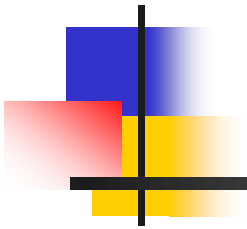
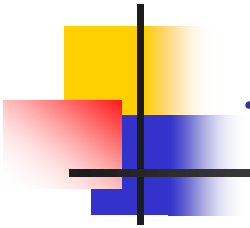


Oracle 5강 – SUB Query

양 명 숙
[now4ever7@gmail.com]



목차

- Sub query란
- Sub query 의 종류
- Scalar sub query(스칼라 서브쿼리)
- inline view

Sub query란

- Sub query
  - Sql문을 작성할 때 질문이 여러 가지가 한꺼번에 나오는 경우
  - 예)emp테이블에서 scott 보다 급여를 많이 받는 사람이 누구일까?
    - Scott의 급여를 알아야 scott보다 급여를 많이 받는 사람의 급여를 조회할 수 있다
    - 원래는 Scott의 급여를 먼저 조회한 후 그 보다 급여를 많이 받는 사람을 한 번 더 조회해야 함 => 2번 sql 작성, 서버에 I/O를 두 번 발생시킨다는 뜻, 성능이 저하된다는 의미
    - => 이런 문제점 보완 : Sub Query(쿼리 안에 또 다른 쿼리가 담겨 있는 것)

Sub Query 문법

Main Query(Outer Query)

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

Sub Query(Inner Query)

Sub query

```
select sal from emp
      where ename='SCOTT';
```

SAL
3000

```
Select ename, sal from emp
Where sal > 3000;
```

- Emp 테이블에서 'SCOTT' 보다 급여를 많이 받는 사람의 이름과 급여를 출력하시오

Main Query

```
Select ename, sal from emp
Where sal > ( select sal from emp
              where ename='SCOTT' );
```

이 부분이 Sub query

```
SQL> Select ename, sal from emp
  2  Where sal > ( select sal from emp
  3            where ename='SCOTT' );

ENAME                SAL
---------- ----------
KING               5000
```

우선 sub query가 먼저 수행되어 결과 값을 Main query로 전해주고 그 값을 받아서 Main Query가 수행됨 (이 순서는 서브쿼리의 종류에 따라 달라짐)

Sub query

- Sub query 작성시 주의사항
  - Sub query 부분은 where 절에 연산자 오른쪽에 위치해야 하며 반드시 괄호로 묶어야 함
  - 특별한 경우(Top-n 분석 등)를 제외하고는 sub query 절에 order by 절이 올 수 없음
  - 단일 행 sub query와 다중 행 sub query에 따라 연산자를 잘 선택해야 함
- Sub query의 종류
  - (1) 단일 행 sub query : Sub query의 결과가 1개의 행만 나오는 것
  - (2) 다중 행 sub query :Sub query의 결과가 2건 이상 출력되는 것
  - (3) 다중 칼럼 sub query : Sub query의 결과가 여러 칼럼인 경우
  - (4) 상호연관 sub query : Main query 값을 sub query에 주고 sub query를 수행한 후 그 결과를 다시 main query로 반환해서 수행하는 sub query

단일 행 sub query

- (1) 단일 행 sub query
  - Sub query의 결과가 1개의 행만 나오는 것
  - Sub query를 수행한 결과가 1건만 나오고 이 결과를 Main query로 전달해서 Main Query를 수행하게 됨
- 단일 행 sub query일 경우 where 절에서 사용되는 연산자

연산자	의미
=	같다
<>, !=	같지 않다
>	크다
>=	크거나 같다
<	작다
<=	작거나 같다

```
select deptno1 from student where name = '이윤나';
```

DEPTNO1
101

단일 행 sub query

- 예제) student 테이블과 department 테이블을 사용하여 이윤나 학생과 1전공(deptno1)이 동일한 학생들의 이름과 1전공 이름을 출력하시오

```
SQL> select s.name "학생이름", d.dname "1전공명"
  2  from student s, department d
  3  where s.deptno1 = d.deptno
  4  and s.deptno1 = (select deptno1 from student
  5                   where name='이윤나' );

학생이름   1전공명
---------- --------------------
서진수     컴퓨터공학과
김신영     컴퓨터공학과
일지매     컴퓨터공학과
이윤나     컴퓨터공학과
```

```
select name, deptno1 from student
where deptno1 = (select deptno1 from student where name = '이윤나');
```

```
select s.name, s.deptno1, d.dname from student s, department d
where s.deptno1 = d.deptno
and deptno1 = (select deptno1 from student where name = '이윤나');
```

단일 행 sub query

- 실습) Professor 테이블에서 입사일이 송도권 교수보다 나중에 입사한 사람의 이름과 입사일, 학과명을 출력하시오.
  - professor, department테이블 이용

- 실습) student 테이블에서 1전공(deptno1)이 101번인 학과의 평균 몸무게보다 몸무게가 많은 학생들의 이름과 몸무게를 출력하시오

교수명	입사일	학과명
양선희	01/09/01	멀티미디어공학과
김현정	02/02/24	소프트웨어공학과
최슬기	09/08/30	전자공학과
박원범	99/12/01	기계공학과
차범철	09/01/28	기계공학과
전미은	10/06/28	문헌정보학과
허	01/05/23	문헌정보학과

7 개의 행이 선택되었습니다.

이름	몸무게
서진수	72
서재수	64
김재수	83
박동호	70
지매	72
김진욱	70
안광훈	82
노정호	62
안은수	63
인영민	69
김주현	81

11 개의 행이 선택되었습니다.

단일 행 sub query

- 예제) Professor 테이블에서 심슨 교수와 같은 입사일에 입사한 교수 중에서 조인형 교수보다 월급을 적게 받는 교수의 이름과 급여, 입사일을 출력하시오.

```
SQL> select name "이름", pay "급여", hiredate "입사일"
  2  from professor
  3  where hiredate = (select hiredate from professor
  4                    where name='심슨')
  5  and pay<(select pay from professor
  6               where name='조인형');

이름          급여 입사일
---------- ---------- --------
김도형            530 81/10/23
```

```
--월급이 가장 많은 사원 조회
select name, deptno, emp_type, pay, position
from emp2
where pay = (select max(pay) from emp2);
```

실습

- emp2 테이블에서 월급이 가장 적은 사원 정보 조회
  - Dept2 테이블과 조인해서 부서명도 조회할 것

NAME	EMP_TYPE	PAY	POSITION	DNAME
강월악	인턴직	20000000		영업4팀
장금강	인턴직	20000000		H/W지원
나한라	인턴직	20000000		S/W지원

• any – 여러 개중 아무거나 하나만 조건을 만족해도 된다는 의미
• all – 서브쿼리에서 반환되는 모든 row 값을 만족해야 함을 의미

다중 행 sub query

- 다중 행 sub query
  - Sub query의 결과가 2건 이상 출력되는 것
  - Sub query의 결과가 여러 건 출력되기 때문에 단일 행 연산자를 사용할 수 없음
  - 다중 행 sub query 연산자

연산자	의미
IN	같은 값을 찾음(포함된 값)
>ANY	최소값을 반환함 (비교 연산)
<ANY	최대값을 반환함 (비교 연산)
<ALL	최소값을 반환함 (비교 연산)
>ALL	최대값을 반환함 (비교 연산)
EXIST,	SUB QUERY의 값이 있을 경우 반환함 연관성 있는 서브쿼리의 경우 사용됨

다중 행 sub query

- ANY, ALL은 연산자의 방향에 따라 최대값, 최소값이 달라짐
- 예) SUB query의 결과값이 (100, 200, 300)으로 나온 경우
  - Sal > ANY(100,200,300) => ANY 자리에 최소값인 100 이 반환됨
    => sal > 100
  - Sal < ANY(100,200,300) => ANY 자리에 최대값인 300 이 반환됨
    => sal < 300
  - Sal > ALL(100,200,300) => 최대값인 300 이 반환됨
    => sal > 300
  - Sal < ALL(100,200,300) => 최소값인 100 이 반환됨
    => sal < 100

다중 행 sub query

```
select name "이름", position "직급",
    to_char(pay, '999,999,999') || ' 원' "연봉"
from emp2
where pay > ( select min(pay) from emp2
      where position='과장' );   --과 동일
```

- 예제) emp2 테이블과 dept2 테이블을 참조하여 근무지역(dept2 테이블의 area 칼럼)이 서울 지사인 모든 사원들의 사번과 이름, 부서번호를 출력하시오

```
SQL> select empno, name, deptno
  2  from emp2
  3  where deptno in(select dcode from dept2
  4                  where area='서울지사');

     EMPNO NAME       DEPTNO
---------- ---------- ------
  19960101 전부장       1000
  19970201 최일도       1000
  19930331 백원만       1001
  19950303 천만득       1002
  20000203 최오대       1010
  20000407 윤주왕       1010

6 개의 행이 선택되었습니다.
```

Sub query 수행하면 1000, 1001, 1002, 1010 4건의 결과

- 예제) emp2 테이블을 사용하여 전체 직원 중 과장 직급의 최소 연봉자보다 연봉이 높은 사람의 이름과 직급, 연봉을 출력하시오. 단, 연봉 출력형식은 천 단위 구분기호와 원 표시를 하시오.

```
SQL> select name "이름", position "직급",
  2        to_char(pay, '999,999,999') || ' 원' "연봉"
  3  from emp2
  4  where pay >ANY ( select pay from emp2
  5           where position='과장' );

이름       직급       연봉
---------- -------- ---------------
나사장     대표이사   100,000,000 원
일지매     부장        75,000,000 원
전부장     부장        72,000,000 원
노정호     부장        68,000,000 원
백원만     차장        60,000,000 원
천만득     과장        56,000,000 원
유관순     과장        51,000,000 원
최일도     과장        50,000,000 원

8 개의 행이 선택되었습니다.
```

Sub query 수행 결과

```
       PAY
----------
 50000000
 56000000
 51000000
 49000000
```

다중 행 sub query

- 근무지역이 경기지사가 아닌 모든 사원들 조회

```
select empno, name, pay, deptno
from emp2
where deptno not in
( select dcode from dept2 d where area = '경기지사')
```

EMPNO	NAME	PAY	DEPTNO
19900101	나사장	100000000	0001
19960101	전부장	72000000	1000
19970201	최일도	50000000	1000
19930331	백원만	60000000	1001
19950303	천만득	56000000	1002
19966102	일지매	75000000	1003
19930402	유관순	51000000	1004
19970112	노정호	68000000	1006
19960212	이윤나	49000000	1007
20000101	이태백	30000000	1008
20000203	최오대	30000000	1010
20000334	박지리	30000000	1011
20000305	정북악	22000000	1008
20000407	윤주왕	22000000	1010
20000308	강월악	20000000	1011
20000119	장금강	20000000	1004

- loc가 DALLAS 가 아닌 모든 사원 조회

```
select empno, ename, sal, deptno
from emp
where deptno != ( select deptno from dept d where loc = 'DALLAS')
```

EMPNO	ENAME	SAL	DEPTNO
7499	ALLEN	1600	30
7521	WARD	1250	30
7654	MARTIN	1250	30
7698	BLAKE	2850	30
7782	CLARK	2450	10
7839	KING	5000	10
7844	TURNER	1500	30
7900	JAMES	950	30
7934	MILLER	1300	10

단일 행 sub query

예제

- 예제) emp2 테이블을 사용하여 전체 직원 중 과장 직급의 최대 연봉자보다 연봉이 높은 사람의 이름과 직급, 연봉을 출력하시오.

이름	직급	연봉
나사장	대표이사	100,000,000 원
전부장	부장	72,000,000 원
백원만	차장	60,000,000 원
일지매	부장	75,000,000 원
노정호	부장	68,000,000 원

```
select name "이름", position "직급",
    to_char(pay, '999,999,999') || ' 원' "연봉"
from emp2
where pay > ALL( select pay from emp2
        where position='과장' );

select name "이름", position "직급",
    to_char(pay, '999,999,999') || ' 원' "연봉"
from emp2
where pay > ( select max(pay) from emp2
        where position='과장' );
```

다중 행 sub query

- 실습)student 테이블을 조회하여 전체 학생 중에서 체중이 4학년 학생들의 체중에서 가장 적게 나가는 학생보다 몸무게가 적은 학생의 이름과 몸무게를 출력하시오.

이름	학년	몸무게
김신영	3	48
신은경	3	42
김문호	2	51
이윤나	1	48
허우	1	51

```
select name "이름", grade "학년", weight "몸무게"
from student
where weight < ALL (select weight from student
                    where grade=4 );
```

```
select name "이름", grade "학년", weight "몸무게"
from student
where weight < (select min(weight) from student
                where grade=4 );
```

과 동일

select min(avg(pay)) from emp2 group by deptno

MIN(AVG(PAY))
25000000

다중 행 sub query

- 예제) emp2 테이블을 조회하여 각 부서별 평균 연봉을 구하고 그 중에서 평균 연봉이 가장 적은 부서의 평균 연봉보다 적게 받는 직원들의 부서명,직원명, 연봉을 출력하시오.

```
SQL> select d.dname "부서명",  e.name "사원명", e.pay  "연봉"
  2  from emp2 e, dept2 d
  3  where e.deptno = d.dcode
  4  and e.pay < ALL ( select avg(pay)
  5                    from emp2
  6                    group by deptno)
  7  order by 3;

부서명               사원명          연봉
-------------------- ---------- ----------
H/W지원              장금강       20000000
영업4팀              강월악       20000000
S/W지원              나한라       20000000
영업2팀              유도봉       22000000
영업1팀              정북악       22000000
영업3팀              윤주왕       22000000

6 개의 행이 선택되었습니다.
```

다중 칼럼 sub query

- 다중 칼럼 sub query (pairwise SubQuery)
  - Sub query의 결과가 여러 칼럼인 경우
  - 주로 primary key를 여러 칼럼을 합쳐서 만들었을 경우 한꺼번에 비교하기 위해서 자주 사용함
- 예제) student 테이블을 조회하여 각 학년별로 최대 키를 가진 학생들의 학년과 이름과 키를 출력하시오.

```
SQL> select grade "학년", name "이름", height "키"
  2  from student
  3  where (grade, height) in ( select grade, max(height)
  4                             from student
  5                             group by grade)
  6  order by 1;

      학년 이름                 키
---------- ---------- ----------
         1 김주현              179
         2 노정호              184
         3 오나라              177
         4 박동호              182
```

Sub query 수행 결과

GRADE	MAX(HEIGHT)
1	179
2	184
4	182
3	177

다중 칼럼 sub query

- 예제) professor 테이블을 조회하여 각 학과별로 입사일이 가장 오래된 교수의 교수번호와 이름, 입사일, 학과명을 출력하시오. 단 학과이름순으로 오름차순 정렬하시오.

```
SQL> select p.profno "교수번호" , p.name "교수명", p.hiredate "입사일", d.dname
"학과명"
  2  from professor p, department d
  3  where p.deptno = d.deptno
  4  and (p.deptno, p.hiredate) IN (select deptno, MIN(hiredate)
  5                                 from professor
  6                                 group by deptno)
  7  order by 4;

  교수번호 교수명     입사일   학과명
---------- ---------- -------- --------------------
      4003 박원범     99/12/01 기계공학과
      2003 주승재     82/04/29 멀티미디어공학과
      4007 허은       01/05/23 문헌정보학과
      3001 김도형     81/10/23 소프트웨어공학과
      4001 심슨       81/10/23 전자공학과
      1001 조인형     80/06/23 컴퓨터공학과
      4005 바비       85/09/18 화학공학과

7 개의 행이 선택되었습니다.
```

다중 칼럼 sub query

- 실습) emp2 테이블을 조회하여 직급별로 해당 직급에서 최대 연봉을 받는 직원의 이름과 직급, 연봉을 출력하시오. 단, 연봉순으로 오름차순 정렬하시오

사원명	직급	연봉
김문호	대리	35000000
천만득	과장	56000000
백원만	차장	60000000
일지매	부장	75000000
나사장	대표이사	100000000

AVG(PAY)
71666666.6666667

```
select name, position, pay
from emp2 a
where position='부장';

select avg(pay) from emp2 b
where b.position='부장';
```

상호 연관 sub query

- 상호 연관 sub query(연관성 있는 서브쿼리)
  - Main query 값을 sub query에 주고 sub query를 수행한 후 그 결과를 다시 main query로 반환해서 수행하는 sub query
- 예제) emp2 테이블을 조회하여 직원들 중에서 자신의 직급의 평균 연봉과 같거나 많이 받는 사람들의 이름, 직급, 현재연봉을 출력하시오.

```
SQL> select name "사원이름", position "직급", pay "급여"
  2  from emp2 a
  3  where pay>=(select avg(pay) from emp2 b
  4                  where a.position = b.position);

사원이름   직급         급여
---------- -------- ----------
나사장     대표이사  100000000
전부장     부장       72000000
백원만     차장       60000000
천만득     과장       56000000
일지매     부장       75000000
김문호     대리       35000000

6 개의 행이 선택되었습니다.
```

• Main query 를 먼저 수행해서 직급을 구해서 sub query 에 전달해 주고, 그 값을 받은 sub query가 수행되어 결과(평균연봉)를 다시 main query로 전달해 줌
그 후에 다시 받은 값을 가지고 Main query가 최종적으로 수행되는 형태
• 이 sub query의 경우는 잘못 사용될 경우 성능 저하의 원인이 될 수 있음
 - inline view 등을 사용하기를 권장

• professor 테이블을 조회하여 교수들 중에서 자신의 학과의 평균 급여보다 많이 받는 사람들의 이름, 부서, 현재급여를 출력하시오

```
select * from emp2 e, dept2 d
where e.deptno = d.dcode
and d.pdept is not null;
```

연관성 있는 서브쿼리

- 연관성 있는 서브쿼리
  - 서브쿼리가 메인쿼리에 독립적이지 않고 연관관계, 즉 조인을 통해 연결되어 있는 쿼리를 말함
  - 서브쿼리와 메인 쿼리 사이에서 조인이 사용
  - 서브쿼리에서 테이블 별칭이 사용됨
  - MAIN-QUERY절에 사용된 테이블이 SUB-QUERY절에 다시 재 사용되는 경우
  - SUB-QUERY에 사용될 조건값을 MAIN-QUERY로부터 참조해야 하는 경우 사용되는 방법
  - 성능저하의 단점
- exists 연산자

특정 칼럼 값이 존재하는지 여부를 체크

  - 서브쿼리가 반환하는 결과에 메인 쿼리에서 추출될 데이터들이 존재하기만 하면 조건을 만족하게 됨
- 예제) 부서 테이블의 pdept 값이 null이 아닌 부서에 속하는 사원 추출

```
select * from emp2 e
where exists (select 1 from dept2 d
    where d.pdept is not null
    and e.deptno = d.dcode);
```

```
select * from emp2
where deptno in (select dcode from dept2
    where pdept is not null);
```

성능면에서는 in 보다 exists 가 월등히 우수함

연관성 있는 서브쿼리

```
select *
from dept
where deptno in(select deptno from emp
        where sal>=5000)
order by deptno;
```

- 월급이 5000 달러 이상인 사원이 속한 부서를 추출하시오

```
select d.*
from dept d
where exists(select 1 from emp e
            where e.deptno=d.deptno
            and e.sal>=5000)
order by d.deptno;
```

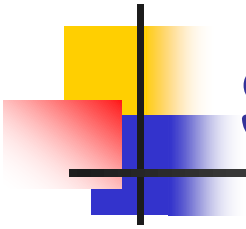
DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK

- in – 어떤 값에 포함되는지 여부를 체크
- in은 () 안에 비교할 값이 올 수도 있고, 서브쿼리가 올수도 있음

- exists – 특정 칼럼 값이 존재하는지 여부를 체크
- exists 는 오직 서브쿼리만 올 수 있음
- 서브쿼리의 where 절에서 비교할 기준 테이블의 컬럼과 조인을 맺어야 하며, 서브쿼리의 결과로 반환되는 행이 1개라도 있을 경우 exists 조건을 만족하게 됨

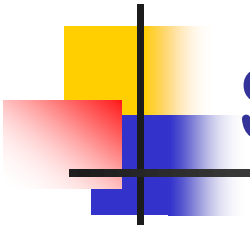
```
select * from dept where deptno in (10, 20);
```

```
select empno, name, pay, deptno
from emp2
where deptno in
( select dcode from dept2 d where area = '경기지사')
```



Sub query 위치별 이름

- Sub query는 오는 위치에 따라서 그 이름이 다름
  - Select (sub query)
    - 1행만 반환할 경우 Scalar sub query
    - Select 절에 오는 서브쿼리로 한번에 결과를 1행씩 반환함
  - From (sub query)
    - Inline View
  - where (sub query)
    - Sub query



Scalar sub query

- scalar subquery
  - 서브 쿼리의 결과 값을 마치 컬럼이나 함수처럼 사용하는 기능 추가
  - 출력된 값은 select list의 한 항목 값으로 대체
  - 공집합이더라도 select list상의 값은 null로 표시
  - 대량의 데이터 처리시 조인으로 대체

Scalar sub query

```
select name "사원이름", dname  "부서이름"
from emp2 e, dept2 d
where e.deptno = d.dcode(+);
```

과 동일

- Scalar sub query
  - Select 절에 오는 서브쿼리로 한번에 결과를 1행씩 반환함
- 예제) emp2 테이블과 dept2 테이블을 조회하여 사원들의 이름과 부서이름을 출력하시오

```
SQL> select name "사원이름",
  2
  3  from emp2 e;
```

```
(select dname from dept2 d
    where e.deptno=d.dcode) "부서이름"
```

Scalar sub query

사원이름	부서이름
나사장	사장실
전부장	경영지원부
최일도	경영지원부
백원만	재무관리팀
천만득	총무팀
일지매	기술부
유관순	H/W지원
김문호	S/W지원
노정호	영업부
이윤나	영업기획팀
이태백	영업1팀

사원이름	부서이름
김설악	영업2팀
최오대	영업3팀
박지리	영업4팀
정북악	영업1팀
유도봉	영업2팀
윤주왕	영업3팀
강월악	영업4팀
장금강	H/W지원
나한라	S/W지원

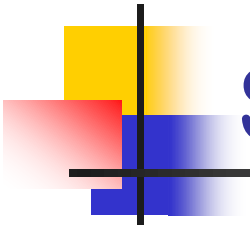
20 개의 행이 선택되었습니다.

Scalar sub query

- 예제
  - 각 부서에 해당하는 사원수 구하기

```
select dname,loc,
            (select count(*) from emp e
                where e.deptno=d.deptno)
from dept d;
```

DNAME	LOC	(SELECTCOUNT(*)FROMEMPWHEREDEPTNO=D.DEPTNO)
ACCOUNTING	NEW YORK	3
RESEARCH	DALLAS	5
SALES	CHICAGO	6
OPERATIONS	BOSTON	1

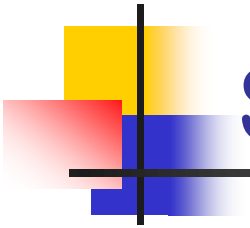


Scalar sub query

- Scalar sub query 는 서브쿼리의 결과가 없을 경우 null을 반환함
- => Scalar sub query는 outer join과 동일함
  - Join 등의 방법은 출력하고자 하는 데이터의 양이 적을 경우 스칼라 서브쿼리보다 속도가 느림
  - 주로 출력하고자 하는 데이터의 양이 적은 경우(예를 들어 코드 테이블 등) 조인보다는 스칼라 서브쿼리를 많이 사용함
  - 데이터의 양이 많은 경우는 조인등의 작업이 훨씬 더 빠를수도 있음

Scalar sub query

- Scalar sub query의 동작 순서
  - 1. Main query를 수행한 후 Scalar sub query에 필요한 값을 제공
  - 2. Scalar sub query를 수행하기 위해 필요한 데이터가 들어있는 블록을 메모리로 로딩함
  - 3. Main query에서 주어진 조건을 가지고 필요한 값을 찾음, 이 결과를 메모리에 입력 값과 출력 값으로 메모리 내의 query execution cache라는 곳에 저장해 둠
    - 입력값은 Main query에서 주어진 값, 출력값은 Scalar sub query 를 수행 후 나온 결과값
  - 4. 다음 조건이 Main query에서 Scalar sub query로 들어오면 해쉬 함수를 이용해서 해당 값이 캐쉬에 존재하는지 찾고 있으면 즉시 결과값을 출력하고 없으면 다시 블록을 엑세스해서 해당 값을 찾은 후 다시 메모리에 캐쉬해 둠
  - 5. Main query가 끝날때까지 반복함



Scalar sub query

- Scalar sub query 가 빠른 이유
  - 찾는 데이터가 메모리에 만들어져 있는 값을 찾아오기 때문
  - 만약 모든 데이터가 메모리에 없거나 또는 데이터량이 많을 경우는 query execution cache 에서 해당 데이터를 찾는 시간이 더 걸리기 때문에 join 보다 속도가 더 걸리게 됨
- Scalar sub query는 데이터 종류가 적고 개수도 비교적 적은 코드성 테이블에서 데이터를 가져올 경우 등에 사용하기를 권장

inline view – from 절에 sub query가 오는 것

inline view(Top-n 분석)

- 의사컬럼 (Pseudocolumn) – 모조, 유령 컬럼
  - 테이블에 있는 일반적인 컬럼처럼 행동하기는 하지만 실제로 테이블에 저장되어 있지 않은 컬럼

– ROWNUM : 쿼리의 결과로 나오는 각각의 row들에 대한 순서 값을 가리키는 의사컬럼
  - 주로 특정 개수나 그 이하의 row를 선택할 때 사용됨

• 상위 5건의 정보만 조회하기
```
select empno, ename, rownum
from emp
where rownum <=5;
```

• 5건의 정보만 조회하기(정렬한 결과에서 조회시)
```
select empno, ename, rownum
from (select empno, ename from emp order by ename)
where rownum <=5;
```

– ROWID : 테이블에 저장된 각각의 row들이 저장된 주소값을 가진 의사컬럼
모든 테이블의 모든 row 들은 오직 자신만의 유일한 ROWID 값을 갖고 있다

```
select rownum, rowid as "ROW_ID", empno, ename, job, sal from emp;
```

EMPNO	ENAME	ROWNUM
7369	SMITH	1
7499	ALLEN	2
7521	WARD	3
7566	JONES	4
7654	MARTIN	5
7698	BLAKE	6
7782	CLARK	7
7788	SCOTT	8
7839	KING	9
7844	TURNER	10

```
SQL> select empno, ename, ROWID, rownum
  2  from emp;

     EMPNO ENAME      ROWID                  ROWNUM
---------- ---------- ------------------ ----------
      7369 SMITH      AAAMfPAAEAAAAAgAAA          1
      7499 ALLEN      AAAMfPAAEAAAAAgAAB          2
      7521 WARD       AAAMfPAAEAAAAAgAAC          3
      7566 JONES      AAAMfPAAEAAAAAgAAD          4
      7654 MARTIN     AAAMfPAAEAAAAAgAAE          5
      7698 BLAKE      AAAMfPAAEAAAAAgAAF          6
      7782 CLARK      AAAMfPAAEAAAAAgAAG          7
      7788 SCOTT      AAAMfPAAEAAAAAgAAH          8
      7839 KING       AAAMfPAAEAAAAAgAAI          9
      7844 TURNER     AAAMfPAAEAAAAAgAAJ         10
      7876 ADAMS      AAAMfPAAEAAAAAgAAK         11
      7900 JAMES      AAAMfPAAEAAAAAgAAL         12
      7902 FORD       AAAMfPAAEAAAAAgAAM         13
      7934 MILLER     AAAMfPAAEAAAAAgAAN         14

14 개의 행이 선택되었습니다.
```

실습

- emp 테이블을 조회하여 직원들 중에서 자신의 job의 평균 연봉(sal)보다 적거나 같게 받는 사람들을 조회하시오.

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	1980-12-17	800		20
7521	WARD	SALESMAN	7698	1981-02-22	1250	500	30
7654	MARTIN	SALESMAN	7698	1981-09-28	1250	1400	30
7782	CLARK	MANAGER	7839	1981-06-09	2450		10
7788	SCOTT	ANALYST	7566	1987-04-19	3000		20
7839	KING	PRESIDENT		1981-11-17	5000		10
7900	JAMES	CLERK	7698	1981-12-03	950		30
7902	FORD	ANALYST	7566	1981-12-03	3000		20

- 각 학과에 해당하는 교수의 수 구하기
- 각 학과에 해당하는 학생수 구하기
  - department , student 테이블

DEPTNO	DNAME	교수의 수
101	컴퓨터공학과	3
102	멀티미디어공학과	3
103	소프트웨어공학과	3
201	전자공학과	2
202	기계공학과	2
203	화학공학과	1
301	문헌정보학과	2
100	컴퓨터정보학부	0
200	메카트로닉스학부	0
300	인문사회학부	0
	학	0
	학	0

DEPTNO	DNAME	학생의 수
101	컴퓨터공학과	4
102	멀티미디어공학과	4
103	소프트웨어공학과	2
201	전자공학과	6
202	기계공학과	2
203	화학공학과	0
301	문헌정보학과	2
100	컴퓨터정보학부	0
200	메카트로닉스학부	0
300	인문사회학부	0
10	공과대학	0
20	인문대학	0

- Professor 테이블에서 월급을 많이 받는 교수 순으로 10명 조회하기
- Student, exam_01 테이블에서 총점이 90이상인 학생들의 정보 조회

실습-emp, dept 테이블 사용

- 1.job이 MANAGER인 사원들 조회(emp)

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7566	JONES	MANAGER	7839	1981-04-02	2975		20
7698	BLAKE	MANAGER	7839	1981-05-01	2850		30
7782	CLARK	MANAGER	7839	1981-06-09	2450		10

- 2. job이 Manager인 모든 사원들보다 입사일이 빠른(작은) 사원 데이터 조회 => all 이용 (emp)
- 3. ALL없이 결과값 출력 <= MIN함수를 써서

ENAME	JOB	HIREDATE
ALLEN	SALESMAN	1981-02-20
WARD	SALESMAN	1981-02-22
MARTIN	SALESMAN	1981-09-28
BLAKE	MANAGER	1981-05-01
TURNER	SALESMAN	1981-09-08
JAMES	CLERK	1981-12-03

ENAME	JOB	HIREDATE
SMITH	CLERK	1980-12-17
ALLEN	SALESMAN	1981-02-20
WARD	SALESMAN	1981-02-22

- 4. sales부서에 근무하는 사원 데이터 조회(emp, dept)
- 5. 평균급여보다 급여를 많이 받는 사원 데이터 가져오기(emp)

ENAME	JOB	HIREDATE	SAL
JONES	MANAGER	1981-04-02	2975
BLAKE	MANAGER	1981-05-01	2850
CLARK	MANAGER	1981-06-09	2450
SCOTT	ANALYST	1987-04-19	3000
KING	PRESIDENT	1981-11-17	5000
FORD	ANALYST	1981-12-03	3000

실습

- 연령대별, 성별 인원수, 백분율 구하기
  - gogak 테이블 이용

연령대	성별	인원수	백분율
0	남	2	6.3%
0	합계	2	6.3%
10	남	2	6.3%
10	여	4	12.5%
10	합계	6	18.8%
20	남	2	6.3%
20	여	2	6.3%
20	합계	4	12.5%
30	남	7	21.9%
30	여	3	9.4%
30	합계	10	31.3%
40	남	4	12.5%
40	여	6	18.8%
40	합계	10	31.3%
total	합계	32	100%

- 연령대를 가로로 출력하기

성별	0대	10대	20대	30대	40대	성별 인원수	백분율
남	2	2	2	7	4	17	53.1%
여		4	2	3	6	15	46.9%
합계	2	6	4	10	10	32	100%

실습

- 학과별, 성별 평균키 구하기
  - student 테이블 이용

- 성별을 가로로 출력하기

학과	남자	여자	평균키
101	181	163	172
102	174	161	170.8
103		165.5	165.5
201	173.2	171	172.8
202	182	177	179.5
301	184	160	172
total	176.3	165.8	172.1

학과	성별	평균키
101	남	181
101	여	163
101	평균	172
102	남	174
102	여	161
102	평균	170.8
103	여	165.5
103	평균	165.5
201	남	173.2
201	여	171
201	평균	172.8
202	남	182
202	여	177
202	평균	179.5
301	남	184
301	여	160
301	평균	172
total	평균	172.1

실습

- 학과별, 성별 평균키 구하기
  - student, department 테이블 이용, join

학과	성별	평균키
기계공학과	남	182
기계공학과	여	177
기계공학과	평균	179.5
멀티미디어공학과	남	174
멀티미디어공학과	여	161
멀티미디어공학과	평균	170.8
문헌정보학과	남	184
문헌정보학과	여	160
문헌정보학과	평균	172
소프트웨어공학과	여	165.5
소프트웨어공학과	평균	165.5
전자공학과	남	173.2
전자공학과	여	171
전자공학과	평균	172.8
컴퓨터공학과	남	181
컴퓨터공학과	여	163
컴퓨터공학과	평균	172
total	평균	172.1

```
--각 부서에 속하는 사원정보를 조회하고, 부서별 평균급여도 출력하시오
select A.department_id "부서번호", A."부서명",
A."사원번호",  A."사원명",  A."입사일", A."급여" , B."부서별 평균급여",
A."급여"-B."부서별 평균급여" as "평균급여 차액"
from
(
 select d.department_id , d.department_name "부서명",
e.employee_id "사원번호",
e.first_name || '-' || e.last_name "사원명",  e.hire_date "입사일",
e.salary+e.salary*nvl(commission_pct, 0) "급여"
from departments d right join employees e
on d.department_id=e.department_id
)A  left join
(
   select d.department_id,
   round(avg(e.salary+e.salary*nvl(commission_pct, 0)), 1) "부서별 평균급여"
   from departments d join employees e
   on  d.department_id=e.department_id
   group by d.DEPARTMENT_ID
)B
on A.department_id = B.department_id
order by 1,3;
```

부서번호	부서명	사원번호	사원명	입사일	급여	부서별 평균급여
10	Administration	200	Jennifer-Whalen	2003/09/17	4400	4400
20	Marketing	201	Michael-Hartstein	2004/02/17	13000	9500
20	Marketing	202	Pat-Fay	2005/08/17	6000	9500
30	Purchasing	114	Den-Raphaely	2002/12/07	11000	4150
30	Purchasing	115	Alexander-Khoo	2003/05/18	3100	4150
30	Purchasing	116	Shelli-Baida	2005/12/24	2900	4150
30	Purchasing	117	Sigal-Tobias	2005/07/24	2800	4150
30	Purchasing	118	Guy-Himuro	2006/11/15	2600	4150
30	Purchasing	119	Karen-Colmenares	2007/08/10	2500	4150
40	Human Resources	203	Susan-Mavris	2002/06/07	6500	6500
50	Shipping	120	Matthew-Weiss	2004/07/18	8000	3475.6
50	Shipping	121	Adam-Fripp	2005/04/10	8200	3475.6
50	Shipping	122	Payam-Kaufling	2003/05/01	7900	3475.6
50	Shipping	123	Shanta-Vollman	2005/10/10	6500	3475.6
50	Shipping	124	Kevin-Mourgos	2007/11/16	5800	3475.6

315: 24 | 34 msecs | Row 1 of 107 total rows | HR@LOCALHOST:1521/ORCL8 | Modified

```
-- job_history  테이블의 정보를 조회하되, job_id에 해당하는 job_title, department_id에 해당하
    는 부서명도 조회하시오. scalar subquery 이용
--job_history, jobs, departments 테이블 이용
select a.employee_id, a.start_date, a.end_date, a.job_id,
(select job_title from jobs b where b.job_id=a.job_id) as "job_title",
a.department_id,
(select department_name from departments c
    where c.department_id = a.department_id) as "부서명"
 from job_history a
 order by a.employee_id;


--
select e.first_name || ' ' || e.last_name as name, e.salary,e.hire_date,
 a.employee_id, a.start_date, a.end_date, a.job_id,
(select job_title from jobs b where b.job_id=a.job_id) as "job_title",
a.department_id,
(select department_name from departments c
    where c.department_id = a.department_id) as "부서명"
 from job_history a  join employees e
 on a.employee_id = e.employee_id
 order by a.employee_id;
```

```
--departments, employees을 조인해서 부서에 해당하는 사원들 정보를 조회하시오
 select d.department_id "부서번호", d.department_name "부서명",
 e.employee_id "사원번호",
 e.first_name || '-' || e.last_name "사원명",  e.hire_date "입사일",
 e.salary+e.salary*nvl(commission_pct, 0) "급여"
 from departments d right join employees e
 on d.department_id=e.department_id
 order by d.department_id;
```

--departments, employees을 조인해서 각 부서에 해당하는 사원들 정보를 조회하시오
--각 부서장명도 출력하시오

```
 /*
  select d.department_id "부서번호", d.department_name "부서명",
  A.MANAGER_ID, E.FIRST_NAME,
 e.employee_id "사원번호",
 e.first_name || '-' || e.last_name "사원명",  e.hire_date "입사일",
 e.salary+e.salary*nvl(commission_pct, 0) "급여"
 from departments d right join employees e
 on d.department_id=e.department_id
 left join departments a
 on a.manager_id=e.employee_id;
*/
 -- => 원하는 결과가 안 나옴
```

```
select V.department_id "부서번호", V.department_name "부서명",
 V."부서장 번호", V."부서장명",
 E.employee_id "사원번호",
 E.first_name || '-' || E.last_name "사원명",  E.hire_date "입사일",
 E.salary+E.salary*nvl(commission_pct, 0) "급여"
 from
 (
 select d.department_id, d.department_name,
 d.manager_id as "부서장 번호",
 E.FIRST_NAME || '-' || e.last_name as "부서장명"
 from departments d join employees e
 on d.manager_id=e.employee_id
 )V right join employees E
 on V.department_id=E.department_id
   order by E.department_id;
```

out join 대상 테이블에 **out join**을 해야 될 테이블이 있을 때 해결하는 방법은 인라인뷰로 한 번 집합을 만들어 주고 밖에서 **out join**을 다시 한 번 해주면 됨

부서번호	부서명	부서장 번호	부서장명	사원번호	사원명	입사일	급여
10	Administration	200	Jennifer-Whalen	200	Jennifer-Whalen	2003/09/17	4400
20	Marketing	201	Michael-Hartstein	202	Pat-Fay	2005/08/17	6000
20	Marketing	201	Michael-Hartstein	201	Michael-Hartstein	2004/02/17	13000
30	Purchasing	114	Den-Raphaely	119	Karen-Colmenares	2007/08/10	2500
30	Purchasing	114	Den-Raphaely	118	Guy-Himuro	2006/11/15	2600
30	Purchasing	114	Den-Raphaely	117	Sigal-Tobias	2005/07/24	2800
30	Purchasing	114	Den-Raphaely	116	Shelli-Baida	2005/12/24	2900
30	Purchasing	114	Den-Raphaely	115	Alexander-Khoo	2003/05/18	3100
30	Purchasing	114	Den-Raphaely	114	Den-Raphaely	2002/12/07	11000
40	Human Resources	203	Susan-Mavris	203	Susan-Mavris	2002/06/07	6500
50	Shipping	121	Adam-Fripp	199	Douglas-Grant	2008/01/13	2600
50	Shipping	121	Adam-Fripp	198	Donald-OConnell	2007/06/21	2600
50	Shipping	121	Adam-Fripp	197	Kevin-Feeney	2006/05/23	3000
50	Shipping	121	Adam-Fripp	196	Alana-Walsh	2006/04/24	3100
50	Shipping	121	Adam-Fripp	194	Samuel-McCain	2006/07/01	3200

279: 18 | 37 msecs | Row 1 of 107 total rows | HR@LOCALHOST:1521/ORCL8 | Modified

조인, group by 복습

- professor, department 테이블을 조인해서 profno(교수번호), name(이름), hiredate(입사일), dname(학과명) 조회하기

- emp2, dept2 테이블을 조인해서 dname(부서명), name(이름), pay(연봉) 조회하되, pay가 6000만원 이상인 데이터만 조회하기

- professor 테이블을 조회하여 각 학과별로 입사일이 가장 오래된(최소값) 교수 조회하기

DEPTNO	MIN(HIREDATE)
101	1980-06-23
102	1982-04-29
103	1981-10-23
201	1981-10-23
202	1999-12-01
203	1985-09-18
301	2001-05-23