

DDL

(Data Definition Language)

1. DDL 이란
2. 테이블 생성하기
3. 제약조건
4. 테이블 수정하기
5. 테이블 삭제하기
6. 시퀀스와 뷰

1. DDL이란?

SQL 문법의 종류

SELECT

테이블에서 원하는 데이터를 조회한다.

DML

테이블에 데이터를 입력/삭제/수정한다.

DDL

테이블 같은 데이터 저장소 객체를 만들거나 수정한다.

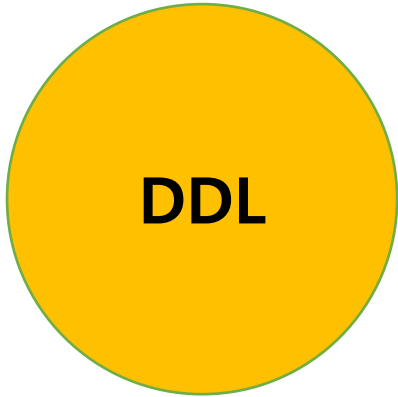
TCL

트랜잭션을 제어한다.

DCL

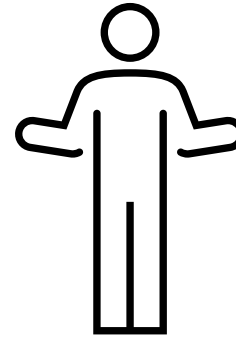
객체에 권한을 부여한다.

1. DDL이란?



테이블 같은 데이터 저장소
객체를 만들거나 수정한다.

테이블과 같은 객체를
생성/수정/삭제 할 수 있습니다.



CREATE

ALTER

DROP

RENAME

TRUNCATE

2. 테이블 생성하기

CREATE

- 새로운 객체(OBJECT) 를 생성할 때 사용하는 명령어 입니다.

CREATE TABLE ...

테이블 생성을 의미합니다.

CREATE USER ...

사용자 계정 생성을 의미합니다.

CREATE SEQUENCE ...

시퀀스 생성을 의미합니다.

CREATE VIEW ...

뷰 생성을 의미합니다.

2. 테이블 생성하기

테이블 생성 전에 필요한 자료형에 대해 알아보시다

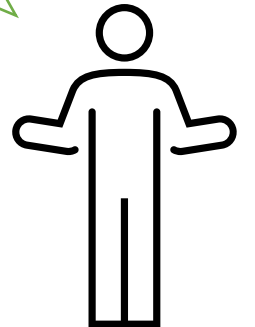
VARCHAR2(n)

NUMBER(n , m)

DATE

실무 팁!

이 외에도 다양한 자료형이 있지만
보통 이 3개의 자료형만 알아도
오라클 데이터베이스는 무난히 사용가능 합니다.



2. 테이블 생성하기

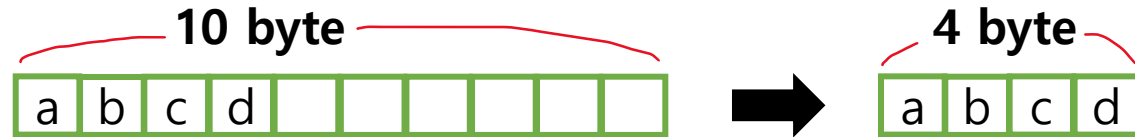
테이블 생성 전에 자료형에 대해 알아봅시다

VARCHAR2(n)

VARCHAR2(n)

문자형 값을 n 바이트까지 입력 받을 수 있는 **가변형 문자열**입니다.

예) varchar2(10) 인 컬럼에 'abcd' 를 입력한 경우



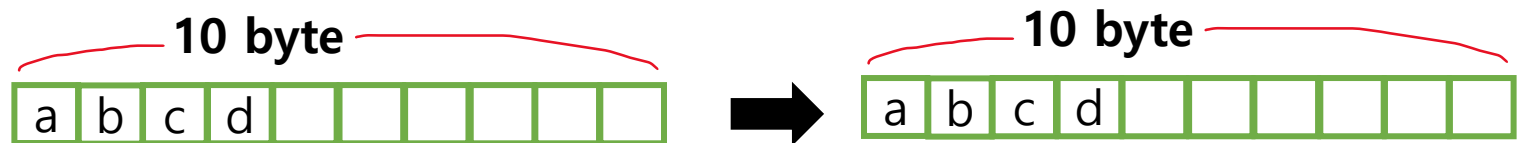
NUMBER(n , m)

DATE

CHAR(n)

문자형 값을 n 바이트까지 입력 받을 수 있는 **고정형 문자열**입니다.

예) char(10) 인 컬럼에 'abcd' 를 입력한 경우



2. 테이블 생성하기

테이블 생성 전에 자료형에 대해 알아봅시다

VARCHAR2(n)

NUMBER(n , m)

DATE

NUMBER(n , m)

숫자형 값을 n자리만큼 입력 받고 m자리만큼 소수를 입력 받습니다.

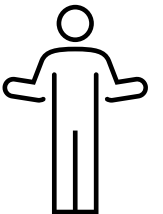
NUMBER로 실수와 정수 모두 표현이 가능합니다.

NUMBER 뒤에 n,m 은 생략이 가능합니다.

예) NUMBER(3) 이면 999까지만 입력이 가능합니다.

실무 팁!

보통은 그냥 NUMBER 로만 사용을 합니다.



2. 테이블 생성하기

테이블 생성 전에 자료형에 대해 알아봅시다

VARCHAR2(n)

NUMBER(n , m)

DATE

DATE

날짜 값을 입력받습니다.

이 외에도 TIMESTAMP라는 자료형도 있습니다.

(둘다 비슷한 날짜형 자료형이며 TIMESTAMP가 좀 더 구체적인 시간을 저장합니다)

```
CREATE TABLE TEST_DATE (
  T_DATE DATE ,                --DATE 형식 자료형
  T_TIMESTAMP TIMESTAMP );    --TIMESTAMP 형식 자료형

INSERT INTO TEST_DATE VALUES (SYSDATE , SYSDATE ) ;
```

리포트 출력 x 실행 결과 x

SQL | 인출된 모든 행: 1(0.025초)

T_DATE	T_TIMESTAMP
23/01/07	23/01/07 20:52:43.000000000

2. 테이블 생성하기

테이블 생성 문법을 분석해봅시다.

테이블명

```
CREATE TABLE QUIZ_TABLE (
```

	컬럼명	자료형	[DEFAULT]	[NULL여부]
			기본은 NULL	기본은 NULL
Q_ID	NUMBER(3, 0)	NOT NULL		
Q_CONTENT	VARCHAR2(200)	NOT NULL		
				NULL 입력하지 마세요
Q_ANSWER	VARCHAR2(100)			
REG_DATE	DATE	DEFAULT SYSDATE		
				따로 값을 입력하지 않는다면
				기본적으로 SYSDATE 를 입력해주세요

```
) ;
```

2. 테이블 생성하기

실제 생성된 TB_MEMBER 테이블의 DDL 입니다.

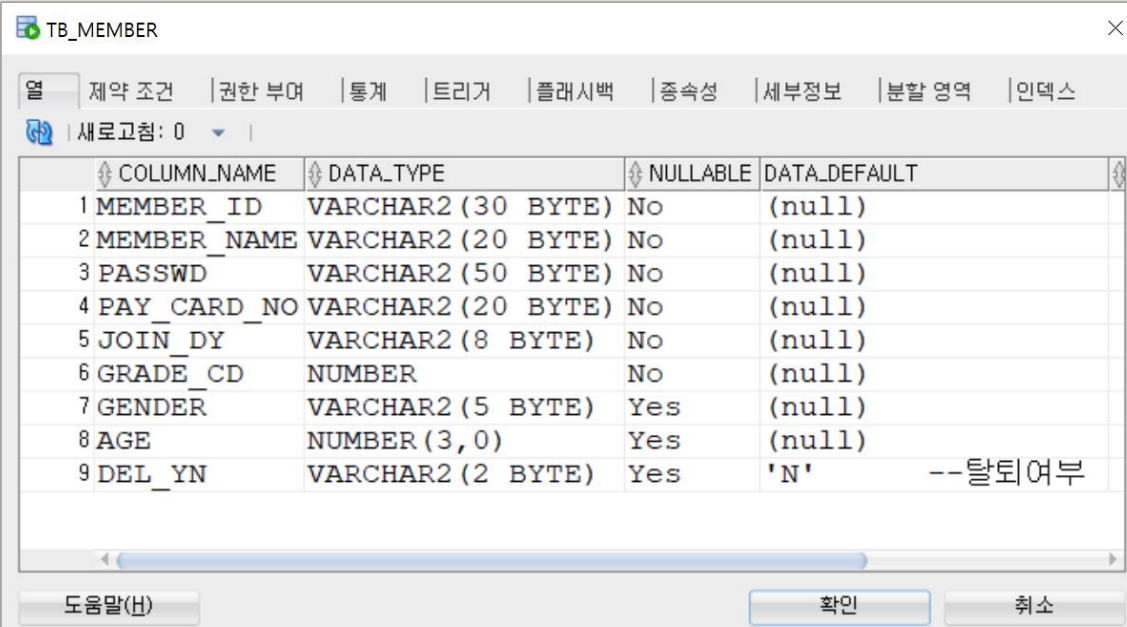
(테이블 생성 정보는 SHIFT + F4 로 볼 수 있습니다)

```
/* TB_MEMBER 테이블 생성 */
CREATE TABLE TB_MEMBER (      --사용자의 정보를 저장하는 테이블

    MEMBER_ID      VARCHAR2(30)    NOT NULL    ,    --회원ID
    MEMBER_NAME     VARCHAR2(20)    NOT NULL    ,    --회원이름
    PASSWD          VARCHAR2(50)    NOT NULL    ,    --패스워드
    PAY_CARD_NO     VARCHAR2(20)    NOT NULL    ,    --결제카드번호
    JOIN_DY         VARCHAR2(8)     NOT NULL    ,    --가입일자
    GRADE_CD        NUMBER          NOT NULL    ,    --등급코드
    GENDER          VARCHAR(5)      ,    --성별
    AGE             NUMBER(3,0)     ,    --나이
    DEL_YN          VARCHAR2(2)     DEFAULT 'N'  --탈퇴여부

) ;
```

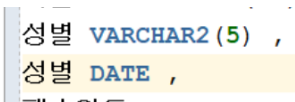
TB_MEMBER --테이블이름을 드래그로 감싸서 SHIFT + F4 입력



	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT
1	MEMBER_ID	VARCHAR2(30 BYTE)	No	(null)
2	MEMBER_NAME	VARCHAR2(20 BYTE)	No	(null)
3	PASSWD	VARCHAR2(50 BYTE)	No	(null)
4	PAY_CARD_NO	VARCHAR2(20 BYTE)	No	(null)
5	JOIN_DY	VARCHAR2(8 BYTE)	No	(null)
6	GRADE_CD	NUMBER	No	(null)
7	GENDER	VARCHAR2(5 BYTE)	Yes	(null)
8	AGE	NUMBER(3,0)	Yes	(null)
9	DEL_YN	VARCHAR2(2 BYTE)	Yes	'N' --탈퇴여부

도움말(H) 확인 취소

잠깐! 테이블 생성시 이름규칙 주의사항

- 1. 대소문자 구분을 안합니다. ex) create table aaa .. => AAA 테이블생성
- 2. 중복되는 테이블명을 쓰면 안됩니다. Ex) create table aaa ... => 에러! 기존 객체가 있습니다.
- 3. 테이블 내에서 컬럼명이 중복되면 안됩니다. Ex)  => 에러! 열명이 중복되었습니다.
- 4. 문자로 시작 , 예약어는 사용 불가능 합니다. (a-z , A-Z , 0-9 , ㄱ-ㅎ , _ , \$, # 특수문자만 사용가능)

2. 테이블 생성하기

실습) 똑같이 작성을 해서 QUIZ_TABLE 을 생성해봅시다.

```
CREATE TABLE QUIZ_TABLE (  
  
    Q_ID          NUMBER(3, 0) NOT NULL ,  
  
    Q_CONTENT     VARCHAR2(200) NOT NULL ,  
  
    Q_ANSWER      VARCHAR2(100) ,  
  
    REG_DATE      DATE DEFAULT SYSDATE  
  
) ;
```

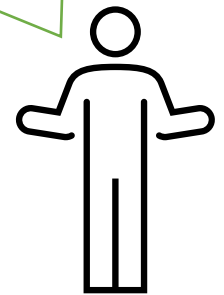
이 테이블은 퀴즈와 답을 저장하는 테이블입니다.

Q_ID 는 퀴즈를 구분하려는 아이디 입니다.

Q_CONTENT 는 문제내용을 입력 받습니다.

Q_ANSWER 는 문제의 답을 입력 받습니다.

REG_DATE는 해당 문제와 답을 입력한 시점입니다.



2. 테이블 생성하기

실습) 생성한 테이블에 다음 데이터를 입력해보세요

```
CREATE TABLE QUIZ_TABLE (  
    Q_ID          NUMBER(3, 0) NOT NULL ,  
    Q_CONTENT     VARCHAR2(200) NOT NULL ,  
    Q_ANSWER      VARCHAR2(100) ,  
    REG_DATE      DATE DEFAULT SYSDATE  
) ;
```

1번 문제, 쥐는 영어로 무엇일까요? 답은 mouse 입니다.

2번 문제, 달력은 영어로 무엇일까요? 답은 calendar 입니다.

3번 문제, 종이는 영어로 무엇일까요? 답은 paper 입니다.

이 세 문제 모두 등록시점은 현재(SYSDATE) 입니다

답)

Q_ID	Q_CONTENT	Q_ANSWER	REG_DATE
1	쥐는 영어로 무엇일까요?	mouse	23/01/07
2	달력은 영어로 무엇일까요?	calendar	23/01/07
3	종이는 영어로 무엇일까요?	paper	23/01/07

REG_DATE 값은 다를 수 있습니다.

제약조건에 대해 알아봅시다.

PRIMARY KEY (PK)

UNIQUE KEY (UK)

NOT NULL

CHECK

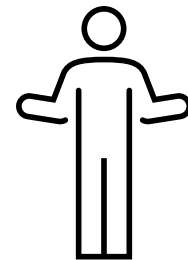
FOREIGN KEY (FK)

제약조건이란

테이블에 입력 가능한 데이터를

조건으로 제약하는 것입니다.

이중 PK 와 FK 는 꼭 이해하고 넘어갑시다



3. 제약조건

PRIMARY KEY (PK) : NOT NULL + UNIQUE

식별자 규칙을 물리적 모델링 한 것으로 **NULL값 입력 불가, 중복 불가**의 특징을 가집니다.

회원

회원ID
비밀번호
이름
연락처

상품

상품ID
상품명
가격



회원

회원ID
비밀번호
이름
연락처

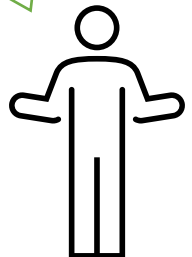
상품

상품ID
상품명
가격

식별자는 테이블에서 각각의 튜플(행)을
유일하게 식별하는 컬럼 집합입니다.

(식별자 = PRIMARY KEY)

* 현재 회원의 테이블명은 TB_MEMBER, 상품의 테이블명은 TB_PRD



3. 제약조건

PRIMARY KEY (PK) : NOT NULL + UNIQUE

특정 컬럼을 식별자로 만들면 **자동으로 NOT NULL + UNIQUE 성질로 바뀌게 됩니다.**

문법)

ALTER TABLE 테이블명 ADD CONSTRAINT 제약조건명 PRIMARY KEY (컬럼) ;

예시)

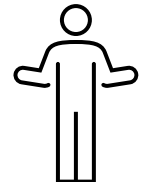
테이블을 변경하다

제약조건을 추가하다

PK를 설정하다

ALTER TABLE QUIZ_TABLE ADD CONSTRAINT PK_QUIZ_TABLE PRIMARY KEY(Q_ID) ;

이제 QUIZ_TABLE 의 Q_ID 컬럼에는
중복 값 입력 불가 + NULL 입력



3. 제약조건

UNIQUE KEY(UK) : UNIQUE

PRIMARY KEY와는 다르게 **NULL 값을 입력할 수 있게 하며**, 중복은 불가능합니다.

문법)

ALTER TABLE 테이블명 ADD CONSTRAINT 제약조건명 UNIQUE (컬럼) ;

예시)

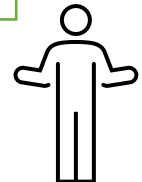
테이블을 변경하다

제약조건을 추가하다

UK를 설정하다

ALTER TABLE QUIZ_TABLE **ADD CONSTRAINT** UK_QUIZ_TABLE **UNIQUE** (Q_CONTENT) ;

이제 QUIZ_TABLE의 Q_CONTENT 컬럼에는
중복 값 입력 불가 + NULL 입력은 가능



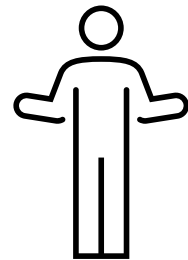
3. 제약조건

NOT NULL

NULL 값이 들어오지 않게 합니다.

```
CREATE TABLE QUIZ_TABLE (  
  
    Q_ID          NUMBER(3, 0) NOT NULL ,  
  
    Q_CONTENT     VARCHAR2(200) NOT NULL ,  
  
    Q_ANSWER      VARCHAR2(100) ,  
  
    REG_DATE      DATE DEFAULT SYSDATE  
  
) ;
```

이 정보(데이터)는 **꼭** 입력되어야 한다!
싶으면 NOT NULL을 설정합니다.



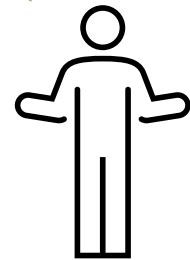
3. 제약조건

CHECK

특정 컬럼에 데이터를 입력할 때 지정한 데이터만 입력할 수 있도록 합니다.

```
ALTER TABLE TB_MEMBER ADD CONSTRAINT CK_DEL_YN CHECK ( DEL_YN IN ( 'Y' , 'N' ) ) ;
```

TB_MEMBER 테이블의
DEL_YN 컬럼에 'Y' , 'N' 값만
입력 가능하도록 검사를 합니다.



3. 제약조건

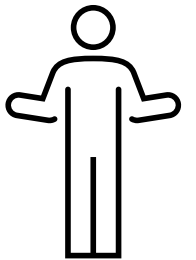
FOREIGN KEY (FK) = 외래키

테이블끼리 연결되어 있는 관계를 물리적 모델링한 것입니다.



FK를 설정하면 **무결성을 지원**해줍니다.

(관계선 = FOREIGN KEY)



3. 제약조건

FOREIGN KEY (FK) = 외래키

테이블끼리 연결되어 있는 관계를 물리적 모델링한 것입니다.

문법)

ALTER TABLE 테이블명 ADD CONSTRAINT 제약조건명

FOREIGN KEY (참조받을컬럼) REFERENCES 참조할테이블(참조할컬럼)

```
-- TB_MEMBER 와 TB_MEMBER_TEL 간 FK 설정
```

```
ALTER TABLE TB_MEMBER_TEL ADD CONSTRAINT FK_MEMBER
```

```
FOREIGN KEY (MEMBER_ID) REFERENCES TB_MEMBER (MEMBER_ID) ;
```

회원

회원ID
비밀번호
이름



회원연락처

회원ID(FK)
구분코드
연락처

실습을 해봅시다. (1/3)

1. 다음 조건에 맞춰 3개의 테이블을 생성 (CREATE) 하세요.
DEFAULT 나 NOT NULL 이 따로 없는 경우는 별도로 설정하지 않음을 의미합니다.

테이블명 : STUDENT

컬럼/자료형 : 학생ID - 가변문자형 10BYTE NOT NULL
: 학생이름 - 가변문자형 20BYTE NOT NULL
: 가입일시 - 날짜형
: 나이 - 숫자형 DEFAULT 0

테이블명 : STUDENT_TEL

컬럼/자료형 : 학생ID - 가변문자형 10BYTE NOT NULL
: 구분코드 - 가변문자형 10BYTE NOT NULL
: 연락처 - 가변문자형 15BYTE NOT NULL

테이블명 : STUDENT_ADDR

컬럼/자료형 : 학생ID - 가변문자형 10BYTE NOT NULL
: 도로명주소 - 가변문자형 200BYTE NOT NULL

실습을 해봅시다. (2/3)

2. 1번 문제에서 만든 3개의 테이블에 각각 주어진 조건에 맞춰 PK (PRIMARY KEY) 제약조건을 추가해주세요.

[힌트 : 만약 두 개 이상의 컬럼을 조합해 PK로 설정하려면 다음과 같은 형식으로 작성해주세요 => PRIMARY KEY(컬럼1 , 컬럼2)]

각 테이블 별로 사용할 PK컬럼과 제약조건 이름은 아래와 같습니다.

STUDENT	테이블의 PK : 학생ID	--제약조건명 : PK_STUDENT
STUDENT_TEL	테이블의 PK : 학생ID + 구분코드	--제약조건명 : PK_STUDENT_TEL
STUDENT_ADDR	테이블의 PK : 학생ID	--제약조건명 : PK_STUDENT_ADDR

실습을 해봅시다. (3/3)

3. 1번 문제에서 만든 3개의 테이블 간의 관계 연결을 위해 FK (FOREIGN KEY) 제약조건을 설정해주세요.

STUDENT_TEL 테이블의 [학생ID] 컬럼은 STUDENT 테이블의 [학생ID] 를 참조하고자 합니다.

제약조건명 : FK_STUDENT_TEL

힌트 : 앞부분이 ALTER TABLE STUDENT_TEL ... 로 시작

STUDENT_ADDR 테이블의 [학생ID] 컬럼은 STUDENT 테이블의 [학생ID] 를 참조하고자 합니다.

제약조건명 : FK_STUDENT_ADDR

힌트 : 앞부분이 ALTER TABLE STUDENT_ADDR ... 로 시작

답안 (1/3)

```
CREATE TABLE STUDENT (  
    학생ID VARCHAR2(10) NOT NULL ,  
    학생이름 VARCHAR2(30) NOT NULL ,  
    가입일시 DATE ,  
    나이 NUMBER DEFAULT 0  
);
```

```
CREATE TABLE STUDENT_TEL (  
    학생ID VARCHAR2(10) NOT NULL ,  
    구분코드 VARCHAR2(10) NOT NULL ,  
    연락처 VARCHAR2(15) NOT NULL  
);
```

```
CREATE TABLE STUDENT_ADDR (  
    학생ID VARCHAR2(10) NOT NULL ,  
    도로명주소 VARCHAR2(200) NOT NULL  
);
```

STUDENT

열 제약 조건 권한 부여 통계 트리거 플래시백 종속성 세부정보 분할 영역 인덱스

새로고침: 0

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT
1 학생ID	VARCHAR2 (10 BYTE)	No	(null)
2 학생이름	VARCHAR2 (30 BYTE)	No	(null)
3 가입일시	DATE	Yes	(null)
4 나이	NUMBER	Yes	0

STUDENT_TEL

열 제약 조건 권한 부여 통계 트리거 플래시백 종속성 세부정보 분할 영역 인덱스

새로고침: 0

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT
1 학생ID	VARCHAR2 (10 BYTE)	No	(null)
2 구분코드	VARCHAR2 (10 BYTE)	No	(null)
3 연락처	VARCHAR2 (15 BYTE)	No	(null)

STUDENT_ADDR

열 제약 조건 권한 부여 통계 트리거 플래시백 종속성 세부정보 분할 영역 인덱스

새로고침: 0

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT
1 학생ID	VARCHAR2 (10 BYTE)	No	(null)
2 도로명주소	VARCHAR2 (200 BYTE)	No	(null)

답안 (2/3)

```
ALTER TABLE STUDENT ADD CONSTRAINT PK_STUDENT PRIMARY KEY ( 학생ID ) ;
```

```
ALTER TABLE STUDENT_TEL ADD CONSTRAINT PK_STUDENT_TEL PRIMARY KEY ( 학생ID , 구분코드 ) ;
```

```
ALTER TABLE STUDENT_ADDR ADD CONSTRAINT PK_STUDENT_ADDR PRIMARY KEY ( 학생ID ) ;
```

답안 (3/3)

```
ALTER TABLE STUDENT_TEL ADD CONSTRAINT FK_STUDENT_TEL  
FOREIGN KEY ( 학생ID ) REFERENCES STUDENT ( 학생ID ) ;  
  
ALTER TABLE STUDENT_ADDR ADD CONSTRAINT FK_STUDENT_ADDR  
FOREIGN KEY ( 학생ID ) REFERENCES STUDENT ( 학생ID ) ;
```


4. 테이블 수정하기

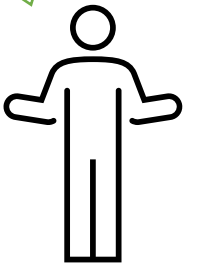
ALTER TABLE ADD 컬럼명..

ALTER TABLE DROP COLUMN 컬럼명 ..

ALTER TABLE MODIFY 컬럼명 ..

ALTER TABLE RENAME 컬럼명 ..

테이블을 잘못 만들어도
삭제할 필요없이 수정이 가능합니다.



4. 테이블 수정하기

ALTER TABLE 테이블명 ADD 컬럼명 자료형 [default] [not null] ;

테이블에 컬럼을 추가합니다.

예) 회원들의 생년월일 정보를 저장할 수 있도록 TB_MEMBER 테이블에 "BIRTH" 컬럼을 추가해주세요. (YYYYMMDD)

```
ALTER TABLE TB_MEMBER ADD ( BIRTH VARCHAR2(8) ) ;
```

Table TB_MEMBER이(가) 변경되었습니다.

MEMBER_ID	MEMBER_NAME	PASSWD	PAY_CARD_NO	JOIN_DY	GRADE_CD	GENDER	AGE	DEL_YN	BIRTH
AAAAA	사용자A	AAAAA	1111-1111-1111-1111	20200101	1	남	(null)	N	(null)
BBBBB	사용자B	BBBBB	2222-2222-2222-2222	20200327	2	여	25	N	(null)
CCCCC	사용자C	CCCCC	3333-3333-3333-3333	20210105	1	남	27	N	(null)
DDDDD	사용자D	DDDDD	4444-4444-4444-4444	20210630	3	여	30	N	(null)
EEEEE	사용자E	EEEEE	5555-5555-5555-5555	20210831	1	남	(null)	N	(null)
FFFFF	사용자F	FFFFF	6666-6666-6666-6666	20220216	3	여	35	N	(null)
GGGGG	사용자G	GGGGG	7777-7777-7777-7777	20220317	2	남	39	N	(null)
HHHHH	사용자H	HHHHH	8888-8888-8888-8888	20220812	5	(null)	44	N	(null)
IIIII	사용자I	IIIII	9999-9999-9999-9999	20230430	4	(null)	52	N	(null)

4. 테이블 수정하기

ALTER TABLE 테이블명 DROP COLUMN 컬럼명 ;

테이블에서 컬럼을 삭제합니다.

예) TB_MEMBER 테이블에서 BIRTH 컬럼을 삭제해주세요.

```
ALTER TABLE TB_MEMBER DROP COLUMN BIRTH ;
```

Table TB_MEMBER01 (가) 변경되었습니다.

MEMBER_ID	MEMBER_NAME	PASSWD	PAY_CARD_NO	JOIN_DY	GRADE_CD	GENDER	AGE	DEL_YN
AAAAA	사용자A	AAAAA	1111-1111-1111-1111	20200101	1	남	(null)	N
BBBBB	사용자B	BBBBB	2222-2222-2222-2222	20200327	2	여	25	N
CCCCC	사용자C	CCCCC	3333-3333-3333-3333	20210105	1	남	27	N
DDDDD	사용자D	DDDDD	4444-4444-4444-4444	20210630	3	여	30	N
EEEEE	사용자E	EEEEE	5555-5555-5555-5555	20210831	1	남	(null)	N
FFFFF	사용자F	FFFFF	6666-6666-6666-6666	20220216	3	여	35	N
GGGGG	사용자G	GGGGG	7777-7777-7777-7777	20220317	2	남	39	N
HHHHH	사용자H	HHHHH	8888-8888-8888-8888	20220812	5	(null)	44	N
IIIII	사용자I	IIIII	9999-9999-9999-9999	20230430	4	(null)	52	N

4. 테이블 수정하기

ALTER TABLE 테이블명 MODIFY (컬럼명 자료형 [DEFAULT] [NOT NULL]);
테이블에서 컬럼 속성을 변경합니다.

예) TB_MEMBER 테이블의 PASSWD 컬럼은 문자형 50BYTE까지 입력을 받을 수 있습니다.

이를 100BYTE 까지 입력받을 수 있도록 컬럼의 성질을 변경해주세요.

```
ALTER TABLE TB_MEMBER MODIFY ( PASSWD VARCHAR2(100) );
```

TB_MEMBER				
열	제약 조건	권한 부여	통계	트리거
새로고침: 0				
COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	
1 MEMBER_ID	VARCHAR2 (30 BYTE)	No	(null)	
2 MEMBER_NAME	VARCHAR2 (20 BYTE)	No	(null)	
3 PASSWD	VARCHAR2 (50 BYTE)	No	(null)	
4 PAY_CARD_NO	VARCHAR2 (20 BYTE)	No	(null)	



TB_MEMBER				
열	제약 조건	권한 부여	통계	트리거
새로고침: 0				
COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	
1 MEMBER_ID	VARCHAR2 (30 BYTE)	No	(null)	
2 MEMBER_NAME	VARCHAR2 (20 BYTE)	No	(null)	
3 PASSWD	VARCHAR2 (100 BYTE)	No	(null)	
4 PAY_CARD_NO	VARCHAR2 (20 BYTE)	No	(null)	

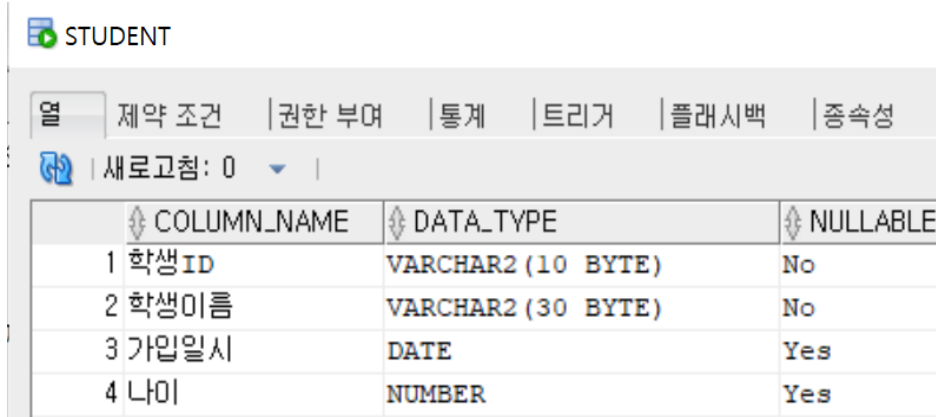
4. 테이블 수정하기

ALTER TABLE 테이블명 RENAME COLUMN 컬럼명 TO 바꿀컬럼명

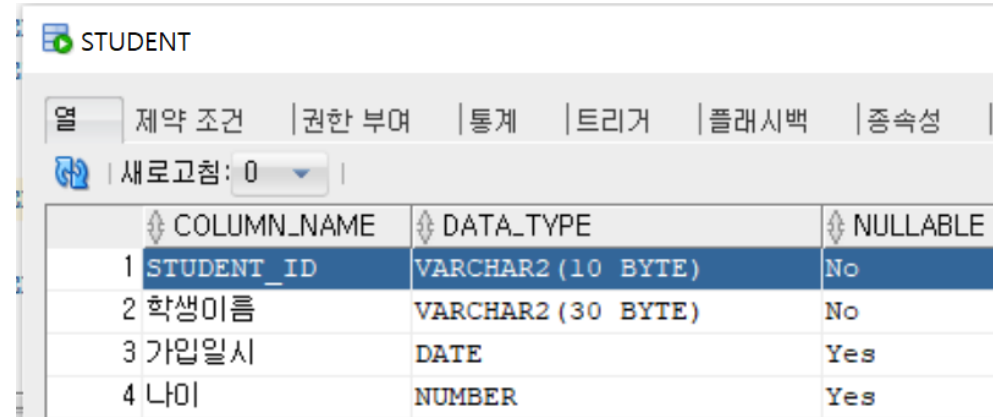
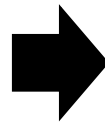
테이블에서 컬럼의 이름을 변경합니다.

예) STUDENT 테이블이 가지고 있는 학생ID 컬럼의 이름을 STUDENT_ID로 변경해주세요.

```
ALTER TABLE STUDENT RENAME COLUMN 학생ID TO STUDENT_ID ;
```



	COLUMN_NAME	DATA_TYPE	NULLABLE
1	학생ID	VARCHAR2 (10 BYTE)	No
2	학생이름	VARCHAR2 (30 BYTE)	No
3	가입일시	DATE	Yes
4	나이	NUMBER	Yes



	COLUMN_NAME	DATA_TYPE	NULLABLE
1	STUDENT_ID	VARCHAR2 (10 BYTE)	No
2	학생이름	VARCHAR2 (30 BYTE)	No
3	가입일시	DATE	Yes
4	나이	NUMBER	Yes

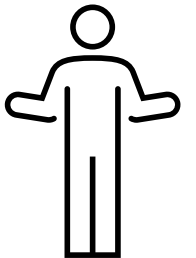
5. 테이블/제약조건 삭제하기

(1)테이블 삭제하기 (drop)

(2)제약조건 삭제하기 (drop constraint)

(3)TRUNCATE , DROP , DELETE 차이 확인

테이블과 제약조건 생성 및 수정을 배웠으니
삭제하는 방법도 알아보시다



5. 테이블/제약조건 삭제하기

DROP TABLE 테이블명 [CASCADE CONSTRAINT]

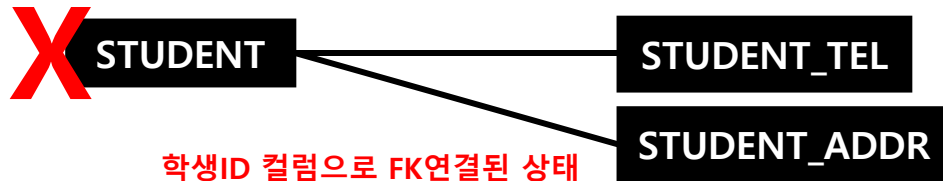
테이블을 영구 삭제합니다. CASCADE CONSTRAINT 옵션을 추가하면 관련 관계선(FK)도 모두 삭제합니다.

예) 아까 만들었던 STUDENT 테이블을 삭제해봅시다.

```
DROP TABLE STUDENT ;
```

오류 보고 -

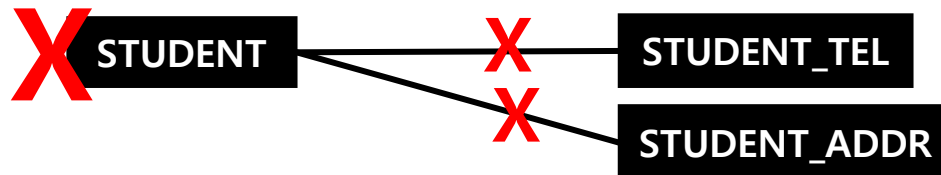
```
ORA-02449: unique/primary keys in table referenced by foreign keys
```



현재 STUDENT 테이블의
학생ID 를 참조하는 테이블이
존재합니다. (STUDENT_TEL , STUDENT_ADDR)

이 경우 테이블 삭제가 불가능합니다.

```
DROP TABLE STUDENT CASCADE CONSTRAINT;
```



CASCADE CONSTRAINT 옵션을 사용하면
STUDENT 연결된 관계선(FK)도 함께 제거합니다.

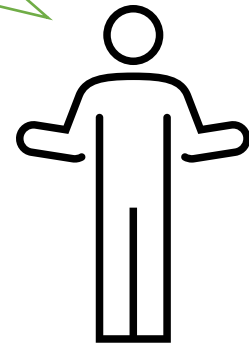
5. 테이블/제약조건 삭제하기

TRUNCATE 는 테이블의 데이터를 삭제합니다.

DELETE 도 테이블의 데이터를 삭제합니다.

DROP은 데이터와 테이블을 삭제합니다.

이 3가지 문법의 차이를 아는 것이 중요합니다.



5. 테이블/제약조건 삭제하기



SELECT * FROM 성적표 ;

학생ID	과목	성적
S0001	국어	90
S0001	수학	85
S0001	영어	100
S0002	국어	100

DELETE FROM 성적표 ;

18개 행 이 (가) 삭제되었습니다.

SQL | 인출된 모든 행: 0(0,002초)

학생ID	과목	성적
------	----	----

TRUNCATE TABLE 성적표 ;

Table 성적표이 (가) 잘렸습니다.

SQL | 인출된 모든 행: 0(0,002초)

학생ID	과목	성적
------	----	----

DROP TABLE 성적표 ;

Table 성적표이 (가) 삭제되었습니다.

DELETE

ROLLBACK; | 롤백 완료.

COMMIT; (COMMIT 시 DELETE 할 데이터 영구삭제)

TRUNCATE

ROLLBACK; (모든 데이터 영구 삭제 / 테이블은 유지)

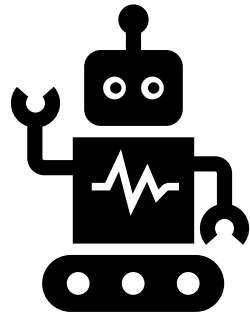
DROP

ROLLBACK; (모든 데이터 영구 삭제 / 테이블도 삭제)

6. 시퀀스와 뷰

시퀀스란?

“순차적인” 을 의미하며 , 오라클에서는 **자동으로 증가하는 값을 만들어주는 객체**



시퀀스

- 1 을 생성해서 보냅니다.
- 2 을 생성해서 보냅니다.
- 3 을 생성해서 보냅니다.
- 4 을 생성해서 보냅니다. ...

6. 시퀀스와 뷰

시퀀스가 필요한 이유

PK의 컬럼 등에 **유일한 일련번호를 만들 때 사용**합니다.

ORDER_NO	MEMBER_ID	PRD_ID	ORDER_DATE	ORDER_CNT	ORDER_PRICE
1	AAAAA	P0002	2023-04-27 00:00:00	1	1500000
2	AAAAA	P0003	2023-04-28 00:00:00	1	600000
3	AAAAA	P0004	2023-04-29 00:00:00	1	800000
4	AAAAA	P0020	2023-04-30 00:00:00	5	25000

TB_ORDER 테이블에서 ORDER_NO 컬럼은

주문 정보를 입력 받을 때 일련의 순서 값을 입력 받아 대상을 식별

```
INSERT INTO TB_ORDER (  
    ORDER_NO  
    , MEMBER_ID  
    , PRD_ID  
    , ORDER_DATE  
    , ORDER_CNT  
    , ORDER_PRICE
```

```
) VALUES (
```

```
    ORDER_NO_SEQ.NEXTVAL  
    , 'BBBBB'  
    , 'P0003'  
    , SYSDATE  
    , 2  
    , 1200000  
);
```

6. 시퀀스와 뷰

시퀀스 생성/사용하기

```
CREATE SEQUENCE ORDER_NO_SEQ  
    INCREMENT BY 1      --증가할 시퀀스 값 (1씩 증가)  
    START WITH 1 ;      --시작할 시퀀스 값 (1부터 시작)
```

시퀀스명.NEXTVAL 를 이용하면 시퀀스 값을 가져올 수 있습니다.

```
SELECT ORDER_NO_SEQ.NEXTVAL FROM DUAL ;
```

ORDER_NO	MEMBER_ID	PRD_ID	ORDER_DATE	ORDER_CNT	ORDER_PRICE
1	AAAAA	P0002	2023-04-27 00:00:00	1	1500000
2	AAAAA	P0003	2023-04-28 00:00:00	1	600000
3	AAAAA	P0004	2023-04-29 00:00:00	1	800000
4	AAAAA	P0020	2023-04-30 00:00:00	5	25000
5	BBBBB	P0003	2023-05-11 10:41:21	2	1200000

```
INSERT INTO TB_ORDER (  
    ORDER_NO  
    , MEMBER_ID  
    , PRD_ID  
    , ORDER_DATE  
    , ORDER_CNT  
    , ORDER_PRICE  
  
    ) VALUES (  
  
    ORDER_NO_SEQ.NEXTVAL  
    , 'BBBBB'  
    , 'P0003'  
    , SYSDATE  
    , 2  
    , 1200000  
  
    ) ;
```



6. 시퀀스와 뷰

시퀀스 삭제하기(DROP)

```
DROP SEQUENCE ORDER_NO_SEQ ;
```

Sequence ORDER_NO_SEQ이 (가) 삭제되었습니다.

6. 시퀀스와 뷰

뷰(VIEW)란?

일종의 “가상테이블” 을 의미합니다.

▶ 회원의 MEMBER_ID , GRADE_CD , TEL_NO (휴대폰) 을 보여주는 쿼리를 작성해봅시다.

```
SELECT A.MEMBER_ID      --회원ID
      , A.GRADE_CD       --등급코드
      , B.TEL_NO         --연락처
FROM TB_MEMBER A
      , TB_MEMBER_TEL B
WHERE A.MEMBER_ID = B.MEMBER_ID
      AND B.TEL_DV_CD = '휴대폰';
```

MEMBER_ID	GRADE_CD	TEL_NO
AAAAA	1	010-1231-1231
BBBBB	2	010-5555-8888

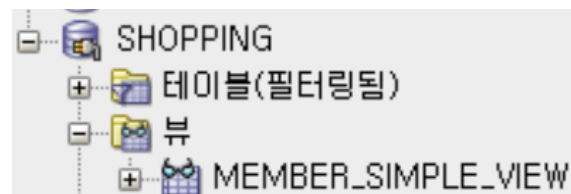
6. 시퀀스와 뷰

뷰 생성하기 (SYSTEM 계정으로 뷰 생성 권한 필요)

```
GRANT CREATE VIEW TO SHOPPING ;
```

```
CREATE VIEW MEMBER_SIMPLE_VIEW AS
SELECT A.MEMBER_ID      --회원ID
      , A.GRADE_CD      --등급코드
      , B.TEL_NO        --연락처
FROM TB_MEMBER A
      , TB_MEMBER_TEL B
WHERE A.MEMBER_ID = B.MEMBER_ID
      AND B.TEL_DV_CD = '휴대폰';
```

View MEMBER_SIMPLE_VIEW이 (가) 생성되었습니다.



6. 시퀀스와 뷰

뷰 사용 원리

테이블처럼 **FROM** 뒤에 뷰 이름을 입력해 사용합니다.

```
SELECT *  
FROM MEMBER_SIMPLE_VIEW ;
```

↓ (1) 뷰 호출

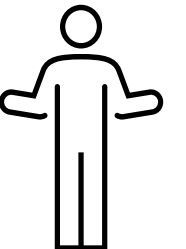
```
CREATE VIEW MEMBER_SIMPLE_VIEW AS  
SELECT A.MEMBER_ID      --회원ID  
      , A.GRADE_CD      --등급코드  
      , B.TEL_NO        --연락처  
FROM TB_MEMBER A  
      , TB_MEMBER_TEL B  
WHERE A.MEMBER_ID = B.MEMBER_ID  
      AND B.TEL_DV_CD = '휴대폰';
```

(2) 해당 뷰에 정의된 쿼리를 재실행

(3) 해당 결과를 출력

MEMBER_ID	GRADE_CD	TEL_NO
AAAAA	1	010-1231-1231
BBBBB	2	010-5555-8888

주의!
절대로 VIEW 에 데이터가
존재하는 게 아닙니다.



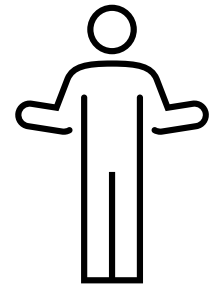
6. 시퀀스와 뷰

뷰를 테이블처럼 활용해 봅시다

```
SELECT *  
FROM MEMBER_SIMPLE_VIEW  
WHERE MEMBER_ID = 'BBBBB';
```

MEMBER_ID	GRADE_CD	TEL_NO
BBBBB	2	010-5555-8888

테이블과 똑같이 사용하면 되므로
뷰를 가상테이블 이라고도 합니다.



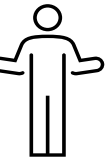
6. 시퀀스와 뷰

인라인 뷰(INLINE VIEW)

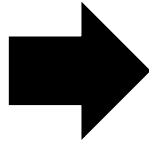
서브쿼리 기술 중 하나로 **FROM 절에 쿼리를 작성해**
가상의 테이블 처럼 사용하는 방식입니다.

실무 팁!

실무에서는 VIEW 보다는
INLINE VIEW 를 자주 사용합니다.



```
CREATE VIEW MEMBER_SIMPLE_VIEW AS
SELECT A.MEMBER_ID      --회원ID
      , A.GRADE_CD      --등급코드
      , B.TEL_NO        --연락처
FROM TB_MEMBER A
      , TB_MEMBER_TEL B
WHERE A.MEMBER_ID = B.MEMBER_ID
      AND B.TEL_DV_CD = '휴대폰';
```



```
SELECT *
FROM (
    SELECT A.MEMBER_ID      --회원ID
          , A.GRADE_CD      --등급코드
          , B.TEL_NO        --연락처
    FROM TB_MEMBER A
          , TB_MEMBER_TEL B
    WHERE A.MEMBER_ID = B.MEMBER_ID
          AND B.TEL_DV_CD = '휴대폰'
)
WHERE MEMBER_ID = 'BBBBB';
```

6. 시퀀스와 뷰

뷰를 사용하는 이유

(1) 자주 사용하는 쿼리를 저장해 놓고 이용할 수 있어 편리하고 연산이 간편해집니다.

```
CREATE VIEW MEMBER_SIMPLE_VIEW AS
SELECT A.MEMBER_ID      --회원ID
      , A.GRADE_CD      --등급코드
      , B.TEL_NO        --연락처
FROM TB_MEMBER A
      , TB_MEMBER_TEL B
WHERE A.MEMBER_ID = B.MEMBER_ID
      AND B.TEL_DV_CD = '휴대폰';
```



```
SELECT *
FROM MEMBER_SIMPLE_VIEW ;
```

6. 시퀀스와 뷰

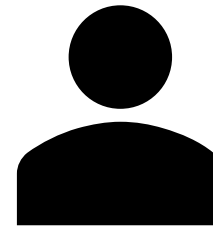
뷰를 사용하는 이유

(2) 원하는 데이터만 보여줄 수 있게 해 **보안 목적으로 사용**할 수 있습니다.

```
CREATE VIEW MEMBER_SIMPLE_VIEW AS
SELECT A.MEMBER_ID      --회원ID
      , A.GRADE_CD      --등급코드
      , B.TEL_NO        --연락처
FROM TB_MEMBER A
      , TB_MEMBER_TEL B
WHERE A.MEMBER_ID = B.MEMBER_ID
      AND B.TEL_DV_CD = '휴대폰';
```

```
SELECT *
FROM MEMBER_SIMPLE_VIEW ;
```

← 접근 가능



이 사용자는
TB_MEMBER 테이블은 접근 불가
MEMBER_SIMPLE_VIEW만 접근 가능

```
SELECT *
FROM TB_MEMBER ;
```

← 접근 불가

6. 시퀀스와 뷰

뷰 삭제하기(DROP)

```
| DROP VIEW MEMBER_SIMPLE_VIEW ;
```

```
| View MEMBER_SIMPLE_VIEW이 (가) 삭제되었습니다.
```

DDL END