Join

-**핵심 포인트**-

서브 쿼리
-select절에 포함된 또 다른 select 문장
-서브 쿼리가 먼저 실행

Join 연산
두 개의 테이블을 사용하는 연산
- cross join : 두 테이블의 칼럼 값에 대한 조건없이 모든 튜플의 결과를 검색
- inner join : 두 테이블의 지정된 칼럼 값이 일치하는 것만 검색
  ex) equi join, non-equi join
- outer join : 두 테이블의 지정된 칼럼 값이 일치하지 않는 것도 검색
  ex) left outer join, right outer join

1. 서브쿼리(SubQuery)

- select절에 포함된 또 다른 select 문장
- 알려지지 않은 데이터의 값을 검색하기 위해 사용
- 구문

```
select 컬럼명
from 테이블명
where 수식 연산자 (select 컬럼명
                   from 테이블명);
```

  - 서브쿼리는 order by절을 포함할 수 없다.
  - order by절은 메인 쿼리문(주 select)의 마지막에 위치한다.
  - 괄호안의 select 문장인 서브쿼리가 먼저 실행되어 메인 쿼리문장의 조건으로 전달되어 결과를 구한다.

[실습1] JONES와 같은 부서에 근무하는 사원들의 부서번호, 사원번호, 이름, 급여를 구하여라.

```
select deptno, empno, ename, sal
from emp
where ename = (select deptno
               from emp
               where ename = 'JONES');
```

[실습2] JONES와 같은 직무를 가진 사원들의 이름과 직무를 구하여라.

```
select ename, job
from emp
where job = (select job
             from emp
             where upper(ename) = 'JONES');
```

1-1 단일행 서브쿼리

- 단일행 서브쿼리는 select 문장으로부터 오직 한 개의 값을 결과로 산출한다.
- 단일행 비교연산자(=, <>, >, >=, <, <=)를 사용한다.
- 서브쿼리는 where절 뿐만 아니라, having 절에서도 사용가능하다.

[실습1] 'CLARK' 보다 급여를 많이 받는 사원의 이름과 직무를 구하여라.

```
select ename, job
from emp
where sal >
      (select sal
       from emp
       where ename = 'CLARK');
```

[실습2] 사원 전체의 평균 급여보다 급여가 적은 사원들의 이름과 급여를 구하여라.

```
select ename, sal
from emp
where sal <
    (select avg(sal)
     from emp);
```

[실습3] emp 테이블에서 부서별로 평균급여가 부서번호 30인 사원들의 평균급여보다 큰 부서의 부서번호와 평균 급여를 구하여라.

```
select deptno, avg(sal)
from emp
group by deptno
having avg(sal) >
     (select avg(sal)
      from emp
      where deptno = 30);
```

1-2 복수행 서브쿼리

- 서브쿼리인 select 문장으로부터 여러 개의 값을 결과로 산출한다.
- 복수행 연산자(in, not in, any, all)을 사용

[실습1] in 연산자

- 메인 쿼리의 비교 조건이 서브쿼리의 결과중에서 하나라도 일치하면 참이 된다.
- 예) 부서별로 이름의 최소값을 가진 사원의 이름과 급여, 부서번호를 구하세요.

```
select ename, sal, deptno
from emp
where ename in
   (select min(ename)
    from emp
    group by deptno);
```

[실습2] any 연산자

- 서브쿼리가 반환하는 각각의 값과 비교하여 하나라도 만족되면 결과에 포함.
- <any는 최대값보다 작음을 나타내고
- >any는 최소값보다 큼을 나타내고
- =any는 in과 동일한 역할을 한다.
- 예) 부서번호가 30번인 사원들의 급여 중 가장 낮은 값보다 높은 급여를 받는 사원의 이름, 급여를 구하세요.

```
select ename, sal
from emp
where sal > any
   (select sal
    from emp
    where deptno = 30);
```

[실습3] all 연산자

- 서브쿼리가 반환하는 모든 값과 비교한다.
- >all은 최대값보다 큼을 나타내고
- <all은 최소값보다 작음을 나타낸다.

예) 부서번호가 30번인 사원들의 급여 중 가장 높은 값보다 더 많은 급여를 받는 사원의 이름, 급여를 구하세요.

```
select ename, sal
from emp
where sal > all
   (select sal
    from emp
    where deptno = 30);
```

2. Join

- inner join : join에 참여하는 모든 테이블에 데이터가 존재하는 경우에만 결과를 출력
  (ex. equi join, non-equi join)
- outer join : 조인에 참가할 때, 한쪽 테이블에는 데이터가 있고, 다른 한쪽 테이블에 없는 경우에 데이터가 있는 쪽 테이블 내용을 전부 출력
  (ex. left outer join, right outer join)
- 두 개 이상의 테이블로부터 데이터를 검색하기 위하여 키 관계를 이용하여 두 개 이상의 테이블을 연결시키는 것을 의미한다.
- where절에 반드시 테이블의 조인 조건을 명시해야 한다.
- 조인 조건이 생략되면 cartesian product가 발생하며, 이것을 cross join이라 한다.

예) select empno, ename, sal, dname
    from emp, dept;

2-1 Equi Join

- 조인하는 테이블 사이에서 공통으로 존재하는 컬럼의 값이 일치되는 행을 찾아 연결하여 결과를 생성
- where절의 조인 조건에 = 연산자를 사용한다.
- 구문

```
select 테이블명1.컬럼명1, 테이블명2.컬럼명2, ....
from 테이블명1, 테이블명2
where 테이블명1.컬럼명1 = 테이블명2.컬럼명2, .... ;
```

- 컬럼명앞에 테이블명을 붙여 사용한다.

[실습1] emp, dept테이블에서 사원 이름, 부서번호, 부서 이름을 구하세요.

```
select emp.ename, emp.deptno, dept.dname
from emp, dept
where emp.deptno = dept.deptno;
```

[실습2] 테이블에 별칭 부여시엔 별칭 꼭 사용해야함.

```
select e.ename, e.deptno, d.dname
from emp e, dept d
where e.deptno = d.deptno;
```

[실습3] scott의 부서와 같은 부서 번호를 갖는 사원의 이름, 부서번호, 부서이름을 구하세요.

- 주 select문에서 테이블 별칭을 사용해도 서브쿼리의 select문에는 영향을 미치지 않는다.

```
select e.ename, e.deptno, d.dname
from emp e, dept d
where e.deptno = d.deptno
and e.deptno =
  (select deptno
   from emp
   where ename = 'SCOTT');
```

2-2 Non-Equi Join

• 조인하는 두 개의 테이블의 특정 컬럼에 직접 일치하지 않을 경우에 사용
• 조인할 테이블사이에 일차키와 외래키의 관계가 없다.
• 조인 조건에 = 이외의 연산자를 사용한다.
• <=, >=, between...and (하한값 먼저 명시) 연산자 사용가능

[실습1] emp, salgrade 테이블로부터 사원의 이름, 직무, 급여, 급여 등급을 구하세요.

```
select e.ename, e.job, e.sal, s.grade
from emp e, salgrade s
where e.sal between s.losal and s.hisal;
```

2-3 Outer Join

• 조인 조건을 만족하지 못하는 행들을 출력하기 위해 사용
• 구문

```
select 테이블명1.컬럼명1, 테이블명2.컬럼명2, ....
from 테이블명1, 테이블명2
where 테이블명1.컬럼명1 = 테이블명2.컬럼명2(+) ;
```

또는

```
select 테이블명1.컬럼명1, 테이블명2.컬럼명2, ....
from 테이블명1, 테이블명2
where 테이블명1.컬럼명1(+) = 테이블명2.컬럼명2 ;
```

- (+) 기호는 조인 조건에서 정보가 부족한 테이블의 칼럼명 뒤에 사용한다.

[실습1] emp, dept테이블에서 사원의 이름, 부서 번호, 부서 이름을 구하세요.

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

(1) left outer join : 왼쪽 테이블이 기준이 되어 모든 데이터를 가져오고, 오른쪽 테이블에서 가져올 수 없는 열은 null로 출력

```
select 테이블명1.컬럼명1, 테이블명2.컬럼명2, ....
from 테이블명1 [ left | right ] outer join 테이블명2
on 조인 조건
```

예) 사원이름과 사원에 해당하는 매니저를 출력

```
select e1.ename, e2.ename
from emp e1 left outer join emp e2
on e1.mgr = e2.empno;
```

(2) right outer join : 오른쪽 테이블이 기준이 되어 모든 데이터를 가져오고, 왼쪽 테이블에서 가져올 수 없는 열은 null로 출력

예) 사원 이름, 부서 번호, 부서 이름을 출력

```
select e.ename, d.deptno, d.dname
from emp e right outer join dept d
on e.deptno = d.deptno;
```