

**DDL**





**DDL**



# DDL

## DDL

- 데이터 정의 언어
- 객체를 만들고, 수정하고, 삭제하는 구문
- CREATE(생성), ALTER(수정), DROP(삭제)

## 오라클 객체 종류

USER, TABLE, VIEW, SEQUENCE, INDEX, PACKAGE,  
PROCEDURAL, FUNCTION, TRIGGER, SYNONYM





**CREATE**



# CREATE

## CREATE

- DDL의 한 종류로 테이블이나 인덱스, 유저 등 다양한 데이터베이스 객체를 생성하는 구문

## 관리자 계정 과 사용자 계정

관리자 계정 : 데이터베이스의 생성과 관리를 담당하는 계정이며, 모든 권한과 책임을 가지는 계정

사용자 계정 : 데이터베이스에 대하여 질의, 갱신, 보고서 작성 등을 수행할 수 있는 계정으로 업무에 필요한 최소한의 권한만 가지는 것을 원칙으로 함



# CREATE

## CREATE – 사용자 만들기

※ USER를 생성은 관리자 계정으로만 가능

표현식

CREATE USER 사용자이름 IDENTIFIED BY 비밀번호;

[ex]

1) CREATE USER kh IDENTIFIED BY kh;

2) CREATE USER test01 IDENTIFIED BY 1234;

※ 관리자계정을 통해서 USER를 만들게 되면 유저는 생성이 되지만 아직 권한이 없어서 접속이 불가능



# CREATE

## CREATE – 사용자 만들기

- 관리자 계정에서 USER를 생성하게 되면 계정은 생성되었지만 권한이 없어서 접속이 불가능하므로 권한을 부여해 주어야 함
- 권한을 부여하거나 회수하는 것을 통해 DATABASE에 접근을 제어함
- 이 때 사용하는 것 구문이 DCL(Data Control Language)로 GRANT(권한 부여), REVOKE(권한해제)가 존재
- 일반적으로 ROLE을 통해서 권한을 부여하고 해제함

## 오라클 ROLE

1. CONNECT : 사용자가 데이터베이스 접속 가능하도록 하기 위한 CREATE SESSION 권한이 있는 ROLE
2. RESOURCE : CREATE구문을 통해 객체를 생성할 수 있는 권한과 INSERT,UPDATE,DELETE 구문을 사용할 수 있는 권한을 모아둔 ROLE



# CREATE

## CREATE – 사용자 만들기

### 권한 부여하기

관리자 계정에서 수행

[표현식]

GRANT 부여할ROLE TO 사용자이름;

[ex]

1) GRANT RESOURCE,CONNECT TO kh;

2) GRANT CONNECT,RESOURCE TO test01;

### 권한 회수

관리자 계정에서 수행

[표현식]

REVOKE 회수할ROLE FROM 사용자이름;

[ex]

1) REVOKE CONNECT FROM kh;

2) REVOKE RESOURCE FROM test01;





# CREATE

## CREATE – 테이블 만들기

### 1. 기본테이블 생성

[표현식]

CREATE TABLE 테이블명 (컬럼명 자료형(크기), 컬럼명 자료형(크기).....)

```
CREATE TABLE MEMBER(  
    MEMBER_ID VARCHAR2(20),  
    MEMBER_PW VARCHAR2(20),  
    MEMBER_NAME VARCHAR2(20)  
);
```

⚡ COLUMN_NAME	⚡ DATA_TYPE	⚡ NULLABLE	DATA_DEFAULT	⚡ COLUMN_ID	⚡ COMMENTS
MEMBER ID	VARCHAR2 (20 BYTE)	Yes	(null)	1	(null)
MEMBER PW	VARCHAR2 (20 BYTE)	Yes	(null)	2	(null)
MEMBER NAME	VARCHAR2 (20 BYTE)	Yes	(null)	3	(null)



# CREATE

## CREATE – 테이블 만들기

### 2. 생성된 테이블 컬럼에 주석 달기

[표현식]

COMMENT ON COLUMN 테이블명.컬럼명 IS '주석내용';

COMMENT ON COLUMN MEMBER.MEMBER\_ID IS '회원 아이디';  
COMMENT ON COLUMN MEMBER.MEMBER\_PW IS '회원 비밀번호';  
COMMENT ON COLUMN MEMBER.MEMBER\_NAME IS '회원 이름';

❖ COLUMN_NAME	❖ DATA_TYPE	❖ NULLABLE	DATA_DEFAULT	❖ COLUMN_ID	❖ COMMENTS
MEMBER ID	VARCHAR2 (20 BYTE)	Yes	(null)	1	회원 아이디
MEMBER PW	VARCHAR2 (20 BYTE)	Yes	(null)	2	회원 비밀번호
MEMBER NAME	VARCHAR2 (20 BYTE)	Yes	(null)	3	회원 이름



# CREATE

## 제약조건

- 테이블 작성 시 각 컬럼에 대한 기록에 대해 제약 조건 설정 가능
- 데이터 무결성을 지키기 위해 제한된 조건

제약조건	설명
NOT NULL	데이터에 NULL을 허용하지 않는다.
UNIQUE	중복된 값을 허용하지 않는다.
PRIMARY KEY	NULL을 허용하지 않고, 중복을 허용하지 않는다. 컬럼의 고유 식별자로 사용하기 위함
FOREIGN KEY	참조되는 테이블의 컬럼 값이 존재하면 허용한다.
CHECK	저장 가능한 데이터 값의 범위나 조건을 지정하여 설정한 값만 허용한다.



# CREATE

## 제약조건 – NOT NULL

- 해당 컬럼에 반드시 값이 기록되어야 하는 경우, 특정 컬럼 값을 저장하거나 수정할 때 NULL 값을 허용하지 않도록 컬럼레벨에서 제한

```
CREATE TABLE USER_NOCONS(  
  USER_NO NUMBER,  
  USER_ID VARCHAR2(20),  
  USER_PW VARCHAR2(30),  
  USER_NAME VARCHAR2(30),  
  GENDER CHAR(6),  
  PHONE CHAR(11)  
);
```

```
INSERT INTO USER_NOCONS  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');
```

```
INSERT INTO USER_NOCONS  
VALUES(2, NULL, NULL, NULL, '남자',  
'01012345678');
```

USE...	USER_ID	USER_PW	USER_NAME	GENDER	PHONE
1	user01	pass01	유저1	남자	01012345678
2	(null)	(null)	(null)	남자	01012345678

# CREATE

## 제약조건 – NOT NULL

```
CREATE TABLE USER_NOTNULL(  
  USER_NO NUMBER,  
  USER_ID VARCHAR2(20) NOT NULL,  
  USER_PW VARCHAR2(30) NOT NULL,  
  USER_NAME VARCHAR2(30) NOT NULL,  
  GENDER CHAR(6),  
  PHONE CHAR(11)  
);
```

```
INSERT INTO USER_NOTNULL  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');
```

```
INSERT INTO USER_NOTNULL  
VALUES(2, NULL, NULL, NULL, '남자',  
'01012345678');
```

```
INSERT INTO USER_NOTNULL VALUES (2, NULL, NULL, NULL, '남자', '01012345678')  
오류 보고 -  
ORA-01400: cannot insert NULL into ("TEST01"."USER_NOTNULL"."USER_ID")
```



# CREATE

## 제약조건 – UNIQUE

- 컬럼 입력 값에 대해 중복을 제한하는 제약조건
- 컬럼레벨과 테이블레벨에서 설정이 가능

```
CREATE TABLE USER_NOCONS(  
  USER_NO NUMBER,  
  USER_ID VARCHAR2(20),  
  USER_PW VARCHAR2(30),  
  USER_NAME VARCHAR2(30),  
  GENDER CHAR(6),  
  PHONE CHAR(11)  
);
```

```
INSERT INTO USER_NOCONS  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');
```

```
INSERT INTO USER_NOCONS  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');
```

USER_NO	USER_ID	USER_PW	USER_NAME	GENDER	PHONE
1	user01	pass01	유저1	남자	01012345678
2	(null)	(null)	(null)	남자	01012345678
1	user01	pass01	유저1	남자	01012345678



# CREATE

## 제약조건 – UNIQUE – 컬럼레벨

```
CREATE TABLE USER_UNIQUE(  
  USER_NO NUMBER,  
  USER_ID VARCHAR2(20) UNIQUE,  
  USER_PW VARCHAR2(30) NOT NULL,  
  USER_NAME VARCHAR2(30),  
  GENDER CHAR(6),  
  PHONE CHAR(11)  
);
```

```
INSERT INTO USER_UNIQUE  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');
```

```
INSERT INTO USER_UNIQUE  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');
```

```
INSERT INTO USER_UNIQUE VALUES (1, 'user01', 'pass01', '유저1', '남자', '01012345678')
```

오류 보고 -

```
ORA-00001: unique constraint (TEST01.SYS_C008069) violated
```



# CREATE

## 제약조건 – UNIQUE – 테이블 레벨

```
CREATE TABLE USER_UNIQUE2(  
  USER_NO NUMBER,  
  USER_ID VARCHAR2(20),  
  USER_PW VARCHAR2(30) NOT NULL,  
  USER_NAME VARCHAR2(30),  
  GENDER CHAR(6),  
  PHONE CHAR(11),  
  UNIQUE (USER_ID)  
);
```

```
INSERT INTO USER_UNIQUE2  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');
```

```
INSERT INTO USER_UNIQUE2  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');
```

```
INSERT INTO USER_UNIQUE2 VALUES (1, 'user01', 'pass01', '유저1', '남자', '01012345678');
```

오류 보고 -

```
ORA-00001: unique constraint (TEST01.SYS C008071) violated
```





# CREATE

## 제약조건 – UNIQUE – NULL

```
INSERT INTO USER_UNIQUE2  
VALUES(1, NULL, 'pass01', '유저1', '남자', '01012345678');
```

```
INSERT INTO USER_UNIQUE2  
VALUES(1, NULL, 'pass01', '유저1', '남자', '01012345678');
```

USE...	USER_ID	USER_PW	USER_NAME	GENDER	PHONE
1	user01	pass01	유저1	남자	01012345678
1	(null)	pass01	유저1	남자	01012345678
1	(null)	pass01	유저1	남자	01012345678

※ UNIQUE 제약조건은 NULL인 경우 중복 저장이 가능

→ NULL도 중복하고 싶지 않은 경우 컬럼레벨에 NOT NULL을 함께 지정하면됨



# CREATE

## 제약조건 – UNIQUE – 컬럼 묶음

```
CREATE TABLE USER_UNIQUE3(  
  USER_NO NUMBER,  
  USER_ID VARCHAR2(20),  
  USER_PW VARCHAR2(30) NOT NULL,  
  USER_NAME VARCHAR2(30),  
  GENDER CHAR(6),  
  PHONE CHAR(11),  
  UNIQUE (USER_NO, USER_ID)  
);
```

```
INSERT INTO USER_UNIQUE3  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');  
INSERT INTO USER_UNIQUE3  
VALUES(2, 'user01', 'pass01', '유저1', '남  
자', '01012345678');  
INSERT INTO USER_UNIQUE3  
VALUES(1, 'user02', 'pass01', '유저1', '남  
자', '01012345678');  
INSERT INTO USER_UNIQUE3  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');
```

### ※ 두 컬럼이 동시에 중복될 때 에러 발생

```
INSERT INTO USER_UNIQUE3 VALUES (1, 'user01', 'pass01', '유저1', '남자', '01012345678')  
오류 보고 -
```

```
ORA-00001: unique constraint (TEST01.SYS C008073) violated
```



# CREATE

## 제약조건 – PRIMARY KEY

- 테이블에서 한 행의 정보를 구분하기 위한 고유 식별자(Identifier)역할
- NOT NULL의 의미와 UNIQUE의 의미를 둘 다 가지고 있음
- 한 테이블 당 한 개만 설정 가능
- 컬럼레벨과 테이블레벨에서 설정 가능



# CREATE

## 제약조건 – PRIMARY KEY – 컬럼레벨

```
CREATE TABLE USER_PK1(  
  USER_NO NUMBER PRIMARY KEY,  
  USER_ID VARCHAR2(20) UNIQUE,  
  USER_PW VARCHAR2(30) NOT NULL,  
  USER_NAME VARCHAR2(30),  
  GENDER CHAR(6),  
  PHONE CHAR(11)  
);
```

```
INSERT INTO USER_PK1  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');
```

```
INSERT INTO USER_PK1  
VALUES(1, 'user02', 'pass01', '유저1', '남  
자', '01012345678');
```

```
INSERT INTO USER_PK1 VALUES (1, 'user02', 'pass01', '유저1', '남자', '01012345678')  
오류 보고 -
```

```
ORA-00001: unique constraint (TEST01.SYS C008075) violated
```



# CREATE

## 제약조건 – PRIMARY KEY – 테이블레벨

```
CREATE TABLE USER_PK2(  
  USER_NO NUMBER,  
  USER_ID VARCHAR2(20) UNIQUE,  
  USER_PW VARCHAR2(30) NOT NULL,  
  USER_NAME VARCHAR2(30),  
  GENDER CHAR(6),  
  PHONE CHAR(11)  
  PRIMARY KEY(USER_NO)  
);
```

```
INSERT INTO USER_PK2  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');
```

```
INSERT INTO USER_PK2  
VALUES(NULL, 'user02', 'pass01', '유저1',  
'남자', '01012345678');
```

```
INSERT INTO USER_PK2 VALUES (NULL, 'user01', 'pass01', '유저1', '남자', '01012345678')
```

오류 보고 -

```
ORA-01400: cannot insert NULL into ("TEST01"."USER_PK2"."USER_NO")
```



# CREATE

## 제약조건 – PRIMARY KEY – 컬럼 묶음

```
CREATE TABLE USER_PK3(  
  USER_NO NUMBER,  
  USER_ID VARCHAR2(20),  
  USER_PW VARCHAR2(30) NOT NULL,  
  USER_NAME VARCHAR2(30),  
  GENDER CHAR(6),  
  PHONE CHAR(11),  
  PRIMARY KEY (USER_NO, USER_ID)  
);
```

```
INSERT INTO USER_PK3  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');  
INSERT INTO USER_PK3  
VALUES(2, 'user01', 'pass01', '유저1', '남  
자', '01012345678');  
INSERT INTO USER_PK3  
VALUES(1, 'user02', 'pass01', '유저1', '남  
자', '01012345678');  
INSERT INTO USER_PK3  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');
```

※ 두 컬럼이 동시에 중복될 때 에러 발생

```
INSERT INTO USER_PK3 VALUES (1, 'user01', 'pass01', '유저1', '남자', '01012345678')
```

오류 보고 -

```
ORA-00001: unique constraint (TEST01.SYS_C008081) violated
```



# CREATE

## 제약조건 – FOREIGN KEY

- 참조 무결성을 유지하기 위한 제약조건
- 참조된 다른 테이블이 제공하는 값만 사용할 수 있도록 제한하는 것
- 참조되는 컬럼과 참조된 컬럼을 통해 테이블간 관계가 형성
- 해당 컬럼은 참조되는 테이블의 컬럼 값 중 하나와 일치하거나 NULL을 가질 수 있음
- 참조되는 테이블의 참조되는 컬럼은 PRIMARY KEY 또는 UNIQUE 제약조건 중에 하나를 가져야 함



# CREATE

## 제약조건 – FOREIGN KEY

참조 되는 테이블 생성 및 데이터 입력

```
CREATE TABLE SHOP_MEMBER(  
  USER_NO NUMBER PRIMARY KEY,  
  USER_ID VARCHAR2(20) UNIQUE,  
  USER_PW VARCHAR2(30) NOT NULL,  
  USER_NAME VARCHAR2(30),  
  GENDER CHAR(6),  
  PHONE CHAR(11)  
);
```

```
INSERT INTO SHOP_MEMBER  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');  
INSERT INTO SHOP_MEMBER  
VALUES(2, 'user02', 'pass02', '유저2', '여  
자', '01022223333');  
INSERT INTO SHOP_MEMBER  
VALUES(3, 'user03', 'pass03', '유저3', '여  
자', '01044445555');
```

USE...	USER_ID	USER_PW	USER_NAME	GENDER	PHONE
1	user01	pass01	유저1	남자	01012345678
2	user02	pass02	유저2	여자	01022223333
3	user03	pass03	유저3	여자	01044445555



# CREATE

## 제약조건 – FOREIGN KEY

테이블 레벨에서 설정

```
CREATE TABLE SHOP_BUY(  
    BUY_NO NUMBER PRIMARY KEY,  
    USER_ID VARCHAR2(20),  
    PRODUCT_NAME VARCHAR2(20),  
    BUY_DATE DATE,  
    FOREIGN KEY (USER_ID) REFERENCES SHOP_MEMBER (USER_ID)  
);
```

컬럼 레벨에서 설정

```
CREATE TABLE SHOP_BUY(  
    BUY_NO NUMBER PRIMARY KEY,  
    USER_ID VARCHAR2(20) REFERENCES SHOP_MEMBER (USER_ID),  
    PRODUCT_NAME VARCHAR2(20),  
    BUY_DATE DATE  
);
```



# CREATE

## 제약조건 – FOREIGN KEY

데이터 삽입

```
INSERT INTO SHOP_BUY(1, 'user01', '축구화',SYSDATE);  
INSERT INTO SHOP_BUY(2, 'user02', '야구화',SYSDATE);  
INSERT INTO SHOP_BUY(3, 'user03', '농구화',SYSDATE);  
INSERT INTO SHOP_BUY(4, 'javalove', '추상화',SYSDATE);
```

```
INSERT INTO SHOP_BUY VALUES(4, 'javalove', '추상화',SYSDATE)
```

오류 보고 -

```
ORA-02291: integrity constraint (TEST01.SYS C008086) violated - parent key not found
```

BUY_NO	USER_ID	PRODUCT_NAME	BUY_DATE
1	user01	축구화	20/02/28
2	user02	야구화	20/02/28
3	user03	농구화	20/02/28

# CREATE

## 제약조건 – FOREIGN KEY – 데이터 삭제

- FOREIGN KEY 제약조건으로 SHOP\_MEMBER의 USER\_ID 컬럼을 참조
- 이때, SHOP\_MEMBER에서 회원을 삭제하려는 경우 SHOP\_BUY의 참조 무결성이 위배
- SHOP\_MEMBER에서 회원이 삭제가 불가능

USE...	USER_ID	USER_PW	USER_NAME	GENDER	PHONE
1	user01	pass01	유저 1	남자	01012345678
2	user02	pass02	유저 2	여자	01022223333
3	user03	pass03	유저 3	여자	01044445555

  

BUY_NO	USER_ID	PRODUCT_NAME	BUY_DATE
1	user01	죽구화	20/02/28
2	user02	야구화	20/02/28
3	user03	농구화	20/02/28

# CREATE

## 제약조건 – FOREIGN KEY – 삭제옵션

부모 테이블의 데이터 삭제 시 자식 테이블의 데이터를 어떠한 방식으로 처리할지 결정하는 옵션

### 1. ON DELETE RESTRICTED

아무것도 지정하지 않는 경우 설정되는 기본 삭제 옵션으로 자식테이블에서 부모테이블의 데이터를 참조하고 있는 경우 데이터 삭제가 불가능 함

### 2. ON DELETE SET NULL

부모 테이블의 데이터 삭제 시 해당 데이터를 참조하고 있던 자식 테이블의 컬럼 값을 NULL로 변경하는 옵션

### 3. ON DELETE CASCADE

부모 테이블의 데이터 삭제 시 해당 데이터를 참조하고 있던 자식 테이블의 데이터까지 모두 삭제하는 옵션



# CREATE

## 제약조건 – FOREIGN KEY

참조 되는 테이블 생성 및 데이터 입력

```
CREATE TABLE SHOP_MEMBER1(  
    USER_NO NUMBER PRIMARY KEY,  
    USER_ID VARCHAR2(20) UNIQUE,  
    USER_PW VARCHAR2(30) NOT NULL,  
    USER_NAME VARCHAR2(30),  
    GENDER CHAR(6),  
    PHONE CHAR(11)  
);
```

```
INSERT INTO SHOP_MEMBER1  
VALUES(1, 'user01', 'pass01', '유저1', '남  
자', '01012345678');  
INSERT INTO SHOP_MEMBER1  
VALUES(2, 'user02', 'pass02', '유저2', '여  
자', '01022223333');  
INSERT INTO SHOP_MEMBER1  
VALUES(3, 'user03', 'pass03', '유저3', '여  
자', '01044445555');
```

USE...	USER_ID	USER_PW	USER_NAME	GENDER	PHONE
1	user01	pass01	유저1	남자	01012345678
2	user02	pass02	유저2	여자	01022223333
3	user03	pass03	유저3	여자	01044445555

# CREATE

## 제약조건 – FOREIGN KEY

ON DELETE SET NULL 설정

```
CREATE TABLE SHOP_BUY2(  
    BUY_NO NUMBER PRIMARY KEY,  
    USER_ID VARCHAR2(20),  
    PRODUCT_NAME VARCHAR2(20),  
    BUY_DATE DATE,  
    FOREIGN KEY (USER_ID) REFERENCES SHOP_MEMBER1 (USER_ID)  
    ON DELETE SET NULL
```

```
);
```

ON DELETE CASCADE 설정

```
CREATE TABLE SHOP_BUY3(  
    BUY_NO NUMBER PRIMARY KEY,  
    USER_ID VARCHAR2(20),  
    PRODUCT_NAME VARCHAR2(20),  
    BUY_DATE DATE,  
    FOREIGN KEY (USER_ID) REFERENCES SHOP_MEMBER1 (USER_ID)  
    ON DELETE CASCADE
```

```
);
```



# CREATE

## 제약조건 – FOREIGN KEY

데이터 삽입

```
INSERT INTO SHOP_BUY2(1, 'user01', '축구화',SYSDATE);  
INSERT INTO SHOP_BUY2(2, 'user02', '야구화',SYSDATE);  
INSERT INTO SHOP_BUY2(3, 'user03', '농구화',SYSDATE);  
INSERT INTO SHOP_BUY3(1, 'user01', '축구화',SYSDATE);  
INSERT INTO SHOP_BUY3(2, 'user02', '야구화',SYSDATE);  
INSERT INTO SHOP_BUY3(3, 'user03', '농구화',SYSDATE);
```

새로 생성한 테이블에 동일한 데이터 입력

BUY_NO	USER_ID	PRODUCT_NAME	BUY_DATE
1	user01	축구화	20/02/28
2	user02	야구화	20/02/28
3	user03	농구화	20/02/28



# CREATE

## 제약조건 – FOREIGN KEY

SHOP\_MEMBER1에서 user01 데이터 삭제

DELETE FROM SHOP\_MEMBER1 WHERE USER\_NO=1;

SHOP\_BUY2 결과  
(ON DELETE SET NULL)

BUY_NO	USER_ID	PRODUCT_NAME	BUY_DATE
1	(null)	죽구화	20/02/28
2	user02	야구화	20/02/28
3	user03	농구화	20/02/28

SHOP\_BUY3 결과  
(ON DELETE CASCADE)

BUY_NO	USER_ID	PRODUCT_NAME	BUY_DATE
2	user02	야구화	20/02/28
3	user03	농구화	20/02/28





# CREATE

## 제약조건 – CHECK

- 해당 컬럼에 입력되거나 수정되는 값을 체크하여, 설정된 값 이외의 값이면 에러 발생
- 비교연산자를 이용하여 조건을 설정하며, 비교값은 리터럴만 사용 가능

```
CREATE TABLE USER_CHECK(  
    USER_NO NUMBER,  
    USER_ID VARCHAR2(20),  
    USER_PW VARCHAR2(30),  
    USER_NAME VARCHAR2(30),  
    GENDER CHAR(6) CHECK (GENDER IN ('남자', '여자')),  
    PHONE CHAR(11)  
);
```

```
INSERT INTO USER_CHECK  
VALUES(1, 'user01', 'pass01', '유저1', '남',  
'01012345678');
```

```
INSERT INTO USER_CHECK VALUES (1, 'user01', 'pass01', '유저1', '남', '01012345678')  
오류 보고 -  
ORA-02290: check constraint (TEST01.SYS_C008109) violated
```



# CREATE

## SUBQUERY를 이용한 CREATE TABLE

- SUBQUERY를 이용해서 SELECT의 조회 결과로 테이블을 생성하는 방법
- 컬럼명과 데이터타입, 값이 복사되고, 제약조건은 NOT NULL만 복사됨

```
CREATE TABLE EMPLOYEE_COPY
```

```
AS
```

```
SELECT EMP_ID,EMP_NAME,DEPT_TITLE,JOB_NAME FROM EMPLOYEE
```

```
LEFT JOIN DEPARTMENT ON (DEPT_CODE = DEPT_ID)
```

```
LEFT JOIN JOB USING(JOB_CODE);
```

	EMP_ID	EMP_NAME	DEPT_TITLE	JOB_NAME
1	214	방명수	인사관리부	사원
2	216	차태연	인사관리부	대리
3	217	전지연	인사관리부	대리
4	219	임시환	회계관리부	차장
5	220	이송석	회계관리부	차장
6	221	유하진	회계관리부	차장
7	206	박나라	해외영업1부	사원
8	210	윤은해	해외영업1부	사원
9	207	하이유	해외영업1부	과장
10	208	김해솔	해외영업1부	과장
11	215	대북론	해외영업1부	과장
12	209	식봉선	해외영업1부	부장

13	203	송은희	해외영업2부	차장
14	204	유재하	해외영업2부	부장
15	205	정형호	해외영업2부	부장
16	211	전하늘	기술지원부	대리
17	212	장우림	기술지원부	대리
18	222	이태림	기술지원부	대리
19	201	송송기	중무부	부장
20	202	송송기	중무부	부장
21	200	선종오	중무부	대표
22	218	이오리	(null)	사원
23	213	하동운	(null)	대리