Chapter 3 데이터 수정

insert
update
delete - 각 행을 삭제
------ 저장을 꼭 해야(처리) 반영됨
commit : 완료 명령어 필요, 안 쓰면 그대로 남아있음.

merge - 양쪽 dataset 공통되어있는 것 수정 and 없는 것 insert
drop - emp 테이블 자체를, 즉 구조를 삭제
truncate - 한꺼번에 전체 행을 날려버림, 구조는 삭제하지 않음

Chapter 4 조인과 서브쿼리

1. Join 이란 ?

- **WHERE 절에서 조인 조건을 작성한다**.
  -> 작성하지 않으면 CARTESIAN PRDOUCT 발생
- 동일한 열 이름이 여러 테이블에 있는 경우,
  열 이름 앞에 테이블 이름이나 테이블 ALIAS 를 붙인다.
(원 테이블 이름 + ALIAS 혼용 불가능함. alias 사용시 무조건 별칭으로 사용)
- N 개의 테이블을 조인하려면, 최소 N-1 개의 조인 조건이 필요하다.

cf. UNION - 수직(상하)으로 합침
JOIN - 수평(좌우)로 합침

1) Cartesian Product

- n 개의 행을 가진 table1 과 M 개를 가진 table2 의 카티시안 곱은 N*M 이다.
  (모든 경우의 수의 행이 출력됨)
- 다음의 경우 카티시안 곱이 생성된다.
  · 조인 조건을 생략한 경우
  · 조인 조건이 부적합할 경우
- 첫 번째 테이블의 모든 행이 두 번째 테이블의 모든 행에 조인되어 처리된다.
- 카티시안 곱이 생성되지 않도록 하려면, WHERE 절의 조인을 유효하게 지정해야 한다.

```
select *
 from emp, dept;
```

---- 14 * 4 건---
카티시안 곱 형태로 나타남. 부서번호가 같다는 것을 명시하지 않았음

```
select *
 from emp, dept;
where deptno = deptno (X)
```

칼럼명 중복 -> alias 사용

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

----------14 건----------

2) EQUI JOIN (등가 join)

SELECT table1.column, table2.column
FROM table1, table2
WHERE table1.column1 = table2. column2 -- 조인조건
and table1.column1 = ... -- 일반조건
(조인조건, 일반조건 순서 상관 X)

- 조인 대상 테이블에서 공통 칼럼을 '=' (equal) 비교를 통해 같은 값을 가지는 행을 연결하여 결과를 생성하는 조인 방법이다.
- SQL 명령문에서 가장 많이 사용하는 방법이다
- FROM 절에 조인하고자 하는 테이블을 모두 명시한다.
- FROM 절에 명시하는 테이블은 테이블 별칭(Alias)을 쓸 수 있다.
- WHERE 절에 두 테이블의 공통 컬럼에 대한 조인 조건을 나열한다.

학생테이블과 학과 테이블을 equi join 하여 이름, 학과번호, 소속학과 이름을 출력하라

```
----------------------------
select name "학생이름", deptno1 "학과번호", dname "학과이름"
 from student, department
where deptno1=deptno ;
----------------------------
table alias 사용 -> select column 명 조심
---------------------------------
select name 학생이름, deptno1 학과번호, dname 학과이름
 from student, department
where deptno1=deptno
 and grade = 4 ;      --일반조건
---------------------------------
```

Q. select 에 alias 안 쓰면? -> 중복된 명이 아닌 이상 괜찮은 것 같다.

@2) 학생테이블과 교수 테이블을 join 하여 학생의 이름, 지도교수번호, 지도교수 이름을 출력하라

```
-----------------------------
select s.name, p.profno, p.name
 from student s , professor p
 where s.PROFNO = p.profno;
---------------------------------------
```

-> 조건이 만족하지 않으면 생략함 = natural join - equi join 특징

cf. outer join 의 목적 -> 생략된 데이터도 출력하기 위해서
방법 : 두 개 중 하나만 기준이 되는 테이블(생략이 없었으면 하는 테이블)의 반대쪽에 (+) 붙이기

```
    select
```

from
where s. profno = p.profno**(+)**
/ 왼쪽이 기준이면 오른쪽에 (+)표시

지도교수 별 지도 학생 수

----------**1. 안되는 경우**----------

```
select s.name, p.profno, p.name, count(s.name)
 from student s , professor p
 where s.PROFNO = p.profno
 group by p.name;
```

---

s.name, p.profno : **group by 에서 정의된 column 이 아니어서!**

----------**2. 되는 경우**----------

```
select p.name, count(s.name)
 from student s , professor p
 where s.PROFNO = p.profno
 group by p.name;
```

---

cf. -> count(s.studno) 가 s.studno 가 not null 이므로 더 정확하게 count 나타내줄 수 있다.

테이블 3 개

```
select s.name "학생이름", d.dname "학과이름", p.name "교수이름"
 from student s, department d, professor p
where s.deptno1 = d.deptno
  and s.profno = p.profno;
```

---

3) Non-Equi Join (=조건이 아닌 조건을 사용할 경우)

테이블을 연결짓는 조인 컬럼에 대한 비교 조건이 <, BETWEEN a AND b 등
'=' 조건이 아닌 연산자를 사용하는 경우의 조인조건이다.

Gogak 테이블과 gift 테이블을 Join 하여 고객의 마일리지 포인트별로 받을 수 있는 상품을 조회하여 고객의 이름과 상품명을 출력하세요

----------틀렸당 실수 조심----------

```
select g1.GNAME as 고객명,
       g2.GNAME as 상품명
 from GOGAK g1, GIFT g2
where g1.POINT between g2.G_START and g2.G_END;
```

---

between A and B 형태 헷갈림. between(,) 이렇게 함..

Gogak 테이블과 gift 테이블을 join 하여 고객이 자기 포인트보다 낮은 포인트의 상품 중 한 가지를 선택할 수 있다고 할 때 선택할 수 있는 모든 경우의 고객명과 상품명을 출력하라

```
select g1.GNAME as 고객명,
       g2.GNAME as 상품명
 from GOGAK g1, GIFT g2
where g1.POINT >= g2.G_START ;
```

G_END 로 기준하면 모든 포인트가 G_END 가 제일 큰 것보다 다 작으므로 기준 삼기 어렵다.
따라서 **G_START 보다 크다고 가정해야 가져갈 수 있는 모든 목록** 찾을 수 있음!

위 문제에서 점주가 제품별 최대 준비수량? - 실습 4

```
select g2.gname, count(g1.gname)
from GOGAK g1, GIFT g2
where g1.POINT >= g2.G_START
group by g2.gname;
```

학생 이름, 점수, 성적(성적이 구간으로 걸쳐져있음)

```
select s.name "이름", e.total "점수", h.grade "학점"
 from student s, exam_01 e, hakjum h
where s.studno = e.studno
  and e.total between h.min_point and h.MAX_POINT ;
```

◆ 실습문제

1. emp2 테이블과 p_grade 테이블을 조회하여 사원의 이름과 직급, 현재 연봉, 해당 직급의 연봉의 하한금액과 상한금액을 출력하세요.

sol)

```
select e.NAME, e.position, e.pay, p.s_pay, p.E_PAY
 from emp2 e, p_grade p
where e.POSITION = p.POSITION;
select * from professor;
```

2. 1 전공이 101 번인 학생들의 학생 이름과 지도교수 이름을 출력하세요

sol)

```
select s.name, p.NAME
 from student s, professor p
where s.PROFNO = p.PROFNO
 and s.deptno1 = 101;
```

Gogak 테이블과 gift 테이블을 join 하여 고객이 자기 포인트보다 낮은 포인트의 상품 중 한 가지를 선택할 수 있다고 할 때, 산악용 자전거를 선택할 수 있는 고객명과 포인트, 상품명을 출력하세요

sol)

```
 select g1.gname, g1.POINT, g2.GNAME
 from gogak g1, gift g2
where g1.point >= g2.G_START
  and g2.gname = '산악용자전거'
```

4. Gogak 테이블과 gift 테이블을 join 하여 고객이 자기 포인트보다 낮은 포인트의 상품 중 한 가지를 선택할 수 있다고 할 때 점주 입장에서는 각 상품별로 최대 몇개의 상품이 필요한지 각 상품별 수량을 출력하세요

sol)

```
select g2.gname, count(g1.gname)
from GOGAK g1, GIFT g2
where g1.POINT >= g2.G_START
group by g2.gname;
```

5. emp2 테이블과 p_grade 테이블을 조회하여 사원들의 이름과 나이, 현재직급, 예상직급을

출력하세요. 예상직급은 나이로 계산하며 소수점 이하는 생략하세요.

sol)

```
select e.name, trunc((sysdate - e.birthday)/365) as "나이", e.POSITION as "현재직급" , p.position as "나이에 따른 예상직급"
  from emp2 e, p_grade p
 where trunc((sysdate - e.birthday)/365) between p.s_age and p.E_age;
```

◆ 전반적인 심화문제

--1. student 에서 각 학생의 학점을 구하고 각 학점별로 분포된 학생 수를 다음과 같이 구하세요
-- A 4
-- B 12
-- C 3
-- D 1

```
select substr(h.grade,1,1), count(s.studno)
 from student s, exam_01 e, hakjum h
where s.studno = e.studno
  and e.total between h.min_point and h.MAX_POINT
group by substr(h.grade,1,1)
order by substr(h.grade,1,1);
```

--2. p_grade 에는 각 직급을 구하는 기준과 각 직급별 기준 연봉의 상, 하한 정보가 존재한다.
--emp 테이블의 각 직원의 근속년수를 기준으로 p_grade 에서 새롭게 정의된 position 정보를
--각 직원의 이름, sal 과 함께 나타내어라.

sol)

```
select e.ename, e.sal, p.position
 from emp e, p_grade p
where trunc(months_between(sysdate,e.hiredate)/12) between p.S_YEAR and E_YEAR;
```

--3. emp 테이블과 SALGRADE 테이블을 이용하여 각 직원의 sal 별 등급을 구하고 각 등급별 평균 연봉을 최종적으로
--나타내어라. 단, 평균연봉은 소수 첫번째 자리에서 반올림하여 정수로 나타낸다.
-- 1 950
-- 2 1267
-- 4 2855
-- 5 5000
-- 3 1550

SOL)

```
select s.grade, round(avg(sal))
 from emp e, salgrade s
where e.sal between s.losal and s.hisal
group by s.grade;
```

--4. student 테이블과 exam_01 테이블을 이용하여 각 학과별 평균 점수와 최고 점수, 최저 점수를 나타내되
--학과이름, 학과번호와 함께 출력되도록 작성하여라.

SOL)

```
select d.dname, s.deptno1, avg(e.TOTAL), max(e.TOTAL), min(e.TOTAL)
 from student s, exam_01 e, department d
where s.STUDNO = e.STUDNO
  and s.DEPTNO1 = d.DEPTNO
group by d.DNAME, s.deptno1;
```

*학과 : dname 과 deptno 일대일관계. group 이 더 세분화되지 않음. 한 개씩 group by 쓸 때랑 같음

-> 함께 group by 에 올리고 select 에도
만일 location 까지 하면 역시 group by 에 올린 후! select 에 설정

--5. 레포트를 작성하고자 한다.
--emp 테이블을 이용하여 각 부서별 직원수를 출력하되 다음과 같은 형식으로 작성하여라.
--
--레포트명 10_직원수 20_직원수 30_학생수

--홍은혜 레포트 3 5 6

뭔가 join 이 **좌우로** 합치는 것이어서 잘 수정하면 될 것 같음

1. 10_직원수 / 20_직원수 / 30_직원수 따로 나눴다.
이걸 어떻게 join 형태로 합치지?
2. dual 사용하려다가멈춤
3. deptno 가 다른 곳. department 테이블도 참조?
4. emp e1, emp e2, emp e3 이렇게 나누는데도 잘 안됨
SOL)
레포트 : 컬럼이름, 홍은혜레포트 : 데이터
방법

```
select deptno, decode(deptno,10,1)
 from emp;
```

/* sum : 1 로,
 count : 모든 문자/숫자/날짜형태
(sm 이랑 count 가 null 을 사용하지 않는 것 유의)

```
select '홍은혜 레포트' as "레포트명",
       sum(decode(deptno,10,1)) as "10_직원수",
       sum(decode(deptno,20,1)) as "20_직원수",
       sum(decode(deptno,30,1)) as "30_직원수"
 from emp;
```

*표현식 사용
*column 별의 조건을 이용한 sum / count 로 표현할 수 있는 방법
 column 처럼 조건을 활용할 수 있다.
***열별(한줄로)** group 형태 표현 방법임 - R, python 이전에 먼저 활용 - **그래프 그릴 때 활용**
cf. group by : 행별 정렬

----------나의 흔적들

```
select count(e.EMPNO), count(e.EMPNO), count(e.empno)
 from emp e
 group by e.deptno
 having e.deptno = 10 and e.deptno = 20 and e.deptno = 30;

select count(E.empno) as "10_직원수", count(e.empno) as "20_직원수", count(e.empno) as
"30_직원수"
 from emp e
 where e.DEPTNO = 10
```

;

```
select count(e1.empno) as "10_직원수", count(e2.empno) as "20_직원수", count(e3.empno) as "30_직원수"
 from EMP e1, EMP e2, EMP e3;
 group by e1.deptno
```

```
select count(empno) as "20_직원수"
 from EMP
 where deptno=20

select count(empno) as "30_직원수"
 from EMP
 where deptno=30
```

```
select count(empno1)
 from EMP;
```

180504 SQL 마지막!

4) OUTER JOIN - 조인조건이 만족하지 않는 정보도 출력할 목적으로 <-> NATURAL JOIN
-EQUI JOIN 조인 조건에서 NULL 값을 가진 행은 출력되지 않음
-NON EQUI JOIN 조인 조건을 만족하지 않는 행은 출력하지 않음
----------NATURAL JOIN--------

-**(+)** : 모두 출력하고자 하는 **기준**의 반대쪽에 기호 사용!
-조건 절에 표시!!! (FROM 절 (X))
-in ANSI Outer JOIN - LEFT OUTER JOIN / RIGHT OUTER JOIN 써야 함.

1. 두 테이블 간 관계에 있는 조건이 여러 개이면, 관계에 대한 모든 조건에 동일한 방향으로 (+)
2. 테이블이 연결된 관계일 때, 연결되어 있는 모든 조건들을 (+) 표시
cf. 맨 마지막장 문제

?full outer join

student 테이블과 professor 테이블을 조인하여 학생 이름과 지도교수 이름을 출력하세요
단, 지도교수가 결정되지 않은 학생의 명단도 함께 출력하세요
-**학생의** 교수 정보 나열

```
select  s.name, p.NAME
 from student s, professor p
where s.PROFNO = p.PROFNO(+);
```

student 테이블과 professor 테이블을 조인하여 학생 이름과 지도교수 이름을 출력하세요
단, 지도교수가 결정되지 않은 학생의 명단도 함께 출력하세요
-**교수의** 학생 정보 나열

```
select  s.name, p.NAME
 from student s, professor p
where s.PROFNO(+) = p.PROFNO;
```

cf - 한 명의 교수가 여러 명을 맡는 경우가 있어서 원래 교수 정원보다 많게 나옴
order by p.name;

지도교수별 지도학생의 수를 출력하라. 단, 지도 학생이 없는 교수도 모두 출력하라

```
select p.name, count(s.name)
 from student s, professor p
where s.PROFNO(+) = p.PROFNO
group by p.name;
```

지도교수별 학생 수를 출력하되, 지도교수의 입사일, 페이, 포지션도 추가하라
-> 지도교수 동명이인이 없을 때, group by 에 추가 column 사용 가능

```
select p.name, count(s.name) , p.hiredate, p.pay, p.position
 from student s, professor p
where s.PROFNO(+) = p.PROFNO
group by p.name, p.hiredate, p.pay, p.position;
```

- 그룹에 대한 정보를 추가하는 건 별로 어렵지 않다

지도학생 X 교수, 지도교수 X 학생 모두 표시
-> UNION 사용하여

5) SELF JOIN

- 본인의 테이블을 두 번 이상 사용
- for 본인의 테이블에 원하는 정보가 다 있을 때,
  한 번의 액세스로(행별 스캔) 불가능한 수행
- 반드시 테이블 알리아스 필요!

부서명과 상위부서명을 같이 출력하라

```
select a.dname "부서명", b.dname "상위부서명"
 from dept2 a, dept2 B
where a.PDEPT = b.Dcode;
```

emp 테이블을 사용하여 각 직원의 이름, 사원번호, 상위관리자 이름을 동시에 출력해보자

```
select a.ename, a.empno, b.ENAME
 from emp a, emp b
where a.mgr = b.empno;
```

◆ 실습 문제

--실습 1

emp2 테이블과 p_grade 를 이용하여 사원이름, 현재직급, 현재직급이 하한연봉, 상한연봉을 함께 출력하여라. 단, 직급이 부여되지 않은 직원들에 대해서도 나타나도록 하여라.

sol)

```
select e.name, e.POSITION, p.s_PAY, p.e_PAY
 from emp2 e, p_grade p
where e.POSITION = p.POSITION(+);
```

--실습 2

professor 테이블에서 교수의 번호, 이름, 입사일, 자신보다 입사일 빠른 사람 인원수를 출력하라. 단, 자신보다 입사일이 빠른 사람수를 오름차순으로 출력하라.

sol)

```
select a.profno, a.name, a.hiredate, count(b.profno) as "선배수"
 from professor a, professor b
where a.hiredate > b.hiredate(+)
group by a.profno, a.name, a.hiredate
order by 4;
```

2. 서브쿼리 Sub Query

1) SubQuery 란?

- subquery 문법

```
SELECT  select_list
 FROM  table 또는 view
WHERE  조건 연산자  (SELECT select_list
                    FROM table
                    WHERE 조건) ;
```

- WHERE 절 오른쪽에 위치해야 하고, 반드시 괄호로 묶어야 함
- 특별한 경우를 제외하고(Top-n 분석 등) Sub Query 절에 Order by 절이 올 수 없음
- 단일행 sub query 와 다중행 wub query 에 따라 연산자를 잘 선택해야 함

목적

where 절에 쓰는 subquery - 일반적

대개 for 그룹을 이용한 정보를 통해 개별 데이터 찾아가기 위함

cf.

sbquery 중 from 절에 쓰이는 건 inline view **for** 일부 추려서, 데이터를 가공한 형태로 사용

select 에 쓰이는 건 scala subquery **for** 쿼리절 결과값을 하나의 컬럼처럼 사용

2) Sub Query 의 몇 가지 유형

(1) Single Row Sub Query (**단일 행** Sub Query)

```
=
<>
>
>=
<
<=
```

틀린 경우

```
select *
  from EMP
 where sal > (select *
               from EMP
              where ename like 'S%');
--------------------------------
```

column 1 개 vs 모든 column *

```
-------------------------
select *
  from EMP
 where sal > (select sal
                from EMP
               where ename like 'S%');
-------------------------
```

(비교하려는 것 : 단일행, 서브쿼리 안의 결과 : 두 행)

>을 in 으로 바꾸기

질의 완성

```
--------------------------------
select *
  from EMP
 where sal > (select avg(sal)
                from EMP
               where ename like 'S%');
-----------------------------------
```

(2) Multi Row Sub Query (다중 행 서브 쿼리)

연산자	의미
IN	**같은 값을 찾음 (or 의미)**
>ANY	최소값을 반환함=>단일행 MIN 사용 가능
<ANY	최대값을 반환함 반환함 =>단일행 MAX 사용 가능
<ALL	최소값을 반환함 =>**단일행 MIN 사용** 가능
>ALL	최대값을 반환함 =>**단일행 MAX 사용** 가능

- 대소비교 : 단일행으로만 가능함

문제

S 로 시작하는 직원들의 연봉과 정확하게 일치하는 사원 찾기

=> 하나로 합칠 수 없음, 각각 전달해야. 단일연산자 사용 불가

```
---------------------------------
select *
  from EMP
 where sal in(select sal
                from EMP
               where ename like 'S%');
-----------------------------------------
```

- in : 다중행 연산 가능하게 하는 연산자.

3) Multi Column Sub Query(다중 컬럼 서브 쿼리) - 그룹별 대소비교 불가능하다 -> 인라인뷰

------emp 테이블에서 각 부서별 최대 연봉을 받는 사람 출력

```
select *
  from EMP
  where (deptno, sal) in (select deptno, max(sal)
                          from EMP
                          group by deptno);
```

--학생테이블, 학년별 최대 키를 갖는 친구의 정보 출력

```
  select *
   from STUDENT
   where (grade, height) in (select grade, max(height)
                             from STUDENT
                             group by grade)   ;
```

메인쿼리에서는 group by 사용 X -> select 에서 모든 column 선택 가능

cf.

```
select *
from student
where height in (select max(height)
                 from student
                group by grade);
```

가 안 되는 이유!

◆ **실습 문제**

1. emp 테이블을 사용하여 smith 와 같은 부서에 있는 사원의 이름과 직업을 출력하세요.

```
select ename, job
  from EMP
  where deptno = (select deptno
                  from EMP
                  where upper(ename) = upper('smith'));
```

2. student 테이블에서 제 1 전공이 101 번인 학과의 평균몸무게보다 몸무게가 많은 학생들의 이름과 몸무게를 출력하세요.

```
select name, weight
  from STUDENT
 where weight > (select avg(weight)
                  from STUDENT
                  where deptno1 = 101);
```

3. professor 테이블에서 직급별 최대 연봉을 갖는 교수의 이름, 입사날짜, 보너스 정보를 출력하세요.

```
select name, hiredate, BONUS
   from PROFESSOR
  where (position, pay) in (select position, max(pay)
                            from PROFESSOR
                           group by position);
```

★ 인라인 뷰 ★

- **그룹 내 대소비교 할 때** 쓰이는 형태! (다중컬럼 서브쿼리 약점 보완)
- **인라인뷰 내에 있는 그룹함수의 결과값은 메인쿼리 절에 그 형태 그대로 전하면 안 됨.**
  (error 발생)
- **따라서 인라인뷰 내에 그룹함수는 반드시 메인쿼리에 알리아스를 통한 전달만이 가능하다.**

+ professor 포지션별로 평균보다 높게 받는 사람은?

----------**옳은 인라인 뷰 형태**----------

```
select *
 from professor p, (select position, avg(pay) as avg_pay
                    from PROFESSOR
                   group by position) i
 where p.position = i.position
   and p.pay > i.avg_pay ;
```

+뷰형태는 반드시 알리아스를 사용해야 인라인뷰 내에 있는 컬럼을 가져올 수 있다.
+그룹함수형태의 column 은 반드시 알리아스를 사용해라

테이블은 아닌데 테이블 형태로 쓰고 싶은 객체 : 뷰

from - 테이블, 뷰 가능.

---

1. 틀린 형태 (연산자 분리)

```
-----------------
select name, hiredate, BONUS
   from PROFESSOR
  where pay > (select avg(pay)
                      from PROFESSOR
                    group by position)
    and position in (select position                -- 항상 참인 조건, 비교로 유효하지 않음
                  from PROFESSOR
                  group by position);
```

2. 오류 형태 (인라인뷰 칼럼 알리아스 X)

```
select *
 from professor p, (select position, avg(pay)
                      from PROFESSOR
                    group by position) i
 where p.position = i.position
   and p.pay > avg(i.pay) ;
-------- -----------------
```

(테이블 알리아스는 컬럼 앞에 사용! 그런데 where 절에는 group 함수 못사용..
미리 구한 걸 땡겨온 목적인데 where 절에 avg 를 쓰는 순간 문법오류 남. avg(i.pay)
인라인뷰 안에서 그룹함수를 사용한 결과값을 메인쿼리 값과 비교할 목적으로 그룹함수를 쓰면 인라인뷰에서의 그룹함수를 가져오라는게 아니라, 다시 구해라는 명령으로 잘못 전달되므로
where 절에는 그룹함수 아예 사용 불가, 따라서 서브쿼리절에 그룹함수를 알리아스로 사용해야 where 절에서 알리아스 사용하면서 가능!)

실습문제

emp 테이블에서 각 부서 내에서 그 부서의 평균연봉보다 높은 직원의 정보를 출력하라.

```
select e.ename, e. deptno, e.sal, a.avg_sal
 from emp e, (select deptno, avg(sal) as avg_sal
              from EMP
              group by deptno ) a
 where e.deptno = a.deptno
   and e.sal > a.avg_sal;
```

다중쿼리보다 인라인뷰가 더 포괄적. max, min 도 역시 사용 가능.

참고

상호연관 서브쿼리 - where 절에 그대로 존재

for 다중컬럼서브쿼리 한계 극복

```
select *
   from PROFESSOR p1
  where pay > (select avg(pay)
                from PROFESSOR p2
                where  p1.position = p2.position
                 group by position);
```

과정

1. main query 먼저 실행
   1 행을 불러옴

2. 이미 메인쿼리의 행의 정보가 서브쿼리 p1.position 으로 들어옴
   -> group by position 필요 없음!

3. 1-2 과정 16 번 모두 반복

서로 필요한 정보를 주고 받음 = 상호연관 서브쿼리
인라인뷰랑 비슷. 데이터가 작을 땐 종종 쓰임.

하지만 너무 많은 데이터가 있는 경우는 사용하지 않음.

(4) 스칼라 서브쿼리 - 참고

- 셀렉트 절에 올리는 서브쿼리
- 박승곤의 포지션을 바꿀 예정인데, 조인형 교수와 항상 똑같은 position 을 유지하도록
- just for 정보 표현

```
--------------------------------
select name, (select position
              from professor
              where name='조인형') as position
 from professor
 where name = '박승곤';
----------------------------
```

★심화문제

1. professor 테이블에서 입사년도(1980, 1990, 2000 년대 등)별 평균연봉보다 높은 연봉을 받는 교수의 이름, 학과명, 연봉을 구하여라.

```
/*
입사년도별평균     대소비교 - 학과명
(교수)             - 교수 1  - 학과
*/
```

-목적이 두 번 : from 절이 두 번
-날짜컬럼 - substr 은 정확히 원본 데이터를 알 때만 사용할 것 => 일반적으론 to_char 이용 권장
-trunc 함수 사용 (날짜형인 경우 year, month 사용 ! 그런데 지금 문자이므로 그냥 사용
정수 첫 번째 자리 버리므로 -1 씀

1.

```
select trunc(to_char(hiredate,'yyyy'),-1), avg(pay)
from professor
group by trunc(to_char(hiredate,'yyyy'),-1);
```

---

sub query 필요
다중컬럼 ~ 원데이터 가공한 정보 사용. 대소비교는 불가능. 오직 =만 -> **inline view** 사용
/원데이터로 그룹에 대한 정보 가질 수 없음 - 새로 만들어서 from 절에 넣음

2.

```
select name, deptno, pay
 from professor p, (select trunc(to_char(hiredate,'yyyy'),-1) as hiredate_yy, avg(pay)  as avg_pay
                    from professor
                    group by trunc(to_char(hiredate,'yyyy'),-1)) i
where trunc(to_char(p.hiredate,'yyyy'),-1) = i.hiredate_yy
 and p.pay > i.avg_pay
;
```

-where 절 안 쓰면 카티시안 곱 나타남
-함수의 경우 컬럼이 쓰인 곳에 테이블 Alias 쓰기
-from 절에 올린 인라인뷰 - 나중에 쓸 column 유의하기 (나중에 where 절에서 잘 살펴보기)
-인라인뷰 알리아스 쓰기(함수, 특히 그룹함수 - 그냥 쓰면 에러발생)

3. 학과명

```
select p.name, d.dname, p.pay
 from professor p, (select trunc(to_char(hiredate,'yyyy'),-1) as hiredate_yy, avg(pay)  as avg_pay
                    from professor
                    group by trunc(to_char(hiredate,'yyyy'),-1)) i ,
      department d
where trunc(to_char(p.hiredate,'yyyy'),-1) = i.hiredate_yy
 and p.pay > i.avg_pay
 and p.deptno = d.deptno
;
```

내 방식

```
select p.name, d.dname, p.pay
  from professor p, (select substr(hiredate,1,1) as ipsa, avg(pay) as avg_pay
                     from professor
                     group by substr(hiredate,1,1)) i,
       department d
 where substr(p.hiredate,1,1) = i.ipsa
   and p.pay > i.avg_pay
   and p.deptno = d.deptno
;
```

cf. 보여주려고

```
 decode(substr(hiredate,1,1),8,'1980 년대',9,'1990 년대',0,'2000 년대',1,'2010 년대')
```

2. emp 테이블에서의 각 부서별 최고연봉자의 이름, 부서명, 연봉을 구하고 상위관리자와 상위관리자의 연봉도 함께 나타내도록 하여라.

부서별 최고연봉 부서별 최고연봉자 - 부서명 - 상위관리자
(emp) - emp1 -부서 1 - emp2

```
select e1.ename, e1.sal, d.dname, e2.ename, e2.sal
  from emp e1, emp e2, dept d
 where (e1.deptno, e1.sal) in (select deptno, max(sal)
                               from EMP
                               group by deptno)
    and e1.deptno = d.deptno
    and e1.mgr = e2.empno(+)
;
```

(KING 은 상위관리자가 없어서 안 나타남. NULL 이라서)

3. student 테이블에서 각 학년별로 본인보다 생년월일이 빠른 학생의 수를 출력하되,
각 학생의 이름, 학과명, 시험성적과 함께 출력하여라.
단, 모든 학생의 정보가 출력될 수 있도록 하자

학년별 생년월일 **비교** - 학과명- 성적
(학생 **1**-학생 **2)** **-** 학과 **-** 성적

1.

```
select s1.name as name, count(s2.name) as count_name
                        from student s1, student s2
                        where s1.grade = s2.grade(+)
                         and s1.BIRTHDAY > s2.BIRTHDAY(+)
                        group by s1.name ;
```

-생략된 정보

e - s1 - s2 **/관계도 그리면서 outerjoin 사용 권장**

I

d

-어느 관계에서 생략되었는지 살펴봐야 함

OUTERJOIN

1. self 조인을 하면서 본인보다 생년월일이 빠른 사람을 찾을 때 **생략이 되었음**
2. 그래서 s1.birthday > s2.birthday(+)을 했지만 여전히 생략됨.
그러나 s1 -s2 관계에 따르면, 조건 중 하나라도 null 이면 가져올 수 없게 된다.
따라서 관계에 있어서 추가된 조건이 있다면, **모든 조건의** 같은 방향에서 (+)를 표현해야 함.

```
select s1.name as name, count(s2.name) as count_name, d.dname, e.total
                        from student s1, student s2, department d, exam_01 e
                        where s1.grade = s2.grade(+)
                         and s1.BIRTHDAY > s2.BIRTHDAY(+)
                         and s1.deptno1 = d.deptno
                         and s1.studno = e.studno
                        group by s1.name, d.dname, e.total ;
```

-s2 for 친구 정보, 따라서 s1 과 join 해야 함. 그런데 친구의 학과/정보가 궁금할 때에는 s2 와 join
-학생의 **이름 - 학과 정보 - 점수** 한꺼번에 묶어도 개인에 대한 개별적인 정보를 가지고 있다. 따라서 이렇게 묶은 것도 개별적인 의미를 가지고 있음.

cf. 학생에 대한 e.total 이 여러개인 경우는 불가능? 이 테이블에서 다 각각 하나씩 가지고 있으므로 그랬던 것 같다.

내방법 수정

```
select ss.name, d.dname, e.total, ss.count_name
from exam_01 e, (select s1.name as name, s1.studno as studno, s1.deptno1 as deptno1, count(s2.name) as count_name
                        from student s1, student s2
                        where s1.BIRTHDAY > s2.BIRTHDAY(+)
                         and s1.grade = s2.grade(+)
                        group by s1.studno, s1.name, s1.deptno1) ss,
      department d
where ss.studno = e.studno
 and ss.deptno1 = d.deptno
;
```

내방법

```
select s.name, d.dname, e.total, ss.count_name
from student s, exam_01 e, (select s1.studno as sstudno, count(s2.name) as count_name
                        from student s1, student s2
                        where s1.BIRTHDAY > s2.BIRTHDAY(+)
                         and s1.grade = s2.grade(+)
                        group by s1.studno) ss,
      department d
where s.studno = ss.sstudno
 and s.studno = e.studno
 and s.deptno1 = d.deptno
;
## group by 문제에 없어도 잘 생각하기!!!
```

테이블 하나 줄여도 되어서 줄였다.

```
select s.name, s.deptno, e.total, s.count_name
from exam_01 e, (select s1.name as name, s1.studno as studno, s1.deptno1 as deptno, count(s2.name) as count_name
                 from student s1, student s2
                 where s1.BIRTHDAY > s2.BIRTHDAY(+)
                  and s1.grade = s2.grade(+)
                 group by s1.studno, s1.deptno1, s1.name) s
where s.studno = e.studno;
```

4. student 테이블에서 같은지역, 같은 성별의 친구가 몇명인지 구하고,
그 학생의 담당 교수이름도 함께 출력되도록 하여라.
단, 같은지역, 같은 성별에 본인은 포함될 수 없다. - **지역번호 이용**

같은지역별 count - 같은 성별 count - 교수이름
(학생 1-학생 2) - (학생 1-학생 3) - 교수이름

##문제 이해를 잘못했다.. 같은 지역이면서 같은 성별

같은지역+성별 - 교수이름
학생 1 - 학생 2 - 교수

1.

```
select s1.name, s2.name
 from student s1, student s2
 where substr(s1.jumin,7,1) = substr(s2.jumin, 7,1)
  and substr(s1.tel,1,instr(s1.tel,')')-1) = substr(s2.tel,1,instr(s2.tel,')')-1);
```

2. 본인은 제외해야 함

```
select s1.name, s2.name
 from student s1, student s2
 where substr(s1.jumin,7,1) = substr(s2.jumin, 7,1)
  and substr(s1.tel,1,instr(s1.tel,')')-1) = substr(s2.tel,1,instr(s2.tel,')')-1)
  and s1.studno != s2.studno ;
```

이름으로도 가능하지만 동명이인의 가능성, 따라서 학번으로 제외하는 것이 좋음

3.

```
select s1.name, count(s2.name)
 from student s1, student s2
 where substr(s1.jumin,7,1) = substr(s2.jumin, 7,1)
  and substr(s1.tel,1,instr(s1.tel,')')-1) = substr(s2.tel,1,instr(s2.tel,')')-1)
  and s1.studno != s2.studno
group by s1.name;
```

4.

```
select s1.name, count(s2.name), p.name
 from student s1, student s2, professor p
 where substr(s1.jumin,7,1) = substr(s2.jumin, 7,1)
  and substr(s1.tel,1,instr(s1.tel,')')-1) = substr(s2.tel,1,instr(s2.tel,')')-1)
  and s1.studno != s2.studno
  and s1.profno = p.PROFNO
group by s1.name, p.name;
```

- 학생별 지도교수가 두명인 일은 없음. 따라서 지도교수도 group by 로 데려와도 학생별이라는 정보 분류 체계 유지됨.

cf. 모든 학생을 출력하세요

s1 - s2

|

p

- 만약 다 표기하고 싶으면 s1-s2 관계의 모두에 (+)표시 & s1-p 관계의 모두에도 (+)표시
- 함수가 있을 때엔 함수 끝이 아니라, 해당 column 에 모두!!!! (+)표시!

```
where substr(s1.jumin,7,1) = substr(s2.jumin(+), 7,1)
  and substr(s1.tel,1,instr(s1.tel,')')-1) = substr(s2.tel(+),1,instr(s2.tel(+),')')-1)
  and s1.studno != s2.studno(+)
  and s1.profno = p.PROFNO(+)
```

```
-------------------------내가 푼 것---------------------
select sss1.studno, sss1.name, sss2.count_jumin, sss3.count_loc, p.name
 from student sss1, professor p, (select ss1.studno, count(ss2.studno) as count_jumin
                                  from student ss1, student ss2
                                 where substr(ss1.jumin,7,1) = substr(ss2.jumin, 7,1)
                                   and ss1.studno <> ss2.studno
                                 group by ss1.STUDNO) sss2                                  , (select s1.studno,
count(s2.studno) as count_loc
                                                                                             from student s1, student
s2
                                                                                            where
substr(s1.tel,1,instr(s1.tel,')')-1) = substr(s2.tel,1,instr(s2.tel,')')-1)
                                                                                              and s1.studno <>
s2.studno
                                                                                            group by s1.studno) sss3
 where sss1.studno = sss2.studno
   and sss1.studno = sss3.studno
   and sss1.profno = p.profno(+);
```

또 테이블 하나 줄였다 (관계 먼저 그려놓고 생각하니까 괜찮다!)

```
select sss2.studno, sss2.name, sss2.count_jumin, sss3.count_loc, p.name
 from professor p, (select ss1.studno, count(ss2.studno) as count_jumin, ss1.PROFNO as profno,
ss1.name
                    from student ss1, student ss2
                    where substr(ss1.jumin,7,1) = substr(ss2.jumin, 7,1)
                    and ss1.studno <> ss2.studno
                    group by ss1.STUDNO, ss1.profno, ss1.name) sss2                          , (select
s1.studno as studno, count(s2.studno) as count_loc
                                                                                          from      student      s1,
student s2
                                                                                          where
substr(s1.tel,1,instr(s1.tel,')')-1) = substr(s2.tel,1,instr(s2.tel,')')-1)
                                                                                           and      s1.studno      <>
s2.studno
```

```
                                                    group   by   s1.studno)
sss3
 where sss2.studno = sss3.studno
   and sss2.profno = p.profno(+);
```

##########하나로 합쳐서 만들면 (3 번처럼) count 한 게 중복으로 뜬다
s1, s2, s3 으로 했을 때 s2~s3 에 대한 조건을 서술하지 않아서 카티시안곱..?

```
select s1.name, count(s2.studno) as count_jumin, count(s3.studno) as count_loc, p.name
                from student s1, student s2, professor p, student s3
                where substr(s1.jumin,7,1) = substr(s2.jumin, 7,1)
                and s1.studno <> s2.studno
                and s1.studno <> s3.studno
                and substr(s1.tel,1,instr(s1.tel,')')-1) = substr(s3.tel,1,instr(s3.tel,')')-1)
                and s1.PROFNO = p.PROFNO
                group by s1.name, p.name;
```

5. 교수에 대한 전체 자료를 다음과 같은 형식으로 만들어 보자
(단, 아우터 조인은 고려하지 않는다)

교수이름 지도학생수 지도학생_성적평균 A_학점자수 B_학점자수 C_학점자수 D_학점자수

심슨 2 79 1 0 0
1

허은 2 83 0 1
1 0

조인형 1 97 1 0
0 0

....

(총 9 명)

교수이름

교수 1 - 학생 1 - 성적 1 - 등급 1

```
select p.name as "교수이름", count(s.name) as "지도학생수",
       round(avg(e.total)) as "지도학생_성적평균",
       count(decode(substr(h.grade,1,1),'A',1)) as "A_학점자수",
       count(decode(substr(h.grade,1,1),'B',1)) as "B_학점자수",
       count(decode(substr(h.grade,1,1),'C',1)) as "C_학점자수",
       count(decode(substr(h.grade,1,1),'D',1)) as "D_학점자수"
 from professor p, student s, exam_01 e, hakjum h
where p.profno = s.PROFNO
  and s.studno = e.STUDNO
  and e.total between h.min_point and h.max_point
group by p.name;
```

cf. OUTERJOIN

- 테이블이 4 개 조인
 p - s - e - h
* s - e, e- h ; 일대일 연결
* p - s 에서 생략이 생김

p.profno = s.PROFNO**(+)** **를 해도 생략이**

학생이 원래 null 인데 표시하려고 가져왔지만,
s - e 에서도 e(+) 를 가져와야 함. (학생이 없는데 학생의 성적을 가져올 수 없음)
e - h 도 null 이므로 h(+)를 가져와야 함. (성적이 없는데 학점을 가져올 수 없음)

따라서

```
where p.profno = s.PROFNO(+)
  and s.studno = e.STUDNO(+)
  and e.total between h.min_point(+) and h.max_point(+)
```

----------나

```
select p.name as "교수이름", count(studno), round(avg(t.total)), count(decode(t.grade,'A',1)) as "A_학점자수", count(decode(t.grade,'B',1)) as "B_학점자수", count(decode(t.grade,'C',1)) as "C_학점자수", count(decode(t.grade,'D',1))as "D_학점자수"
from professor p, (select s.STUDNO as studno, s.PROFNO as profno, e.TOTAL as total, substr(h.GRADE,1,1) as grade
                   from student s, exam_01 e, hakjum h
                  where s.studno = e.STUDNO
                   and e.total between h.MIN_POINT and h.MAX_POINT) t
where p.PROFNO = t.profno
group by p.name;
```