06강 서브쿼리

서브 쿼리의 개념

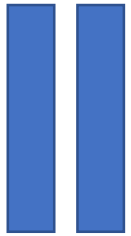
- SCOTT 사원의 급여보다 많은 사원을 검색한다고 가정한다면
쿼리문은 두개가 만들어져야 한다.
첫번째 SCOTT 사원의 급여를 조회하는 쿼리가 필요하다.
두번째 위에서 구한 급여와 비교해서 사원목록을 조회하는 쿼리가 필요하다.

위 두개의 쿼리를 하나의 쿼리문의 합칠 수가 있는데
최종적인 결과를 얻기 위한 쿼리를 main 쿼리라고 하고
mina쿼리를 보조하기 위해서 사용되는 쿼리를 sub쿼리라고 한다.

서브 쿼리의 개념

```
SELECT salary
FROM employee
WHERE ename='SCOTT';
```

```
SELECT ename, salary
FROM employee
WHERE salary > 3000;
```



```
SELECT ename, salary
FROM employee
WHERE salary > (SELECT salary
                FROM employee
                WHERE ename='SCOTT');
```

서브 쿼리의 개념

- 서브쿼리의 역할은 메인 쿼리 내부에서 메인 쿼리가 필요로 하는 값을 만들어 전달하는 역할을 한다.

- 서브쿼리가 만들어 내는 결과값의 개수에 따라
  - 단일 행 서브쿼리
  - 다중 행 서브쿼리

로 분류한다.

- 서브쿼리는 괄호()로 묶고 비교 조건의 오른쪽에 넣는다.
- 단일행 서브쿼리는 단일행 비교 연산만
- 다중 행 서브쿼리는 다중행 비교 연산만 사용해야 한다.

단일행 서브쿼리

- 예제: 최소 급여를 받는 사원의 이름, 담당업무, 급여를 출력하기

```
SELECT ename, job, salary
FROM employee
WHERE salary=(SELECT MIN(salary)
              FROM employee);
```

단일행 서브쿼리

- 서브쿼리는 결과값을 메인쿼리의 조건에 비교를 위해 사용되므로

WHERE 절에 주로 사용된다

- 그룹화의 조건도 서브쿼리의 용도에 대상이 된다.

HAVING 절에도 사용가능

- 예제 각 부서별 최소급여가 30번 부서의 최소급여보다 큰 부서의 부서번호와 그때의 최소급여를 출력해 보자

```
SELECT dno, min(salary)
FROM employee
GROUP BY dno
HAVING MIN(salary) > (SELECT MIN(SALARY)
                      FROM employee
                      WHERE dno=30);
```

다중행 서브쿼리

- 다중 행 서브 쿼리는 서브쿼리의 반환 결과가 하나 이상의 행일 때 사용하는 서브 쿼리
- 반드시 다중 행 연산자와 함께 사용해야 한다.

다중 행 연산자	의미
IN	서브 쿼리의 결과중 하나라도 일치하면 True
ANY, SOME	서브 쿼리의 결과와 하나 이상 일치하면 True
ALL	서브 쿼리의 결과와 모든 값이 일치해야 True
EXSIT	비교 조건이 서브 쿼리의 결과중 만족하는 값이 하나라도 존재하면 True

다중행 서브쿼리

- IN 연산자 => 메인 쿼리의 비교 조건에서 서브 쿼리의 출력 결과와 하나라도 일치하면 메인 쿼리의 WHERE 절이 참이 되게 하는 연산자

- 예제: 부서별 최소 급여를 받는 사원 번호와 이름을 출력하는 쿼리문

```
SELECT eno, ename
FROM employee
WHERE salary = (SELECT min(salary)
                FROM employee
                GROUP BY dno);
```

에러 발생

```
SELECT eno, ename
FROM employee
WHERE salary IN(SELECT min(salary)
                FROM employee
                GROUP BY dno);
```

에러 해소

다중행 서브쿼리

- ANY 연산자 => 서브 쿼리가 반환하는 각각의 값과 비교 (OR)
  - <ANY : 최대값보다 작음
  - >ANY : 최소값보다 큼
  - =ANY : IN과 동일
  - ANY와 SOME는 같은 역할을 하는 연산자

- 예제: 직급이 SALESMAN이 아니면서 급여가 SALESMAN보다 낮은 사원을 출력

```
SELECT *
FROM employee
WHERE salary < ANY (SELECT salary
                    FROM employee
                    WHERE job='SALESMAN')
AND job<>'SALESMAN';
```

```
SELECT MAX(salary)
FROM employee
WHERE job='SALESMAN';
```

다중행 서브쿼리

- ALL 연산자 => 서브 쿼리가 반환하는 모든 값과 비교 (AND)
  - < ALL : 최소값보다 작음
  - > ALL : 최대값보다 큼

- 예제: 직급이 SALESMAN이 아니면서 급여가 SALESMAN보다 급여가 적은 사원을 출력

```
SELECT *
FROM employee
WHERE salary < ALL (SELECT salary
                    FROM employee
                    WHERE job='SALESMAN')
AND job<>'SALESMAN';
```

```
SELECT MIN(salary)
FROM employee
WHERE job='SALESMAN';
```

다중행 서브쿼리

- 서브쿼리 값의 범위