtf';
```

```
SQL> ALTER TABLESPACE MY_SPACE OFFLINE;

Tablespace 'MY_SPACE' altered.

SQL> ALTER TABLESPACE MY_SPACE ADD DATAFILE '/tibero/s/my_file01.dtf';
TBR-24038: Tablespace 'MY_SPACE' is offline.

SQL> ALTER TABLESPACE MY_SPACE ONLINE;

Tablespace 'MY_SPACE' altered.

SQL> ALTER TABLESPACE MY_SPACE ADD DATAFILE '/tibero/s/my_file01.dtf';

Tablespace 'MY_SPACE' altered.
```

- 미디어 복구 수행

```
ALTER TABLESPACE MY_SPACE OFFLINE IMMEDIATE;
ALTER DTABASE RECOVER AUTOMATIC TABLESPACE MY_SPACE;
ALTER TABLESPACE MY_SPACE ONLINE;
```

```
SQL> ALTER TABLESPACE MY_SPACE OFFLINE IMMEDIATE;

Tablespace 'MY_SPACE' altered.

SQL> ALTER DTABASE RECOVER AUTOMATIC TABLESPACE MY_SPACE;
TBR-7001: General syntax error.
at line 1, column 7 of null:
ALTER DTABASE RECOVER AUTOMATIC TABLESPACE MY_SPACE
      ^^^^^^^

SQL> ALTER DATABASE RECOVER AUTOMATIC TABLESPACE MY_SPACE;

Database altered.

SQL> ALTER TABLESPACE MY_SPACE ONLINE;

Tablespace 'MY_SPACE' altered.
```

3. 테이블 스페이스 정보 조회

3.1. 수행

수행내역
모든 TABLESPACE의 정보 조회
현재 사용자에 속한 TABLESPACE 정보 조회
모든 TABLESPACE에 대한 간략한 정보
DATAFILE & TABLESPACE 같이 조회

3.2. 결과

1. 모든 TABLESPACE의 정보 조회

```
DESC DBA_TABLESPACES
SELECT TABLESPACE_NAME FROM DBA_TABLESPACES;
```

```
SQL> DESC DBA_TABLESPACES

COLUMN_NAME                              TYPE                 CONSTRAINT
---------------------------------------- -------------------- --------------------
TABLESPACE_NAME                          VARCHAR(128)
TS_ID                                    NUMBER
DATAFILE_COUNT                           NUMBER
BLOCK_SIZE                               NUMBER
NEXT_EXTENT                              NUMBER
STATUS                                   VARCHAR(9)
CONTENTS                                 VARCHAR(9)
LOGGING                                  VARCHAR(9)
FORCE_LOGGING                            VARCHAR(3)
ALLOCATION_TYPE                          VARCHAR(7)
ENCRYPTED                                VARCHAR(3)
INITIAL_EXTENT                           NUMBER
MIN_EXTENTS                              NUMBER
MAX_EXTENTS                              NUMBER
EXTENT_MANAGEMENT                        VARCHAR(5)
SEGMENT_SPACE_MANAGEMENT                 VARCHAR(4)
```

```
SQL> SELECT TABLESPACE_NAME FROM DBA_TABLESPACES;

TABLESPACE_NAME
--------------------
SYSTEM
UNDO
TEMP
USR
SYSSUB
MY_SPACE
MY_SPACE2

7 rows selected.
```

2. 현재 사용자에 속한 TABLESPACE 정보 조회

```
DESC USER_TABLESPACES
SELECT TABLESPACE_NAME FROM USER_TABLESPACES;
```

```
SQL> DESC USER_TABLESPACES

COLUMN_NAME                               TYPE               CONSTRAINT
----------------------------------------- ------------------ --------------------
TABLESPACE_NAME                           VARCHAR(128)
TS_ID                                     NUMBER
DATAFILE_COUNT                            NUMBER
BLOCK_SIZE                                NUMBER
NEXT_EXTENT                               NUMBER
STATUS                                    VARCHAR(9)
CONTENTS                                  VARCHAR(9)
LOGGING                                   VARCHAR(9)
FORCE_LOGGING                             VARCHAR(3)
ALLOCATION_TYPE                           VARCHAR(7)
ENCRYPTED                                 VARCHAR(3)
INITIAL_EXTENT                            NUMBER
MIN_EXTENTS                               NUMBER
MAX_EXTENTS                               NUMBER
EXTENT_MANAGEMENT                         VARCHAR(5)
SEGMENT_SPACE_MANAGEMENT                  VARCHAR(4)
```

```
SQL> SELECT TABLESPACE_NAME FROM USER_TABLESPACES;

TABLESPACE_NAME
--------------------
SYSTEM
UNDO
TEMP
USR
SYSSUB
MY_SPACE
MY_SPACE2

7 rows selected.
```

3. 모든 TABLESPACE에 대한 간략한 정보

```
DESC V$TABLESPACE
COL NAME FOR A20
SELECT * FROM V$TABLESPACE;
```

```
SQL> DESC V$TABLESPACE

COLUMN_NAME                              TYPE               CONSTRAINT
---------------------------------------- ------------------ --------------------
TS#                                      NUMBER
NAME                                     VARCHAR(128)
TYPE                                     VARCHAR(4)
BIGFILE                                  CHAR(2)
FLASHBACK_ON                             CHAR(2)
```

```
SQL> COL NAME FOR A20
SQL> SELECT * FROM V$TABLESPACE;

       TS# NAME                 TYPE BIGFILE FLASHBACK_ON
---------- -------------------- ---- ------- ------------
         0 SYSTEM               DATA NO      NO
         1 UNDO                 UNDO NO      NO
         2 TEMP                 TEMP NO      NO
         3 USR                  DATA NO      NO
         4 SYSSUB               DATA NO      NO
         5 MY_SPACE             DATA NO      NO
         6 MY_SPACE2            DATA NO      NO

7 rows selected.
```

4. DATAFILE & TABLESPACE 같이 조회

```
DESC DBA_DATAFILES;
COL TABLESPACE_NAME FOR A20
COL FILE_NAME FOR A20
SELECT TABLESPACE_NAME, FILE_NAME FROM DBA_DATAFILES;
```

```
SQL> DESC DBA_DATAFILES;

COLUMN_NAME                              TYPE               CONSTRAINT
---------------------------------------- ------------------ --------------------
FILE_NAME                                VARCHAR(256)
FILE_ID                                  NUMBER
TABLESPACE_NAME                          VARCHAR(128)
BYTES                                    NUMBER
BLOCKS                                   NUMBER
STATUS                                   CHAR(9)
RELATIVE_FNO                             NUMBER
AUTOEXTENSIBLE                           VARCHAR(3)
MAXBYTES                                 NUMBER
MAXBLOCKS                                NUMBER
INCREMENT_BY                             NUMBER
```

```
SQL> COL TABLESPACE_NAME FOR A20
SQL> COL FILE_NAME FOR A20
SQL> SELECT TABLESPACE_NAME, FILE_NAME FROM DBA_DATAFILES;

TABLESPACE_NAME      FILE_NAME
-------------------- --------------------
SYSTEM               /tibero/tbdata/tiber
                     o/system001.dtf

UNDO                 /tibero/tbdata/tiber
                     o/undo001.dtf

USR                  /tibero/tbdata/tiber
                     o/usr001.dtf

SYSSUB               /tibero/tbdata/tiber
                     o/syssub001.dtf

MY_SPACE             /tibero/s/my_file.dt
                     f

MY_SPACE2            /tibero/s/my_file1.d
                     tf

MY_SPACE2            /tibero/s/my_file2.d
                     tf

MY_SPACE             /tibero/s/my_file01.
                     dtf


8 rows selected.
```

4. 로그 파일 생성

4.1. 수행

수행내역
두 개의 로그 멤버로 구성된 로그 그룹 추가
특정 로그 그룹 지칭하여 로그 멤버 추가
기존 로그 그룹에 새로운 로그 멤버 추가
$TB_SID.tip에 저장된 아카이브 로그

4.2. 결과

1. 두 개의 로그 멤버로 구성된 로그 그룹 추가

```
ALTER DATABASE ADD LOGFILE(
'/tibero/s/log/my_log21.log',
'/tibero/s/log/my_log22.log') SIZE 512K;
```

```
SQL> ALTER DATABASE ADD LOGFILE(
   2 '/tibero/s/log/my_log21.log',
   3 '/tibero/s/log/my_log22.log') SIZE 512K;

Database altered.

SQL> q
Disconnected.
[tibero@T1:/tibero/s/log]$ ls -al
total 1028
drwxrwx--- 1 root vboxsf      0 Nov 29 23:58 .
drwxrwx--- 1 root vboxsf   4096 Nov 29 23:57 ..
-rwxrwx--- 1 root vboxsf 524288 Nov 29 23:58 my_log21.log
-rwxrwx--- 1 root vboxsf 524288 Nov 29 23:58 my_log22.log
```

2. 특정 로그 그룹 지칭하여 로그 멤버 추가 (7)

```
ALTER DATABASE ADD LOGFILE GROUP 7(
'/tibero/s/log/my_log31.log',
'/tibero/s/log/my_log32.log') SIZE 512K;
```

```
SQL> ALTER DATABASE ADD LOGFILE GROUP 7(
   2 '/tibero/s/log/my_log31.log',
   3 '/tibero/s/log/my_log32.log') SIZE 512K;

Database altered.

SQL> q
Disconnected.
[tibero@T1:/tibero/s/log]$ ls -al
total 1028
drwxrwx--- 1 root vboxsf      0 Nov 30 00:03 .
drwxrwx--- 1 root vboxsf   4096 Nov 29 23:57 ..
-rwxrwx--- 1 root vboxsf 524288 Nov 30 00:03 my_log31.log
-rwxrwx--- 1 root vboxsf 524288 Nov 30 00:03 my_log32.log
```

3. 기존 로그 그룹(group 7)에 새로운 로그 멤버 추가

```
ALTER DATABASE ADD LOGFILE MEMBER
'/tibero/s/log/my_log33.log' TO GROUP 7;
```

```
SQL> ALTER DATABASE ADD LOGFILE MEMBER
   2 '/tibero/s/log/my_log33.log' TO GROUP 7;

Database altered.

SQL> q
Disconnected.
[tibero@T1:/tibero/s/log]$ ls -al
total 1540
drwxrwx--- 1 root vboxsf      0 Nov 30 00:04 .
drwxrwx--- 1 root vboxsf   4096 Nov 29 23:57 ..
-rwxrwx--- 1 root vboxsf 524288 Nov 30 00:03 my_log31.log
-rwxrwx--- 1 root vboxsf 524288 Nov 30 00:03 my_log32.log
-rwxrwx--- 1 root vboxsf 524288 Nov 30 00:04 my_log33.log
```

4. $TB_SID.tip에 저장된 아카이브 로그(LOG_ARCHIVE_DEST)

```
cd /$TB_HOME/config
cat tibero.tip

cd /tibero/tbdata/tibero/arch
```

```
DB_NAME=tibero
LISTENER_PORT=8629
CONTROL_FILES="/tibero/tbdata/tibero/c1.ctl","/tibero/tbdata/tibero/c2.ctl"
DB_CREATE_FILE_DEST=/tibero/tbdata/tibero
LOG_ARCHIVE_DEST=/tibero/tbdata/tibero/arch
MAX_SESSION_COUNT=20
TOTAL_SHM_SIZE=600M
MEMORY_TARGET=1G
[tibero@T1://tibero/tibero6/config]$ cd /tibero/tbdata/tibero/arch
[tibero@T1:/tibero/tbdata/tibero/arch]$ ls -al
total 51200
drwxr-xr-x 2 tibero dba       54 Nov 28 12:42 .
drwxr-xr-x 6 tibero dba      197 Nov 29 11:07 ..
-rw------- 1 tibero dba 26214400 Nov 28 10:59 log-t0-r0-s1.arc
-rw------- 1 tibero dba 26214400 Nov 28 12:42 log-t0-r0-s2.arc
```

5. 로그 파일 제거

5.1. 수행

수행내역
로그 그룹 내의 하나의 로그 멤버 제거
로그 그룹 내에 남겨진 로그 멤버가 하나도 없는 경우 -ERROR
로그 그룹 제거

5.2. 결과

0. 기존 로그 멤버

```
[tibero@T1:/tibero/s/log]$ ls
my_log31.log  my_log32.log  my_log33.log
```

1. 로그 그룹 내의 하나의 로그 멤버 제거

```
ALTER DATABASE DROP LOGFILE MEMBER '/tibero/s/log/my_log31.log';
ALTER DATABASE DROP LOGFILE MEMBER '/tibero/s/log/my_log32.log';
```

```
SQL> ALTER DATABASE DROP LOGFILE MEMBER '/tibero/s/log/my_log31.log';

Database altered.

SQL> ALTER DATABASE DROP LOGFILE MEMBER '/tibero/s/log/my_log32.log';

Database altered.
```

2. 로그 그룹 내에 남겨진 로그 멤버가 하나도 없는 경우 -ERROR

```
ALTER DATABASE DROP LOGFILE MEMBER '/tibero/s/log/my_log33.log';
```

```
SQL> ALTER DATABASE DROP LOGFILE MEMBER '/tibero/s/log/my_log33.log';
TBR-24024: Unable to drop log member cannot drop last member of log file.
```

3. 로그 그룹 제거

```
ALTER DATABASE DROP LOGFILE GROUP 7;
```

```
SQL> ALTER DATABASE DROP LOGFILE GROUP 7;

Database altered.
```

6. 로그 파일 조회

6.1. 수행

수행내역
로그 그룹 정보 조회
V $LOG로그 그룹과 그룹 크기(MB), 로그 파일 개수 조회
로그 파일 정보 조회
V$LOGFILE로 로그 그룹, 로그 파일 (+경로) 조회
로그 그룹과 파일 같은 테이블로 조회

6.2. 결과

1. 로그 그룹 정보 조회

```
DESC V$LOG
```

```
SQL> DESC V$LOG

COLUMN_NAME                                       TYPE                 CONSTRAINT
----------------------------------------- ------------------ -----------
THREAD#                                           NUMBER
GROUP#                                            NUMBER
SEQUENCE#                                         NUMBER
BYTES                                             NUMBER
MEMBERS                                           NUMBER
ARCHIVED                                          VARCHAR(3)
STATUS                                            VARCHAR(8)
FIRST_CHANGE#                                     NUMBER
FIRST_TIME                                        DATE
```

2. V$LOG로그 그룹과 그룹 크기(MB), 로그 파일 개수 조회

```
SELECT GROUP#, BYTES/1024/1024 MB, MEMBERS FROM V$LOG;
```

```
SQL> SELECT GROUP#, BYTES/1024/1024 MB, MEMBERS FROM V$LOG;

    GROUP#         MB    MEMBERS
---------- ---------- ----------
         0         25          2
         1         25          2
         2         25          2
         3         25          2
         4         25          2

5 rows selected.
```

3. 로그 파일 정보 조회

```
DESC V$LOGFILE
```

```
SQL> DESC V$LOGFILE

COLUMN_NAME                                       TYPE                 CONSTRAINT
------------------------------------------------- -------------------- -----------
GROUP#                                            NUMBER
STATUS                                            VARCHAR(7)
TYPE                                              CHAR(6)
MEMBER                                            VARCHAR(256)
```

4. V$LOGFILE로 로그 그룹, 로그 파일 (+경로) 조회

```
COL MEMBER FOR A30
SELECT GROUP#,MEMBER FROM V$LOGFILE;
```

```
    GROUP# MEMBER
---------- ------------------------------
         0 /tibero/tbdata/tibero/redo1/lo
           g01.log

         0 /tibero/tbdata/tibero/redo2/lo
           g02.log

         1 /tibero/tbdata/tibero/redo1/lo
           g11.log

         1 /tibero/tbdata/tibero/redo2/lo
           g12.log

         2 /tibero/tbdata/tibero/redo1/lo
           g21.log

         2 /tibero/tbdata/tibero/redo2/lo
           g22.log

         3 /tibero/tbdata/tibero/redo1/lo
           g31.log

         3 /tibero/tbdata/tibero/redo2/lo
           g32.log


    GROUP# MEMBER
---------- ------------------------------
         4 /tibero/tbdata/tibero/redo1/lo
           g41.log

         4 /tibero/tbdata/tibero/redo2/lo
           g42.log
```

5. 로그 그룹과 파일 같은 테이블로 조회

```
SELECT A.GROUP#, A.ARCHIVED, B.MEMBER
FROM V$LOG A, V$LOGFILE B
ORDER BY 1;
```

```
    GROUP# ARCHIVED MEMBER
---------- -------- ------------------------------
         0 YES      /tibero/tbdata/tibero/redo1/lo
                    g01.log

         0 YES      /tibero/tbdata/tibero/redo2/lo
                    g32.log

         0 YES      /tibero/tbdata/tibero/redo1/lo
                    g31.log

         0 YES      /tibero/tbdata/tibero/redo2/lo
                    g22.log

         0 YES      /tibero/tbdata/tibero/redo1/lo
                    g21.log

         0 YES      /tibero/tbdata/tibero/redo2/lo
                    g42.log

         0 YES      /tibero/tbdata/tibero/redo2/lo
                    g12.log

         0 YES      /tibero/tbdata/tibero/redo1/lo
                    g11.log
```

...

7. 컨트롤 파일 백업

7.1. 수행

수행내역
현재 컨트롤 파일 정보 조회
컨트롤 파일 백업 및 로그 스위치 수행
티베로 종료 및 컨트롤 파일 삭제
티베로 기동하여 마운트 모드 및 장애 상황 확인
티베로 종료 후 노마운트 기동 및 컨트롤 파일 복구
컨트롤 파일 복구 확인

7.2. 결과

1. 현재 컨트롤 파일 정보 조회

```
DESC V$CONTROLFILE

SET TIME ON
COL NAME FOR A30
SELECT * FROM V$CONTROLFILE;
```

```
SQL> DESC V$CONTROLFILE

COLUMN_NAME                              TYPE                CONSTRAINT
---------------------------------------- ------------------- -----------
STATUS                                   NUMBER
NAME                                     VARCHAR(256)
BLKSIZE                                  NUMBER
BLOCKS                                   NUMBER
```

```
SQL> SET TIME ON
01:30:52 SQL> COL NAME FOR A30
01:30:58 SQL> SELECT * FROM V$CONTROLFILE;

    STATUS NAME                              BLKSIZE     BLOCKS
---------- ------------------------------ ---------- ----------
         0 /tibero/tbdata/tibero/c1.ctl        16384       1393
         0 /tibero/tbdata/tibero/c2.ctl        16384       1393

2 rows selected.
```

2. 컨트롤 파일 백업 및 로그 스위치 수행

```
ALTER DATABASE BACKUP CONTROLFILE TO TRACE AS '/tibero/s/backup_on/ctl.sql'
REUSE RESETLOGS;
ALTER SYSTEM SWTICH LOGFILE;
ALTER SYSTEM SWTICH LOGFILE;
ALTER SYSTEM SWTICH LOGFILE;
ALTER SYSTEM SWTICH LOGFILE;
ALTER SYSTEM SWTICH LOGFILE;
```

```
SQL> SET TIME ON
01:46:47 SQL> ALTER DATABASE BACKUP CONTROLFILE TO TRACE AS '/tibero/s/backup_on
/ctl.sql' REUSE RESETLOGS;

Database altered.

01:47:07 SQL> ALTER SYSTEM SWITCH LOGFILE;

System altered.

01:47:17 SQL> ALTER SYSTEM SWITCH LOGFILE;

System altered.

01:47:23 SQL> ALTER SYSTEM SWITCH LOGFILE;

System altered.

01:47:29 SQL> ALTER SYSTEM SWITCH LOGFILE;

System altered.

01:47:35 SQL> ALTER SYSTEM SWITCH LOGFILE;

System altered.
```

3. 티베로 종료 및 컨트롤 파일 삭제

```
tbdown
cd /tibero/tbdata/tibero
rm *.ctl
```

```
[tibero@T1:/tibero/s/backup_on]$ tbdown

Tibero instance terminated (NORMAL mode).

[tibero@T1:/tibero/s/backup_on]$ cd /tibero/tbdata/tibero
[tibero@T1:/tibero/tbdata/tibero]$ ls -al
total 1703452
drwxr-xr-x 6 tibero dba       197 Nov 30 00:53 .
drwxr-xr-x 3 tibero dba        20 Nov 28 10:57 ..
drwxr-xr-x 2 tibero dba       174 Nov 30 01:47 arch
-rw------- 1 tibero dba  22822912 Nov 30 01:47 c1.ctl
-rw------- 1 tibero dba  22822912 Nov 30 01:47 c2.ctl
drwx------ 2 tibero dba        70 Nov 30 00:53 java
-r-------- 1 tibero dba        24 Nov 30 00:52 .passwd
drwxr-xr-x 2 tibero dba        91 Nov 30 00:52 redo1
drwxr-xr-x 2 tibero dba        91 Nov 30 00:52 redo2
-rw------- 1 tibero dba 398458880 Nov 30 01:47 syssub001.dtf
-rw------- 1 tibero dba 398458880 Nov 30 01:47 system001.dtf
-rw------- 1 tibero dba 398458880 Nov 30 00:52 temp001.dtf
-rw------- 1 tibero dba 398458880 Nov 30 01:47 undo001.dtf
-rw------- 1 tibero dba 104857600 Nov 30 01:47 usr001.dtf
[tibero@T1:/tibero/tbdata/tibero]$ rm *.ctl
[tibero@T1:/tibero/tbdata/tibero]$ ls
arch  redo1  syssub001.dtf  temp001.dtf  usr001.dtf
java  redo2  system001.dtf  undo001.dtf
```

4. 티베로 기동하여 마운트 모드 및 장애 상황 확인

```
tbboot
```

```
[tibero@T1:/tibero/tbdata/tibero]$ tbboot
Listener port = 8629

*************************************************************
*   Warning: Control file open failed
*    /tibero/tbdata/tibero/c1.ctl
*************************************************************


*************************************************************
*   Warning: Control file open failed
*    /tibero/tbdata/tibero/c2.ctl
*************************************************************


*************************************************************
* Critical Warning : Raise svmode failed. The reason is
*   TBR-24003 :  Unable to read control file.
*   Current server mode is NOMOUNT.
*************************************************************


Tibero 6

TmaxData Corporation Copyright (c) 2008-. All rights reserved.
Tibero instance started suspended at NOMOUNT mode.
```

5. 티베로 종료 후 노마운트 기동 및 컨트롤 파일 복구

```
tbdown
tbboot nomount
tbsql sys/tibero

SET TIME ON
@/tibero/s/backup_on/ctl.sql
```

```
[tibero@T1:/tibero/tbdata/tibero]$ tbdown

Tibero instance terminated (NORMAL mode).

[tibero@T1:/tibero/tbdata/tibero]$ tbboot nomount
Listener port = 8629

Tibero 6

TmaxData Corporation Copyright (c) 2008-. All rights reserved.
Tibero instance started up (NOMOUNT mode).
[tibero@T1:/tibero/tbdata/tibero]$ tbsql sys/tibero

tbSQL 6

TmaxData Corporation Copyright (c) 2008-. All rights reserved.

Connected to Tibero.

SQL> SET TIME ON
01:48:58 SQL> @/tibero/s/backup_on/ctl.sql
01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13
 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:1
3 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:
13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49:13 01:49
:13 01:49:13 01:49:13

Control File created.
```

6. 컨트롤 파일 복구 확인

```
COL NAME FOR A30
SELECT * FROM CONTROLFILE;
```

```
01:52:25 SQL> COL NAME FOR A30
01:52:34 SQL> SELECT * FROM V$CONTROLFILE;

    STATUS NAME                           BLKSIZE     BLOCKS
---------- ------------------------------ ---------- ----------
         0 /tibero/tbdata/tibero/c1.ctl        16384       1413
         0 /tibero/tbdata/tibero/c2.ctl        16384       1413

2 rows selected.
```

8. 테이블 생성

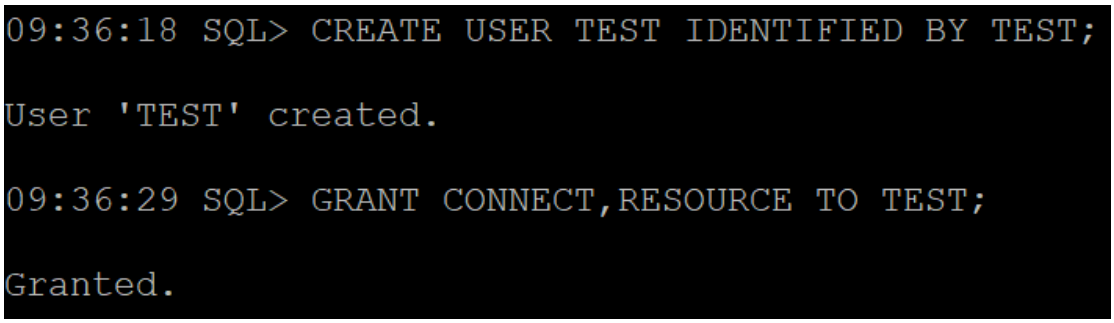
8.1. 수행

수행내역
사용자 생성 및 권한 부여
사용자 접속 후 DEPT, EMP 테이블 생성
테이블 속성 조회
다른 사용자에 같은 이름의 테이블 생성

8.2. 결과

1. 사용자 생성 및 권한 부여

```
CREATE USER TEST IDENTIFIED BY TEST
GRANT RESOURCE, CONNECT BY TEST
```



2. 사용자 접속 후 DEPT, EMP 테이블 생성

```
CONN TEST/TEST

/*DEPT TABLE*/
CREATE TABLE TEST.DEPT
(
DEPTNO NUMBER PRIMARY KEY,  --기본키
DEPTNAME VARCHAR(20),
PDEPTNO NUMBER
)
TABLESPACE MY_SPACE  --저장될 TABLESPACE 지정
PCTFREE 5 INITRANS 3;

/*EMP TABLE*/
CREATE TABLE TEST.EMP
(
EMPNO NUMBER PRIMARY KEY,
ENAME VARCHAR(16) NOT NULL,
ADDR VARCHAR(24),
SALARY NUMBER,
DEPTNO NUMBER,
CHECK (SALARY >= 10000), --SALARY 10000이상
FOREIGN KEY (DEPTNO) REFERENCES TEST.DEPT(DEPTNO) --DEPT TABLE의 DEPTNO를 외래키로 지정
)
TABLESPACE MY_SPACE
PCTFREE 5 INITRANS 3; --남겨둘 여유공간 지정
```

```
10:17:02 SQL> CREATE TABLE TEST.DEPT
(
DEPTNO NUMBER PRIMARY KEY,  --기본키
DEPTNAME VARCHAR(20),
PDEPTNO NUMBER
)
TABLESPACE MY_SPACE  --저장될 TABLESPACE 지정
PCTFREE 5 INITRANS 3;10:17:03    2 10:17:03    3 10:17:03    4 10:17:03    5 10:17
:03    6 10:17:03    7 10:17:03    8

Table 'TEST.DEPT' created.

10:17:04 SQL> CREATE TABLE TEST.EMP
(
EMPNO NUMBER PRIMARY KEY,
ENAME VARCHAR(16) NOT NULL,
ADDR VARCHAR(24),
SALARY NUMBER,
DEPTNO NUMBER,
CHECK (SALARY >= 10000), --SALARY 10000이상
FOREIGN KEY (DEPTNO) REFERENCES DEPT(DEPTNO) --DEPT TABLE의 DEPTNO를 외래키로 지정
)
TABLESPACE MY_SPACE
10:17:14    2 PCTFREE 5 INITRANS 3;10:17:14    3 10:17:14    4 10:17:14    5 10:17
:14    6 10:17:14    7 10:17:14    8 10:17:14    9 10:17:14   10 10:17:14   11 10:
17:14   12

Table 'TEST.EMP' created.
```

3. 테이블 속성 조회

```
DESC DEPT
DESC EMP
```

```
09:45:09 SQL> DESC DEPT

COLUMN_NAME                              TYPE               CONSTRAINT
---------------------------------------- ------------------ --------------------
DEPTNO                                   NUMBER             PRIMARY KEY
DEPTNAME                                 VARCHAR(20)
PDEPTNO                                  NUMBER


INDEX_NAME                     TYPE                     COLUMN_NAME
------------------------------ ------------------------ ----------------------
_SYS_CON32800212               NORMAL                   DEPTNO

09:45:13 SQL> DESC EMP

COLUMN_NAME                              TYPE               CONSTRAINT
---------------------------------------- ------------------ --------------------
EMPNO                                    NUMBER             PRIMARY KEY
ENAME                                    VARCHAR(16)        NOT NULL
ADDR                                     VARCHAR(24)
SALARY                                   NUMBER             CHECK(SALARY >= 1000
                                                            0)

DEPTNO                                   NUMBER             REFERENTIAL


INDEX_NAME                     TYPE                     COLUMN_NAME
------------------------------ ------------------------ ----------------------
_SYS_CON32900785               NORMAL                   EMPNO
```

5. 다른 사용자에 같은 이름의 테이블 생성

```
CREATE USER TEST2 IDENTIFIED BY TEST2;
GRANT CONNECT, RESOURCE TO TEST2;
CONN TEST2/TEST2

CREATE TABLE TEST2.DEPT
(
DEPTNO NUMBER PRIMARY KEY,  --기본키
DEPTNAME VARCHAR(20),
PDEPTNO NUMBER
)
TABLESPACE MY_SPACE  --저장될 TABLESPACE 지정
PCTFREE 5 INITRANS 3;

COL OWNER FOR A20
COL TABLE_NAME FOR A20
SELECT OWNER, TABLE_NAME FROM DBA_TABLES WHERE TABLE_NAME='DEPT';
```

```
10:21:35 SQL> COL OWNER FOR A20
10:21:39 SQL> COL TABLE_NAME FOR A20
10:21:44 SQL> SELECT OWNER, TABLE_NAME FROM DBA_TABLES WHERE TABLE_NAME='DEPT';

OWNER                TABLE_NAME
-------------------- --------------------
TEST                 DEPT
TEST2                DEPT

2 rows selected.
```

9. 테이블 생성 -ERROR

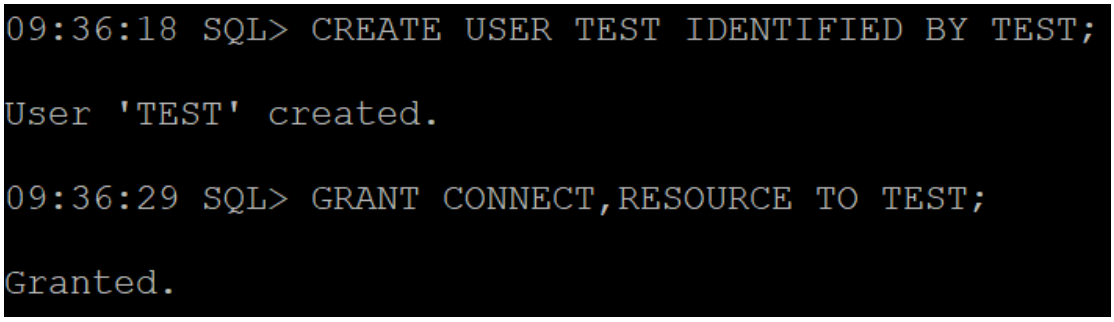
9.1. 수행

수행내역
사용자 생성 및 권한 부여
같은 사용자에 같은 이름의 테이블 생성 - ERROR
부모키가 있는 테이블 없이 테이블 생성 - ERROR

9.2. 결과

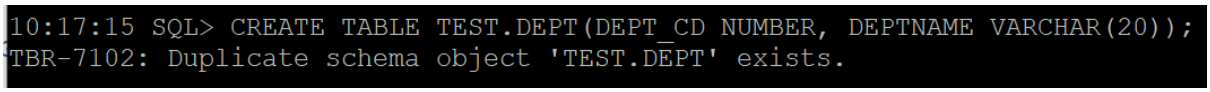
1. 사용자 생성 및 권한 부여

```
CREATE USER TEST IDENTIFIED BY TEST
GRANT RESOURCE, CONNECT BY TEST
```



2. 같은 사용자에 같은 이름의 테이블 생성 - ERROR

```
CREATE TABLE TEST.DEPT(DEPT_CD NUMBER, DEPTNAME VARCHAR(20));
```



3. 부모키가 있는 테이블 없이 테이블 생성 - ERROR

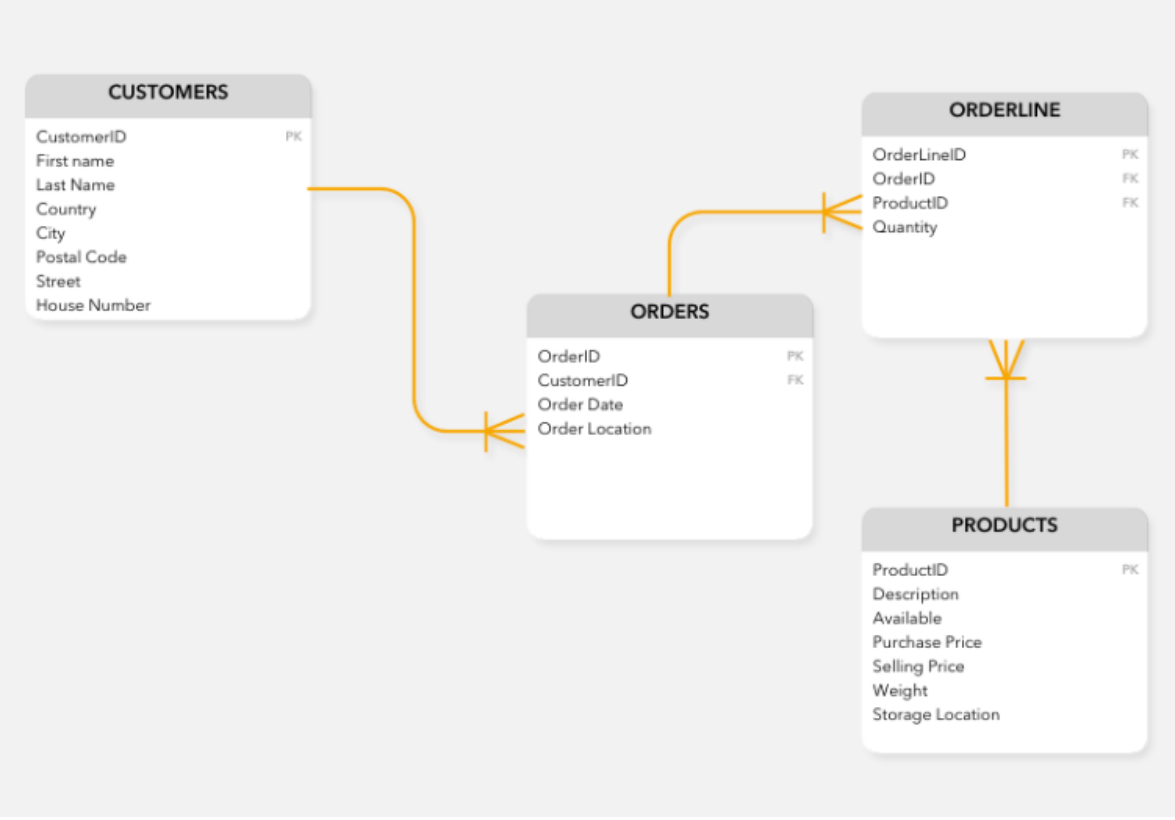
```
CREATE TABLE TEST.EMP
(
EMPNO NUMBER PRIMARY KEY,
ENAME VARCHAR(16) NOT NULL,
ADDR VARCHAR(24),
SALARY NUMBER,
DEPTNO NUMBER,
CHECK (SALARY >= 10000), --SALARY 10000이상
FOREIGN KEY (DEPTNO) REFERENCES TEST.DEPT(DEPTNO) --DEPT TABLE의 DEPTNO를 외래키로 지정
)
TABLESPACE MY_SPACE
PCTFREE 5 INITRANS 3;
```

```
10:30:26 SQL> CREATE TABLE TEST.EMP
(
EMPNO NUMBER PRIMARY KEY,
ENAME VARCHAR(16) NOT NULL,
ADDR VARCHAR(24),
SALARY NUMBER,
DEPTNO NUMBER,
CHECK (SALARY >= 10000), --SALARY 10000이 상
FOREIGN KEY (DEPTNO) REFERENCES TEST.DEPT(DEPTNO) --DEPT TABLE의  DEPTNO를  외 래 키 로  지 정
)
TABLESPACE MY_SPACE
PCTFREE 5 INITRANS 3;10:30:59    2 10:30:59    3 10:30:59    4 10:30:59    5 10:30:59
  6 10:30:59    7 10:30:59    8 10:30:59    9 10:30:59    10 10:30:59    11 10:30:59    12

TBR-7071: Schema object 'TEST.DEPT' was not found or is invalid.
```

10. 테이블 생성 - ERD

10.1. 수행



수행내역
ERD 한 것과 같이 고객과 주문 테이블 생성
CUSTOMER 테이블 생성
ORDERS 테이블 생성
PRODUCTS 테이블 생성
ORDERLINE 테이블 생성
테이블 정보 조회

10.2. 결과

1. CUSTOMER 테이블 생성 (ORDERS 테이블의 부모 테이블)

```
CREATE TABLE CUSTOMERS(
CUSTOMERID NUMBER PRIMARY KEY,
FIRST_NAME VARCHAR2(20),
LAST_NAME VARCHAR2(20),
COUNTRY VARCHAR2(20),
CITY VARCHAR2(20),
POSTAL_CODE NUMBER,
STREET VARCHAR2(20),
HOUSE_NUMBER NUMBER)
;
```

```
15:24:16 SQL> CREATE TABLE CUSTOMERS(
15:24:32    2 CUSTOMERID NUMBER PRIMARY KEY,
15:24:57    3 FIRST_NAME VARCHAR2(20),
15:25:25    4 LAST_NAME VARCHAR2(20),
15:25:34    5 COUNTRY VARCHAR2(20),
15:25:46    6 CITY VARCHAR2(20),
15:25:56    7 POSTAL_CODE NUMBER,
15:26:05    8 STREET VARCHAR2(20),
15:26:17    9 HOUSE_NUMBER NUMBER)
15:27:27   10 ;

Table 'CUSTOMERS' created.
```

2. ORDERS 테이블 생성

- ORDERLINE 테이블의 부모 테이블, CUSTIMERS 테이블의 자식 테이블

```
CREATE TABLE ORDERS(
ORDERID NUMBER PRIMARY KEY,
CUSTOMERID NUMBER,
ORDER_DATE DATE,
ORDER_LOCATION VARCHAR2(20),
FOREIGN KEY (CUSTOMERID) REFERENCES CUSTOMERS(CUSTOMERID));
```

```
15:27:45 SQL> CREATE TABLE ORDERS(
15:28:11    2 ORDERID NUMBER PRIMARY KEY,
15:28:21    3 CUSTOMERID NUMBER,
15:28:35    4 ORDER_DATE DATE,
15:28:44    5 ORDER_LOCATION VARCHAR2(20),
15:29:05    6 FOREIGN KEY (CUSTOMERID) REFERENCES CUSTOMERS(CUSTOMERID));

Table 'ORDERS' created.
```

3. PRODUCTS 테이블 생성 (ORDERLINE 테이블의 부모 테이블)

```
CREATE TABLE PRODUCTS(
PRODUCTID NUMBER PRIMARY KEY,
```

```
DESCRIP VARCHAR2(20),
AVAIL VARCHAR2(20),
P_PRICE NUMBER,
S_PRICE NUMBER,
WEIGHT NUMBER,
S_LOC VARCHAR2(20));
```

```
15:37:38 SQL> CREATE TABLE PRODUCTS(
15:37:51   2 PRODUCTID NUMBER PRIMARY KEY,
15:37:58   3 DESCRIP VARCHAR2(20),
15:38:05   4 AVAIL VARCHAR2(20),
15:38:12   5 P_PRICE NUMBER,
15:38:16   6 S_PRICE NUMBER,
15:38:21   7 WEIGHT NUMBER,
15:38:24   8 S_LOC VARCHAR2(20));

Table 'PRODUCTS' created.
```

4. ORDERLINE 테이블 생성 (ORDER,PRODUCTS 테이블의 자식테이블)

```
CREATE TABLE ORDERLINE(
ORDERLINEID NUMBER PRIMARY KEY,
ORDERID NUMBER,
PRODUCTID NUMBER,
QUANTITY NUMBER,
FOREIGN KEY (ORDERID) REFERENCES ORDERS(ORDERID)
FOREIGN KEY (PRODUCTID) REFERENCES PRODUCTS(PRODUCTID));
```

```
15:40:22 SQL> CREATE TABLE ORDERLINE(
15:40:27   2 ORDERLINEID NUMBER PRIMARY KEY,
15:40:34   3 ORDERID NUMBER,
15:40:37   4 PRODUCTID NUMBER,
15:40:41   5 QUANTITY NUMBER,
15:40:45   6 FOREIGN KEY (ORDERID) REFERENCES ORDERS(ORDERID),
15:41:00   7 FOREIGN KEY (PRODUCTID) REFERENCES PRODUCTS(PRODUCTID));

Table 'ORDERLINE' created.
```

5. 테이블 정보 조회

```
DESC CUSTOMERS
DESC ORDERS
DESC PRODUCTS
DESC ORDERLINE
```

```
15:46:37 SQL> DESC CUSTOMERS

COLUMN_NAME                              TYPE                 CONSTRAINT
---------------------------------------- -------------------- --------------------
CUSTOMERID                               NUMBER               PRIMARY KEY
FIRST_NAME                               VARCHAR(20)
LAST_NAME                                VARCHAR(20)
COUNTRY                                  VARCHAR(20)
CITY                                     VARCHAR(20)
POSTAL_CODE                              NUMBER
STREET                                   VARCHAR(20)
HOUSE_NUMBER                             NUMBER


INDEX_NAME                       TYPE                     COLUMN_NAME
-------------------------------- ------------------------ ----------------------
_SYS_CON33900258                 NORMAL                   CUSTOMERID

15:46:51 SQL> DESC ORDERS

COLUMN_NAME                              TYPE                 CONSTRAINT
---------------------------------------- -------------------- --------------------
ORDERID                                  NUMBER               PRIMARY KEY
CUSTOMERID                               NUMBER               REFERENTIAL
ORDER_DATE                               DATE
ORDER_LOCATION                           VARCHAR(20)


INDEX_NAME                       TYPE                     COLUMN_NAME
-------------------------------- ------------------------ ----------------------
_SYS_CON34000822                 NORMAL                   ORDERID
```

```
15:46:55 SQL> DESC PRODUCTS

COLUMN_NAME                              TYPE                 CONSTRAINT
---------------------------------------- -------------------- --------------------
PRODUCTID                                NUMBER               PRIMARY KEY
DESCRIP                                  VARCHAR(20)
AVAIL                                    VARCHAR(20)
P_PRICE                                  NUMBER
S_PRICE                                  NUMBER
WEIGHT                                   NUMBER
S_LOC                                    VARCHAR(20)


INDEX_NAME                       TYPE                     COLUMN_NAME
-------------------------------- ------------------------ ----------------------
_SYS_CON34200917                 NORMAL                   PRODUCTID

15:47:06 SQL> DESC ORDERLINE

COLUMN_NAME                              TYPE                 CONSTRAINT
---------------------------------------- -------------------- --------------------
ORDERLINEID                              NUMBER               PRIMARY KEY
ORDERID                                  NUMBER               REFERENTIAL
PRODUCTID                                NUMBER               REFERENTIAL
QUANTITY                                 NUMBER


INDEX_NAME                       TYPE                     COLUMN_NAME
-------------------------------- ------------------------ ----------------------
_SYS_CON34500549                 NORMAL                   ORDERLINEID
```

11. 테이블 변경 & 삭제

11.1. 수행

수행내역
컬럼 속성 변경
컬럼 이름 변경
디스크 블록 파라미터 값 변경
부모 테이블 먼저 제거 - ERROR O
부모 테이블 먼저 제거 - ERROR X

11.2. 결과

- 테이블 변경 - ALTER TABLE ~

1. 칼럼 속성 변경

```
DESC TEST.EMP
ALTER TABLE TEST.EMP
MODIFY (SALARY DEFAULT 5000 NOT NULL);
DESC TEST.EMP
```

```
15:56:46 SQL> DESC TEST.EMP

COLUMN_NAME                              TYPE               CONSTRAINT
---------------------------------------- ------------------ --------------------
EMPNO                                    NUMBER             PRIMARY KEY
ENAME                                    VARCHAR(16)        NOT NULL
ADDR                                     VARCHAR(24)
SALARY                                   NUMBER             CHECK(SALARY >= 1000
                                                            0)

DEPTNO                                   NUMBER             REFERENTIAL


INDEX_NAME                      TYPE                    COLUMN_NAME
------------------------------- ----------------------- ----------------------
_TEST_CON34900916               NORMAL                  EMPNO

15:57:01 SQL> ALTER TABLE TEST.EMP
15:57:08    2 MODIFY (SALARY DEFAULT 5000 NOT NULL);

Table 'TEST.EMP' altered.

15:57:23 SQL> DESC TEST.EMP

COLUMN_NAME                              TYPE               CONSTRAINT
---------------------------------------- ------------------ --------------------
EMPNO                                    NUMBER             PRIMARY KEY
ENAME                                    VARCHAR(16)        NOT NULL
ADDR                                     VARCHAR(24)
SALARY                                   NUMBER             CHECK(SALARY >= 1000
                                                            0)

                                                            NOT NULL
DEPTNO                                   NUMBER             REFERENTIAL


INDEX_NAME                      TYPE                    COLUMN_NAME
```

2. 칼럼 이름 변경

```
ALTER TABLE TEST.EMP RENAME COLUMN ADDR TO ADDRESS;
```

```
15:57:33 SQL> ALTER TABLE TEST.EMP RENAME COLUMN ADDR TO ADDRESS;

Table 'TEST.EMP' altered.

15:57:55 SQL> DESC TEST.EMP

COLUMN_NAME                              TYPE               CONSTRAINT
---------------------------------------- ------------------ --------------------
EMPNO                                    NUMBER             PRIMARY KEY
ENAME                                    VARCHAR(16)        NOT NULL
ADDRESS                                  VARCHAR(24)
SALARY                                   NUMBER             CHECK(SALARY >= 1000
                                                            0)

                                                            NOT NULL
DEPTNO                                   NUMBER             REFERENTIAL


INDEX_NAME                     TYPE                     COLUMN_NAME
------------------------------ ------------------------ ----------------------
_TEST_CON34900916              NORMAL                   EMPNO
```

3. 디스크 블록 파라미터 값 변경

```
ALTER TABLE TEST.EMP PCTFREE 10;
SELECT TABLE_NAME, PCT_FREE FROM DBA_TABLES WHERE OWNER='TEST';
```

```
16:06:06 SQL> SELECT TABLE_NAME, PCT_FREE FROM DBA_TABLES WHERE OWNER='TEST';

TABLE_NAME             PCT_FREE
-------------------- ----------
EMP                          10
DEPT                          5

2 rows selected.
```

4. 부모 테이블 먼저 제거 - ERROR O

```
DROP TABLE TEST.DEPT;
```

```
15:59:13 SQL> DROP TABLE TEST.DEPT;
TBR-7156: Referenced by foreign keys.
```

4. 부모 테이블 먼저 제거 - ERROR X

```
DROP TABLE TEST.DEPT CASCADE CONSTRAINTS;
```

```
15:59:36 SQL> DROP TABLE TEST.DEPT CASCADE CONSTRAINTS;

Table 'TEST.DEPT' dropped.

15:59:44 SQL> DESC TEST.DEPT
TBS-70020: Specified object was not found.
```

12. 테이블 정보 조회 정적 뷰

12.1. 수행

수행내역
모든 테이블의 정보 조회
현재 사용자에 속한 테이블 정보 조회
현재 사용자가 접근 가능한 테이블의 정보 조회
컬럼 정보 조회
테이블과 컬럼 정보 하나의 테이블로 조회

12.2. 결과

1. 모든 테이블의 정보 조회

```
DESC DBA_TABLES
```

```
16:06:38 SQL> DESC DBA_TABLES

COLUMN_NAME                              TYPE                 CONSTRAINT
---------------------------------------- -------------------- --------------------
OWNER                                    VARCHAR(128)
TABLE_NAME                               VARCHAR(128)
TABLESPACE_NAME                          VARCHAR(128)
PCT_FREE                                 NUMBER
INI_TRANS                                NUMBER
LOGGING                                  VARCHAR(3)
NUM_ROWS                                 NUMBER
BLOCKS                                   NUMBER
AVG_ROW_LEN                              NUMBER
DEGREE                                   NUMBER
SAMPLE_SIZE                              NUMBER
LAST_ANALYZED                            DATE
PARTITIONED                              VARCHAR(3)
BUFFER_POOL                              VARCHAR(7)
ROW_MOVEMENT                             VARCHAR(8)
DURATION                                 VARCHAR(11)
COMPRESSION                              VARCHAR(3)
COMPRESS_FOR                             VARCHAR(12)
DROPPED                                  VARCHAR(3)
READ_ONLY                                VARCHAR(3)
TEMPORARY                                VARCHAR(3)
MAX_EXTENTS                              NUMBER
IOT_TYPE                                 VARCHAR(12)
INITIAL_EXTENT                           NUMBER
NEXT_EXTENT                              NUMBER
MIN_EXTENTS                              NUMBER
IS_VIRTUAL                               VARCHAR(1)
```

2. 현재 사용자에 속한 테이블 정보 조회

```
DESC USER_TABLES
```

```
16:09:11 SQL> DESC USER_TABLES

COLUMN_NAME                              TYPE                 CONSTRAINT
---------------------------------------- -------------------- --------------------
TABLE_NAME                               VARCHAR(128)
TABLESPACE_NAME                          VARCHAR(128)
PCT_FREE                                 NUMBER
INI_TRANS                                NUMBER
LOGGING                                  VARCHAR(3)
NUM_ROWS                                 NUMBER
BLOCKS                                   NUMBER
AVG_ROW_LEN                              NUMBER
DEGREE                                   NUMBER
SAMPLE_SIZE                              NUMBER
LAST_ANALYZED                            DATE
PARTITIONED                              VARCHAR(3)
BUFFER_POOL                              VARCHAR(7)
ROW_MOVEMENT                             VARCHAR(8)
DURATION                                 VARCHAR(11)
COMPRESSION                              VARCHAR(3)
COMPRESS_FOR                             VARCHAR(12)
DROPPED                                  VARCHAR(3)
READ_ONLY                                VARCHAR(3)
TEMPORARY                                VARCHAR(3)
MAX_EXTENTS                              NUMBER
IOT_TYPE                                 VARCHAR(12)
INITIAL_EXTENT                           NUMBER
NEXT_EXTENT                              NUMBER
MIN_EXTENTS                              NUMBER
IS_VIRTUAL                               VARCHAR(1)
```

3. 현재 사용자가 접근 가능한 테이블의 정보 조회

```
DESC ALL_TABLES
```

```
16:09:15 SQL> DESC ALL_tABLES

COLUMN_NAME                              TYPE                CONSTRAINT
---------------------------------------- ------------------- --------------------
OWNER                                    VARCHAR(128)
TABLE_NAME                               VARCHAR(128)
TABLESPACE_NAME                          VARCHAR(128)
PCT_FREE                                 NUMBER
INI_TRANS                                NUMBER
LOGGING                                  VARCHAR(3)
NUM_ROWS                                 NUMBER
BLOCKS                                   NUMBER
AVG_ROW_LEN                              NUMBER
DEGREE                                   NUMBER
SAMPLE_SIZE                              NUMBER
LAST_ANALYZED                            DATE
PARTITIONED                              VARCHAR(3)
BUFFER_POOL                              VARCHAR(7)
ROW_MOVEMENT                             VARCHAR(8)
DURATION                                 VARCHAR(11)
COMPRESSION                              VARCHAR(3)
COMPRESS_FOR                             VARCHAR(12)
DROPPED                                  VARCHAR(3)
READ_ONLY                                VARCHAR(3)
TEMPORARY                                VARCHAR(3)
MAX_EXTENTS                              NUMBER
IOT_TYPE                                 VARCHAR(12)
INITIAL_EXTENT                           NUMBER
NEXT_EXTENT                              NUMBER
MIN_EXTENTS                              NUMBER
IS_VIRTUAL                               VARCHAR(1)
```

4. 컬럼 정보 조회

```
DESC DBA_TBL_COLUMNS
DESC USER_TBL_COLUMNS
DESC ALL_TBL_COLUMNS
```

```
16:09:24 SQL> DESC DBA_TBL_COLUMNS

COLUMN_NAME                              TYPE                CONSTRAINT
---------------------------------------- ------------------- --------------------
OWNER                                    VARCHAR(128)
TABLE_NAME                               VARCHAR(128)
COLUMN_NAME                              VARCHAR(128)
DATA_TYPE                                VARCHAR(65532)
DATA_TYPE_OWNER                          VARCHAR(128)
DATA_LENGTH                              NUMBER
DATA_PRECISION                           NUMBER
DATA_SCALE                               NUMBER
NULLABLE                                 VARCHAR(1)
COLUMN_ID                                NUMBER
DATA_DEFAULT                             LONG
DEFAULT_LENGTH                           NUMBER
CHAR_COL_DECL_LENGTH                     NUMBER
CHAR_LENGTH                              NUMBER
CHAR_USED                                VARCHAR(1)
VIRTUAL_COLUMN                           VARCHAR(1)
```

```
16:09:35 SQL> DESC USER_TBL_COLUMNS

COLUMN_NAME                              TYPE               CONSTRAINT
---------------------------------------- ------------------ --------------------
TABLE_NAME                               VARCHAR(128)
COLUMN_NAME                              VARCHAR(128)
DATA_TYPE                                VARCHAR(65532)
DATA_TYPE_OWNER                          VARCHAR(128)
DATA_LENGTH                              NUMBER
DATA_PRECISION                           NUMBER
DATA_SCALE                               NUMBER
NULLABLE                                 VARCHAR(1)
COLUMN_ID                                NUMBER
DATA_DEFAULT                             LONG
DEFAULT_LENGTH                           NUMBER
CHAR_COL_DECL_LENGTH                     NUMBER
CHAR_LENGTH                              NUMBER
CHAR_USED                                VARCHAR(1)
VIRTUAL_COLUMN                           VARCHAR(1)
```

```
16:09:51 SQL> DESC ALL_TBL_COLUMNS

COLUMN_NAME                              TYPE               CONSTRAINT
---------------------------------------- ------------------ --------------------
OWNER                                    VARCHAR(128)
TABLE_NAME                               VARCHAR(128)
COLUMN_NAME                              VARCHAR(128)
DATA_TYPE                                VARCHAR(65532)
DATA_TYPE_OWNER                          VARCHAR(128)
DATA_LENGTH                              NUMBER
DATA_PRECISION                           NUMBER
DATA_SCALE                               NUMBER
NULLABLE                                 VARCHAR(1)
COLUMN_ID                                NUMBER
DATA_DEFAULT                             LONG
DEFAULT_LENGTH                           NUMBER
CHAR_COL_DECL_LENGTH                     NUMBER
CHAR_LENGTH                              NUMBER
CHAR_USED                                VARCHAR(1)
VIRTUAL_COLUMN                           VARCHAR(1)
OBJ_ID                                   NUMBER
```

5. 테이블과 컬럼 정보 하나의 테이블로 조회

```
COL COLUMN_NAME FOR A20
SELECT A.OWNER, A.TABLE_NAME, B.COLUMN_NAME
FROM DBA_TABLES A, DBA_TBL_COLUMNS B
WHERE A.OWNER='TEST' AND A.TABLE_NAME=B.TABLE_NAME;
```

```
16:37:51 SQL> COL COLUMN_NAME FOR A20
16:38:04 SQL> ^C
16:38:07 SQL> SELECT A.OWNER, A.TABLE_NAME, B.COLUMN_NAME
FROM DBA_TABLES A, DBA_TBL_COLUMNS B
WHERE A.OWNER='TEST' AND A.TABLE_NAME=B.TABLE_NAME;16:38:19

OWNER                TABLE_NAME           COLUMN_NAME
-------------------- -------------------- --------------------
TEST                 EMP                  ENAME
TEST                 EMP                  ADDR
TEST                 EMP                  SALARY
TEST                 EMP                  DEPTNO
TEST                 EMP                  EMPNO
TEST                 DEPT                 DEPTNO
TEST                 DEPT                 DEPTNAME
TEST                 DEPT                 PDEPTNO

8 rows selected.
```

13. 테이블 압축

13.1. 수행

수행내역
TEST 사용자 생성 및 권한 부여
압축이 지정된 테이블 생성
partition별 압축 지정하는 테이블 생성
TEST 사용자 테이블 압축 확인
기존 테이블 압축 및 압축 해제
테이블의 추가적인 DML에 대한 압축 여부를 변경
압축된 테이블에 DDL 수행 - ERROR

13.2. 결과

1. TEST 사용자 생성 및 권한 부여

```
CREATE USER TEST IDENTIFIED BY TEST;
GRANT CONNECT, RESOURCE TO TEST

CONN TEST/TEST
```

```
SQL> CREATE USER TEST IDENTIFIED BY TEST;

User 'TEST' created.

SQL> GRANT RESOURCE, CONNECT TO TEST;

Granted.

SQL> CONN TEST/TEST
Connected to Tibero.
```

2. 압축이 지정된 테이블 생성

```
CREATE TABLE TEST.EMP_PART(
EMPNO DECIMAL(4),
ENAME VARCHAR(10),
JOB VARCHAR(9),
MGR DECIMAL(4),
HIREDATE VARCHAR(14),
SAL NUMBER(7,2),
COMM NUMBER(7,2),
DEPTNO NUMBER(2))
COMPRESS;
```

```
SQL> CREATE TABLE TEST.EMP(
  2 EMPNO DECIMAL(4),
  3 ENAME VARCHAR(10),
  4 JOB VARCHAR(9),
  5 MGR DECIMAL(4),
  6 HIREDATE VARCHAR(14),
  7 SAL NUMBER(7,2),
  8 COMM NUMBER(7,2),
  9 DEPTNO NUMBER(2))
 10 COMPRESS;

Table 'TEST.EMP' created.
```

3. partition별 압축 지정하는 테이블 생성

```
CREATE TABLE TEST.EMP_PART(
EMPNO DECIMAL(4),
ENAME VARCHAR(10),
JOB VARCHAR(9),
MGR DECIMAL(4),
HIREDATE VARCHAR(14),
SAL NUMBER(7,2),
COMM NUMBER(7,2),
DEPTNO NUMBER(2))
COMPRESS
PARTITION BY RANGE(EMPNO)
(PARTITION EMP_PART1 VALUES LESS THAN(500),
```

```
PARTITION EMP_PART2 VALUES LESS THAN(1000) NOCOMPRESS,
PARTITION EMP_PART3 VALUES LESS THAN(1500),
PARTITION EMP_PART4 VALUES LESS THAN(2000) NOCOMPRESS,
PARTITION EMP_PART5 VALUES LESS THAN(MAXVALUE));
```

```
SQL> CREATE TABLE TEST.EMP_PART(
EMPNO DECIMAL(4),
ENAME VARCHAR(10),
JOB VARCHAR(9),
MGR DECIMAL(4),
HIREDATE VARCHAR(14),
SAL NUMBER(7,2),
COMM NUMBER(7,2),
DEPTNO NUMBER(2))
COMPRESS
PARTITION BY RANGE(EMPNO)
(PARTITION EMP_PART1 VALUES LESS THAN(500),
PARTITION EMP_PART2 VALUES LESS THAN(1000) NOCOMPRESS,
PARTITION EMP_PART3 VALUES LESS THAN(1500),
PARTITION EMP_PART4 VALUES LESS THAN(2000) NOCOMPRESS,
PARTITION EMP_PART5 VALUES LESS THAN(MAXVALUE));   2
 8    9   10   11   12   13   14   15   16

Table 'TEST.EMP_PART' created.
```

4. TEST 사용자 테이블 압축 확인

```
/*테이블 조회 권한 부여*/
CONN SYS/TIBERO
GRANT SELECT ON USER_TABLES TO TEST;
GRANT SELECT ON DBA_TABLES TO TEST;
CONN TEST/TEST

/*테이블 압축 확인*/
SELECT TABLE_NAME, COMPRESSION FROM USER_TABLES WHERE TABLE_NAME LIKE 'EMP%';
```

```
SQL> CONN SYS/TIBERO
Connected to Tibero.

SQL> GRANT SELECT ON USER_TABLES TO TEST;

Granted.

SQL> GRANT SELECT ON DBA_TABLES TO TEST;

Granted.

SQL> CONN TEST/TEST
Connected to Tibero.
```

```
SQL> SELECT TABLE_NAME, COMPRESSION FROM USER_TABLES WHERE TABLE_NAME LIKE 'EMP%
';

TABLE_NAME           COMPRESSION
-------------------- -----------
EMP                  YES
EMP_PART             YES

2 rows selected.
```

5. 기존 테이블 압축 및 압축 해제

```
ALTER TABLE EMP MOVE NOCOMPRESS;
ALTER TABLE EMP_PART MOVE PARTITION EMP_PART1 NOCOMPRESS;
```

```
SQL> ALTER TABLE EMP MOVE NOCOMPRESS;

Table 'EMP' altered.
```

```
SQL> ALTER TABLE EMP_PART MOVE PARTITION EMP_PART1 NOCOMPRESS;

Table 'EMP_PART' altered.
```

6. 테이블의 추가적인 DML에 대한 압축 여부를 변경

```
ALTER TABLE EMP COMPRESS;
```

```
SQL> ALTER TABLE EMP COMPRESS;

Table 'EMP' altered.
```

7. 압축된 테이블에 DDL(칼럼 변경, 삭제) 수행 - ERROR

```
ALTER TABLE TEST.EMP
MODIFY (SALARY DEFAULT 5000 NOT NULL);
```

```
SQL> ALTER TABLE TEST.EMP
  2 MODIFY (SALARY DEFAULT 5000 NOT NULL);
TBR-7002: Unsupported DDL.
```

14. INDEX ORGANIZED TABLE 생성

14.1. 수행

수행내역
TEST 사용자 생성 및 권한 부여
일반 테이블 생성
INDEX ORGANIZED TABLE 생성
IOT TABLE의 제약조건 - ERROR
일반 테이블과 IOT 테이블 비교 (ROWID)

14.2. 결과

1. TEST 사용자 생성 및 권한 부여

```
SET TIME ON
CREATE USER TEST IDENTIFIED BY TEST;
GRANT CONNECT, RESOURCE TO TEST;
CONN TEST/TEST
```

```
22:34:39 SQL> CREATE USER TEST IDENTIFIED BY TEST;

User 'TEST' created.

22:34:51 SQL> GRANT CONNECT, RESOURCE TO TEST;

Granted.

22:34:56 SQL> CONN TEST/TEST
Connected to Tibero.
```

2. 일반 테이블 생성

```
CREATE TABLE TEST.TBL
(COL1 NUMBER,
COL2 VARCHAR2(20),
```

```
COL3 VARCHAR2(10),
COL4 VARCHAR2(10),
PRIMARY KEY (COL1,COL2));
```

3.INDEX ORGANIZED TABLE 생성

```
CREATE TABLE TEST.TBL_IOT
(COL1 NUMBER,
COL2 VARCHAR2(20),
COL3 VARCHAR2(10),
COL4 VARCHAR2(10),
PRIMARY KEY (COL1,COL2))
ORGANIZATION INDEX
PCTTHRESHOLD 40
OVERFLOW;
```

```
22:39:16 SQL> CREATE TABLE TEST.TBL_IOT
22:39:27    2 (COL1 NUMBER,
22:39:31    3 COL2 VARCHAR2(20),
22:39:35    4 COL3 VARCHAR2(10),
22:39:41    5 COL4 VARCHAR2(10),
22:39:47    6 PRIMARY KEY (COL1,COL2))
22:39:53    7 ORGANIZATION INDEX
22:40:00    8 PCTTHRESHOLD 40
22:40:08    9 OVERFLOW;

Table 'TEST.TBL_IOT' created.
```

4. IOT TABLE의 제약조건 - ERROR

- IOT TABLE은 LOB,LONG 타입의 칼럼 포함할 수 없음

```
CREATE TABLE TEST.TBL_IOT2
(COL1 NUMBER PRIMARY KEY,
COL2 LONG)
ORGANIZATION INDEX
PCTTHRESHOLD 40
OVERFLOW;
```

```
22:44:27 SQL> CREATE TABLE TEST.TBL_IOT2
22:44:34    2 (COL1 NUMBER PRIMARY KEY,
22:44:42    3 COL2 LONG)
22:44:48    4 ORGANIZATION INDEX
22:44:52    5 PCTTHRESHOLD 40
22:44:58    6 OVERFLOW;
TBR-7482: Index-Organized Table cannot include a LOB or LONG type column.
```

- IOT TABLE은 PRIMARY KEY를 반드시 포함해야함.

```
CREATE TABLE TEST.TBL_IOT2
(COL1 NUMBER)
ORGANIZATION INDEX
```

```
PCTTHRESHOLD 40
OVERFLOW;
```

```
22:45:12 SQL> CREATE TABLE TEST.TBL_IOT2
22:45:21    2 (COL1 NUMBER)
22:45:27    3 ORGANIZATION INDEX
22:45:30    4 PCTTHRESHOLD 40
22:45:37    5 OVERFLOW;
TBR-7263: Primary key constraint was not found.
```

5. 일반 테이블과 IOT 테이블 비교 (ROWID)

```
/*일반 테이블 ROWID출력*/
INSERT INTO TEST.TBL VALUES(1,'STUDENT','A','B');
INSERT INTO TEST.TBL VALUES(2,'PROFESSOR','E','F');
SELECT ROWID, TEST.TBL.* FROM TEST.TBL;
/*IOT TABLE ROWID 출력*/
INSERT INTO TEST.TBL_IOT VALUES(1,'STUDENT','A','B');
INSERT INTO TEST.TBL_IOT VALUES(2,'PROFESSOR','E','F');
SELECT ROWID, TEST.TBL_IOT.* FROM TEST.TBL_IOT;
```

```
22:56:26 SQL> SELECT ROWID, TEST.TBL.* FROM TEST.TBL;

ROWID                  COL1 COL2                 COL3       COL4
------------------ ---------- -------------------- ---------- ----------
AAAAq8AACAAAABmAAA          1 STUDENT              A          B
AAAAq8AACAAAABmAAB          2 PROFESSOR            E          F

2 rows selected.
```

```
22:57:20 SQL> SELECT ROWID, TEST.TBL_IOT.* FROM TEST.TBL_IOT;
TBR-8026: Invalid identifier.
at line 1, column 9 of null:
SELECT ROWID, TEST.TBL_IOT.* FROM TEST.TBL_IOT
       ^
```

- IOT TABLE은 일반 테이블과 다르게 ROWID가 없고 PRIMARY KEY로 ORDERING된다.

15. 제약조건 생성

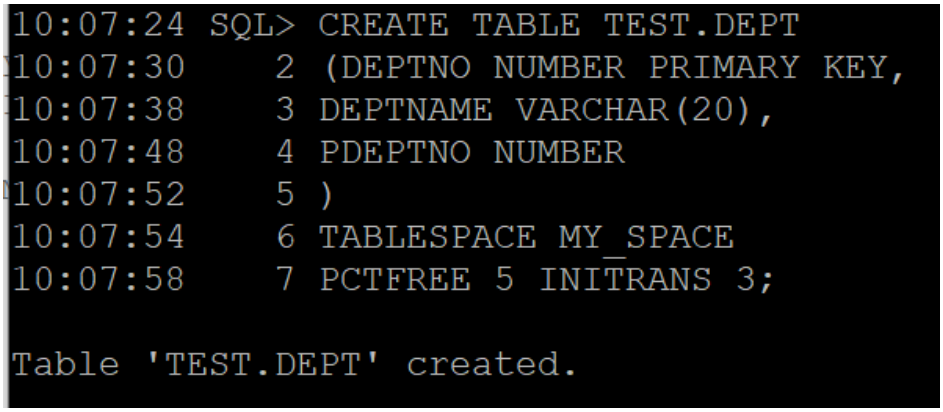
15.1. 수행

수행내역
제약조건 이름 설정
제약조건 컬럼 단위로 선언
제약조건 테이블 단위로 선언 - ERROR O
제약조건 테이블 단위로 선언 - ERROR X

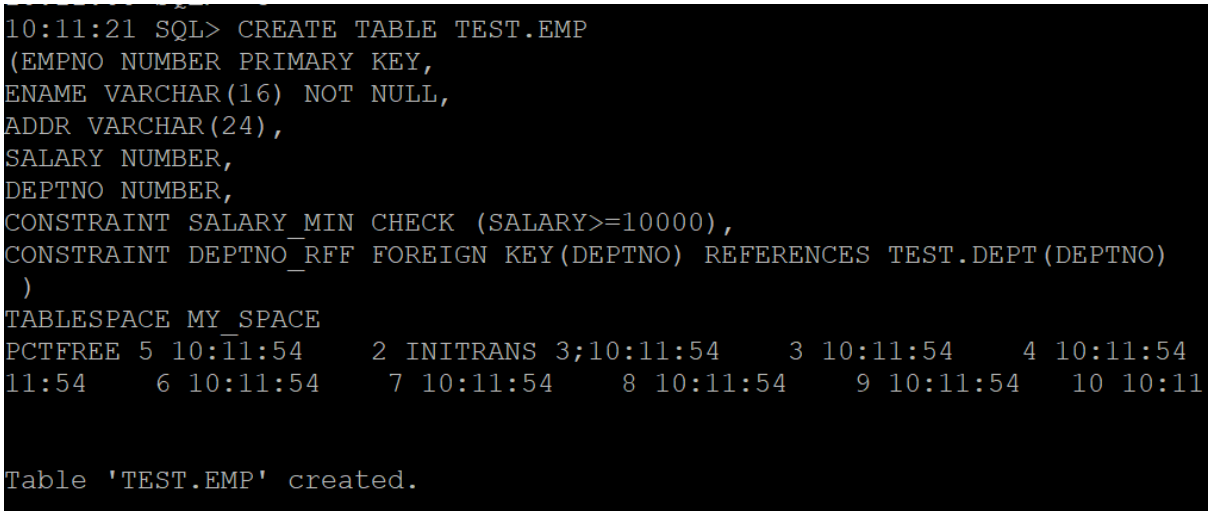
15.2. 결과

1. 제약조건 이름 설정

```
CREATE TABLE TEST.DEPT
(DEPTNO NUMBER PRIMARY KEY,
DEPTNAME VARCHAR(20),
PDEPTNO NUMBER
)
TABLESPACE MY_SPACE
PCTFREE 5 INITRANS 3;
```




```
CREATE TABLE TEST.EMP
(EMPNO NUMBER PRIMARY KEY,
ENAME VARCHAR(16) NOT NULL,
ADDR VARCHAR(24),
SALARY NUMBER,
DEPTNO NUMBER,
CONSTRAINT SALARY_MIN CHECK (SALARY>=10000),
CONSTRAINT DEPTNO_RFF FOREIGN KEY(DEPTNO) REFERENCES TEST.DEPT(DEPTNO)
 )
TABLESPACE MY_SPACE
PCTFREE 5 INITRANS 3;
```



- EMP 테이블에 SALARY_MIN, DEPTNO_RFF 제약조건 설정

2. 제약조건 컬럼 단위로 선언

```
CREATE TABLE TEST.TEMP_PROD
(
       PROD_ID     NUMBER(6)      CONSTRAINT PROD_ID_PK PRIMARY KEY,
       PROD_NAME   VARCHAR2(50)   CONSTRAINT PROD_NAME_NN NOT NULL,
       PROD_COST   VARCHAR2(30)   CONSTRAINT PROD_COST_NN NOT NULL,
       PROD_PID    NUMBER(6) ,
       PROD_DATE   DATE           CONSTRAINT PROD_DATE_NN NOT NULL
);

DESC TEST.TEMP_PROD
```

```
10:20:58 SQL> CREATE TABLE TEST.TEMP_PROD
(
       PROD_ID     NUMBER(6)      CONSTRAINT PROD_ID_PK PRIMAR
Y KEY,
       PROD_NAME   VARCHAR2(50)   CONSTRAINT PROD_NAME_NN NOT
NULL,
       PROD_COST   VARCHAR2(30)   CONSTRAINT PROD_COST_NN NOT
NULL,
       PROD_PID    NUMBER(6) ,
       PROD_DATE   DATE           CONSTRAINT PROD_DATE_NN NOT
NULL
);10:21:15    2 10:21:15    3 10:21:15    4 10:21:15    5 10:2
1:15    6 10:21:15    7 10:21:15    8

Table 'TEST.TEMP_PROD' created.

10:21:16 SQL> DESC TEST.TEMP_PROD

COLUMN_NAME                               TYPE               CONSTRAINT
----------------------------------------- ------------------ ----------------------
PROD_ID                                   NUMBER(6)          PRIMARY KEY
PROD_NAME                                 VARCHAR(50)        NOT NULL
PROD_COST                                 VARCHAR(30)        NOT NULL
PROD_PID                                  NUMBER(6)
PROD_DATE                                 DATE               NOT NULL


INDEX_NAME                      TYPE                     COLUMN_NAME
------------------------------- ------------------------ ----------------------
PROD_ID_PK                      NORMAL                   PROD_ID
```

3. 제약조건 테이블 단위로 선언 - ERROR O

```
CREATE TABLE TEST.TEMP_PROD
(PROD_ID NUMBER(6),
PROD_NAME VARCHAR2(50) CONSTRAINT PROD_ID_PK PRIMARY KEY(PROD_ID, PROD_NAME),
PROD_COST VARCHAR2(30) CONSTRAINT PROD_COST_NN NOT NULL,
PROD_PID NUMBER(6),
PROD_DATE DATE CONSTRAINT PROD_DATE_NN NOT NULL);
```

```
10:29:53 SQL> CREATE TABLE TEST.TEMP_PROD
10:30:04    2 (PROD_ID NUMBER(6),
10:30:18    3 PROD_NAME VARCHAR2(50) CONSTRAINT PROD_ID_PK PRI
MARY KEY(PROD_ID, PROD_NAME),
10:31:12    4 PROD_COST VARCHAR2(30) CONSTRAINT PROD_COST_NN N
OT NULL,
10:31:27    5 PROD_PID NUMBER(6),
10:31:34    6 PROD_DATE DATE CONSTRAINT PROD_DATE_NN NOT NULL)
;
TBR-7001: General syntax error.
at line 3, column 57 of null:
PROD_NAME VARCHAR2(50) CONSTRAINT PROD_ID_PK PRIMARY KEY(PROD_
ID, PROD_NAME),
                                                        ^
```

- 두개 이상의 칼럼의 제약조건을 선언하려면 모든 칼럼 선언한 후 정의해야함

4. 제약조건 테이블 단위로 선언 - ERROR X

```
CREATE TABLE TEST.TEMP_PROD
(CONSTRAINT PROD_ID_PK PRIMARY KEY(PROD_ID,PROD_NAME),
PROD_ID NUMBER(6),
PROD_NAME VARCHAR2(50) CONSTRAINT PROD_NAME_NN NOT NULL,
PROD_COST VARCHAR2(30) CONSTRAINT PROD_COST_NN NOT NULL,
PROD_PID NUMBER(6),
PROD_DATE DATE CONSTRAINT PROD_DATE_NN NOT NULL);
```

```
10:25:09 SQL> CREATE TABLE TEST.TEMP_PROD
10:25:14    2 (PROD_ID NUMBER(6),
10:25:19    3 PROD_NAME VARCHAR2(50) CONSTRAINT PROD_NAME_NN N
OT NULL,
10:25:37    4 CONSTRAINT PROD_ID_PK PRIMARY KEY(PROD_ID, PROD_
NAME),
10:25:56    5 PROD_COST VARCHAR2(30) CONSTRAINT PROD_COST_NN N
OT NULL,
10:26:16    6 PROD_PID NUMBER(6),
10:26:22    7 PROD_DATE DATE CONSTRAINT PROD_DATE_NN NOT NULL)
;

Table 'TEST.TEMP_PROD' created.

10:26:36 SQL> DESC TEST.TEMP_PROD

COLUMN_NAME                              TYPE              CONSTRAINT
---------------------------------------- ----------------- --------------------
PROD_ID                                  NUMBER(6)         PRIMARY KEY
PROD_NAME                                VARCHAR(50)       PRIMARY KEY
                                                           NOT NULL
PROD_COST                                VARCHAR(30)       NOT NULL
PROD_PID                                 NUMBER(6)
PROD_DATE                                DATE              NOT NULL


INDEX_NAME                     TYPE                    COLUMN_NAME
------------------------------ ----------------------- --------------------
PROD_ID_PK                     NORMAL                  PROD_ID
                                                       PROD_NAME
```

- 두 개 이상의 컬럼의 제약조건 선언은 어느 위치에 해도 에러 X

16. 제약조건 변경 및 삭제

- 기본 키, 유일 키 제약조건을 제외하고는 반드시 제약조건의 이름이 있어야 한다.

16.1. 수행

수행내역
제약조건 이름 변경
제약조건 새로 추가
제약조건 제거
제약조건 확인

16.2. 결과

1. 제약조건 이름 변경

```
ALTER TABLE TEST.EMP
RENAME CONSTRAINT SALARY_MIN TO SAL_MIN;

CONN SYS/TIBERO
GRANT SELECT ON DBA_CONSTRAINTS TO TEST; --제약조건TABLE 조회할 권한 부여

CONN TEST/TEST
SELECT CONSTRAINT_NAME FROM DBA_CONSTRAINTS WHERE TABLE_NAME='EMP';
```

```
10:41:54 SQL> ALTER TABLE TEST.EMP
10:41:57    2 RENAME CONSTRAINT SALARY_MIN TO SAL_MIN;

Table 'TEST.EMP' altered.
```

```
10:44:16 SQL> SELECT CONSTRAINT_NAME FROM DBA_CONSTRAINTS WHERE TABLE_NAME='EMP';

CONSTRAINT_NAME
--------------------------------------------------------------------------------
_TEST_CON37200701
SAL_MIN
DEPTNO_RFF
_TEST_CON37300577

4 rows selected.
```

2. 제약조건 새로 추가

```
ALTER TABLE TEST.EMP
ADD CONSTRAINT SALARY_MAX CHECK (SALARY>=50000);

ALTER TABLE TEST.EMP
ADD UNIQUE(ENAME, DEPTNO);

/*바뀐 제약조건 확인*/
SELECT CONSTRAINT_NAME FROM DBA_CONSTRAINTS WHERE TABLE_NAME='EMP';
```

```
10:57:28 SQL> ALTER TABLE TEST.EMP
10:57:34    2 ADD CONSTRAINT SALARY_MAX CHECK (SALARY>=50000);

Table 'TEST.EMP' altered.

10:58:01 SQL> ALTER TABLE TEST.EMP
10:58:10    2 ADD UNIQUE(ENAME, DEPTNO);

Table 'TEST.EMP' altered.
```

```
10:58:19 SQL> SELECT CONSTRAINT_NAME FROM DBA_CONSTRAINTS WHERE TABLE_NAME='EMP'

CONSTRAINT_NAME
--------------------------------------------------------------------------------
_TEST_CON37200701
SAL_MIN
SALARY_MAX
_TEST_CON38900400
DEPTNO_RFF
_TEST_CON37300577

6 rows selected.
```

3. 제약조건 제거

```
ALTER TABLE TEST.EMP
DROP PRIMARY KEY;

ALTER TABLE TEST.EMP
DROP CONSTRAINT SALARY_MAX;
```

```
11:19:36 SQL> ALTER TABLE TEST.EMP
11:19:40    2 DROP PRIMARY KEY;

Table 'TEST.EMP' altered.

11:19:44 SQL> ALTER TABLE TEST.EMP
11:19:49    2 DROP CONSTRAINT SALARY_MAX;

Table 'TEST.EMP' altered.
```

4. 제약조건 확인

```
SELECT CONSTRAINT_NAME FROM DBA_CONSTRAINTS WHERE TABLE_NAME='EMP';
```

```
11:20:45 SQL> SELECT CONSTRAINT_NAME FROM DBA_CONSTRAINTS WHERE TABLE_NAME=
'EMP';

CONSTRAINT_NAME
--------------------------------------------------------------------------------
-----
SAL_MIN
_TEST_CON38900400
DEPTNO_RFF
_TEST_CON37300577

4 rows selected.
```

17. 제약조건 관리하기 (DISABLE NOVALIDATE)

- DISABLE NOVALIDATE : 일시적으로 제약조건 사용하지 않는 것

17.1. 수행

수행내역
TEST 사용자 생성 및 권한 부여
TEST 사용자에 테이블 생성
PRIMARY KEY에 중복된 데이터 삽입 - ERROR O
DISABLE NOVALIDATE 사용
PRIMARY KEY에 중복된 데이터 삽입 - ERROR X

17.2. 결과

1. TEST 사용자 생성 및 권한 부여

```
CREATE USER TEST IDENTIFIED BY TEST;
GRANT CONNECT, RESOURCE TO TEST;
CONN TEST/TEST
```

```
15:18:24 SQL> CREATE USER TEST IDENTIFIED BY TEST;

User 'TEST' created.

15:18:47 SQL> GRANT CONNECT, RESOURCE TO TEST;

Granted.

15:18:52 SQL> CONN TEST/TEST
Connected to Tibero.
```

2. TEST 사용자에 테이블 생성

```
CREATE TABLE TEST.TBL
(NO NUMBER,
NAME VARCHAR2(20),
CONSTRAINT NO_PK PRIMARY KEY(NO));
```

```
15:44:36 SQL> CREATE TABLE TEST.TBL
15:44:41    2 (NO NUMBER,
15:44:44    3 NAME VARCHAR2(20),
15:44:48    4 CONSTRAINT NO_PK PRIMARY KEY(NO));

Table 'TEST.TBL' created.
```

3. PRIMARY KEY에 중복된 데이터 삽입 - ERROR O

```
INSERT INTO TEST.TBL VALUES(1,'AAA');
INSERT INTO TEST.TBL VALUES(2,'BBB');
INSERT INTO TEST.TBL VALUES(3,'CCC');
INSERT INTO TEST.TBL VALUES(1,'DDD');
```

```
15:44:57 SQL> INSERT INTO TEST.TBL VALUES(1,'AAA');

1 row inserted.

15:45:06 SQL> INSERT INTO TEST.TBL VALUES(2,'BBB');

1 row inserted.

15:45:38 SQL> INSERT INTO TEST.TBL VALUES(3,'CCC');

1 row inserted.

15:45:46 SQL> INSERT INTO TEST.TBL VALUES(1,'DDD');
TBR-10007: UNIQUE constraint violation ('TEST'.'NO_PK').
```

4. DISABLE NOVALIDATE 사용

```
ALTER TABLE TEST.TBL
DISABLE NOVALIDATE CONSTRAINT NO_PK;
```

```
15:46:00 SQL> ALTER TABLE TEST.TBL
15:46:16    2 DISABLE NOVALIDATE CONSTRAINT NO_PK;

Table 'TEST.TBL' altered.
```

18. 제약조건 관리하기 (DISABLE VALIDATE)

- DISABLE VALIDATE : 테이블에 칼럼 추가, 삭제, 변경 불가

18.1. 수행

수행내역
칼럼 내용 변경 불가하게 제약조건 변경
데이터 조회
해당 테이블에 칼럼 추가 - ERROR
해당 테이블에 칼럼 삭제 - ERROR

18.2. 결과

1. 칼럼 내용 변경 불가하게 제약조건 변경

```
ALTER TABLE TEST.TBL
DISABLE VALIDATE CONSTRAINT NO_PK;
```

```
16:10:27 SQL> ALTER TABLE TEST.TBL
16:10:32    2 DISABLE VALIDATE CONSTRAINT NO_PK;

Table 'TEST.TBL' altered.
```

2. 데이터 조회

```
SELECT * FROM TEST.TBL;
```

```
16:10:41 SQL> SELECT * FROM TEST.TBL;

        NO NAME
---------- --------------------
         1 AAA
         2 BBB
         3 CCC

3 rows selected.
```

3. 해당 테이블에 칼럼 추가 - ERROR

```
INSERT INTO TEST.TBL VALUES(4,'DDD');
```

```
16:12:01 SQL> INSERT INTO TEST.TBL VALUES(4,'DDD');
TBR-8091: INSERT/UPDATE/DELETE cannot be performed on tables with constrain
ts (TBL.NO_PK) in the DISABLE VALIDATE state.
```

4. 해당 테이블에 칼럼 삭제 - ERROR

```
DELETE FROM TEST.TBL WHERE NAME='CCC';
```

```
TBR-8091: INSERT/UPDATE/DELETE cannot be performed on tables with constrain
ts (TBL.NO_PK) in the DISABLE VALIDATE state.
```

19. 제약조건 관리하기 (ENABLE NOVALIDATE)

- ENABLE NOVALIDATE : ENABLE 하고 나서 입력되는 데이터만 검사

19.1. 수행

수행내역
NOT NULL 제약조건이 있는 테이블 생성
DISABLE 시킨 후 NULL 값을 입력
ENABLE NOVALIDATE 제약조건으로 변경
해당 테이블에 NULL값 추가 - ERROR

19.2. 결과

1. NOT NULL 제약조건이 있는 테이블 생성

```
CREATE TABLE TEST.TBL2
(NO NUMBER CONSTRAINT NO_NN NOT NULL,
NAME VARCHAR2(20));

INSERT INTO TEST.TBL2 VALUES(1,'AAA');
INSERT INTO TEST.TBL2 VALUES(2,'BBB');
INSERT INTO TEST.TBL2 VALUES(3,'CCC');
```

```
16:29:19 SQL> CREATE TABLE TEST.TBL2
16:29:25    2 (NO NUMBER CONSTRAINT NO_NN NOT NULL,
16:29:32    3 NAME VARCHAR2(20));

Table 'TEST.TBL2' created.
```

2. DISABLE 시킨 후 NULL 값을 입력

```
ALTER TABLE TEST.TBL2
DISABLE CONSTRAINT NO_NN;

INSERT INTO TEST.TBL2 VALUES(NULL,'DDD');
SELECT * FROM TEST.TBL2;
```

```
16:29:42 SQL> INSERT INTO TEST.TBL2 VALUES(1,'AAA');

1 row inserted.

16:30:03 SQL> INSERT INTO TEST.TBL2 VALUES(2,'BBB');

1 row inserted.

16:30:14 SQL> INSERT INTO TEST.TBL2 VALUES(3,'CCC');

1 row inserted.
```

```
16:30:28 SQL> ALTER TABLE TEST.TBL2
16:30:46    2 DISABLE CONSTRAINT NO_NN;

Table 'TEST.TBL2' altered.

16:30:56 SQL> INSERT INTO TEST.TBL2 VALUES(NULL,'DDD');

1 row inserted.

16:31:12 SQL> SELECT * FROM TEST.TBL2;

        NO NAME
---------- --------------------
         1 AAA
         2 BBB
         3 CCC
           DDD

4 rows selected.
```

3. ENABLE NOVALIDATE 제약조건으로 변경

```
ALTER TABLE TEST.TBL2
ENABLE NOVALIDATE CONSTRAINT NO_NN;

SELECT * FROM TEST.TBL2;
```

```
16:31:19 SQL> ALTER TABLE TEST.EBL2
16:31:57    2 ENABLE NOVALIDATE CONSTRAINT NO_NN;
TBR-7071: Schema object 'TEST.EBL2' was not found or is invalid.

16:32:10 SQL> ALTER TABLE TEST.TBL2
16:32:15    2 ENABLE NOVALIDATE CONSTRAINT NO_NN;

Table 'TEST.TBL2' altered.

16:32:22 SQL> SELECT * FROM TEST.TBL2;

        NO NAME
---------- --------------------
         1 AAA
         2 BBB
         3 CCC
           DDD

4 rows selected.
```

4. 해당 테이블에 NULL값 추가 - ERROR

```
INSERT INTO TEST.TBL2 VALUES(NULL,'EEE');
```

```
16:32:29 SQL> INSERT INTO TEST.TBL2 VALUES(NULL,'EEE');
TBR-10005: NOT NULL constraint violation ('TEST'.'TBL2'.'NO').
```

→ 원래 있던 NULL값 (잘못된 데이터)는 탐지하지 못함.

20. 제약조건 관리하기 (ENABLE VALIDATE)

- ENABLE VALIDATE : 기존데이터든 이후에 입력되는 신규 데이터든 모든 데이터를 전부 검사

20.1. 수행

수행내역
잘못된 데이터가 들어있는 테이블 조회
DISABLE 로 제약조건 변경
ENABLE VALIDATE - 잘못된 데이터가 들어있기 때문에 ERROR

20.2. 결과

1. 잘못된 데이터가 들어있는 테이블 조회

```
DESC TEST.TBL2
SELECT * FROM TEST.TBL2;
```

```
17:20:01 SQL> DESC TEST.TBL2

COLUMN_NAME                              TYPE                 CONSTRAINT

---------------------------------------- ------------------ -----------
--
NO                                       NUMBER               NOT NULL
NAME                                     VARCHAR(20)
```

```
16:36:43 SQL> SELECT * FROM TEST.TBL2;

        NO NAME
---------- --------------------
         1 AAA
         2 BBB
         3 CCC
           DDD

4 rows selected.
```

2. DISABLE 로 제약조건 변경

```
ALTER TABLE TEST.TBL2
DISABLE CONSTRAINT NO_NN;
```

```
17:18:41 SQL> ALTER TABLE TEST.TBL2
17:19:18    2 DISABLE CONSTRAINT NO_NN;

Table 'TEST.TBL2' altered.
```

3. ENABLE VALIDATE - 잘못된 데이터가 들어있기 때문에 ERROR

```
ALTER TABLE TEST.TBL2
ENABLE VALIDATE CONSTRAINT NO_NN;
```

```
17:19:26 SQL> ALTER TABLE TEST.TBL2
17:19:31    2 ENABLE VALIDATE CONSTRAINT NO_NN;
TBR-7045: A column contains a NULL value: cannot create or enable the constrai
nt.
```

21. 디스크 블록 파라미터 설정

- PCTFREE : 새로 객체 삽입하고 삭제하는 경우 기존의 디스크 블록에 반영됨
- INITRANS : 기존의 디스크 블록에 반영되지 않고 새로 할당된 디스크 블록에만 반영됨

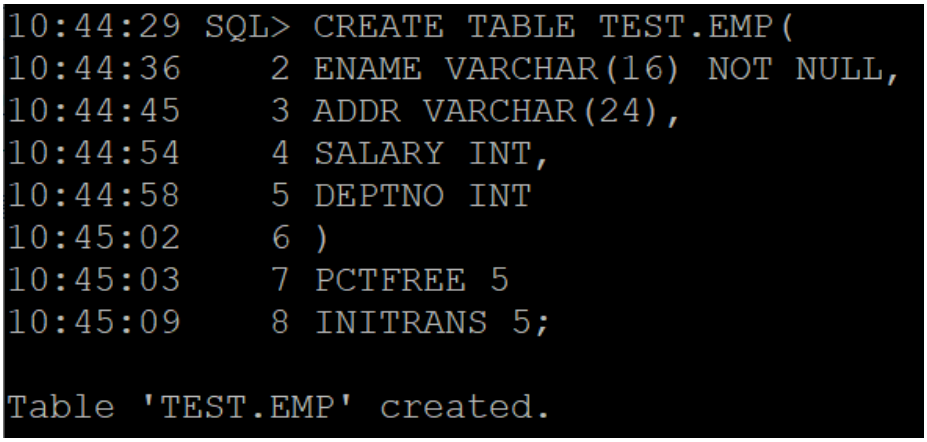
21.1. 수행

수행내역
PCTREE, INITRANS 파라미터 설정된 테이블 생성
디스크 블록 파라미터 변경
인덱스 설정할 때 INITRANS 파라미터 값 설정

21.2. 결과

1. PCTREE, INITRANS 파라미터 설정된 테이블 생성

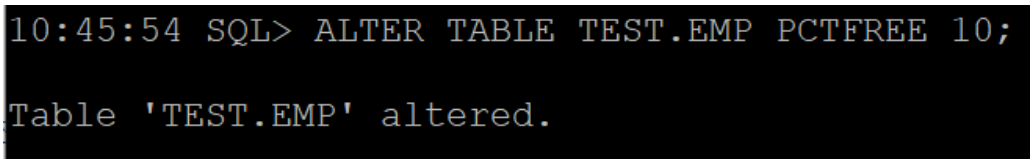
```
CREATE TABLE TEST.EMP(
ENAME VARCHAR(16) NOT NULL,
ADDR VARCHAR(24),
SALARY INT,
DEPTNO INT
)
PCTFREE 5
INITRANS 5;
```



2. 디스크 블록 파라미터 변경

```
ALTER TABLE TEST.EMP PCTFREE 10;

/*변경 확인*/
CONN SYS/TIBERO
DESC DBA_TABLES
GRANT SELECT ON DBA_TABLES TO TEST;
CONN TEST/TEST
SELECT PCT_FREE, INI_TRANS FROM DBA_TABLES WHERE TABLE_NAME='EMP';
```



```
10:49:58 SQL> SELECT PCT_FREE, INI_TRANS FROM DBA_TABLES WHERE TABLE_NAME='EMP';

  PCT_FREE  INI_TRANS
---------- ----------
        10          5

1 row selected.
```

3. 인덱스 설정할 때 INITRANS 파라미터 값 설정

```
CREATE INDEX EMP_DEPTNO_IDX ON TEST.EMP(DEPTNO) INITRANS 5;
DESC TEST.EMP
```

```
10:54:25 SQL> CREATE INDEX EMP_DEPTNO_IDX ON TEST.EMP(DEPTNO) INITRANS 5;

Index 'EMP_DEPTNO_IDX' created.

10:54:42 SQL> DESC TEST.EMP

COLUMN_NAME                              TYPE               CONSTRAINT
---------------------------------------- ------------------ --------------------
ENAME                                    VARCHAR(16)        NOT NULL
ADDR                                     VARCHAR(24)
SALARY                                   NUMBER(38)
DEPTNO                                   NUMBER(38)


INDEX_NAME                       TYPE                     COLUMN_NAME
-------------------------------- ------------------------ ------------------------
EMP_DEPTNO_IDX                   NORMAL                   DEPTNO
```

22. 인덱스 HINT

- INDEX는 WHERE절에 많이 사용되는 칼럼에 INDEX를 지정하여 비용을 줄이고 공간

22.1. 수행

수행내역
TEST 사용자에 테이블 스페이스 지정
TEST 사용자에 대용량 테이블 생성
INDEX 생성
INDEX HINT 사용하여 성능 확인
NO_INDEX HINT 사용하여 성능 확인
인덱스의 사용여부 모니터링

22.2. 결과

1. TEST 사용자에 테이블 스페이스 지정

```
/*테이블 스페이스 생성*/
CREATE TABLESPACE TEST_SPACE
  DATAFILE '/tibero/share/TEST_001.dtf' SIZE 500M
```

```
  AUTOEXTEND ON NEXT 10M MAXSIZE 32G
  EXTENT MANAGEMENT LOCAL
  UNIFORM SIZE 256K;

/*INDEX 테이블 스페이스 생성*/
CREATE TABLESPACE TEST_SPACE_IDX
  DATAFILE '/tibero/share/TEST_IDX_001.dtf' SIZE 500M
  AUTOEXTEND ON NEXT 10M MAXSIZE 32G
  EXTENT MANAGEMENT LOCAL
  UNIFORM SIZE 256K;

/*TEST 사용자 생성*/
CREATE USER TEST IDENTIFIED BY 'TEST' DEFAULT TABLESPACE TEST_SPACE;
GRANT DBA TO TEST;
GRANT CONNECT, RESOURCE TO TEST WITH ADMIN OPTION;

/*TEST 사용자 접속*/
CONN TEST/TEST

/*임시 테이브스페이스 생성*/
CREATE TEMPORARY TABLESPACE TEST_TEMP
  TEMPFILE '/tibero/share/TEST_temp.dtf' SIZE 500M
  AUTOEXTEND ON NEXT 10M MAXSIZE 32G
  EXTENT MANAGEMENT LOCAL
  UNIFORM SIZE 256K;

ALTER DATABASE DEFAULT TEMPORARY TABLESPACE TEST_TEMP;
SELECT USERNAME, DEFAULT_TEMP_TABLESPACE FROM DBA_USERS WHERE USERNAME='TEST';
```

```
SQL> SELECT USERNAME, DEFAULT_TEMP_TABLESPACE FROM DBA_USERS WHERE USERNAME='TEST';

USERNAME
--------------------------------------------------------------------------------
DEFAULT_TEMP_TABLESPACE
--------------------------------------------------------------------------------
TEST
TEST_TEMP


1 row selected.
```

2. TEST 사용자에 대용량 테이블(TEST1) 생성

```
create table TEST.TEST1 (
name varchar(20),
sex varchar(1)
) TABLESPACE TEST_SPACE;

insert into TEST.TEST1 select 'HUMAN'||to_char(lpad(level,8,'0')), TO_CHAR(ROUND(DBMS_RANDOM.VALUE(1,2), 0)) from
dual connect by level <= 70000000;
commit;
```

3. 인덱스 생성

```
create index IDX_01_TEST_TEST1 on TEST.TEST1(sex) TABLESPACE TEST_SPACE_IDX;
DESC TEST.TEST1;
```

```
SQL> DESC TEST.TEST1;

COLUMN_NAME                              TYPE               CONSTRAINT
---------------------------------------- ------------------ --------------------
NAME                                     VARCHAR(20)
SEX                                      VARCHAR(1)


INDEX_NAME                     TYPE                    COLUMN_NAME
------------------------------ ----------------------- ----------------------
IDX_01_TEST_TEST1              NORMAL                  SEX
```

4. INDEX HINT 사용하여 성능 확인

```
SET AUTOT TRACEONLY EXP PLANSTAT
-- type1 : ALIAS 사용
SELECT /*+ INDEX(A1 IDX_01_TEST_TEST1) */
 A1.NAME
FROM TEST.TEST1 A1 WHERE A1.SEX = '1';
```

```
SQL> SET AUTOT TRACEONLY EXP PLANSTAT
SQL> -- type1 : ALIAS 사용
SELECT /*+ INDEX(A1 IDX_01_TEST_TEST1) */
 A1.NAME
FROM TEST.TEST1 A1 WHERE A1.SEX = '1';SQL>    2    3

SQL ID: bg9x7s0vf0s64
Child number: 234
Plan hash value: 1755549671

Execution Plan
----------------------------------------------------------------------------------
   1  TABLE ACCESS (ROWID): TEST1 (Cost:227924, %%CPU:0, Rows:31681238)
   2    INDEX (RANGE SCAN): IDX_01_TEST_TEST1 (Cost:53805, %%CPU:0, Rows:3168123
8)



Predicate Information
----------------------------------------------------------------------------------
   2 - access: ("A1"."SEX" = '1') (0.562)


Note
----------------------------------------------------------------------------------
   2 - dynamic sampling used for this table (98 blocks)


Execution Stat
----------------------------------------------------------------------------------
   1  TABLE ACCESS (ROWID): TEST1 (Time:0. ms, Rows:0, Starts:0)
   2    INDEX (RANGE SCAN): IDX_01_TEST_TEST1 (Time:0. ms, Rows:0, Starts:0)
```

5. 인덱스의 사용여부 모니터링

```
ALTER INDEX IDX_01_TEST_TEST1 MONITORING USAGE;

SET AUTOT OFF
SELECT USED FROM V$OBJECT_USAGE;
```

```
SQL> ALTER INDEX IDX_01_TEST_TEST1 MONITORING USAGE;

Index 'IDX_01_TEST_TEST1' altered.
```

```
SQL> SET AUTOT OFF
SQL> SELECT USED FROM V$OBJECT_USAGE;

USED
----
N

1 row selected.
```

23. VIEW

23.1. 수행

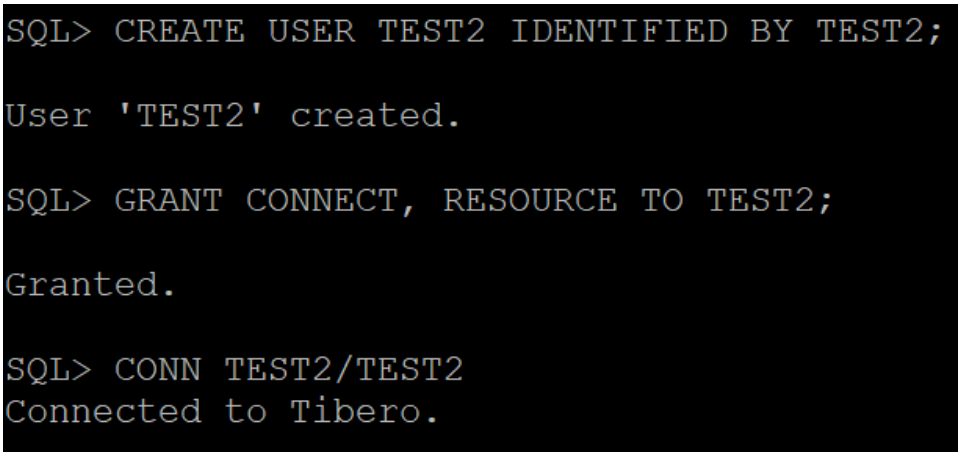
수행내역
TEST2 사용자 생성
EMP,DEPT 테이블 생성
데이터 입력 및 조회
EMP테이블의 뷰 생성
조인 테이블의 뷰 생성
뷰 변경
뷰 정보 조회

23.2. 결과

1. TEST2 사용자 생성

```
CREATE USER TEST2 IDENTIFIED BY TEST2;
GRANT CONNECT, RESOURCE TO TEST;

CONN TEST2/TEST2
```



2. EMP,DEPT 테이블 생성

```
CREATE TABLE TEST2.DEPT
  (
      DEPTNO    NUMBER PRIMARY KEY,
      DEPTNAME  VARCHAR(20),
      PDEPTNO   NUMBER
  )
  TABLESPACE my_space
  PCTFREE 5 INITRANS 3;

CREATE TABLE TEST2.EMP
  (
      EMPNO     NUMBER PRIMARY KEY,
      ENAME     VARCHAR(16) NOT NULL,
      ADDR      VARCHAR(24),
      SALARY    NUMBER,
      DEPTNO    NUMBER,
      CHECK (SALARY >= 10000),
      FOREIGN KEY (DEPTNO) REFERENCES DEPT(DEPTNO)
  )
  TABLESPACE my_space
  PCTFREE 5 INITRANS 3;
```

```
SQL> CREATE TABLE TEST2.DEPT
  (
      DEPTNO    NUMBER PRIMARY KEY,
      DEPTNAME  VARCHAR(20),
      PDEPTNO   NUMBER
  )
  TABLESPACE my_space
  PCTFREE 5 INITRANS 3;   2    3    4    5    6    7

Table 'TEST2.DEPT' created.

SQL> CREATE TABLE TEST2.EMP
  (
      EMPNO     NUMBER PRIMARY KEY,
      ENAME     VARCHAR(16) NOT NULL,
      ADDR      VARCHAR(24),
      SALARY    NUMBER,
      DEPTNO    NUMBER,
      CHECK (SALARY >= 10000),
      FOREIGN KEY (DEPTNO) REFERENCES DEPT(DEPTNO)
  )
  TABLESPACE my_space
  PCTFREE 5 INITRANS 3;   2    3    4    5    6    7

Table 'TEST2.EMP' created.
```

3. 데이터 입력 및 조회

```
INSERT INTO TEST2.DEPT VALUES(1,'AAA',10);
INSERT INTO TEST2.DEPT VALUES(2,'BBB',20);
INSERT INTO TEST2.DEPT VALUES(3,'CCC',30);

INSERT INTO TEST2.EMP VALUES(10,'A','SEOUL',15000,1);
INSERT INTO TEST2.EMP VALUES(11,'B','YONGIN',20000,2);
INSERT INTO TEST2.EMP VALUES(12,'C','INCHEON',10000,3);

SELECT * FROM TEST2.DEPT;
SELECT * FROM TEST2.EMP;
```

```
SQL> SELECT * FROM TEST2.DEPT;

    DEPTNO DEPTNAME                PDEPTNO
---------- -------------------- ----------
         1 AAA                          10
         2 BBB                          20
         3 CCC                          30

3 rows selected.
```

```
SQL> SELECT * FROM TEST2.EMP;

     EMPNO ENAME           ADDR                      SALARY     DEPTNO
---------- --------------- ------------------------ ---------- ----------
        10 A               SEOUL                     15000          1
        11 B               YONGIN                    20000          2
        12 C               INCHEON                   10000          3

3 rows selected.
```

4. EMP테이블의 뷰 생성

```
CREATE VIEW TEST2.MANAGER AS
       SELECT * FROM TEST2.EMP
       WHERE DEPTNO = 1;

SELECT * FROM TEST2.MANAGER;
```

```
SQL> CREATE VIEW TEST2.MANAGER AS
  2  SELECT * FROM TEST2.EMP
  3  WHERE DEPTNO=1;

View 'TEST2.MANAGER' created.
```

```
SQL> SELECT * FROM TEST2.MANAGER;

     EMPNO ENAME           ADDR                      SALARY     DEPTNO
---------- --------------- ------------------------ ---------- ----------
        10 A               SEOUL                     15000          1

1 row selected.
```

5. 조인 테이블의 뷰 생성

```
CREATE VIEW TEST2.EMP_DEPT AS
SELECT E.EMPNO, E.ENAME, E.SALARY, D.DEPTNO, D.PDEPTNO
FROM EMP E, DEPT D
WHERE E.DEPTNO=D.DEPTNO;
```

```
SELECT * FROM TEST2.EMP_DEPT;
```

```
SQL> CREATE VIEW TEST2.EMP_DEPT AS
  2 SELECT E.EMPNO, E.ENAME, E.SALARY, D.DEPTNO, D.PDEPTNO
  3 FROM EMP E, DEPT D
  4 WHERE E.DEPTNO=D.DEPTNO;

View 'TEST2.EMP_DEPT' created.
```

```
SQL> SELECT * FROM TEST2.EMP_DEPT;

     EMPNO ENAME                SALARY     DEPTNO    PDEPTNO
---------- ---------------- ---------- ---------- ----------
        10 A                     15000          1         10
        11 B                     20000          2         20
        12 C                     10000          3         30

3 rows selected.
```

6. 뷰 변경

```
CREATE OR REPLACE VIEW MANAGER AS
SELECT * FROM TEST2.EMP
WHERE DEPTNO=2;

SELECT * FROM MANAGER;
```

```
SQL> CREATE OR REPLACE VIEW MANAGER AS
  2 SELECT * FROM TEST2.EMP
  3 WHERE DEPTNO=2;

View 'MANAGER' created.

SQL> SELECT * FROM MANAGER;

     EMPNO ENAME           ADDR                      SALARY     DEPTNO
---------- ---------------- ------------------------ ---------- ----------
        11 B                YONGIN                      20000          2

1 row selected.
```

7. 뷰 정보 조회

```
CONN SYS/TIBERO
--뷰 정보 조회 권한 부여
GRANT SELECT ON DBA_VIEWS TO TEST2;
CONN TEST2/TEST2

SET LINESIZE 2000
COL OWNER FOR A20
```

```
COL VIEW_NAME FOR A20
SELECT * FROM DBA_VIEWS WHERE OWNER='TEST2';
```

```
SQL> SET LINESIZE 2000
SQL> SELECT * FROM DBA_VIEWS WHERE OWNER='TEST2';

OWNER                VIEW_NAME            TEXT
-------------------- -------------------- --------------------------------------------------------------------------------
TEST2                MANAGER              SELECT "EMPNO", "ENAME", "ADDR", "SALARY", "DEPTNO" FROM TEST2.EMP
                                          WHERE DEPTNO=

TEST2                EMP_DEPT             SELECT E.EMPNO, E.ENAME, E.SALARY, D.DEPTNO, D.PDEPTNO
                                          FROM EMP E, DEPT D
                                          WHERE


2 rows selected.
```

24. SEQUENCE

24.1. 수행

수행내역
TEST2에 시퀀스 생성 권한 부여
시퀀스의 생성 (증가 시퀀스)
시퀀스의 생성(감소 시퀀스)
시퀀스 값으로 데이터 추가 - NEXTVAL
시퀀스 값으로 데이터 추가 - CURRVAL
시퀀스 변경후 확인
시퀀스 정보 조회

24.2. 결과

1. TEST2에 시퀀스 생성 권한 부여

```
CREATE USER TEST2 IDENTIFIED BY TEST2;
GRANT CONNECT, RESOURCE TO TEST2;
GRANT CREATE SEQUENCE TO TEST2;
CONN TEST2/TEST2
```

```
SQL> CREATE USER TEST2 IDENTIFIED BY TEST2;

User 'TEST2' created.

SQL> GRANT CONNECT, RESOURCE TO TEST2;

Granted.

SQL> GRANT CREATE SEQUENCE TO TEST2;

Granted.

SQL> CONN TEST2/TEST2
Connected to Tibero.
```

2. 시퀀스의 생성 (증가 시퀀스)

```
CREATE SEQUENCE IN_NEW_ID
MINVALUE 1000
MAXVALUE 9999
INCREMENT BY 10
CACHE 100
NOCYCLE;
```

```
SQL> CREATE SEQUENCE IN_NEW_ID
  2 MINVALUE 1000
  3 MAXVALUE 9999
  4 INCREMENT BY 10
  5 CACHE 100
  6 NOCYCLE;

Sequence 'IN_NEW_ID' created.
```

3. 시퀀스의 생성 (감소 시퀀스)

```
CREATE SEQUENCE DE_NEW_ID
MINVALUE 1000
MAXVALUE 9999
INCREMENT BY -10
CACHE 100
NOCYCLE;
```

4. 시퀀스 값으로 데이터 추가 - NEXTVAL

```
CREATE TABLE TEST2.EMP_ID (ID NUMBER, NAME VARCHAR(30));
INSERT INTO TEST2.EMP_ID VALUES(IN_NEW_ID.NEXTVAL,'JMP');
INSERT INTO TEST2.EMP_ID VALUES(IN_NEW_ID.NEXTVAL,'SYJ');
SELECT * FROM TEST2.EMP_ID;
```

```
SQL> SELECT * FROM TEST2.EMP_ID;

        ID NAME
---------- ------------------------------
      1000 JMP
      1010 SYJ

2 rows selected.
```

5. 시퀀스 값으로 데이터 추가 - CURRVAL

```
INSERT INTO TEST2.EMP_ID VALUES(IN_NEW_ID.CURRVAL,'JWP');
SELECT * FROM TEST2.EMP_ID;
```

```
SQL> INSERT INTO TEST2.EMP_ID VALUES(IN_NEW_ID.CURRVAL,'JWP');

1 row inserted.

SQL> SELECT * FROM TEST2.EMP_ID;

        ID NAME
---------- ------------------------------
      1000 JMP
      1010 SYJ
      1010 JWP

3 rows selected.
```

6. 시퀀스 변경후 확인

```
ALTER SEQUENCE IN_NEW_ID
MAXVALUE 99999
INCREMENT BY 1
CACHE 200;

INSERT INTO TEST2.EMP_ID VALUES(IN_NEW_ID.NEXTVAL,'AAA');
SELECT * FROM TEST2.EMP_ID;
```

```
SQL> ALTER SEQUENCE IN_NEW_ID
   2 MAXVALUE 99999
   3 INCREMENT BY 1
   4 CACHE 200;

Sequence 'IN_NEW_ID' altered.

SQL> INSERT INTO TEST2.EMP_ID VALUES(IN_NEW_ID.NEXTVAL,'AAA');

1 row inserted.

SQL> SELECT * FROM TEST2.EMP_ID;

        ID NAME
---------- --------------------------------
      1000 JMP
      1010 SYJ
      1010 JWP
      1011 AAA
```

7. 시퀀스 정보 조회

```
CONN SYS/TIBERO
/*시퀀스 정보 조회 권한 부여*/
GRANT SELECT ON DBA_SEQUENCES TO TEST2;
CONN TEST2/TEST2

SET LINESIZE 2000
COL SEQUENCE_OWNER FOR A20
COL SEQUENCE_NAME FOR A20
SELECT * FROM DBA_SEQUENCES WHERE SEQUENCE_OWNER='TEST2';
```

```
SQL> SET LINESIZE 2000
SQL> COL SEQUENCE_OWNER FOR A20
SQL> COL SEQUENCE_NAME FOR A20
SQL> SELECT * FROM DBA_SEQUENCES WHERE SEQUENCE_OWNER='TEST2';

SEQUENCE_OWNER       SEQUENCE_NAME         MIN_VALUE  MAX_VALUE INCREMENT_BY CYCLE_FLAG ORDER_FLAG IF_AVAIL CACHE_SIZE LAST_NUMBER
-------------------- -------------------- ---------- ---------- ------------ ---------- ---------- -------- ---------- -----------
TEST2                DE_NEW_ID                  1000       9999          -10 N          N          Y               100        9999
TEST2                IN_NEW_ID                  1000      99999            1 N          N          Y               200        1211

2 rows selected.
```

25. 트리거

25.1. 수행

수행내역
TEST2에 트리거 생성 권한 부여
TEST2에 상품, 입고, 판매 테이블 생성
상품 테이블에 데이터 삽입
입고 테이블에 상품이 입고 되면 상품 테이블에 상품의 재고 수량 수정되는 트리거 작성
입고 테이블에 데이터 삽입
조회하기

25.2. 결과

1. TEST2에 트리거 생성 권한 부여

```
CREATE USER TEST2 IDENTIFIED BY TEST2;
GRANT CONNECT, RESOURCE TO TEST2;
GRANT CREATE TRIGGER TO TEST2;
CONN TEST2/TEST2
```

2. TEST2에 상품, 입고, 판매 테이블 생성

```
/*상품 테이블*/
CREATE TABLE TEST2.PRODUCT(
CODE VARCHAR(6) NOT NULL PRIMARY KEY,
NAME VARCHAR2(30) NOT NULL
,COMP VARCHAR2(30) NOT NULL,
PRICE NUMBER,
AMOUNT NUMBER DEFAULT 0);

/*입고 테이블*/
CREATE TABLE TEST2.IN_PRO(
IN_NO NUMBER PRIMARY KEY,
IN_CODE VARCHAR2(6) NOT NULL CONSTRAINT FK_IBGO_NO REFERENCES TEST2.PRODUCT(CODE),
IN_DATE DATE,
IN_AMOUNT NUMBER,
IN_PRICE NUMBER);

/*판매 테이블*/
CREATE TABLE TEST2.SALE(
S_NO NUMBER PRIMARY KEY,
S_CODE VARCHAR2(6) NOT NULL CONSTRAINT FK_PAN_NO REFERENCES TEST2.PRODUCT(CODE)
 ,S_DATE DATE,
S_AMOUNT NUMBER,
S_PRICE NUMBER);
```

```
16:54:44 SQL> CREATE TABLE TEST2.PRODUCT(
16:54:53    2 CODE VARCHAR(6) NOT NULL PRIMARY KEY,
16:55:07    3 NAME VARCHAR2(30) NOT NULL
16:55:15    4 ,COMP VARCHAR2(30) NOT NULL,
16:55:34    5 PRICE NUMBER,
16:55:39    6 AMOUNT NUMBER DEFAULT 0);

Table 'TEST2.PRODUCT' created.

16:55:45 SQL> CREATE TABLE TEST2.IN_PRO(
16:55:57    2 NO NUMBER PRIMARY KEY,
16:56:03    3 CODE VARCHAR2(6) NOT NULL CONSTRAINT FK_IBGO_NO REFERENCES TEST2.PROD
UCT(CODE),
16:57:30    4 IN_DATE DATE,
16:57:36    5 AMOUNT NUMBER,
16:57:40    6 IN_PRICE NUMBER);

Table 'TEST2.IN_PRO' created.
```

```
16:59:05 SQL> CREATE TABLE TEST2.SALE(
NO NUMBER PRIMARY KEY,
CODE VARCHAR2(6) NOT NULL CONSTRAINT FK_PAN_NO REFERENCES TEST2.PRODUCT(CODE)
 ,S_DATE DATE,
AMOUNT NUMBER,
S_PRICE NUMBER);16:59:28    2 16:59:28    3 16:59:28    4 16:59:28    5 16:59:28
 6

Table 'TEST2.SALE' created.
```

3. 상품 테이블에 데이터 삽입

```
INSERT INTO TEST2.PRODUCT VALUES('AAA','AA','A',10000,'');
INSERT INTO TEST2.PRODUCT VALUES('BBB','BB','B',15000,'');
INSERT INTO TEST2.PRODUCT VALUES('CCC','CC','C',6000,'');
INSERT INTO TEST2.PRODUCT VALUES('DDD','DD','D',5000,'');
INSERT INTO TEST2.PRODUCT VALUES('EEE','EE','E',2000,'');
COMMIT;

COL COMP FOR A10
COL NAME FOR A10
SELECT * FROM TEST2.PRODUCT;
```

```
17:02:06 SQL> COL COMP FOR A10
17:05:13 SQL> COL NAME FOR A10
17:05:17 SQL> SELECT * FROM TEST2.PRODUCT;

CODE   NAME       COMP            PRICE     AMOUNT
------ ---------- ---------- ---------- ----------
AAA    AA         A                10000
BBB    BB         B                15000
CCC    CC         C                 6000
DDD    DD         D                 5000
EEE    EE         E                 2000

5 rows selected.
```

4. 입고 테이블에 상품이 입고 되면 상품 테이블에 상품의 재고 수량 수정되는 트리거 작성 - ERROR

```
CREATE OR REPLACE TRIGGER INSIPGO
AFTER
INSERT ON TEST2.IN_PRO
FOR EACH ROW
BEGIN
UPDATE TEST2.PRODUCT
SET AMOUNT=AMOUNT+:=NEW.AMOUNT
WHERE CODE=:=NEW.CODE;
END;
/
```

```
SQL> SET SERVEROUTPUT ON
SQL> CREATE OR REPLACE TRIGGER INSIPGO
AFTER
INSERT ON TEST2.IN_PRO
FOR EACH ROW
BEGIN
UPDATE TEST2.PRODUCT
SET AMOUNT=AMOUNT+:=NEW.AMOUNT
WHERE CODE=:=NEW.CODE;
END;
/   2    3    4    5    6    7    8    9   10

Warning: TRIGGER created with compilation errors.
```

26. 파티션

- ALTER TABLE 문에 의해 새로 만들어진 파티션은 기존의 마지막 파티션의 범위보다 높은 범위여야 함

26.1. 수행

수행내역
TABLESPACE 3개 생성
PARTITION 생성
PARTITION 추가 후 확인
PARTITION 제거 후 확인

26.2. 결과

1. TABLESPACE 3개 생성

```
CREATE TABLESPACE TEST_PART1 DATAFILE 'TEST_PART1.DBF' SIZE 100M;
CREATE TABLESPACE TEST_PART2 DATAFILE 'TEST_PART2.DBF' SIZE 100M;
CREATE TABLESPACE TEST_PART3 DATAFILE 'TEST_PART3.DBF' SIZE 100M;
SELECT TABLESPACE_NAME FROM DBA_TABLESPACES;
```

```
SQL> CREATE TABLESPACE TEST_PART1 DATAFILE 'TEST_PART1.DBF' SIZE 100M;

Tablespace 'TEST_PART1' created.

SQL> CREATE TABLESPACE TEST_PART2 DATAFILE 'TEST_PART2.DBF' SIZE 100M;
C
Tablespace 'TEST_PART2' created.

SQL> CREATE TABLESPACE TEST_PART3 DATAFILE 'TEST_PART3.DBF' SIZE 100M;
TBS-70003: Invalid command. Enter HELP or HELP <command>.
at line 1, column 1:
CCREATE TABLESPACE TEST_PART3 DATAFILE 'TEST_PART3.DBF' SIZE 100M;
^^^^^^^

SQL> CREATE TABLESPACE TEST_PART3 DATAFILE 'TEST_PART3.DBF' SIZE 100M;

Tablespace 'TEST_PART3' created.
```

```
SQL> SELECT TABLESPACE_NAME FROM DBA_TABLESPACES;

TABLESPACE_NAME
--------------------------------------------------
SYSTEM
UNDO
TEMP
USR
SYSSUB
TEST_PART1
TEST_SPACE
TEST_SPACE_IDX
TEST_TEMP
TEST_PART2
TEST_PART3

11 rows selected.
```

2. PARTITION 생성

```
CREATE TABLE PARTITIONED_TABLE1(C1 NUMBER, C2 CLOB, C3 NUMBER)
PARTITION BY RANGE(C1, C3)
(PARTITION PART1 VALUES LESS THAN(30,40) TABLESPACE TEST_PART1,
PARTITION PART2 VALUES LESS THAN(50,60) TABLESPACE TEST_PART2,
PARTITION PART3 VALUES LESS THAN(60,70) TABLESPACE TEST_PART3);
```

```
SQL> CREATE TABLE PARTITIONED_TABLE1(C1 NUMBER, C2 CLOB, C3 NUMBER)
PARTITION BY RANGE(C1, C3)
(PARTITION PART1 VALUES LESS THAN(30,40) TABLESPACE TEST_PART1,
PARTITION PART2 VALUES LESS THAN(50,60) TABLESPACE TEST_PART2,
PARTITION PART3 VALUES LESS THAN(60,70) TABLESPACE TEST_PART3);   2

Table 'PARTITIONED_TABLE1' created.
```

3. PARTITION 추가 후 확인

```
ALTER TABLE PARTITIONED_TABLE1 ADD PARTITION PART4
VALUES LESS THAN(70,80);

COL TABLE_NAME FOR A20
COL PARTITION_NAME FOR A20
SELECT TABLE_NAME, PARTITION_NAME FROM USER_TAB_PARTITIONS WHERE
TABLE_NAME='PARTITIONED_TABLE1';
```

```
SQL> COL TABLE_NAME FOR A20
SQL> COL PARTITION_NAME FOR A20
SQL> SELECT TABLE_NAME, PARTITION_NAME FROM USER_TAB_PARTITIONS WHERE TABLE_NAME='P
ARTITIONED_TABLE1';

TABLE_NAME           PARTITION_NAME
-------------------- --------------------
PARTITIONED_TABLE1   PART4
PARTITIONED_TABLE1   PART1
PARTITIONED_TABLE1   PART2
PARTITIONED_TABLE1   PART3

4 rows selected.
```

4. PARTITION 제거 후 확인

```
ALTER TABLE PARTITIONED_TABLE1 DROP PARTITION PART4;
SELECT TABLE_NAME, PARTITION_NAME FROM USER_TAB_PARTITIONS WHERE
TABLE_NAME='PARTITIONED_TABLE1';
```

```
SQL> ALTER TABLE PARTITIONED_TABLE1 DROP PARTITION PART4;

Table 'PARTITIONED_TABLE1' altered.

SQL> SELECT TABLE_NAME, PARTITION_NAME FROM USER_TAB_PARTITIONS WHERE TABLE_NAME='P
ARTITIONED_TABLE1';

TABLE_NAME           PARTITION_NAME
-------------------- --------------------
PARTITIONED_TABLE1   PART1
PARTITIONED_TABLE1   PART2
PARTITIONED_TABLE1   PART3

3 rows selected.
```

27. 파티션 - ERROR

- ALTER TABLE 문에 의해 새로 만들어진 파티션은 기존의 마지막 파티션의 범위보다 높은 범위여야 함

26.1. 수행

수행내역
TABLESPACE 3개 생성
PARTITION 생성
PARTITION 추가

26.2. 결과

1. TABLESPACE 3개 생성

```
CREATE TABLESPACE TEST_PART1 DATAFILE 'TEST_PART1.DBF' SIZE 100M;
CREATE TABLESPACE TEST_PART2 DATAFILE 'TEST_PART2.DBF' SIZE 100M;
CREATE TABLESPACE TEST_PART3 DATAFILE 'TEST_PART3.DBF' SIZE 100M;
SELECT TABLESPACE_NAME FROM DBA_TABLESPACES;
```

```
SQL> CREATE TABLESPACE TEST_PART1 DATAFILE 'TEST_PART1.DBF' SIZE 100M;

Tablespace 'TEST_PART1' created.

SQL> CREATE TABLESPACE TEST_PART2 DATAFILE 'TEST_PART2.DBF' SIZE 100M;
C
Tablespace 'TEST_PART2' created.

SQL> CREATE TABLESPACE TEST_PART3 DATAFILE 'TEST_PART3.DBF' SIZE 100M;
TBS-70003: Invalid command. Enter HELP or HELP <command>.
at line 1, column 1:
CCREATE TABLESPACE TEST_PART3 DATAFILE 'TEST_PART3.DBF' SIZE 100M;
^^^^^^^

SQL> CREATE TABLESPACE TEST_PART3 DATAFILE 'TEST_PART3.DBF' SIZE 100M;

Tablespace 'TEST_PART3' created.
```

2. PARTITION 생성

```
CREATE TABLE PARTITIONED_TABLE2 (C1 NUMBER, C2 CLOB, C3 NUMBER)
PARTITION BY RANGE (C1, C3)
(
PARTITION PART1 VALUES LESS THAN (50, 20) TABLESPACE TEST_PART1,
PARTITION PART2 VALUES LESS THAN (30, 10) TABLESPACE TEST_PART2,
PARTITION DEF_PART VALUES LESS THAN (MAXVALUE, MAXVALUE) TABLESPACE TEST_PART3
);
```

```
SQL> CREATE TABLE PARTITIONED_TABLE2 (C1 NUMBER, C2 CLOB, C3 NUMBER)
             PARTITION BY RANGE (C1, C3)
             (
                PARTITION PART1 VALUES LESS THAN (50, 20),
                PARTITION PART2 VALUES LESS THAN (30, 10),
                PARTITION DEF_PART VALUES LESS THAN (MAXVALUE, MAXVALUE)
             );   2    3    4    5    6    7
TBR-7163: Specified partition values are incorrect.
```

- VALUES LESS THAN : '이전 PARTITION에 들어가지 않고 ~ 보다 작은 값을 갖는 데이터를 포함하는 파티션'
  이므로 앞 파티션의 범위가 뒤 파티션을 포함하면 안됨

3. PARTITION 추가

```
CREATE TABLE PARTITIONED_TABLE3 (C1 NUMBER, C2 CLOB, C3 NUMBER)
 PARTITION BY RANGE (C1, C3)
(
PARTITION PART1 VALUES LESS THAN (50, 20) TABLESPACE TEST_PART1,
PARTITION PART2 VALUES LESS THAN (60, 70) TABLESPACE TEST_PART2
);

ALTER TABLE PARTITIONED_TABLE3 ADD PARTITION PART3
VALUES LESS THAN (80, 40);

ALTER TABLE PARTITIONED_TABLE3 ADD PARTITION PART3
VALUES LESS THAN (70, 100);
```

```
SQL> CREATE TABLE PARTITIONED_TABLE3 (C1 NUMBER, C2 CLOB, C3 NUMBER)
             PARTITION BY RANGE (C1, C3)
             (
                PARTITION PART1 VALUES LESS THAN (50, 20),
                PARTITION PART2 VALUES LESS THAN (60, 70)
             );   2    3    4    5    6

Table 'PARTITIONED_TABLE3' created.

SQL> ALTER TABLE PARTITIONED_TABLE3 ADD PARTITION PART3
            VALUES LESS THAN (80, 40);   2

Table 'PARTITIONED_TABLE3' altered.

SQL> ALTER TABLE PARTITIONED_TABLE3 ADD PARTITION PART3
            VALUES LESS THAN (70, 100);   2
TBR-7163: Specified partition values are incorrect.
```

- 기존 PART3 파티션의 범위는 40~80이므로 80보다 높은 범위로 만들어야함

28. 인덱스 파티션

28.1. 수행

수행내역
TABLESPACE 3개 생성
로컬 파티션 생성
글로벌 파티션 생성
인덱스 파티션 확인

28.2. 결과

1. TABLESPACE 3개 생성

```
CREATE TABLESPACE TEST_PART1 DATAFILE 'TEST_PART1.DBF' SIZE 100M;
CREATE TABLESPACE TEST_PART2 DATAFILE 'TEST_PART2.DBF' SIZE 100M;
CREATE TABLESPACE TEST_PART3 DATAFILE 'TEST_PART3.DBF' SIZE 100M;
SELECT TABLESPACE_NAME FROM DBA_TABLESPACES;
```

```
SQL> CREATE TABLESPACE TEST_PART1 DATAFILE 'TEST_PART1.DBF' SIZE 100M;

Tablespace 'TEST_PART1' created.

SQL> CREATE TABLESPACE TEST_PART2 DATAFILE 'TEST_PART2.DBF' SIZE 100M;
C
Tablespace 'TEST_PART2' created.

SQL> CREATE TABLESPACE TEST_PART3 DATAFILE 'TEST_PART3.DBF' SIZE 100M;
TBS-70003: Invalid command. Enter HELP or HELP <command>.
at line 1, column 1:
CCREATE TABLESPACE TEST_PART3 DATAFILE 'TEST_PART3.DBF' SIZE 100M;
^^^^^^^

SQL> CREATE TABLESPACE TEST_PART3 DATAFILE 'TEST_PART3.DBF' SIZE 100M;

Tablespace 'TEST_PART3' created.
```

2. 로컬 파티션 생성

- 테이블 파티션에 들어가는 키로 파티션을 나누는 방법
- 테이블의 한 파티션과 1:1 대응

```
CREATE TABLE PARTITIONED_TABLE1 (C1 NUMBER, C2 CLOB, C3 NUMBER)
PARTITION BY RANGE (C1, C3)
(
PARTITION PART1 VALUES LESS THAN (30, 40) TABLESPACE TEST_PART1,
PARTITION PART2 VALUES LESS THAN (50, 60) TABLESPACE TEST_PART2,
PARTITION DEF_PART VALUES LESS THAN (MAXVALUE, MAXVALUE) TABLESPACE TEST_PART3
);

CREATE INDEX PARTITIONED_INDEX1 ON PARTITIONED_TABLE1 (C1) LOCAL
(
PARTITION IPART1 INITRANS 3,
PARTITION IPART2 PCTFREE 10,
PARTITION IPART3
);
```

```
SQL> CREATE TABLE PARTITIONED_TABLE1 (C1 NUMBER, C2 CLOB, C3 NUMBER)
PARTITION BY RANGE (C1, C3)
(
PARTITION PART1 VALUES LESS THAN (30, 40) TABLESPACE TEST_PART1,
PARTITION PART2 VALUES LESS THAN (50, 60) TABLESPACE TEST_PART2,
PARTITION DEF_PART VALUES LESS THAN (MAXVALUE, MAXVALUE) TABLESPACE TEST_PART3
);   2    3    4    5    6    7

Table 'PARTITIONED_TABLE1' created.

SQL> CREATE INDEX PARTITIONED_INDEX1 ON PARTITIONED_TABLE1 (C1) LOCAL
(
PARTITION IPART1 INITRANS 3,
PARTITION IPART2 PCTFREE 10,
PARTITION IPART3
);   2    3    4    5    6

Index 'PARTITIONED_INDEX1' created.
```

3. 글로벌 파티션 생성

- 테이블과는 무관하게 인덱스에 따로 파티션 설정

```
CREATE TABLE PARTITIONED_TABLE1 (C1 NUMBER, C2 CLOB, C3 NUMBER)
PARTITION BY RANGE (C1, C3)
(
PARTITION PART1 VALUES LESS THAN (30, 40) TABLESPACE TEST_PART1,
PARTITION PART2 VALUES LESS THAN (50, 60) TABLESPACE TEST_PART2,
PARTITION DEF_PART VALUES LESS THAN (MAXVALUE, MAXVALUE) TABLESPACE TEST_PART3
);

CREATE INDEX PARTITIONED_INDEX2 ON PARTITIONED_TABLE1 (C3, C1)
GLOBAL PARTITION BY RANGE (C3)
(
PARTITION IPART1 VALUES LESS THAN (20) INITRANS 3,
PARTITION IPART2 VALUES LESS THAN (70) PCTFREE 10,
PARTITION IPART3 VALUES LESS THAN (MAXVALUE)
);
```

```
SQL> CREATE TABLE PARTITIONED_TABLE1 (C1 NUMBER, C2 CLOB, C3 NUMBER)
PARTITION BY RANGE (C1, C3)
(
PARTITION PART1 VALUES LESS THAN (30, 40) TABLESPACE TEST_PART1,
PARTITION PART2 VALUES LESS THAN (50, 60) TABLESPACE TEST_PART2,
PARTITION DEF_PART VALUES LESS THAN (MAXVALUE, MAXVALUE) TABLESPACE TEST_PART3
);   2    3    4    5    6    7

Table 'PARTITIONED_TABLE1' created.
```

```
SQL> CREATE INDEX PARTITIONED_INDEX2 ON PARTITIONED_TABLE1 (C3, C1)
GLOBAL PARTITION BY RANGE (C3)
(
PARTITION IPART1 VALUES LESS THAN (20) INITRANS 3,
PARTITION IPART2 VALUES LESS THAN (70) PCTFREE 10,
PARTITION IPART3 VALUES LESS THAN (MAXVALUE)
);   2    3    4    5    6    7

Index 'PARTITIONED_INDEX2' created.
```

4.인덱스 파티션 확인

```
SELECT OWNER, INDEX_NAME, TABLE_NAME FROM DBA_PART_INDEXES WHERE OWNER='TEST';
```

```
SQL> SELECT OWNER, INDEX_NAME, TABLE_NAME FROM DBA_PART_INDEXES WHERE OWNER='TEST';

OWNER
--------------------------------------------------------------------------------
INDEX_NAME
--------------------------------------------------------------------------------
TABLE_NAME
--------------------
TEST
PARTITIONED_INDEX1
PARTITIONED_TABLE1

TEST
PARTITIONED_INDEX2
PARTITIONED_TABLE1

TEST
_TEST_LIDX284200
PARTITIONED_TABLE3

TEST
_TEST_LIDX285500
PARTITIONED_TABLE1


4 rows selected.
```