# 4. 데이터 조작어(DML)(4)[조인(Join)]

### 4.4 조인(Join)

- 둘 이상의 테이블을 연결하여 데이터를 검색하는 방법 입니다.
- 보통 둘이상의 행들의 공통된 값 Primary Key 및 Foreign Key 값을 사용하여 조인합니다.
- 그러므로 두 개의 테이블을 SELECT문장 안에서 조인하려면 적어도 하나의 칼럼이 그 두 테이블 사이에서 공유 되어야 합니다.

**** 조인 종류****

- Equijoin(동등조인, 내부조인)
- Non-equijoin
- Outer join
- Self join

=======조인방법=======================================

1) 똑같은 컬럼명이 존재하는  테이블이 있을 경우는 반드시 컬럼명 앞에 테이블명을 붙인다.

2) n개의 테이블을 조인하려면 최소한 n-1번의 조인 조건문이 필요하다.

```
SELECT  table1.colum1[,  table2.column2  ...]
FROM       table1,  table2
WHERE  table1.colum1=table2.column2;
```

또는 join절을 이용한 명시적 조인

```
SELECT  table1.colum1[,table2.colum2...]
FROM  table1  JOIN  table2
ON    table1.column1  =  talbe2.column2;
```

**** Equi Join ****

조인조건이 정확히 일치하는 경우에 사용(pk와 fk를 사용하여)

부서이름

```
SQL>  SELECT  e.ename,  d.dname
        FROM  emp  e,  dept  d
        WHERE  e.deptno  =  d.  deptno;
```

- WHERE 절에 조인 조건을 작성하고 column명 앞에 테이블명을 적습니다.

### 1) EQUI JOIN

- EQUI JOIN이란 조인 조건에서 '='를 사용하여 값들이 정확하게 일치하는 경우에 사용하는 조인을 의미
- 대부분 PK와 FK의 관계를 이용하여 조인한다.
- 다른 말로 단순 조인 또는 내부 조인이라고 한다.

**[실습]**

사원의 부서 이름을 결정하기 위해 EMP테이블의 DEPTNO와 DEPT테이블의 DEPTNO와 값을 비교해야 한다. EMP테이블과 DEPT테이블 사이의 관계는 양쪽 테이블의 DEPTNO열이 같아 야 한다. 이들이 PK와 FK로 연결되어 있다.

```
SELECT empno, ename, emp.deptno, dept.deptno, dname, loc
FROM emp JOIN dept
ON emp.deptno = dept.deptno;
```

또는

```
SELECT empno, ename, emp.deptno, dept.deptno, dname, loc
FROM EMP, DEPT
WHERE EMP.DEPTNO=DEPT.DEPTNO;
```

**테이블에서 ALIAS사용**********

- 테이블 별칭을 사용하여 긴 테이블명을 간단하게 사용한다.
- 테이블 별칭은 30자까지 사용가능하나 짧을 수록 좋다.
- FROM 절에서 별칭이 사용되면 SELECT문 전체에서 사용 가능하다.
- 테이블의 별칭은 현재 SELECT문에서만 유용하다.

```
SELECT E.EMPNO, E.ENAME, E.JOB, E.DEPTNO, D.DEPTNO, D.DNAME, D.LOC FROM EMP E
JOIN DEPT D
ON E.DEPTNO = D.DEPTNO ORDER BY D.DEPTNO;
```

**[실습]**
EMP와 DEPT 테이블에서 업무가 MANAGER인 사원의 이름, 업무, 부서명,
근무지를 출력하세요.

- 아래는 inline view를 이용하여

```
SELECT ENAME, JOB, A.DEPTNO, B.LOC FROM (
SELECT  ENAME,JOB,DEPTNO  FROM EMP WHERE JOB='MANAGER') A, DEPT B
WHERE A.DEPTNO=B.DEPTNO;
```

- 또는 join절을 이용해서 구할 수도 있다.

```
SELECT  ENAME,JOB,A.DEPTNO,LOC FROM EMP A
JOIN DEPT B ON A.DEPTNO= B.DEPTNO
AND A.JOB='MANAGER';
```

==========

**1_1) AND연산자를 사용하여 추가적인 검색 조건**

- 조인 이외의 WHERE절에 추가적인 조인 조건을 가질 수 있다.

**[실습]**

**SALESMAN의 사원번호, 이름, 급여, 부서명, 근무지를 출력하여라.**

**** Non - Equijoin ****

조인 조건이 정확히 일치하지 않는 경우에 사용 (등급,학점)

- Non-equijoin은 테이블의 어떤 column도 join할 테이블의 column에 일치하지 않을 때 사용하고 조인조건은 동등( = )이외의 연산자를 갖습니다.

  (BETWEEN AND, IS NULL, IS NOT NULL, IN, NOT IN)

**- WHERE절 뒤의 조건절에서 두 개 이상의 테이블을 연결할 때**
**조건이 EQUAL(=) 이 아닌 다른 연산 기호로 만들어지는 경우에 사용.**

**- EMP와 SALGRADE 테이블 사이 관련성을 살펴보면, EMP테이블의 어떠한 컬럼도 직접적으로 SALGRADE테이블의 한 컬럼에 상응하지 않는다.**

**따라서 두 테이블의 관련성은 EMP테이블의 SAL컬럼이 SALGRADE의 LOSAL과 HISAL 컬럼 사이에 있다. 이 경우 NON-EQUIJOIN이다.**

**조인 조건은 = 연산자 이외의 BETWEEN~AND 연산자를 이용한다.**

```
SELECT EMPNO, ENAME, SAL FROM EMP;
SELECT GRADE, LOSAL, HISAL FROM SALGRADE;
```

**위 두 테이블 정보를 NON-EQUIJOIN하면**

```
SELECT EMPNO,ENAME,SAL,GRADE,LOSAL,HISAL FROM EMP E, SALGRADE S
WHERE E.SAL BETWEEN S.LOSAL AND S.HISAL
```

```
select empno,ename,sal,grade,losal,hisal from emp e,salgrade s
where e.sal between s.losal and s.hisal and e.deptno=10;
```

** Self Join **

- 때때로 자체적으로 테이블을 조인할 필요가 있다.
  가령 각 사원들의 관리자 이름을 알기 위해서 자체적으로 EMP테이블을 조인하는 것이 필요하다.

```
SELECT E.EMPNO, E.ENAME, E.MGR, E2.EMPNO "MGR NO", E2.ENAME "MGR NAME"
FROM EMP E JOIN EMP E2
ON E.MGR=E2.EMPNO;
```

- self join을 사용하여 한 테이블의 행들을 같은 테이블의 행들과 조인한다.
- 같은 테이블에 대해 두개의 alias를 작성하여 테이블을 구분함으로
  from 절에 두 개의 테이블을 사용하는 것 같이 한다.
- 컬럼에 대해서도 어떤 테이블에서 왔는지 반드시 별칭을 기술해야 한다.
- 테이블 하나를 두 개 또는 그 이상으로 self join할 수 있다.

**[문제]** emp테이블에서 "누구의 관리자는 누구이다"는 내용을 출력하세요.

** Out(외부) Join **

- 두 개 이상의 테이블을 EQUI JOIN방법으로 조인하는 경우, 한 쪽 테이블에 일치하는 행이 없으면 다른 쪽 테이블을 NULL로 하여 그 값을 보여준다.

- SELECT 컬럼1, 컬럼2... FROM 테이블1, 테이블2
  WHERE 테이블1.컬럼 = 테이블2.컬럼(+)
...여기서 (+) 기호는 NULL열이 작성되는 쪽에 붙는다.

이 경우 기준이 되는 테이블은 테이블1이다. 두 개의 테이블 가운데 기준 테이블을 중심으로 테이블1.컬럼과 테이블2.컬럼의 EQUAL조건에서 만족하지 않는 항목이 있더라도 테이블2 항목을 NULL로 설정 하여 출력하게 된다.

```
select e.empno, e.ename, e.job, e.deptno, d.deptno, d.dname, d.loc
from dept d, emp e   where d.deptno=e.deptno(+);
```

위의 조인문을 아래와 같이 join절을 이용할 경우는

```
SELECT e.empno, e.ename, e.job, e.deptno, d.deptno, d.dname, d.loc
FROM dept d LEFT OUTER JOIN emp e
ON d.deptno=e.deptno;
```

** LEFT OUTER JOIN **

- 왼쪽 테이블에 조인시킬 컬럼의 값이 없는 경우 사용합니다.

```
SQL> SELECT DISTINCT (e.deptno), d.deptno
     FROM dept d LEFT OUTER JOIN emp e
     ON d.deptno = e.deptno;
```

** RIGHT OUTER JOIN **

- 오른쪽에 테이블에 조인시킬 컬럼의 값이 없는 경우 사용 합니다.

```
SQL> SELECT DISTINCT (a.deptno), b.deptno
       FROM emp a RIGHT OUTER JOIN dept b
       ON a.deptno = b.deptno;
```

**** FULL OUTER JOIN ****

- 양쪽 테이블에 다 outer join 을 거는것을 TWO-WAY OUTER JOIN 또는 FULL OUTER JOIN이라 한다.

```
SQL> SELECT DISTINCT (a.deptno), b.deptno
       FROM emp a FULL OUTER JOIN dept b
       ON a.deptno = b.deptno
```

-- 위의 세 문장의 결과는 아래와 같습니다.

```
   DEPTNO     DEPTNO
---------- ----------
       10         10
       20         20
       30         30
                  40
```

LEFT OUTER JOIN과 RIGHT OUTER JOIN의 테이블 순서를 바꾸어 가면서 테스트를 하시면 쉽게 이해를 하실 수 있습니다.

```
SQL>  SELECT DISTINCT(a.deptno), b.deptno
  FROM emp a, dept b
  WHERE  a.deptno(+) = b.deptno
```

※ 다음의 쿼리를 한번 잘 보시기 바랍니다.

```
SQL>  SELECT DISTINCT(a.deptno), b.deptno
        FROM emp a, dept b
        WHERE  a.deptno(+) = b.deptno
            AND a.ename LIKE '%';
 EPTNO   DEPTNO
---------- ----------
       10         10
       20         20
       30         30
```

쿼리 결과를 잘 보면 out조인이 되지 않은 것을 알 수 있습니다.
위 쿼리를 out조인이 되기 위해서는 아래와 같이 고쳐야 합니다.

```
SQL> SELECT DISTINCT(a.deptno), b.deptno
        FROM emp a, dept b
        WHERE   a.deptno(+) = b.deptno
            AND a.ename(+) LIKE '%';
```

| DEPT | DEPTNO |
|---|---|
| 10 | 10 |
| 20 | 20 |
| 30 | 30 |
| | 40 |

OUT조인 조건이 걸려있는 테이블에는 다른 조건절이 들어와도
똑같이 OUT조인 연산자인 (+)를 해주어야 합니다.

참고] 위를 join절로 고치면?

```
select distinct(a.deptno),b.deptno
from emp a right outer join dept b
on a.deptno=b.deptno and a.ename like '%'
```

**[종합문제]**

1. emp테이블에서 모든 사원에 대한 이름, 부서번호, 부서명을 출력하는 문장을 작성하세요.
   (부서번호순으로 오름차순 정렬하라.)

2. emp테이블에서 NEW YORK에서 근무하고 있는 사원에 대하여 이름, 업무, 급여, 부서명을 출력하는 SELECT문을 작성하세요.

3. EMP테이블에서 보너스를 받는 사원에 대하여 이름, 부서명, 위치를 출력하는 SELECT문을 작성하세요.

4. EMP테이블에서 이름 중 L자가 있는 사원에 대해 이름, 업무, 부서명, 위치를 출력하는 문장을 작성하세요.

5. 아래의 결과를 출력하는 문장을 작성하세요
   (관리자가 없는 King을 포함하여 모든 사원을 출력)

```
------------------------------------------------------------------------
Emplyee          Emp#        Manager        Mgr#
------------------------------------------------------------------------
KING              7839
BLAKE             7698       KING           7839
CKARK             7782       KING           7839
.....
-----------------------------------------
14ROWS  SELECTED.
```

** INNER JOIN **

- 일반 조인시 ,를 생략하고 INNER JOIN을 추가하고 WHERE절 대신 ON절을 사용해야 합니다.
- INNER는 생략 가능합니다.

- 아래의 두 SQL 문장을 비교해 보세요

-- INNER JOIN을 사용한 문장

```
SQL> SELECT e.empno, e.ename FROM dept d INNER JOIN emp e ON d.deptno = e.deptno;
```

-- 일반적인 SQL 문장

```
SQL> SELECT e.empno, e.ename FROM dept d, emp e WHERE d.deptno = e.deptno;
```

** NATURAL JOIN **

- Equijoin과 동일하다고 보시면 됩니다.
- 두 테이블의 동일한 이름을 가지는 칼럼은 모두 조인 이 됩니다.
- 동일한 컬럼을 내부적으로 찾게 되므로 테이블 Alias를 주면 오류가 발생 합니다.
- 동일한 컬럼이 두개 이상일 경우 JOIN~USING문장으로 조인되는 컬럼을 제어 할 수 있습니다.
- 아래의 두 SQL 문장을 비교해 보세요

-- NATURAL JOIN을 사용한 SQL 문장

```
SQL> SELECT empno, ename, deptno FROM emp NATURAL JOIN dept
```

-- Oracle9i 이전에 일반적인 SQL 문장

```
SQL> SELECT e.empno, e.ename, d.deptno FROM emp e, dept d WHERE e.deptno = d.deptno
```

** JOIN ~ USING **

- NATURAL JOIN의 단점은 동일한 이름을 가지는 칼럼은 모두 조인이 되었는데 USING 문을 사용하면 컬럼을 선택해서 조인을 할 수가 있습니다.
- USING절 안에 포함되는 컬럼에 Alias를 지정하면 오류가 발생 합니다.

-- 일반적인 사용 방법

```
SQL> SELECT e.empno, e.ename, deptno FROM emp e JOIN dept d USING(deptno)
```

** ON 구문 **

- 조인 조건을 지정 할 수 있습니다.
- 모든 논리 연산 및 서브쿼리를 지정할 수 있습니다.

-- 테스트를 위해 scott유저에서 아래 insert문장을 실행시켜 주세요

```
INSERT INTO bonus(ename, job, sal) VALUES('SMITH', 'CLERK', 500);
```

-- ON절 사용 예제 (multi - table joins)

```
SQL> SELECT e.empno, e.ename, e.sal
        FROM emp e JOIN dept d ON (e.deptno = d.deptno)
        JOIN bonus b ON (b.ename = e.ename)
        WHERE e.sal IS NOT NULL
```