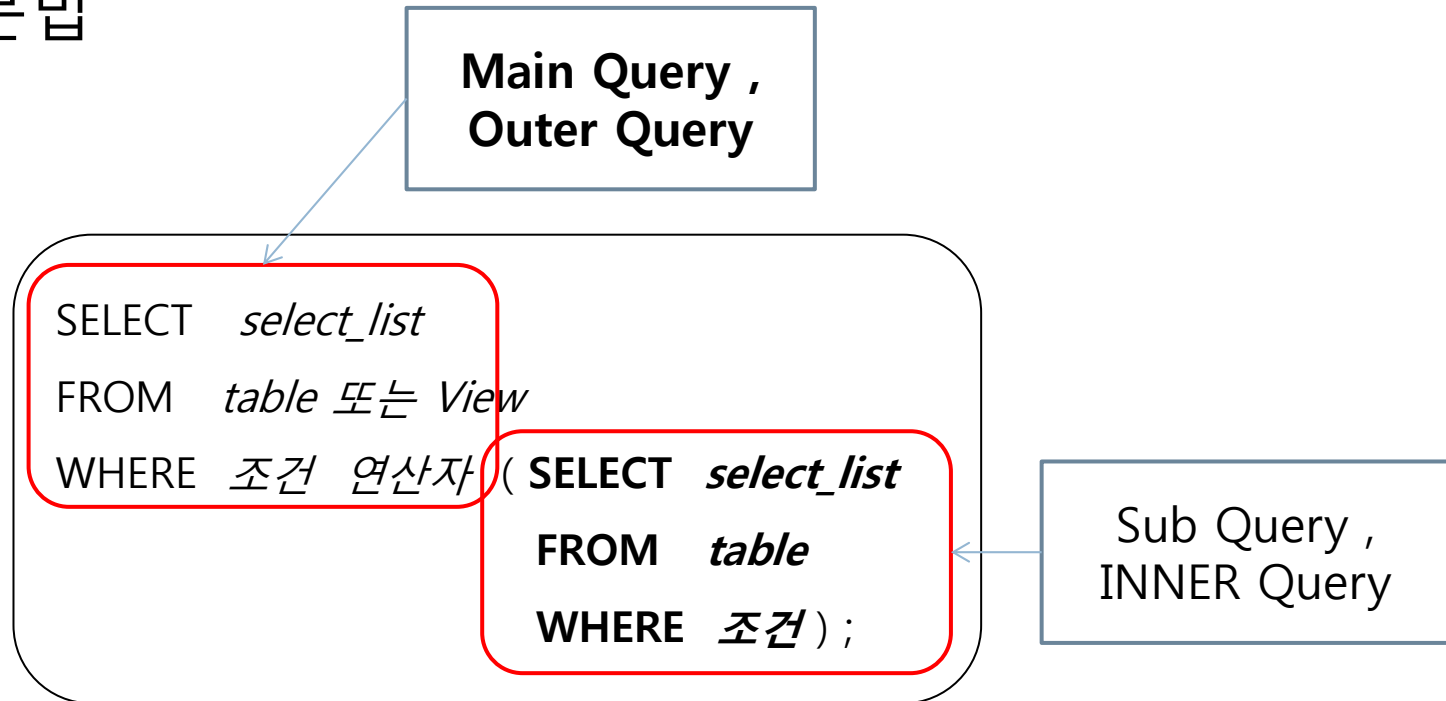


4-2. Sub query

1. Sub Query란?
2. Sub Query 종류
3. Scalar Sub Query(스칼라 서브쿼리)

Sub Query란

- Sub Query 필요성
 - ▣ SQL 작성할 때 여러 가지 조건이 한꺼번에 나오는 경우
- 문법



- 예 : Emp 테이블에서 'SCOTT' 보다 급여를 많이 받는 사람의 이름과 급여를 출력하세요.

```
SCOTT>SELECT ename , sal
2 FROM emp
3 WHERE sal > ( SELECT sal
4               FROM emp
5               WHERE ename='SCOTT') ;
```

□ Sub Query 작성 시 주의 사항

- Sub Query 부분은 Where 절에 연산자 오른쪽에 위치해야 하며 반드시 괄호로 묶어야 합니다.
- 특별한 경우 (Top-n 분석 등)를 제외하고는 Sub Query 절에 Order by 절이 올 수 없습니다.
- 단일 행 Sub Query 와 다중 행 Sub Query 에 따라 연산자를 잘 선택해야 합니다.

Sub Query의 종류

□ 단일 행 Sub Query

연산자	의 미
=	같다 (Equal to)
<>	같지 않다 (Not Equal to)
>	크다 (Greater Than)
>=	크거나 같다 (Greater Than or Equal to)
<	작다 (Less Than)
<=	작거나 같다 (Less Than or Equal to)

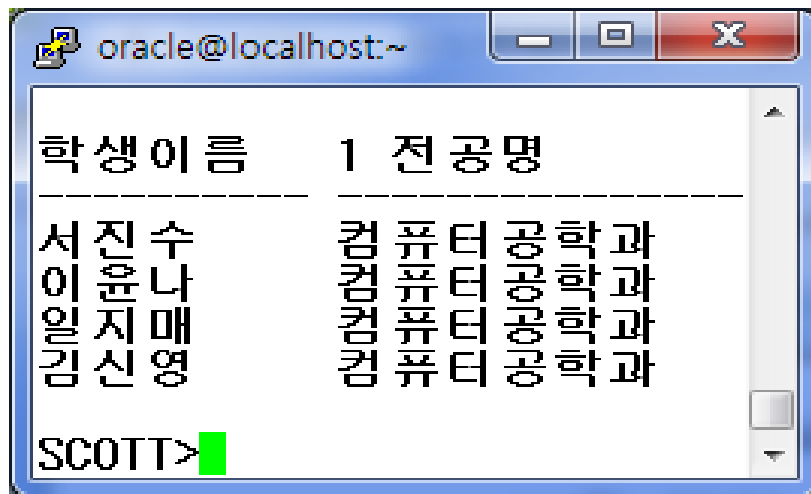
□ 단일 행 Sub Query 연습문제

(1) 연습문제 1:

Student 테이블과 department 테이블을 사용하여 이윤나 학생과 1 전공 (deptno1)이 동일한 학생들의 이름과 1전공 이름을 출력하세요.

(2) 단일 행 Sub Query 연습문제 2:

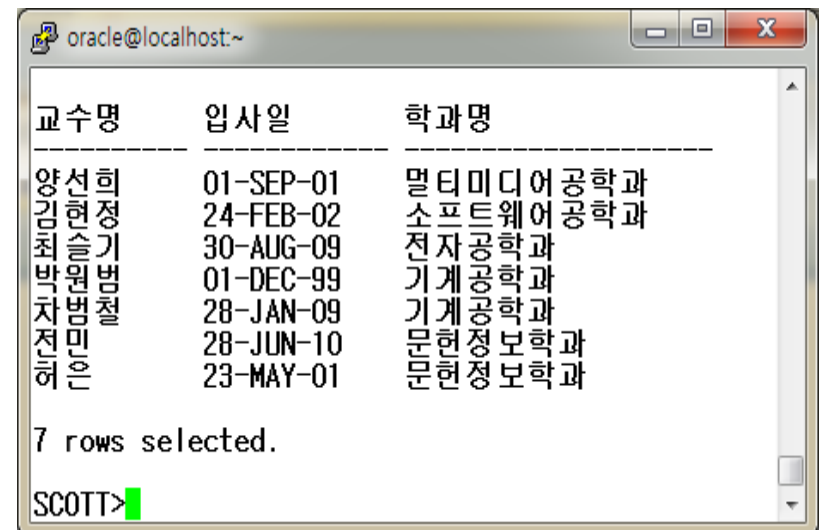
Professor 테이블에서 입사일이 송도권 교수보다 나중에 입사한 사람의 이름과 입사일, 학과명을 출력하세요.



oracle@localhost:~

학생이름	1 전공명
서진수	컴퓨터공학과
이윤나	컴퓨터공학과
일지매	컴퓨터공학과
김신영	컴퓨터공학과

SCOTT>



oracle@localhost:~

교수명	입사일	학과명
양선희	01-SEP-01	멀티미디어공학과
김현정	24-FEB-02	소프트웨어공학과
최슬기	30-AUG-09	전자공학과
박원범	01-DEC-99	기계공학과
차범철	28-JAN-09	기계공학과
전민호	28-JUN-10	문헌정보학과
허은	23-MAY-01	문헌정보학과

7 rows selected.

SCOTT>

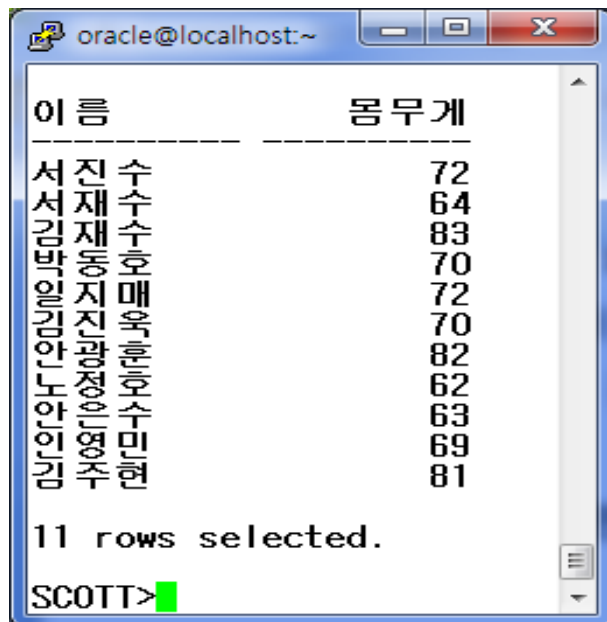
□ 단일 행 Sub Query 연습문제-2

(3)연습 문제 3:

Student 테이블에서 1 전공(deptno1)이 101번 인 학과의 평균 몸무게보다 몸무게가 많은 학생들의 이름과 몸무게를 출력하세요.

(4) 연습문제 4:

Professor 테이블에서 심슨 교수와 같은 입사일에 입사한 교수 중에서 조인 형 교수보다 월급을 적게 받는 교수의 이름과 급여, 입사일을 출력하세요.

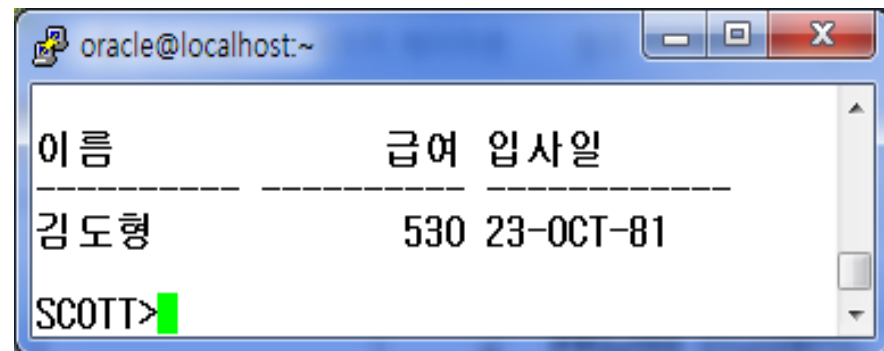


oracle@localhost:~

이름	몸무게
서진수	72
서재수	64
김재수	83
박동호	70
박지매	72
김진욱	70
안광훈	82
안영호	62
안영수	63
안민현	69
김주현	81

11 rows selected.

SCOTT>



oracle@localhost:~

이름	급여	입사일
김도형	530	23-OCT-81

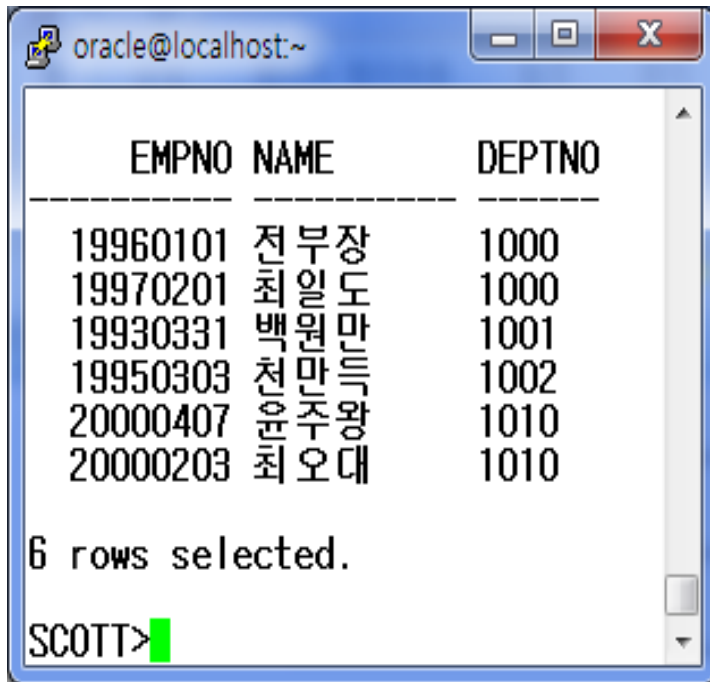
SCOTT>

□ 다중 행 Sub Query

연산자	의 미
IN	같은 값을 찾음
>ANY	최소값을 반환함
<ANY	최대값을 반환함
<ALL	최소값을 반환함
>ALL	최대값을 반환함
EXIST	Sub Query 의 값이 있을 경우 반환함

(1) 다중 행 Sub Query 예 1:

Emp2 테이블과 Dept2 테이블을 참조하여 근무지역(dept2 테이블의 area 컬럼)이 서울 지사인 모든 사원들의 사번과 이름, 부서번호를 출력하세요.



EMPNO	NAME	DEPTNO
19960101	전 부장	1000
19970201	최 일 도	1000
19930331	백 원 만	1001
19950303	천 만 덕	1002
20000407	윤 주 왕	1010
20000203	최 오 대	1010

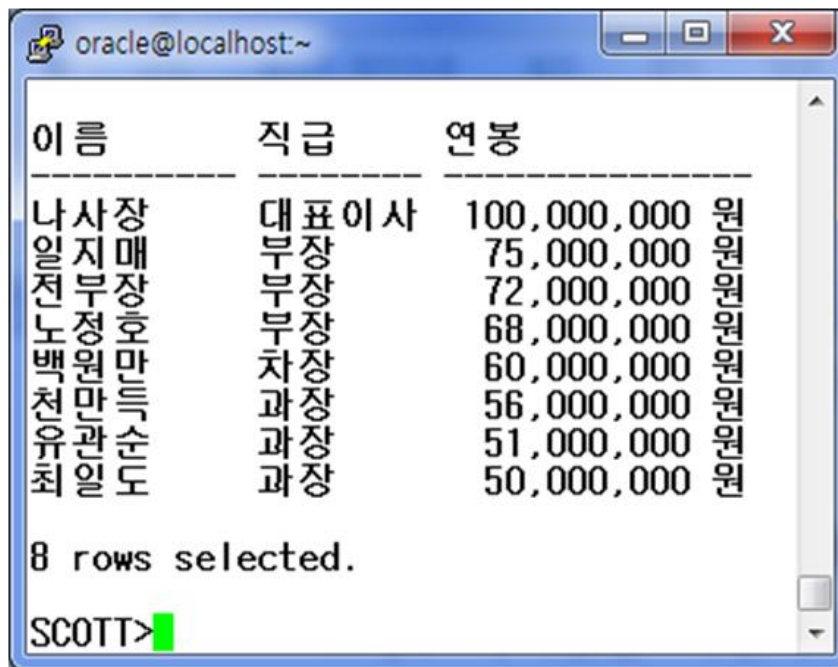
6 rows selected.

SCOTT>

```
SELECT empno, name, deptno
FROM emp2
WHERE deptno IN (SELECT dcode
                  FROM dept2
                  WHERE area='서울지사') ;
```

(2) 다중 행 Sub Query 연습문제 1:

Emp2 테이블을 사용하여 전체 직원 중 과장 직급의 최소 연봉자보다 연봉이 높은 사람의 이름과 직급, 연봉을 출력하세요. 단 연봉 출력 형식은 아래와 같이 천 단위 구분기호와 원 표시를 하세요.

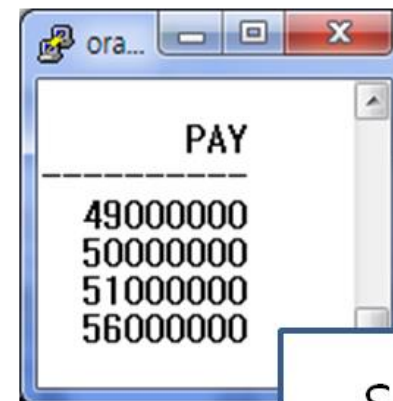


oracle@localhost:~

이름	직급	연봉
나사	대표이사	100,000,000 원
일지매	부부장	75,000,000 원
전부장	부부장	72,000,000 원
노정호	부부장	68,000,000 원
백원만	차장	60,000,000 원
천만	과장	56,000,000 원
유관순	과장	51,000,000 원
최일도	과장	50,000,000 원

8 rows selected.

SCOTT>



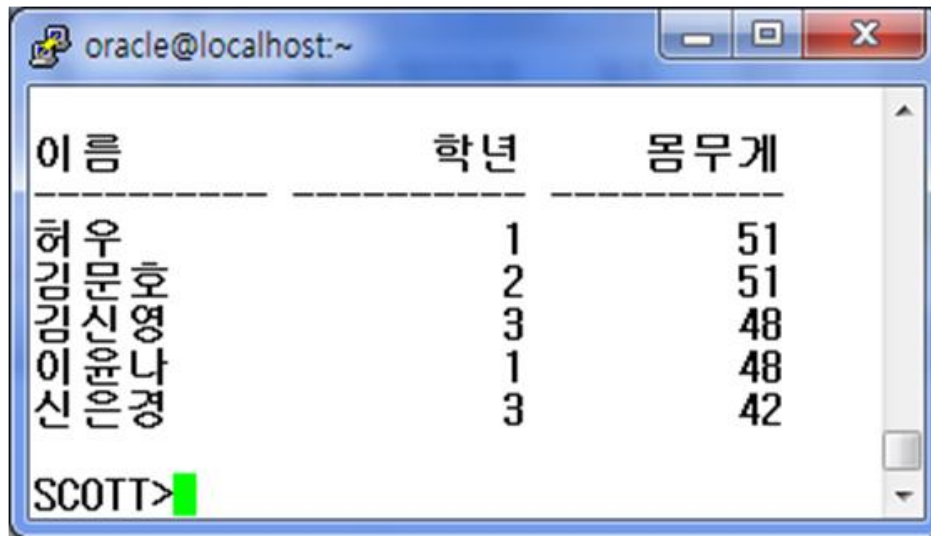
ora...

PAY
490000000
500000000
510000000
560000000

Sub Query
수행 결과화면

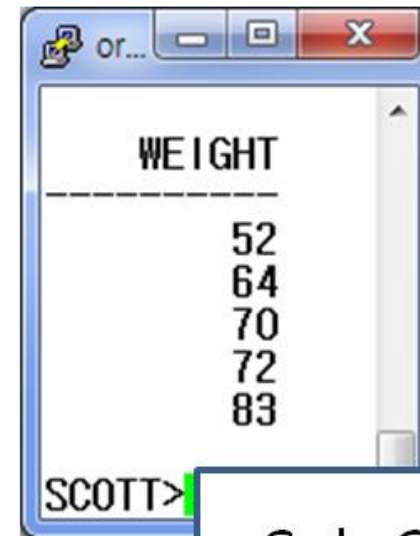
(3) 다중 행 Sub Query 연습문제 2:

Student 테이블을 조회하여 전체 학생 중에서 체중이 4학년 학생들의 체중에서 가장 적게 나가는 학생보다 몸무게가 적은 학생의 이름과 학년과 몸무게를 출력하세요



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

SCOTT>

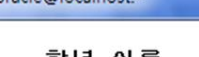


WEIGHT
52
64
70
72
83

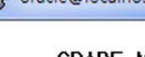
SCOTT>

Sub Query
수행 결과

(1) 다중 컬럼 Sub Query 예 1 :



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



```
oracle@localhost:~  
-----  
GRADE  MAX(HEIGHT)  
-----  
1      179  
2      184  
3      177  
4      182  
SCOTT>
```

```
SELECT grade "학년" ,name "이름" , height "키"
FROM student
WHERE (grade,height) IN (SELECT grade, MAX(height)
                        FROM student
                        GROUP BY grade )
ORDER BY 1 ;
```

(2) 다중 컬럼 Sub Query 연습문제 1 :

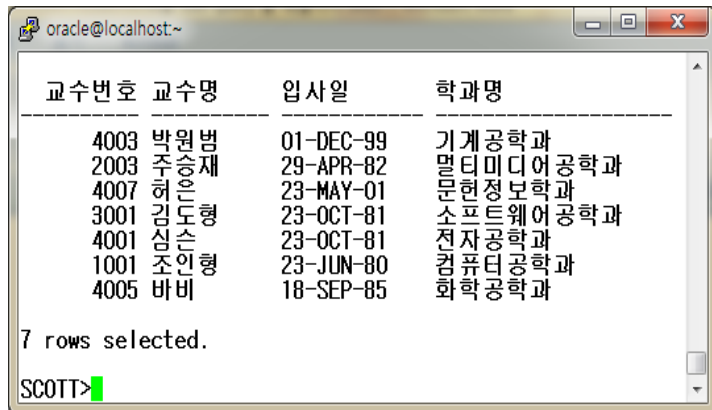
Professor 테이블을 조회하여 각 학과별로 입사일이 가장 오래된 교수의 교수번호와 이름, 학과명을 출력하세요. (학과이름순으로 오름차순 정렬하세요)

(3) 다중 컬럼 Sub Query 연습문제 2:

Emp2 테이블을 조회하여 직급별로 해당 직급에서 최대 연봉을 받는 직원의 이름과 직급, 연봉을 출력하세요. 연봉순으로 오름차순 정렬하세요.

(4) 다중 컬럼 Sub Query 연습문제 3:

Emp2 테이블을 조회하여 각 부서별 평균 연봉을 구하고 그 중에서 평균 연봉이 가장 적은 부서의 평균 연봉보다 적게 받는 직원들의 부서명, 직원명, 연봉을 출력하세요

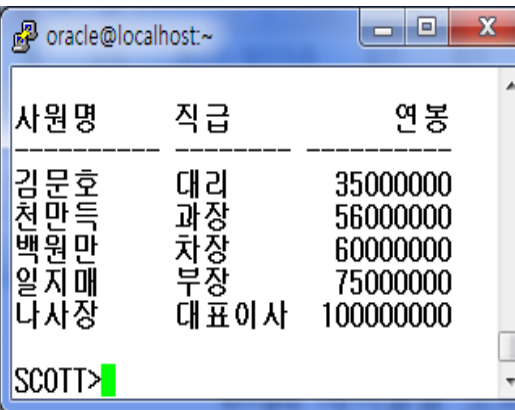


oracle@localhost:~

교수번호	교수명	입사일	학과명
4003	박원범	01-DEC-99	기계공학
2003	주승재	29-APR-82	기계공학
4007	허도형	23-MAY-01	기계공학
3001	김도형	23-OCT-81	기계공학
4001	심스	23-OCT-81	기계공학
1001	조인형	23-JUN-80	기계공학
4005	바비	18-SEP-85	기계공학

7 rows selected.

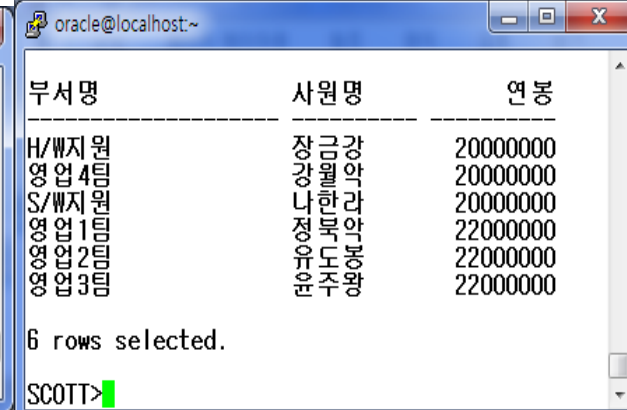
SCOTT>



oracle@localhost:~

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

SCOTT>



oracle@localhost:~

부서명	직원명	연봉
H/W지원	장강	20000000
영업4팀	김월악	20000000
S/W지원	나한라	20000000
영업1팀	정복악	22000000
영업2팀	유동주	22000000
영업3팀	윤주왕	22000000

6 rows selected.

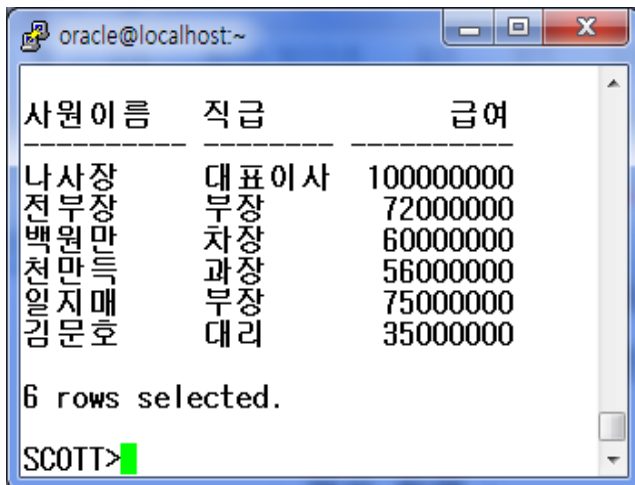
SCOTT>

□ 상호 연관 Sub Query

- Main Query 값을 Sub Query에 주고 Sub Query를 수행한 후 그 결과 값을 다시 Main Query로 반환

상호 연관 Sub Query 예 :

Emp2 테이블을 조회해서 직원 들 중에서 자신의 직급의 평균연봉과 같거나 많이 받는 사람들의 이름과 직급, 현재 연봉을 출력하세요.



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

6 rows selected.

SCOTT>

```
SELECT name "사원이름", position "직급" , pay "급여"
FROM emp2 a
WHERE pay >= ( SELECT AVG(pay)
                FROM emp2 b
                WHERE a.position=b.position) ;
```

Scalar Sub Query (스칼라 서브쿼리)

□ 스칼라 서버 쿼리

- select 절에 오는 서버 쿼리로 한번에 결과를 1행씩 반환

- 예 :

emp2 테이블과 dept2
테이블을 조회하여 사원
들의 이름과 부서이름을
출력하세요.

```
SELECT name "사원이름",  
       (SELECT dname  
        FROM dept2 d  
        WHERE e.deptno=d.dcode) "부서이름"  
FROM emp2 e;
```

- 스칼라 쿼리는 Join 방법으로 가능, 데이터 양이 적을 때 Join보다 실행 속도가 빠름

□ 스칼라 서브 쿼리의 실행 순서

1. Main Query 를 수행한 후 Scalar Sub Query 에 필요한 값을 제공.
2. Scalar Sub Query 를 수행하기 위해 필요한 데이터가 들어있는 블록을 메모리로 로딩.
3. Main Query 에서 주어진 조건을 가지고 필요한 값을 검색후 이 결과를 메모리에 입력값과 출력값으로 메모리 내의 query execution cache 라는 곳에 저장함. 여기서 입력값은 Main Query 에서 주어진 값이고 출력값은 Scalar Sub Query 를 수행 후 나온 결과값임. 이 값을 저장하는 캐쉬 값을 지정하는 파라미터는 `_query_execution_cache_max_size` 임.
4. 다음 조건이 Main Query 에서 Scalar Sub Query 로 들어오면 해쉬 함수를 이용해서 해당 값이 캐쉬에 존재하는 지 찾고 있으면 즉시 결과 값을 출력하고 없으면 다시 블록을 액세스 해서 해당 값을 찾은 후 다시 메모리에 캐싱 함.
5. Main Query 가 끝날 때까지 반복.

스칼라 서브 쿼리는 주로 코드성 테이블을 조회 할 경우에 적합함

□ 실행 계획 비교

PuTTY (오프라인)

Execution Plan

Plan hash value: 3092771810

스칼라 서브 쿼리의 실행계획

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		20	240	3 (0)	00:00:01
1	TABLE ACCESS BY INDEX ROWID	DEPT2	1	14	1 (0)	00:00:01
* 2	INDEX UNIQUE SCAN	SYS_C0014054	1		0 (0)	00:00:01
3	TABLE ACCESS FULL	EMP2	20	240	3 (0)	00:00:01

PuTTY (오프라인)

Execution Plan

Plan hash value: 1691961045

Join 쿼리의 실행계획

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		20	520	6 (17)	00:00:01
1	MERGE JOIN		20	520	6 (17)	00:00:01
2	TABLE ACCESS BY INDEX ROWID	DEPT2	13	182	2 (0)	00:00:01
3	INDEX FULL SCAN	SYS_C0014054	13		1 (0)	00:00:01
* 4	SORT JOIN		20	240	4 (25)	00:00:01
5	TABLE ACCESS FULL	EMP2	20	240	3 (0)	00:00:01