

데이터베이스

관계형 데이터베이스

08. 테이블 구조 생성, 변경 및 삭제하는 DDL

first
coding

01 테이블 구조 정의하는 CREATE TABLE

- 지금까지는 오라클에서 학습용으로 제공해 주는 테이블을 사용하여 다양한 조회를 해 보았습니다.
- 이번 장에서는 DDL(Data Definition Language)을 사용하여 테이블 구조 자체를 새롭게 생성, 수정, 삭제해 보겠습니다.
- 우선 CREATE TABLE 명령어로 새로운 테이블을 생성해 보겠습니다.
- 다음은 CREATE TABLE 문의 기본 형식입니다.

식별이름
CREATE TABLE *table_name*

(*column_name*, *data_type* *expr*, ...);

숫자 : number

문자 : char, varchar

날짜 : date

도메인 정의 -> 타입, 사이즈

제약조건 처리

제약조건의 정의

- 도메인 제약

기본키

외래키

01 테이블 구조 정의하는 CREATE TABLE

CREATE TABLE 테이블

```
2 ( 컬럼1 데이터 타입 ,  
3   컬럼2 데이터 타입 ,  
4   컬럼3 데이터 타입 ) ;
```

키워드

SCOTT> CREATE TABLE ddl_test

```
2 ( no NUMBER(3) ,  
3   name VARCHAR2(10),  
4   birth DATE DEFAULT SYSDATE ) ;
```

제약



DDL_TEST

NO	NAME	BIRTH

- 데이터베이스에는 문자, 숫자, 날짜, 이미지 등과 같은 다양한 형태의 데이터가 저장됩니다.

이름	비고
★ CHAR(size)	고정 길이 문자 데이터. VARCHAR2와 동일한 형태의 자료를 저장할 수 있고, 입력된 자료의 길이와는 상관없이 정해진 길이만큼 저장 영역 차지. 최소 크기는 1
★ VARCHAR2(size)	Up to 2000 Bytes 가변 길이 문자 데이터. 실제 입력된 문자열의 길이만큼 저장 영역을 차지. 최대 크기는 명시해야 하며, 최소 크기는 1
NUMBER	Internal Number Format 최고 40자리까지의 숫자를 저장할 수 있습니다. 이때 소수점이나 부호는 길이에 포함되지 않는다.
NUMBER(w)	W자리까지의 수치로 최대 38자리까지 가능하다. (38자리가 유효 숫자이다.)
NUMVER(w, d)	W는 전체 길이, d는 소수점 이하 자릿수이다. 소수점은 자릿수에 포함되지 않는다.
★ DATE	BC 4712년 1월 1일~AD 4712년 12월 31일까지의 날짜

02 데이터 형

파일처리는 자바 라이브러리로 나중에 해볼 예정

- 데이터베이스에는 문자, 숫자, 날짜, 이미지 등과 같은 다양한 형태의 데이터가 저장됩니다.

이름	비고
★ LONG 문자	가변 길이의 문자형 데이터 타입, 최대 크기는 2GB
LOB	2GB까지의 가변 길이 바이너리 데이터를 저장시킬 수 있습니다. 이미지 문서, 실행 파일을 저장할 수 있습니다. mp4, mp3, exe
ROWID	ROWID는 Tree-piece Format을 가짐. ROWID는 DB에 저장되어 있지 않으며, DB Data도 아니다.
BFILE	대용량의 바이너리 데이터를 파일 형태로 저장 최대 4GB
★ TIMESTAMP(n)	DATE 형의 확장된 형태 날짜 + 시간
INTERVAL YEAR TO MONTH	년과 월을 이용하여 기간을 저장
INTERVAL DAY TO SECOND	일, 시, 분, 초를 이용하여 기간을 저장 두 날짜 값의 정확한 차이를 표현하는데 유용하다.

- 지금까지 실습에 사용했던 사원 테이블과 유사한 구조의 사원번호, 사원이름, 급여 3개의 칼럼으로 구성된 EMP01 테이블을 생성해 봅시다.

1. CREATE TABLE 명령어로 EMP01 테이블을 새롭게 생성합니다.

```
CREATE TABLE EMP01(  
  EMPNO NUMBER(4),  
  ENAME VARCHAR2(20),  
  SAL NUMBER(7, 2));
```

- CREATE TABLE 문에서 **서브 쿼리를 사용**하여 이미 존재하는 테이블과 동일한 구조와 내용을 갖는 새로운 테이블을 생성할 수 있습니다.
- 1. CREATE TABLE 명령어 다음에 컬럼을 일일이 정의하는 대신 AS 절을 추가하여 EMP 테이블과 동일한 내용과 구조를 갖는 EMP02 테이블을 생성해 봅시다.

```
CREATE TABLE EMP02 => 컬럼들 생성  
AS                  쿼플(행) 들 까지 같이 복사해 온다.  
SELECT * FROM EMP;  
서브쿼리
```

04 원하는 컬럼으로 구성된 복제 테이블 생성하기

- 기존 테이블에서 원하는 컬럼만 선택적으로 복사해서 생성할 수도 있습니다.
- 1. 서브 쿼리문의 SELECT 절에 * 대신 원하는 컬럼명을 명시하면 기존 테이블에서 일부의 컬럼만 복사할 수 있습니다.

```
CREATE TABLE EMP03  
AS  
SELECT EMPNO, ENAME FROM EMP;
```


04 원하는 행으로 구성된 복제 테이블 생성하기

- 기존 테이블에서 원하는 행만 선택적으로 복사해서 생성할 수도 있습니다.
- 1. 서브 쿼리문의 SELECT 문을 구성할 때 WHERE 절을 추가하여 원하는 조건을 제시하면 기존 테이블에서 일부의 행만 복사합니다.

```
CREATE TABLE EMP05  
AS  
SELECT * FROM EMP  
WHERE DEPTNO=10;
```

- 서브 쿼리를 이용하여 테이블을 복사하되 데이터는 복사하지 않고 기존 테이블의 구조만 복사하는 것을 살펴봅시다.
- 테이블의 구조만 복사하는 것은 별도의 명령이 있는 것이 아닙니다. 이 역시 서브 쿼리를 이용해야 하는데 WHERE 조건 절에 항상 거짓이 되는 조건을 지정하게 되면 테이블에서 얻어질 수 있는 로우가 없게 되므로 빈 테이블이 생성되게 됩니다.

```
CREATE TABLE EMP06  
AS  
SELECT * FROM EMP WHERE 1=0;
```

- WHERE 1=0; 조건은 항상 거짓입니다. 이를 이용하여 테이블의 데이터는 가져오지 않고 구조만 복사하게 됩니다.

06 테이블 구조 변경하는 ALTER TABLE

- ALTER TABLE 명령문은 기존 테이블의 구조를 변경하기 위한 DDL 명령문입니다. 테이블에 대한 구조 변경은 컬럼의 추가, 삭제, 컬럼의 타입이나 길이를 변경할 때 사용합니다. 테이블의 구조를 변경하게 되면 기존에 저장되어 있던 데이터에 영향을 주게 됩니다.
- ALTER TABLE로 칼럼 추가, 수정, 삭제하기 위해서는 다음과 같은 명령어를 사용합니다.
 - ADD COLUMN 절을 사용하여 새로운 칼럼을 추가한다.
 - MODIFY COLUMN 절을 사용하여 기존 칼럼을 수정한다.
 - DROP COLUMN 절을 사용하여 기존 칼럼을 삭제한다.

- ALTER TABLE ADD 문은 기존 테이블에 새로운 컬럼을 추가합니다.
- 새로운 컬럼은 테이블 맨 마지막에 추가되므로 자신이 원하는 위치에 만들어 넣을 수 없습니다.
- 또한 이미 이전에 추가해 놓은 로우가 존재한다면 그 로우에도 컬럼이 추가되지만, 컬럼 값은 NULL 값으로 입력됩니다. => null 값을 지정해줘야해서 번거롭다

```
ALTER TABLE table_name  
ADD (column_name, data_type expr, ...);
```

개발 환경에서는 ALTER 테이블 사용을 거의 안한다

입력테스트

질의테스트

에서는 드롭하고 그 구조에 맞게 새로 만든다.

06 EMP01 테이블에 JOB 컬럼 추가하기

- EMP01 테이블에 문자 타입의 직급(JOB) 칼럼을 추가해 봅시다.

```
ALTER TABLE EMP01  
ADD(JOB VARCHAR2(9));
```

- ALTER TABLE MODIFY 문을 다음과 같은 형식으로 사용하면 테이블에 이미 존재하는 컬럼을 변경할 수 있게 됩니다.

```
ALTER TABLE table_name  
MODIFY (column_name, data_type expr, ...);
```

- 컬럼을 변경한다는 것은 컬럼에 대해서 데이터 타입이나 크기, 기본 값들을 변경한다는 의미입니다.

- 1. 직급(JOB) 컬럼을 최대 30글자까지 저장할 수 있게 변경해 보도록 하자.

```
ALTER TABLE EMP01  
MODIFY(JOB VARCHAR2(30));
```

- 테이블에 이미 존재하는 컬럼을 삭제해 봅시다.
- ALTER TABLE ~ DROP COLUMN 명령어로 컬럼을 삭제할 수 있습니다.

```
ALTER TABLE table_name  
DROP COLUMN column_name;
```


- 1. EMP01 테이블의 직급 컬럼을 삭제해 보도록 합시다.

```
ALTER TABLE EMP01  
DROP COLUMN JOB;
```

07 테이블 구조 삭제하는 DROP TABLE

- DROP TABLE문은 기존 테이블을 제거합니다.

```
DROP TABLE table_name;
```

- 1. CREATE TABLE을 학습할 때 만들어 놓았던 EMP01 테이블을 삭제해 봅시다.

```
DROP TABLE EMP01;
```

08 테이블의 모든 로우를 제거하는 TRUNCATE

- 기존에 사용하던 테이블의 모든 로우를 제거하기 위한 명령어로 TRUNCATE가 제공됩니다.

```
TRUNCATE table_name
```

- 테이블 EMP02 에 저장된 데이터를 확인하였으면 이번에는 테이블의 모든 로우를 제거해 보도록 하겠습니다.

```
TRUNCATE TABLE EMP02;
```

- 기존에 사용하던 테이블의 이름을 변경하기 위한 명령어로 RENAME이 제공됩니다.

```
RENAME old_name TO new_name
```

- EMP02 테이블의 이름을 TEST 란 이름으로 변경합니다.

```
RENAME EMP02 TO TEST;
```

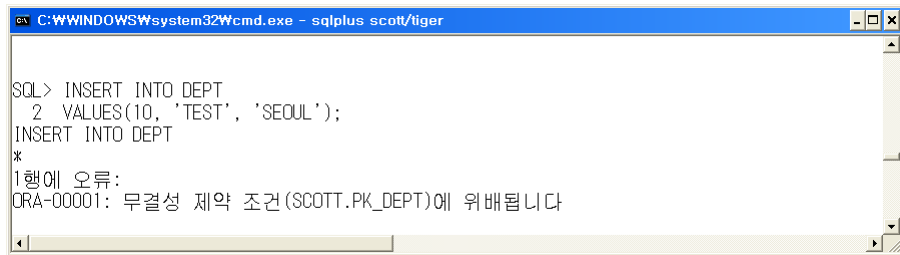
08-2. 데이터 무결성을 위한 제약 조건

01 무결성 제약 조건의 개념과 종류

- 데이터 무결성 제약 조건(Data Integrity Constraint Rule)이란 테이블에 부적절한 자료가 입력되는 것을 방지하기 위해서 테이블을 생성할 때 각 컬럼에 대해서 정의하는 여러 가지 규칙을 말합니다.

무결성 제약 조건	역할
NOT NULL	NULL을 허용하지 않는다.
UNIQUE	중복된 값을 허용하지 않는다. 항상 유일한 값을 갖도록 한다.
PRIMARY KEY	NULL을 허용하지 않고 중복된 값을 허용하지 않는다. NOT NULL 조건과 UNIQUE 조건을 결합한 형태이다.
FOREIGN KEY	참조되는 테이블의 칼럼의 값이 존재하면 허용한다.
CHECK	저장 가능한 데이터 값의 범위나 조건을 지정하여 설정한 값만을 허용한다.

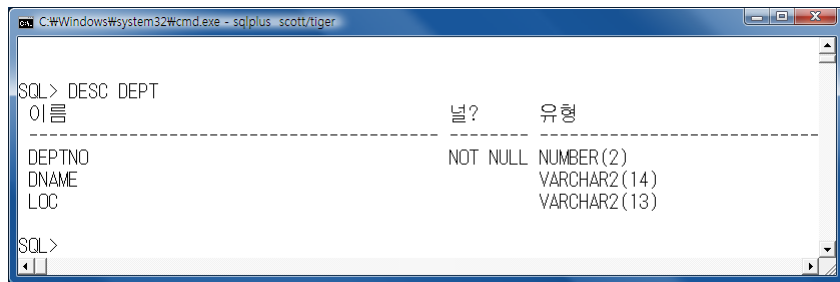
- 아래의 그림은 EMP 테이블에 INSERT 작업 중 무결성 제약 조건을 위반했을 때 나타나는 에러 메시지입니다.



```
C:\WINDOWS\system32\cmd.exe - sqlplus scott/tiger

SQL> INSERT INTO DEPT
  2  VALUES(10, 'TEST', 'SEOUL');
INSERT INTO DEPT
*
1행에 오류:
ORA-00001: 무결성 제약 조건 (SCOTT.PK_DEPT)에 위반됩니다
```

- DESC 명령어로는 NOT NULL 제약조건만 확인할 수 있고 DEPTNO 컬럼에 기본 키 제약 조건이 지정된 것을 알 수 없습니다.



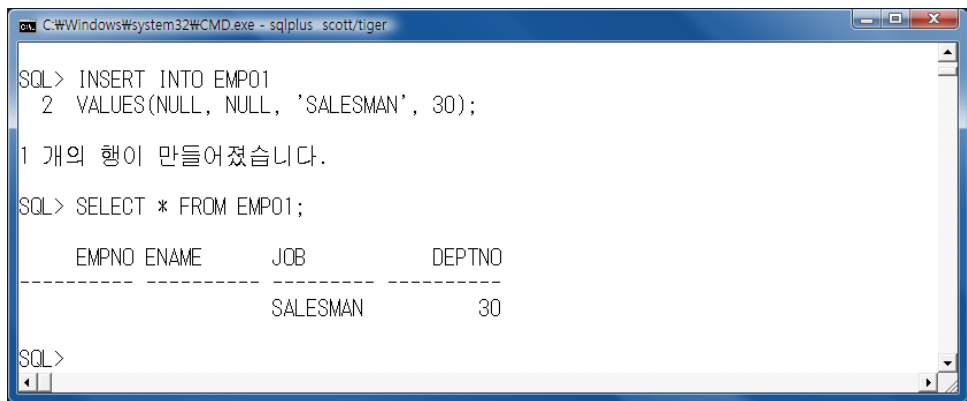
```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger

SQL> DESC DEPT
이름              불?      유형
-----
DEPTNO            NOT NULL  NUMBER(2)
DNAME              VARCHAR2(14)
LOC               VARCHAR2(13)

SQL>
```

03 필수 입력을 위한 NOT NULL 제약 조건

- 새로운 사원이 입사하여 사원의 정보를 입력하는데 사원번호와 사원 명이 불 분명하여 데이터가 저장되지 않았다면 누구의 직급인지, 누구의 부서번호인지를 모르게 되므로 자료로서의 의미를 갖기 어렵습니다.



```
C:\Windows\system32\CMD.exe - sqlplus scott/tiger

SQL> INSERT INTO EMP01
2  VALUES(NULL, NULL, 'SALESMAN', 30);

1 개의 행이 만들어졌습니다.

SQL> SELECT * FROM EMP01;

   EMPNO ENAME      JOB          DEPTNO
-----
          SALESMAN          30

SQL>
```

- 따라서 사원의 정보를 입력할 때 반드시 입력해야하는 선택이 아닌 필수 입력을 요구하는 컬럼이 있다면 위와 같이 NULL 값이 저장되지 못하도록 제약 조건을 설정해야 합니다.

- NOT NULL 제약 조건을 지정하지 않으면 위 예에서처럼 NULL 값이 저장됩니다.
- 특정 컬럼에 NULL 값이 저장되지 못하도록 하려면 NOT NULL 제한 조건을 설정해야 합니다.
- 이제 제약 조건을 설정하는 방법을 살펴봅시다.
- 제약 조건을 설정하는 방법은 컬럼 레벨과 테이블 레벨 두 가지 방식이 있습니다. NOT NULL 제약 조건은 컬럼 레벨로만 정의할 수 있습니다.

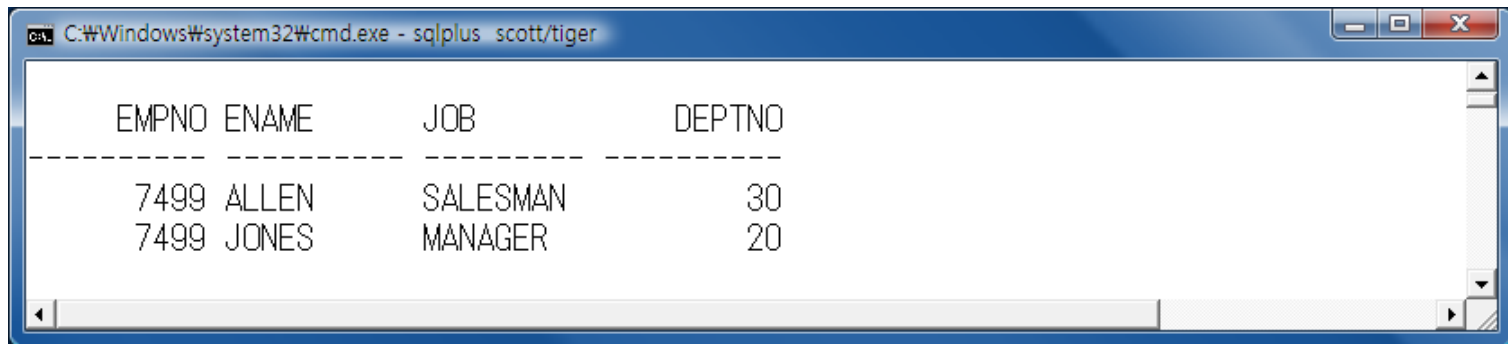
column_name data_type constraint_type

- 사원 테이블(EMP02)을 사원번호, 사원명, 직급, 부서번호 4개의 칼럼으로 구성하되 이번에는 사원번호와 사원명에 NOT NULL 조건을 지정하도록 합시다. 제약 조건은 칼럼명과 자료형을 기술한 후에 연이어서 NOT NULL을 기술하면 됩니다.
 - 지금까지 실습에 사용했던 사원 테이블과 유사한 구조의 사원번호, 사원명, 직급, 부서번호 4개의 칼럼으로 구성된 EMP02 테이블을 생성하되 EMPNO와 EMPNAME 컬럼에 NOT NULL 제약 조건 설정해 봅시다.

```
CREATE TABLE EMP02(  
  EMPNO NUMBER(4) NOT NULL,  
  ENAME VARCHAR2(10) NOT NULL,  
  JOB VARCHAR2(9),  
  DEPTNO NUMBER(2)  
);
```

04 유일한 값만 허용하는 UNIQUE 제약 조건

- UNIQUE 제약 조건이란 특정 칼럼에 대해 자료가 중복되지 않게 하는 것입니다.
- 즉, 지정된 칼럼에는 유일한 값이 수록되게 하는 것입니다.
- 새로운 사원이 입사하여 이 사원의 정보를 입력했는데, 이미 존재하는 사원의 번호와 동일한 사원번호로 입력하였더니 성공적으로 추가된다면 어떻게 될까요?



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - sqlplus scott/tiger". Inside the window, a table is displayed with the following data:

EMPNO	ENAME	JOB	DEPTNO
7499	ALLEN	SALESMAN	30
7499	JONES	MANAGER	20

The table has four columns: EMPNO, ENAME, JOB, and DEPTNO. The first row shows an employee with EMPNO 7499, ENAME ALLEN, JOB SALESMAN, and DEPTNO 30. The second row shows an employee with EMPNO 7499, ENAME JONES, JOB MANAGER, and DEPTNO 20. The duplicate EMPNO value (7499) is highlighted in the original image.

04 UNIQUE 제약조건을 설정하여 테이블 생성하기

- 다음은 사원 테이블의 사원번호를 유일키로 지정한 예입니다.
- 지금까지 실습에 사용했던 사원 테이블과 유사한 구조의 사원번호, 사원명, 직급, 부서번호 4개의 칼럼으로 구성된 EMP03 테이블을 생성하되 사원번호를 유일키로 지정합시다. 제약 조건은 칼럼명과 자료형을 기술한 후에 연이어서 UNIQUE를 기술하면 됩니다.

```
CREATE TABLE EMP03(  
  EMPNO NUMBER(4) UNIQUE,  
  ENAME VARCHAR2(10) NOT NULL,  
  JOB VARCHAR2(9),  
  DEPTNO NUMBER(2)  
);
```

- 이번에는 제약 조건명을 지정하는 방법을 살펴보도록 합시다.

```
column_name data_type CONSTRAINT constraint_name  
constraint_type
```

- 사용자 제약 조건 명을 설정하기 위해서는 CONSTRAINT라는 키워드와 함께 제약 조건 명을 기술하면 된다는 것을 확인할 수 있습니다.
- 제약 조건 명(constraining_name)은 다음과 같은 명명 규칙을 준수해서 작성하는 것이 좋습니다.

[테이블명]_[컬럼명]_[제약 조건 유형]

- 사원 테이블 EMP04에 대해서 사원 번호를 저장하는 컬럼 EMPNO에 대한 유일 키 제약 조건 명인 EMP04_EMPNO_UK를 지정합니다.

EMP04_EMPNO_UK

테이블명

컬럼명

제약 조건유형

- 사용자 제약 조건 명을 설정하기 위해서는 CONSTRAINT라는 키워드와 함께 제약 조건 명을 기술해야 합니다.

05 컬럼 레벨로 제약 조건명 명시하기

- 지금까지 실습에 사용했던 사원 테이블과 유사한 구조의 사원번호, 사원명, 직급, 부서번호 4개의 칼럼으로 구성된 EMP04 테이블을 생성하되 사원번호에는 유일키로 사원명은 NOT NULL 제약조건을 설정해 보시다.

```
CREATE TABLE EMP04(  
    EMPNO NUMBER(4) CONSTRAINT EMP04_EMPNO_UK UNIQUE,  
    ENAME VARCHAR2(10) CONSTRAINT EMP04_ENAME_NN NOT NULL,  
    JOB VARCHAR2(9),  
    DEPTNO NUMBER(2)  
);
```

06 데이터 구분을 위한 PRIMARY KEY 제약 조건

- 테이블 내의 해당 행을 다른 행과 구분할 수 있도록 하는 칼럼은 반드시 존재해야 합니다.
식별 기능을 갖는 칼럼은 유일하면서도 NULL 값을 허용하지 말아야 합니다.
- 즉, UNIQUE 제약 조건과 NOT NULL 제약 조건을 모두 갖고 있어야 하는데 이러한 두 가지 제약 조건을 모두 갖는 것이 기본 키(PRIMARY KEY) 제약 조건입니다.

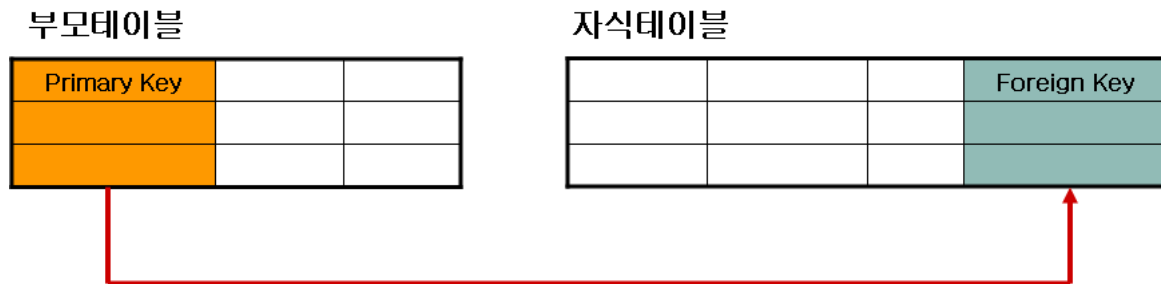
06 PRIMARY KEY 제약 조건 설정하기

- 다음은 사원 테이블의 사원번호를 기본 키로 지정한 예입니다. 제약 조건은 칼럼명과 자료형을 기술한 후에 연이어서 PRIMARY KEY를 기술하면 됩니다.
- 지금까지 실습에 사용했던 사원 테이블과 유사한 구조의 사원번호, 사원명, 직급, 부서번호 4개의 칼럼으로 구성된 테이블을 생성하되 기본 키 제약 조건을 설정해 봅시다.

```
CREATE TABLE EMP05(  
    EMPNO NUMBER(4) CONSTRAINT EMP05_EMPNO_PK PRIMARY KEY ,  
    ENAME VARCHAR2(10) CONSTRAINT EMP05_ENAME_NN NOT NULL,  
    JOB VARCHAR2(9),  
    DEPTNO NUMBER(2)  
);
```

07 참조 무결성을 위한 FOREIGN KEY 제약 조건

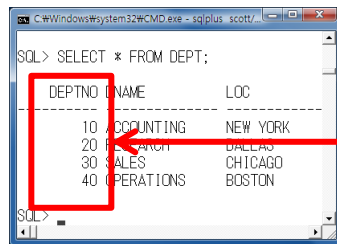
- 부모 키가 되기 위한 칼럼은 반드시 부모 테이블의 기본 키(PRIMARY KEY)나 유일키(UNIQUE)로 설정되어 있어야 한다.



- 우리가 지금까지 학습할 때 사용한 오라클이 제공해주는 EMP 테이블과 DEPT 테이블을 보면 부모 테이블인 부서 테이블(DEPT)의 부서번호(DEPTNO)는 기본 키(PRIMARY KEY)로 설정되어 있고, 이를 참조할 수 있도록 하기 위해서 자식 테이블인 사원 테이블(EMP)에서 부서번호(DEPTNO)에 외래 키 (FOREIGN KEY) 제약조건을 설정해 놓은 상태입니다.

07 참조 무결성을 위한 FOREIGN KEY 제약 조건

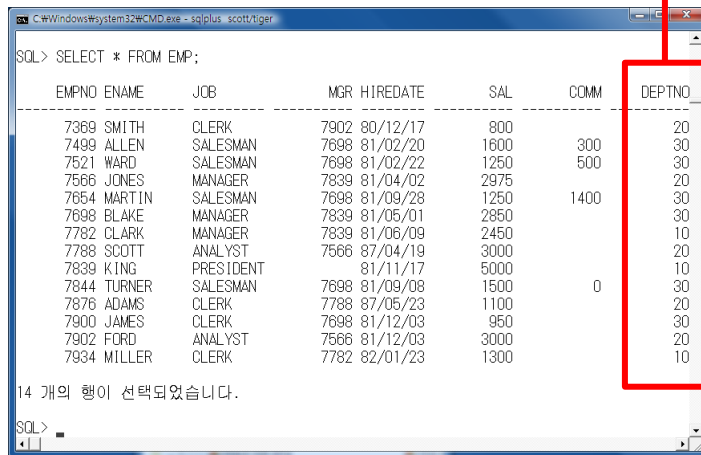
- 부서(dept) 테이블의 기본 키인 부서 번호(deptno) 컬럼을 부모 키라고 함



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

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

- 사원(emp) 테이블의
부서 번호(deptno) 컬럼은
외래 키로 지정해야만
참조의 무결성이 설정됨



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

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800		20
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7566	JONES	MANAGER	7839	81/04/02	2975		20
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850		30
7782	CLARK	MANAGER	7839	81/06/09	2450		10
7788	SCOTT	ANALYST	7566	87/04/19	3000		20
7839	KING	PRESIDENT		81/11/17	5000		10
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
7876	ADAMS	CLERK	7788	87/05/23	1100		20
7900	JAMES	CLERK	7698	81/12/03	950		30
7902	FORD	ANALYST	7566	81/12/03	3000		20
7934	MILLER	CLERK	7782	82/01/23	1300		10

14 개의 행이 선택되었습니다.

07 참조 무결성을 위한 FOREIGN KEY 제약 조건

- 자식 테이블(EMP)에 참조의 무결성을 위해 특정 컬럼에 외래 키를 설정하였다면 새로운 데이터를 추가할 때마다 부모 테이블에 부모 키로 설정된 컬럼을 살핍니다.
- 부모 키로 설정된 컬럼에 존재하는 값만 추가하고 존재하지 않는 값이라면 추가하지 않습니다.
- 이렇게 함으로서 자식 테이블이 부모 테이블을 참조하는데 아무런 문제가 없도록 합니다.

- CHECK 제약 조건은 입력되는 값을 체크하여 설정된 값 이외의 값이 들어오면 오류 메시지와 함께 명령이 수행되지 못하게 하는 것입니다.
- 조건으로 데이터의 값의 범위나 특정 패턴의 숫자나 문자 값을 설정할 수 있습니다.
- 예를 들어 사원 테이블에 급여 컬럼을 생성하되 급여 컬럼 값은 500에서 5000사이의 값만 저장할 수 있도록 하거나 성별을 저장하는 컬럼으로 GENDER 를 정의하고, 이 컬럼에는 남자는 M, 여자는 F 둘 중의 하나만 저장할 수 있도록 제약을 주려면 CHECK 제약조건을 지정해야 합니다.

08 <실습하기> CHECK 제약 조건 설정하기

- 사원 테이블에 급여 컬럼을 생성하되 급여 컬럼 값은 500에서 5000사이의 값만 저장할 수 있도록 하고 성별을 저장하는 컬럼으로 GENDER 를 정의하고, 이 컬럼에는 남자는 M, 여자는 F 둘 중의 하나만 저장할 수 있도록 CHECK 제약조건을 지정해 봅시다. 사원번호, 사원명, 직급, 부서번호, 직급, 성별 6개의 칼럼으로 구성된 테이블을 생성하되 기본 키 제약 조건, 외래키 제약 조건은 물로 CHECK 제약 조건도 설정해 봅시다.

```
CREATE TABLE EMP07(  
    EMPNO NUMBER(4)  
    CONSTRAINT EMP07_EMPNO_PK PRIMARY KEY ,  
    ENAME VARCHAR2(10)  
    CONSTRAINT EMP07_ENAME_NN NOT NULL ,  
    SAL NUMBER(7, 2)  
    CONSTRAINT EMP07_SAL_CK CHECK(SAL BETWEEN 500 AND 5000),  
    GENDER VARCHAR2(1)  
    CONSTRAINT EMP07_GENDER_CK CHECK (GENDER IN('M', 'F'))  
);
```


- 디폴트는 아무런 값을 입력 하지 않았을 때 디폴트제약의 값이 입력이 됩니다.

```
CREATE TABLE DEPT01(  
    DEPTNO NUMBER(2) PRIMARY KEY,  
    DNAME VARCHAR2(14),  
    LOC VARCHAR2(13) DEFAULT 'SEOUL'  
);
```

10 테이블 레벨 방식으로 제약 조건 지정하기

- 지금까지 제약 조건을 지정하는 방식을 컬럼 레벨의 제약 조건 지정이라고 합니다.
- 컬럼 레벨 제약 조건
 - CREATE TABLE로 테이블을 생성하면서 컬럼을 정의하게 되는데 하나의 컬럼 정의가 다 마무리되기 전에 컬럼명 다음에 타입을 지정하고 그 뒤에 연이어서 제약 조건을 지정하는 방식입니다.
- 테이블 레벨의 제약 조건
 - 컬럼을 모두 정의하고 나서 테이블 정의를 마무리 짓기 전에 따로 생성된 컬럼들에 대한 제약 조건을 한꺼번에 지정하는 것입니다.

10 테이블 레벨 방식으로 제약 조건 지정하기

- 다음은 테이블 레벨 정의 방식의 기본 형식입니다.

```
CREATE TABLE table_name (  
    column_name1 datatype1,  
    column_name2 datatype2, . . .  
    [CONSTRAINT constraint_name] constraint_type (column_name)  
)
```

- 테이블 레벨에서 칼럼의 제약 조건을 정의할 때 주의할 것은 NOT NULL 조건은 테이블 레벨 정의 방법으로 제약 조건을 지정할 수 없다는 점입니다.

```
CREATE TABLE EMP02(  
    EMPNO NUMBER(4),  
    ENAME VARCHAR2(10) NOT NULL,  
    JOB VARCHAR2(9),  
    DEPTNO NUMBER(4),  
    PRIMARY KEY(EMPNO),  
    UNIQUE(JOB),  
    FOREIGN KEY(DEPTNO) REFERENCES DEPT(DEPTNO)  
);
```

```
CREATE TABLE EMP03(  
    EMPNO NUMBER(4) CONSTRAINT EMP03_ENAME_NN NOT NULL,  
    ENAME VARCHAR2(10),  
    JOB VARCHAR2(9),  
    DEPTNO NUMBER(4),  
    CONSTRAINT EMP03_EMPNO_PK PRIMARY KEY(EMPNO),  
    CONSTRAINT EMP03_JOB_UK UNIQUE(JOB),  
    CONSTRAINT EMP03_DEPTNO_FK FOREIGN KEY(DEPTNO)  
    REFERENCES DEPT(DEPTNO)  
);
```