**[문제1] PT7-15, PT7-11 ~ PT7-15 예를 사용하여 학생(학번, 이름, 학과코드) 테이블을 생성하고 3명을 입력, 학과(학과코드, 학과명) 테이블에서 3행을 입력한 후 PT7-11 ~ PT7-15 예와 동일 내용으로 SQL을 Test 하고 결과를 확인할 것.**

SQL> create table c\_test1(

2 no NUMBER,

3 name VARCHAR2(6),

4 deptno NUMBER);

Table created.

SQL> CREATE TABLE c\_test2 (

2 no NUMBER,

3 name VARCHAR2(10) ) ;

Table created.

**\*일단 두개의 테이블을 생성합니다.**

SQL> alter table c\_test1

2 add constraint ctest1\_deptno\_fk FOREIGN KEY(deptno)

3 references c\_test2(no);

references c\_test2(no)

\*

ERROR at line 3:

ORA-02270: no matching unique or primary key for this column-list

**유니크 제약조건이 걸려있지 않거나 기본키(유니크 제약조건 + Not NULL)가 아니면 외래키로 지정이 불가능.**

**그래서 유니크 제약조건을 걸어줍니다.**

SQL> ALTER TABLE c\_test1

2 ADD CONSTRAINT ctest1\_deptno\_fk FOREIGN KEY(deptno)

3 REFERENCES c\_test2(no)

4 ON DELETE CASCADE ;

Table altered.

**ON DELETE CASCADE로 연관관계의 주인인 기본키인 no가 삭제되면 deptno가 외래키로 갖고있는 튜플이 다 삭제된다는 것을 확인.**

**\*c\_test2에 데이터 삽입**

SQL> INSERT INTO c\_test2 VALUES (10,'AAAA') ;

1 row created.

SQL> INSERT INTO c\_test2 VALUES (10,'BBBB') ;

INSERT INTO c\_test2 VALUES (10,'BBBB')

\*

ERROR at line 1:

ORA-00001: unique constraint (USER23.CTEST2\_NO\_UK) violated

**기본 키로 정한 10이 같이 들어가되면 유니크 제약조건을 포함하고 있는 PRIMARY KEY제약조건에 의해서 오류가 발생하는 것을 확인.**

SQL> INSERT INTO c\_test2 VALUES (20,'BBBB') ;

1 row created.

SQL> INSERT INTO c\_test2 VALUES (30,'CCCC') ;

1. row created.

**\*c\_test1에 데이터 삽입**

SQL> INSERT INTO c\_test1 VALUES (1,'apple', 10) ;

1 row created.

SQL> INSERT INTO c\_test1 VALUES (2,'banana', 20) ;

1 row created.

SQL> INSERT INTO c\_test1 VALUES (3,'cherry', 30) ;

1 row created.

SQL> INSERT INTO c\_test1 VALUES (4, 'peach', 40) ;

INSERT INTO c\_test1 VALUES (4, 'peach', 40)

\*

ERROR at line 1:

ORA-02291: integrity constraint (USER23.CTEST1\_DEPTNO\_FK) violated - parent key

not found

**외래키가 40인 데이터를 추가하려고 하면 에러가 납니다.**

**기본적으로 pk값에 한해서 와야 하는데 외래키가 이상한 값이 와버리니까 참조하지 못하는 값인 것입니다.**

SQL> DELETE FROM c\_test2 WHERE no =10;

1 row deleted.

**기본키가 10인 튜플에 대해 삭제를 강행하면 참조하고 있던 c\_test1의 튜플도 삭제가 됩니다.**

SQL> SELECT \* FROM c\_test1 ;

NO NAME DEPTNO

---------- ------ ----------

2 banana 20

3 cherry 30

**[실행결과]**

외래키를 지정할 때, 기본키를 기준으로 외래키를 생성할 수 있습니다. 외래키는 유니크 제약조건 + NOT NULL 제약조건이 둘중에 하나가 걸려 있을 때 지정이 가능하며 기본키가 걸려있는 튜플을 삭제할 때 연관된 외래키도 삭제가 됩니다.

여기서 중요한 점은 외래키 있는 튜플은 삭제하면 삭제되지 않으며 외래키가 있는 테이블에 기본키 중에 없는 외래키를 삽입하려고 하면 에러가 발생합니다.

**[문제2] PT7-19, PT7-16 ~ PT7-19의 예를 사용하여 SQL를 실행하고 결과가 동일한 지를 확인할 것.(제약조건 관리하기)**

**t\_novalidate 테이블 생성**

SQL> CREATE TABLE t\_novalidate(

2 no NUMBER PRIMARY KEY,

3 name VARCHAR(10) NOT NULL);

Table created.

**생성한 테이블에 데이터 삽입.**

SQL> INSERT INTO t\_novalidate VALUES(1, 'AAA') ;

1 row created.

SQL> INSERT INTO t\_novalidate VALUES(2, 'BBB') ;

1 row created.

SQL> INSERT INTO t\_novalidate VALUES(3, 'CCC') ;

1 row created.

**t\_validate 테이블 생성**

SQL> CREATE TABLE t\_validate

2 ( no NUMBER CONSTRAINT tv\_no\_pk PRIMARY KEY,

3 name VARCHAR2(10) CONSTRAINT tv\_name\_nn NOT NULL ) ;

Table created.

**생성한 테이블에 데이터 삽입.**

SQL> INSERT INTO t\_validate VALUES(1, 'AAA') ;

1 row created.

SQL> INSERT INTO t\_validate VALUES(2, 'BBB') ;

1 row created.

SQL> INSERT INTO t\_validate VALUES(3, 'CCC') ;

1 row created.

**t\_enable 테이블 생성**

SQL> CREATE TABLE t\_enable

2 ( no NUMBER CONSTRAINT te\_no\_pk PRIMARY KEY,

3 name VARCHAR2(10) CONSTRAINT te\_name\_nn NOT NULL ) ;

Table created.

SQL> INSERT INTO t\_enable VALUES(1,'AAA');

1 row created.

SQL> INSERT INTO t\_enable VALUES(2,'BBB');

1 row created.

SQL> INSERT INTO t\_enable VALUES(3,NULL);

INSERT INTO t\_enable VALUES(3,NULL)

\*

ERROR at line 1:

ORA-01400: cannot insert NULL into ("USER23"."T\_ENABLE"."NAME")

**\*역시 NOT NULL 제약조건이 걸려서 확인이 되지 않음.**

**=============제약 조건 DISABLE 하기==============**

SQL> INSERT INTO t\_novalidate VALUES(1,'DDD');

INSERT INTO t\_novalidate VALUES(1,'DDD')

\*

ERROR at line 1:

ORA-00001: unique constraint (USER23.SYS\_C0014873) violated

**기본키는 유일해야 하는데 중복된 값을 입력하려고 하면 에러가 발생합니다.**

SQL> ALTER TABLE t\_novalidate

2 DISABLE NOVALIDATE CONSTRAINT SYS\_C0014873 ;

Table altered.

**그러면 이상태에서 DISABLE NOVALIDATE CONSTRAINT SYS\_C0014873 ; 구문으로 해당 제약 조건을 disable 시켜줄 수 있습니다.**

SQL> INSERT INTO t\_novalidate VALUES(1,'DDD') ;

1 row created.

**이제 data가 들어가는 것을 확인할 수 있습니다.**

SQL> INSERT INTO t\_validate VALUES(4,NULL);

INSERT INTO t\_validate VALUES(4,NULL)

\*

ERROR at line 1:

ORA-01400: cannot insert NULL into ("USER23"."T\_VALIDATE"."NAME")

**\* name VARCHAR2(10) CONSTRAINT tv\_name\_nn NOT NULL ) 아까 이 구문으로 테이블을 생성할 때 not null 제약 조건을 지정해주었는데 null을 넣으려고 하니 에러가 발생했습니다.**

**\* 이번엔 disable 해주겠습니다.**

SQL> ALTER TABLE t\_validate

2 DISABLE VALIDATE CONSTRAINT tv\_name\_nn ;

Table altered.

**\*여기서는 system에서 알려주는 값과 다르게 tv\_name\_nn을 지정해주었는데 이는 제약조건의 이름을 정해준 것을 다시 쓰는 것 입니다.**

SQL> INSERT INTO t\_validate VALUES(4,NULL);

INSERT INTO t\_validate VALUES(4,NULL)

\*

ERROR at line 1:

ORA-25128: No insert/update/delete on table with constraint (USER23.TV\_NAME\_NN)

disabled and validated

**\*이번에는 데이터를 넣어주려고 했더니 데이터를 변경할 수 없게 하는 옵션이 지정되어있다고 나옵니다.**

**DISABLE VALIDATE 이게 해당 컬럼의 데이터를 변경하거나 삭제 삽입을 하지 못하게 하는 옵션 설정인 것 같습니다.**

**데이터 변경이 안되게끔 하는 옵션, 해당칼럼의 내용을 변경할 수없다.**

**insert, update, delete 작업을 수행할 수 없음**

**11g 에서의 read only의 개념과 같음**

INSERT INTO t\_validate VALUES(4,'DDD')

\*

ERROR at line 1:

ORA-25128: No insert/update/delete on table with constraint (USER23.TV\_NAME\_NN)

disabled and validated

**\*값을 주고 삽입을 해도 여전히 안되는 것을 확인할 수 있습니다.**

SQL> INSERT INTO t\_validate (no) VALUES(4) ;

INSERT INTO t\_validate (no) VALUES(4)

\*

ERROR at line 1:

ORA-25128: No insert/update/delete on table with constraint (USER23.TV\_NAME\_NN)

disabled and validated

**[실행결과]**

처음엔 DISABLE VALIDATE CONSTRAINT 제약조건 이름; 에대해서 헷갈렸는데 VALIDATE는 제약조건을 비활성화하고 삽입 수정 삭제를 못하게 끔 하는 조건이 걸립니다.

반면에 NOVALIDATE는 일단 강제로 뚫고 들어가 제약조건을 비활성화 시키는 역할을 한다는 것을 알았습니다.

**기본적으로 DISABLE만 써주고 VALIDATE OR NOVALIDATE 를 적어주지 않으면 NOVALIDATE를 기본으로 가져갑니다.**

**[문제3] PT7-23, PT7-20 ~ PT7-23의 예를 사용하여 SQL를 실행하고 결과가 동일한 지를 확인할 것.**

SQL> ALTER TABLE t\_enable DISABLE CONSTRAINT te\_name\_nn ;

Table altered.

SQL> INSERT INTO t\_enable VALUES(3,NULL);

1 row created.

**\* 입력이 잘 되어지는 것을 확인할 수 있습니다.**

**=====================ENABLE로 활성화 하기======================**

SQL> ALTER TABLE t\_enable

2 ENABLE NOVALIDATE CONSTRAINT te\_name\_nn ;

Table altered.

**\* 이렇게 제약조건을 활성화 시킬 수 있습니다.**

SQL> SELECT \* FROM t\_enable ;

NO NAME

---------- ----------

1 AAA

2 BBB

3

**\* 이렇게 ENABLE로 활성화 해주었는데 잘못된 데이터 튜플은 검사하지 않고 제약조건이 걸린 이 후 부터 검사하게 됩니다.**

SQL> INSERT INTO t\_enable VALUES(4,NULL);

INSERT INTO t\_enable VALUES(4,NULL)

\*

ERROR at line 1:

ORA-01400: cannot insert NULL into ("USER23"."T\_ENABLE"."NAME")

**다시 제약조건을 풀어줍니다.**

SQL> ALTER TABLE t\_enable DISABLE CONSTRAINT te\_name\_nn ;

Table altered.

**하지만 기존에 데이터도 문제가 있는지 검사하기 위해서**

SQL> ALTER TABLE t\_enable ENABLE VALIDATE CONSTRAINT te\_name\_nn ;

ALTER TABLE t\_enable ENABLE VALIDATE CONSTRAINT te\_name\_nn

\*

ERROR at line 1:

ORA-02293: cannot validate (USER23.TE\_NAME\_NN) - check constraint violated

**\* ENABLE VALIDATE CONSTRAINT te\_name\_nn 구문으로 문제가 있는지 파악할 수 있습니다.**

**[실행결과]**

여기서도 VALIDATE와 NOVALIDATE의 차이에 대해 알았습니다. NOVALIDATE는 기존에 데이터에 문제가 있는지 파악하지 않고 활성화 걸린 시점부터 삽입되어지는 데이터에 대해서만 체크합니다.

그리고 VALIDATE는 활성화 시킬 때, 기존 테이블 안에 데이터가 문제가 있는지 없는지 판단할 수 있습니다.

**[문제4] PT7-28~29, tcons 테이블, dept2 테이블, emp2 테이블을 생성하고, PT7-28~29에서 연습문제1, 2, 3, 4를 처리하기 위한 SQL를 실습하고 결과를 확인할 것.**

테이블이(가) 표시된 사진

자동 생성된 설명

**- dept2(dcode, dname)을 사전에 생성함(3개의 행을 입력)**

SQL> create table dept2 (decode VARCHAR(6), dname VARCHAR(20));

Table created.

SQL> insert into dept2 values (1, 'AAA');

1 row created.

SQL> insert into dept2 values (2, 'BBB');

1 row created.

SQL> insert into dept2 values (3, 'CCC');

1 row created.

SQL> ALTER TABLE dept2 ADD CONSTRAINT dept2\_uk UNIQUE(decode);

Table altered.

**\* 제약 조건으로 외래키로 가지고 오고 싶으면 UNIQUE가 걸려야 하는데 안걸려서 유니크 제약조건을 걸어줬습니다.**

SQL> create table tcons(

2 no NUMBER(5) CONSTRAINT tcons\_no\_pk PRIMARY KEY,

3 name VARCHAR2(20) CONSTRAINT tcons\_name\_nn NOT NULL,

4 jumina VARCHAR2(13) CONSTRAINT tcons\_jumin\_nn NOT NULL,

5 juminb VARCHAR2(13) CONSTRAINT tcons\_jumin\_uk UNIQUE,

6 area NUMBER(1) CONSTRAINT tcons\_area\_ck CHECK (area<5),

7 deptno VARCHAR(6) CONSTRAINT tcons\_deptno\_fk REFERENCES dept2(decode));

Table created.

**2. tcons 테이블의 name 컬럼이 emp2 테이블의 ename 컬럼의 값을 참조하도록 참조키 제약조건을 추가 설정하는 쿼리를 쓰세요.(tcons 테이블이 자식테이블입니다)**

**(단, emp2테이블은 emp테이블에서 empno, ename 컬럼 만을 복사하여 생성)**

SQL> create table emp2 AS select empno,ename from emp;

Table created.

SQL> ALTER TABLE emp2 ADD CONSTRAINT ename\_primary\_key PRIMARY KEY(ename);

Table altered.

SQL> ALTER TABLE tcons ADD CONSTRAINT emp\_tcons\_fr FOREIGN KEY(name) REFERENCES emp2(ename);

Table altered.

**테이블을 복사해서 EMP2를 만들고 추가하려고 했더니 에러가 났습니다. 뭐지 하고 보다가 유니크나, 기본키 제약조건이 없는 것을 인지하고 바로 이번엔 기본키 제약조건을 설정해준 다음, 외래키 제약조건을 걸어줬는데 되었습니다.**

**3. tcons 테이블의 jumin 컬럼에 만들어져 있는 제약조건을 "사용안함"으로 변경하되 해당 테이블의 데이터에 DML 까지 안되도록 변경하는 쿼리를 작성하라.**

**(제약조건 이름은 tcons\_jumin\_nn 임)**

SQL> ALTER TABLE tcons DISABLE VALIDATE CONSTRAINT tcons\_jumin\_nn;

Table altered.

**VALIDATE 변경, 삭제, 삽입이 되지 않는 read only와 비슷한 구조로 설정했습니다.**

**4. tcons 테이블에 설정되어 있는 제약조건 중 자신이 생성한 제약 조건들을 테이블명 , 컬럼명 ,제약조건명으로 검색하는 쿼리를 작성하라.**

SQL> COL table\_name format a10

SQL> COL constraint\_name format a20

SQL> COL column\_name format a15

SQL> SELECT table\_name, constraint\_name, column\_name FROM user\_cons\_columns WHERE table\_name='TCONS';

TABLE\_NAME CONSTRAINT\_NAME COLUMN\_NAME

---------- -------------------- ---------------

TCONS EMP\_TCONS\_FR NAME

TCONS TCONS\_DEPTNO\_FK DEPTNO

TCONS TCONS\_JUMIN\_UK JUMINB

TCONS TCONS\_NO\_PK NO

TCONS TCONS\_AREA\_CK AREA

TCONS TCONS\_JUMIN\_NN JUMINA

TCONS TCONS\_NAME\_NN NAME

**[실행결과]**

앞의 내용을 종합적으로 이해한 내용을 바탕으로 했습니다. 이상 없습니다.