* Sub Query
  + 서브 쿼리는 하나의 SELECT 문장의 절 안에 포함된 또 하나의 SELECT 문장입니다.
  + 서브 쿼리를 포함하고 있는 쿼리를 메인 쿼리, 포함된 또 하나의 쿼리를 서브 쿼리라 합니다.
  + 서브 쿼리는 비교 연산자의 오른쪽에 기술해야 하고 반드시 괄호로 둘러쌓아야 합니다.
  + 칼럼 출력시, 질의(where)의 값으로, table을 대신하는 방식으로 사용할 수 있습니다.
  + select ename, sal+ifnull(comm,0) total from emp where total between 2500 and 3000;
    - **(조건에서 사용되는 서브쿼리)**는 메인쿼리가 실행되기 전 실행이 됩니다.
      * 서로 참조하고 있는 테이블의 필드를 검색에 조건으로 사용해야 되는경우
      * join 할 테이블의 필드를 검색조건으로 필요로 하는데 필드를 출력용으로 사용하지 않을때는 서브쿼리가 join보다 좋은 방법이다.
        + 예) esearch부서에 근무하는 사원을 알고싶은데 부서정보는 출력안할 때
      * 그룹함수의 결과를 검색조건으로 사용해야 할 때
        + 예) 가장 높은 급여를 받는 사원
      * 다수의 검색 결과를 다시 검색조건으로 사용할 때
        + 예) 부서별로 급여를 가장 많이 받는 사원
    - **(필드로 사용되는 서브쿼리)**는 해당 row가 출력 될 때마다 실행됩니다
      * 필드로 출력되는 내역이 다른 테이블의 조건으로 사용될 때
      * join과 아주 유사하게 사용할 수 있다
      * 서브쿼리가 실행되면 꼭 1개의 값만 반환되어야한다. (single row)
      * 레코드를 실행 할 때 마다 서브쿼리를 실행하기 때문에 성능체크 필요
      * select \*,(select dname from dept limit 0,1) 부서이름 from emp
      * emp select \*,(select dname from dept where deptno=emp.deptno) 부서이름 from emp;
      * 에 급여등급 출력
      * emp에 부서이름 출력
      * select emp.\* dept.dname from emp left join dept using(deptno);
    - **(from절에 있는 table로 사용되는 서브쿼리)는** 서브쿼리가 먼저 실행되고, 꼭 별칭을 지어주어야한다.
    - table에 있는 레코드(물리적으로 저장된)를 where의 조건에 맞는 것만 필드의 내용으로 출력하는 것이 질의인데.. 이미 출력된 내용을 출력의 검색 조건으로 사용하는 것은 불가능하다!
    - total을 출력한 내역을 다시 table로 사용하면 출력조건으로 사용가능하다  
      table로사용하는 서브쿼리
    - select \* from (select ename, sal+ifnull(comm,0) total from emp) as a where a.total between 2500 and 3000;
  + 서브 쿼리는 메인 쿼리가 실행되기 이전에 한번만 실행이 됩니다.
  + 'SMITH'씨와 동일한 부서에서 일하는 직원의 이름을 출력하세요.

SELECT ename FROM EMP

WHERE deptno = (SELECT deptno FROM EMP

WHERE ename = 'SMITH');

* + 연습문제)JONES이 일하는 부서의 이름(DEPT)은?
  + SELECT DNAME FROM DEPT

WHERE DEPTNO = ( SELECT DEPTNO

FROM EMP

WHERE ENAME='JONES' );

1. Single Row Sub Query
   1. 단일 행(Single Row) 서브 쿼리는 수행 결과가 오직 하나의 로우(행, row)만을 반환하는 서브 쿼리를 갖는 것을 말합니다.
   2. 단일 행 서브 쿼리문에서는 이렇게 오직 하나의 로우(행, row)로 반환되어야하고  
       서브 쿼리의 결과는 메인 쿼리에 보내게 되는데 메인 쿼리의 WHERE 절에서는 단일 행 비교 연산자인 =, >, >=, <, <=, <>를 사용해야 합니다.
   3. KING과 같은 부서에서 근무하는 사원의 이름과 부서 번호를 출력하는 SQL 문을 작성해 보시오.

SELECT ename, deptno FROM EMP

WHERE deptno=(

SELECT deptno

FROM EMP

WHERE ename='KING' )

;

* 1. 문제1) MARTIN와 동일한 직급을 가진 사원의 이름을 출력하세요.
     1. SELECT ename FROM EMP

WHERE ename<>'MARTIN'

AND job =( SELECT job

FROM EMP

WHERE ename='MARTIN');

* 1. 문제2) MARTIN의 급여와 동일하거나 더 많이 받는 사원 명과 급여를 출력하시오.
     1. SELECT ename, sal FROM EMP

WHERE ename<>'MARTIN'

AND sal >=( SELECT sal

FROM EMP

WHERE ename='MARTIN')

ORDER BY sal;

* 1. 문제3) DALLAS에서 근무하는 사원의 이름, 부서 번호를 출력하시오.
     1. SELECT ename, deptno FROM EMP

WHERE deptno = ( SELECT deptno

FROM DEPT

WHERE loc = 'DALLAS');

* 1. 문제4) SALES(영업부) 부서에서 근무하는 모든 사원의 이름과 급여를 출력하시오.
     1. SELECT ename, sal FROM EMP

WHERE deptno = ( SELECT deptno

FROM DEPT

WHERE dname= 'SALES');

* 1. 문제5) 직속상관이 KING인 사원의 이름과 급여를 출력하시오.
     1. SELECT ename, sal FROM EMP

WHERE mgr =( SELECT empno

FROM EMP

WHERE ename='KING');

1. Single Row Sub Query Used Aggregate Functions
   1. Aggregate Functions는 집계 함수로 우리가 흔히 말하는 group 함수이다.
   2. 문제1) 평균 급여보다 더 많은 급여를 받는 사원을 검색하세요.
      1. SELECT ename, sal FROM EMP

WHERE sal > ( SELECT AVG(sal)

FROM EMP);

1. Multiple Row Sub Query
   1. 다중 행 서브 쿼리는 서브 쿼리에서 반환되는 결과가 하나 이상의 행일 때 사용하는 서브 쿼리입니다. 다중 행 서브 쿼리는 반드시 다중 행 연산자(Multiple Row Operator)와 함께 사용해야 합니다.
   2. IN
      1. 메인 쿼리의 비교 조건(‘=’ 연산자로 비교할 경우)이 서브 쿼리의 결과 중에서 하나라도 일치하면 참입니다.
      2. 예제1) 급여를 3000 이상 받는 사원이 소속된 부서와 동일한 부서에서 근무하는 사원 이름, 급여, 부서명을 출력하세요

SELECT e.ename, e.sal, d.dname

FROM EMP e,DEPT d

WHERE e.deptno = d.deptno

AND e.deptno IN( SELECT deptno

FROM EMP

WHERE sal>=3000);

* + 1. 문제1-1)부서별로 가장 급여를 많이 받는 사원의 정보(사원 번호, 사원 이름, 급여, 부서 번호)를 출력하시오.(IN 연산자 이용)
       1. 힌트) 부서별 가장 급여를 많이 받는 사원 : GROUP BY
       2. SELECT EMPNO,ENAME,DEPTNO FROM EMP

WHERE SAL IN (

SELECT MAX(sal) FROM EMP

GROUP BY DEPTNO)

* + 1. 문제1-2)직급(JOB)이 MANAGER인 사람이 속한 부서의 부서 번호와 부서명, 지역을 출력하시오.
       1. SELECT DEPTNO,DNAME,LOC FROM DEPT

WHERE DEPTNO IN (

SELECT DEPTNO FROM EMP

WHERE LOWER(JOB) = 'manager')

* 1. ALL
     1. 메인 쿼리의 비교 조건이 서브 쿼리의 검색 결과와 모든 값이 일치하면 참입니다.
     2. 찾아진 값에 대해서 AND 연산을 해서 모두 참이면 참이 되는 셈이 됩니다. > ALL 은 “모든 비교값 보다 크냐”고 묻는 것이 되므로 최대값보다 더 크면 참이 됩니다. ( ALL(30,40,50) 50이 선택됨)
     3. 예제2) 30번 소속 사원들 중에서 급여를 가장 많이 받는 사원보다 더 많은 급여를 받는 사람의 이름, 급여를 출력하세요.
        1. > ALL() 사용

SELECT ENAME, SAL FROM EMP

WHERE SAL > ALL( SELECT SAL

FROM EMP

WHERE DEPTNO =30);

* + - 1. MAX() 사용

SELECT ENAME, SAL FROM EMP

WHERE SAL > ( SELECT MAX(SAL)

FROM EMP

WHERE DEPTNO =30);

* + 1. 문제2-1)영업 사원들 보다 급여를 많이 받는 사원들의 이름과 급여, 직급(담당 업무)을 출력하되 영업 사원은 출력하지 않습니다.
       1. SELECT ENAME,SAL,JOB FROM EMP

WHERE SAL > ALL(

SELECT SAL FROM EMP

WHERE LOWER(JOB) = 'salesman')

* + - 1. SELECT ENAME,SAL,JOB FROM EMP

WHERE SAL >(

SELECT MAX(SAL) FROM EMP

WHERE LOWER(JOB) = 'salesman')

* 1. ANY, SOME
     1. 메인 쿼리의 비교 조건이 서브 쿼리의 검색 결과와 하나 이상이 일치하면 참입니다.
     2. > ANY() 는 찾아진 값에 대해서 하나라도 크면 참이 되는 셈이 됩니다. 그러므로 찾아진 값 중에서 가장 작은 값 즉, 최소 값 보다 크면 참이 됩니다. ( ANY(30,40,50) 30이 선택됨)
     3. 예제3-1) 부서번호가 30번인 사원들중 급여가 가장 낮은 사람보다 높은 급여를 받는 사원의 이름과 급여를 출력하세요.
        1. MIN()을 사용한 구문

SELECT ENAME, SAL FROM EMP

WHERE SAL > (

SELECT MIN(SAL) FROM EMP

WHERE DEPTNO=30);

* + - 1. ANY()를 이용

SELECT ENAME, SAL FROM EMP

WHERE SAL > ANY(

SELECT SAL FROM EMP

WHERE DEPTNO=30);

* + 1. 문제3-1) 영업 사원들의 최소 급여보다 많이 받는 사원들의 이름과 급여와 직급(담당 업무)을 출력하되 영업사원은 출력하지 않습니다.
       1. SELECT ENAME,SAL,JOB FROM EMP

WHERE LOWER(JOB)<> 'salesman'

AND SAL > ANY(

SELECT SAL FROM EMP

WHERE LOWER(JOB) = 'salesman')

* + 1. 3-2) SMITH와 동일한 직급을 가진 사원의 이름과 직급을 출력합니다.(단일행 서브 쿼리 이용)
       1. SELECT ENAME, JOB FROM EMP

WHERE JOB =(

SELECT JOB FROM EMP

WHERE ENAME ='SMITH')

* + 1. 3-3) 직급이 ‘SALESMAN’인 사원이 받는 급여들의 최대 급여보다 많이 받는 사원들의 이름과 급여를 출력하되 부서번호가 20번인 사원은 제외합니다.(ALL 연산자 이용)
    2. 3-4) 직급이 ‘SALESMAN’인 사원이 받는 급여들의 최소 급여보다 많이 받는 사원들의 이름과 급여를 출력하되 부서번호가 20번인 사원은 제외한다.(ANY 연산자 이용)
  1. EXIST

메인 쿼리의 비교 조건이 서브 쿼리의 결과 중에서 만족하는 값이 하나라도 존재하면 참입니다.

* + 1. SELECT 칼럼

FROM 테이블

WHERE EXISTS (서브 쿼리)

* + - 1. 서브 쿼리의 결과가 하나 이상 존재할 경우 그 조건으로 메인 쿼리를 실행한다.
    1. SELECT 칼럼

FROM A

WHERE EXISTS (

SELECT \* FROM B

WHERE A.key = B.key );

* + - 1. 마치 join을 하듯 "A.key = B.key" 라는 구문이 있는 데 이를 "set 한다" 라고 한다.
      2. 서브 쿼리의 set 한 검색 결과가 존재할 경우 set의 조건으로 메인 쿼리를 실행한다.
      3. Where의 조건으로 Inner Join이 필요하지만 결과는 한 개의 테이블에서 해결될 경우 사용하면 성능이 향상된다.
      4. EXISTS()는 IN()과 거의 똑같이 사용 가능하지만 IN()은 결과를 실제로 출력하지만 EXISTS()는 결과가 실제로 존재 하는지 만 검사해서 성능이 IN()보다 좋다.
    1. 예1-1) EMP 테이블에서 사용 중인 DEPT 테이블의 deptno와 dname을 출력하라. IN() 을 이용

SELECT dname, deptno FROM dept

WHERE deptno IN (

SELECT DISTINCT(deptno) FROM EMP);

* + 1. 예1-2) 위의 문제를 EXITS()이용

SELECT dname, deptno FROM dept

WHERE EXISTS (

SELECT \* FROM EMP

WHERE emp.deptno = dept. deptno );

* + 1. 예2-1) EMP 테이블에서 사용 하지 않는 DEPT 테이블의 deptno와 dname을 출력하라. NOT IN() 을 이용

SELECT dname, deptno FROM dept

WHERE deptno NOT IN (

SELECT DISTINCT(deptno) FROM EMP);

* + 1. 예2-2)위의 문제를 NO EXISTS() 이용

SELECT dname, deptno FROM dept

WHERE NOT EXISTS (

SELECT \* FROM EMP

WHERE emp.deptno =dept. deptno );

* + 1. 문제1) 커미션을 받는 부서명을 출력하라 (EXITS 이용)

SELECT dname FROM dept

WHERE EXISTS (

SELECT \* FROM EMP

WHERE emp.deptno =dept. deptno

AND emp.comm>0);

* + - 1. SELECT DNAME,DEPTNO FROM DEPT

WHERE DEPTNO IN (

SELECT DEPTNO FROM EMP

WHERE COMM>0)

* + 1. 문제2)부서명이 'ACCOUNTING'인 사원의 사원 번호와 이름, 급여를 출력하세요.(문제 실수)
       1. SELECT ename, sal , job FROM EMP e

WHERE EXISTS (

SELECT \* FROM DEPT d

WHERE e.deptno =d.deptno

AND dname = 'ACCOUNTING');

문제3) 50번인 부서 번호가 있으면 SMITH씨의 부서를 50번으로 바꾸세요. (50번 부서가 없기 때문에 실행 0)

UPDATE EMP SET deptno = 50

WHERE ename = 'SMITH'

AND EXISTS(

SELECT \*FROM DEPT

WHERE deptno=50

) ;