

오라클

- 김서연 -

관계형 데이터베이스

- ▶ 테이블의 형태로 데이터를 저장한다.
 - 테이블은 우리가 흔히 엑셀에서 작성하는 표, 웹 사이트의 게시판처럼 구성된 형태를 의미합니다.

성명	전화번호	주소	부서코드	부서명	부서전화번호	부서위치	부서장
김미리	010-22-5551	인천시	01011	인사과	02-1234-2222	서울시 구로구 구로동	김서연
장동건	010-22-5667	성남시	01012	총무부	02-789-1456	서울시 강남구	서유미
홍길동	018-777-2585	서대문구	01011	인사과	02-1234-2222	서울시 구로구 구로동	김서연
김범룡	010-22-555	구로구	01011	총무부	02-789-1456	서울시 강남구	서유미
박정운	012-784-455	안양시	01012	인사과	02-1234-2222	서울시 구로구 구로동	김서연

레코드

컬럼

관계형 데이터베이스

▶ 테이블 정규화

- 위의 형태와 같이 테이블을 작성하면 지금은 5개의 레코드만 정의 하였지만 레코드가 쌓일수록 부서명, 부서전화번호, 부서위치, 부서장 데이터는 중복된 데이터로 쌓일 것입니다. 이렇게 데이터가 중복될 경우 디스크 공간을 많이 차지하거나 수정상에 문제가 발생하는 등 여러 가지 발생할 수 있는 문제점 들이 많이 있습니다. 따라서 관계형 데이터베이스에서는 데이터의 성격 별로 테이블을 나누어서 설계합니다. 이를 **테이블 정규화**라 표현합니다.
- 이렇게 테이블을 정규화시킬 때는 성격별로 나누어 줍니다. 적절하게 어떤걸 어떻게 나누어야 하는지 바로 바로 판단하는건 역시 노하우가 쌓여야 가능한 일이겠죠!^^

관계형 데이터베이스

- ▶ 테이블 정규화 테이블 정규화는 다음과 같이 작업

인사				부서			
성명	전화번호	주소	부서코드	부서코드	부서명	부서전화번호	부서위치
김미리	010-22-5551	인천시	01011	01011	인사과	02-1234-2222	서울시 구로구 구로동
장동건	010-22-5667	성남시	01012	01012	총무부	02-789-1456	서울시 강남구
홍길동	018-777-2585	서대문구	01011				
김범룡	010-22-555	구로구	01011				
박정은	012-784-455	안양시	01012				

- PK는 각 레코드를 구분하기 위한 고유한 값을 갖고 있는 키이며 기본키 또는 Primary key라고 한다.
- 관계형 데이터베이스에서는 테이블은 정규화한 후 다른 테이블에서 기본키를 참조하는 경우가 흔히 있으며 위의 인사 테이블 처럼 기본키를 참조하는 컬럼을 FK라 한다. FK는 외래키 또는 Foreign Key라 한다. FK는 기본키를 데이터를 참조하여 사용하므로 중복된 값이 저장될 수 있다.

테이블 개요

- ▶ 테이블은 오라클에서 데이터를 저장하는 대표적인 오브젝트이다.
- ▶ 관계형 데이터 베이스에서 SQL문을 이용하여 데이터를 접근하며
- ▶ DDL , DML, Query, DCL등으로 나뉜다.
- ▶ DDL은 테이블을 정의하기 위해 사용하는 언어이며 다음과 같은 종류가 있다.
 - Create문 - 테이블 생성
 - Alter문 - 테이블 수정(컬럼 추가, 삭제, 변경)
 - Drop문 - 테이블 삭제

테이블 생성

- ▶ 사용자 테이블과 데이터 디렉터리로 구분된다.
 - 사용자 테이블 - 사용자가 시스템을 활용하며 발생하는 데이터를 저장할 때 사용하는 테이블
 - 데이터 디렉터리 - 오라클에서 발생하는 모든 데이터베이스의 정보를 저장하는 테이블
- ▶ 테이블 생성 구문

```
Create table 테이블명(  
    컬럼명 데이터타입(크기) [제약조건],  
    컬럼명 데이터타입 (크기),  
    컬럼명 데이터타입 (크기))
```

```
create table dept(  
    deptno varchar2(20) primary key,  
    deptname varchar2(20),  
    loc varchar2(20),  
    tel varchar2(20),  
    mgr varchar2(20))
```

테이블 생성

- 테이블 구조 보기 - 테이블을 생성한 후 다음과 같은 명령어를 이용하여 구조를 확인할 수 있다.


```
SQL> desc dept;
```

Name	Null?	Type
DEPTNO	NOT NULL	VARCHAR2(20)
DEPTNAME		VARCHAR2(20)
LOC		VARCHAR2(20)
TEL		VARCHAR2(20)
MGR		VARCHAR2(20)

- 데이터 타입
 - Varchar2 - 가변문자열 저장
 - Char - 고정문자열 저장
 - Number(전체자리수,소수자리수)
 - Date - 날짜형식

테이블생성

```
create table employee(  
    id varchar2(20) primary key,  
    name varchar2(20),  
    pass varchar2(20),  
    hiredate date,  
    grade varchar2(20),  
    point number,  
    deptno varchar2(20));
```



테이블 변경

- ▶ 테이블의 구조를 변경할 수 있다.
- ▶ 컬럼 추가

Alter table 테이블명
Add (컬럼명 데이터타입)

```
alter table dept  
add (cratedate date);
```

- 테이블 구조 확인

```
SQL> desc dept;
```

Name	Null?	Type
DEPTNO	NOT NULL	VARCHAR2(20)
DEPTNAME		VARCHAR2(20)
LOC		VARCHAR2(20)
TEL		VARCHAR2(20)
MGR		VARCHAR2(20)
CRATEDATE		DATE

테이블 변경

▶ 컬럼 수정

Alter table 테이블명
modify (컬럼명 데이터타입)

```
alter table dept  
modify (mgr varchar2(10))
```

- 기존에 정의된 컬럼의 타입이나 길이를 수정할 수 있다.

```
SQL> desc dept;
```

Name	Null?	Type
DEPTNO	NOT NULL	VARCHAR2(20)
DEPTNAME		VARCHAR2(20)
LOC		VARCHAR2(20)
TEL		VARCHAR2(20)
MGR		VARCHAR2(10)
CRATEDATE		DATE

테이블 변경

▶ 컬럼 삭제

Alter table 테이블명
drop (컬럼명)

```
alter table dept  
drop(CRATEDATE);
```

- 기존에 정의된 컬럼을 제거할 때 사용

```
SQL> desc dept;
```

Name	Null?	Type
DEPTNO	NOT NULL	VARCHAR2(20)
DEPTNAME		VARCHAR2(20)
LOC		VARCHAR2(20)
TEL		VARCHAR2(20)
MGR		VARCHAR2(10)

테이블 삭제

- ▶ 테이블의 모든 데이터와 테이블 구조를 삭제한다.
- ▶ 커밋 되지 않은 모든 데이터를 커밋한다.
- ▶ 구문
 - Drop table 테이블명

제약조건

- ▶ 데이터가 테이블에 저장될 때 데이터의 무결성을 위하여 규칙을 정의할 수 있다. 이를 제약조건이라 한다. 제약조건은 테이블을 생성하며 정의하거나 생성 후에 `alter table`문을 이용하여 추가할 수 있다.
- ▶ 제약조건의 종류
 - NOT NULL
 - UNIQUE
 - PRIMARY KEY
 - FOREIGN KEY
 - CHECK

제약조건

▶ 제약조건의 추가

- 테이블 정의할 때 컬럼을 정의하며 추가하거나 테이블의 마지막에서 제약조건을 추가할 수 있다.

```
create table test(  
    id varchar2(10) not null,  
    data1 number);
```

- 테이블이 모두 정의된 후 추가하는 경우

Alter table 테이블명

Add constraint 제약조건이름 추가할제약조건

- 생성된 제약조건은 삭제해야할 수 있으므로 제약조건을 정의할 때 제약조건의 이름을 정의한다.

제약조건

▶ 제약조건의 삭제

- 제약조건을 삭제하는 경우 역시 테이블 수정 명령문을 이용하여 삭제하며 생성할 때 정의한 제약조건의 이름을 이용하여 삭제한다.

`alter table` 테이블명

`drop constraint` 제약조건명

▶ 제약조건의 조회

제약조건이 저장되는 `user_constraints`에서 정보를 조회한다.

NOT NULL

- ▶ 컬럼에 데이터가 저장될 때 NULL이 포함되지 못하도록 지정한다.
- ▶ 컬럼 수준으로 not null제약조건을 정의함

```
create table test(  
    id varchar2(10) not null,  
    data1 number);
```

- ▶ 다음과 같이 null을 insert하려 하면 에러가 발생한다.

```
SQL> insert into test values(null, 1000);  
insert into test values(null, 1000)  
*
```

ERROR at line 1:

ORA-01400: cannot insert NULL into ("SCOTT"."TEST"."ID")

UNIQUE

- ▶ 중복된 값이 저장될 수 없는 컬럼인 경우 unique제약 조건을 정의하여 제약한다.
- ▶ 예를 들어 dept테이블의 deptname같은 경우 중복될 수 없다. 이미 테이블이 만들어진 상태에서 제약조건을 추가할 수 있다.

```
alter table dept  
add constraint uk_dname unique(deptname)
```

- ▶ 중복된 값이 저장되는 경우 다음과 같이 에러를 발생시킨다.

```
SQL> insert into dept values('002','전산실','구로디지털','02-111-2222','9411221');  
insert into dept values('002','전산실','구로디지털','02-111-2222','9411221')
```

*

ERROR at line 1:

ORA-00001: unique constraint (SCOTT.UK_DNAME) violated

PRIMARY KEY

- ▶ 테이블에 저장된 레코드를 식별하기 위한 제약조건으로 not null, unique의 특성을 함께 갖고 있는 제약조건이다.
- ▶ 보통 테이블을 정의하며 컬럼 수준으로 제약조건을 정의한다.

```
create table dept(  
    deptno varchar2(20) primary key,  
    deptname varchar2(20),  
    loc varchar2(20),  
    tel varchar2(20),  
    mgr varchar2(20))■
```

FOREIGN KEY

- ▶ 관계형 데이터베이스는 데이터 관리를 테이블로 하며 성격에 따라 여러 테이블로 데이터를 나누어 관리한다 했습니다. 이때 이미 작성된 테이블의 기본키를 참조해서 쓰는 경우 foreign key 제약 조건을 설정한다.
- ▶ 테이블에 저장된 값을 가져와야 하므로 데이터의 무결성을 위해 적용
- ▶ 제약조건을 정의할때 어떤 테이블의 어떤 기본키를 참조하는지 정의해야 한다.
- ▶ Foreign key 제약조건이 설정된 컬럼에는 기본키와 동일한 값이나 null이 저장된다.

FOREIGN KEY

Alter table 테이블명

Add constraint 제약조건명 foreign key(foreign key제약조건을 적용할 컬럼)
references 테이블명(기본키 - foreign key에서 참조할 기본키)

다음과 같은 형식으로 제약조건을 설정한다.

```
alter table employee  
add constraint fk_emp_deptno foreign key (deptno)  
references kitridept(deptno);
```

기본키 테이블의 기본키에 없는 데이터를 입력하면 “parent key not found” 에러가 발생한다.

```
SQL> insert into employee values ('9410222', '김서연', '1222', '1994/3/27', '3급'  
2                                     , 10000, '100');
```

```
insert into employee values ('9410222', '김서연', '1222', '1994/3/27', '3급'
```

```
*
```

```
ERROR at line 1:
```

```
ORA-02291: integrity constraint (SCOTT.FK_EMP_DEPTNO) violated - parent key not fou
```

조인

- ▶ 여러 테이블의 컬럼을 조회하여 결과를 만들고 싶은 경우 조인을 적용한다.
- ▶ 조인은 관계형 데이터베이스에서 매우 중요한 개념이며 통상적으로 기본키와 외래키를 이용하여 조인한다.
- ▶ 조인은 외래키값을 기본키와 비교하여 일치하는 데이터를 가져오는 것이므로 반드시 어떤 키를 비교해야 하는지 조건에 명시해야 하며 이를 조인조건이라 한다.
- ▶ 조인조건을 명시 하지 않는 경우 조인을 적용한 테이블의 모든 결과가 출력된다.
- ▶ n 개의 테이블을 조인하는 경우 $n-1$ 개의 조인조건이 필요하다.

조인

- ▶ 조인의 종류
 - Equijoin
 - Non-equijoin
 - Outer join
 - Self join

EquiJoin

- ▶ 기본키와 외래키를 비교하여 정확하게 일치하는 값에 해당하는 컬럼을 반환하는 조인

사번	고객명	직업	관리자	입사일	급여	보너스	부서코드
7369	SMITH	CLERK	7902	1980-12-17	800		20
7499	ALLEN	SALESMAN	7698	1981-02-20	1600	300	30
7521	WARD	SALESMAN	7698	1981-02-22	1250	500	30
7566	JONES	MANAGER	7839	1981-04-02	2975		20
7654	MARTIN	SALESMAN	7698	1981-09-28	1250	1400	30
7698	BLAKE	MANAGER	7839	1981-05-01	2850		30
7782	CLARK	MANAGER	7839	1981-06-09	2450		10
7788	SCOTT	ANALYST	7566	1982-12-09	3000		20
7839	KING	PRESIDENT		1981-11-17	5000		10
7844	TURNER	SALESMAN	7698	1981-09-08	1500	0	30
7876	ADAMS	CLERK	7788	1983-01-12	1100		20
7900	JAMES	CLERK	7698	1981-12-03	950		30
7902	FORD	ANALYST	7566	1981-12-03	3000		20
7934	MILLER	CLERK	7782	1982-01-23	1300		10

DEPTNO	DNAME	LOC_
10	ACCOUNTING	A1
20	RESEARCH	B1
30	SALES	C1
40	OPERATIONS	A1
50	INSA	

EquiJoin

- ▶ 조인형식

Select 테이블명.컬럼명, 테이블명.컬럼명...

From 테이블1, 테이블2...

Where 조인조건