subquery

subquery는 select절, from절, where절, having절, order by절에 선언 가능

where절 subquery는 연산자 오른쪽에 ()안에 선언합니다.

single row subquery는 >, =, <>, >=, <, <= single row operator와 함께 선언합니다.

multiple row subquery는 in, any, all multiple row operator와 함께 선언합니다.

()안에 선언되는 subquery의 select문은 order by절을 제외하고 모두 사용가능, 그룹함수도 사용 가능

conn scott/oracle

Q> ADAMS 사원보다 급여가 높은 사원이름과 급여 조회

select ename, sal

from emp

where sal > (ADAMS 사원의 급여);

select ename, sal

from emp

where sal > (select sal

from emp

where ename = 'ADAMS' );

Q> 7369사원과 동일한 직무를 담당하는 사원의 이름, 부서번호, 직무, 급여를 조회 (7369사원은 결과에서 제외하시오)

select ename, deptno, job, sal

from emp

where job = (select job

from emp

where empno = 7369)

and empno <> 7369;

Q> allen사원과 동일한 부서에 근무하면서 급여를 더 많이 받는 사원 검색

(사원이름, 부서번호, 직무, 급여)

select ename, deptno, job, sal

from emp

where deptno = (select deptno

from emp

where ename='ALLEN')

and sal > (select sal

from emp

where ename='ALLEN');

Q> emp테이블에서 급여를 제일 적게 받는 사원의 이름, 부서, 직무, 급여를 검색

select ename, deptno, job, sal

from emp

where sal = (select min(sal)

from emp);

Q> 부서별 최대 급여를 받는 사원들의 이름, 부서, 직무, 급여를 검색

select ename, deptno, job, sal

from emp

where sal in (select max(sal)

from emp

group by deptno);

update emp

set deptno = 10 , sal=2850

where empno = 8000;

select \* from emp where empno = 8000;

commit;

select ename, deptno, job, sal

from emp

where (deptno, sal) in (select deptno, max(sal)

from emp

group by deptno); --multiple row + multiple column subquery

Q> 직무가 salesman인 모든 사원보다 급여를 많이 받는 사원검색

(이름, 직무, 급여 - 단 salesman직무 제외)

select ename, deptno, job, sal

from emp

where sal >all (select sal

from emp

where job = 'SALESMAN')

and job <> 'SALESMAN';

select ename, deptno, job, sal

from emp

where sal >(select max(sal)

from emp

where job = 'SALESMAN')

and job <> 'SALESMAN';

Q> 직무가 salesman인 최소 한명의 사원보다 급여를 많이 받는 사원검색

(이름, 직무, 급여 - 단 salesman직무 제외)

select ename, deptno, job, sal

from emp

where sal >any (select sal

from emp

where job = 'SALESMAN')

and job <> 'SALESMAN';

select ename, deptno, job, sal

from emp

where sal >(select min(sal)

from emp

where job = 'SALESMAN')

and job <> 'SALESMAN';

Q> (각 부서별) 전체 사원의 평균 급여보다 급여를 많이 받는 사원 검색

select ename, deptno, sal

from emp

where sal > (select avg(sal)

from emp);

Q> 각 부서별 부서의 평균 급여보다 급여를 많이 받는 사원 검색

join 또는 co-related subquery

select ename, deptno, sal

from emp a

where sal > (select avg(sal)

from emp b

where a.deptno = b.deptno); --co-related subuqery

select a.ename, a.deptno, a.sal

from emp a, (select deptno, avg(sal) avgsal

from emp

group by deptno) b

where a.deptno = b.deptno and a.sal > b.avgsal;

상관관계 서브쿼리는 main query의 행수만큼 반복 수행합니다.

서브쿼리는 main query의 컬럼을 참조해서 수행하므로 ...

conn hr/oracle

Q> 전체 평균 급여보다 높은 부서별 평균 급여를 검색

##################################################

join -하나이상의 테이블로부터 동일한 속성의 컬럼값이 일치할때 레코드를 결합해서 검색 (테이블의 하나의 레코드가 다른 테이블의 하나의 레코드와 결합)

조인문법 -where절에 조인 조건 선언

sql 1999 표준 조인 문법 (from절 조인 선언)

join 종류 :

equi join(inner join) : 조인할 두 테이블에서 동일한 속성의 컬럼값이 일치할때 (=연산자 사용)

non-equi join : 조인할 두 테이블에서 동일한 속성의 컬럼이 존재하지 않은 경우 (=가 아닌 다른 연산자를 사용해서 조인 조건을 선언)

self-join : 조인할 두 테이블이 동일한 테이블(하나의 테이블에서 PK를 참조하는 FK가 존재하는 경우로서 자기참조 가능한 테이블)로부터 레코드를 결합해서 검색

outer-join : equi join에서 조인할 두 테이블에서 동일한 속성의 컬럼값이 일치할때 레코드를 결합해서 검색 결과를 생성하므로, 조인할 컬럼중 하나가 null인경우는 조인 결과집합에서 레코드가 누락되지 않고 결과집합에 가져오기 위해 수행하는 join

cartesian product(cross join) - 조인조건이 누락되면 조인할 테이블의 레코드가 전체 레코드에 대해서 한번씩 조인됨

conn hr/oracle

Q> 사원번호, 사원이름, 소속부서번호, 소속부서이름을 검색

select employee\_id, last\_name, department\_id, department\_name

from employees , departments; --조인조건 누락, 컬럼정의 모호함 오류

select a.employee\_id, a.last\_name, a.department\_id, b.department\_name

from employees a , departments b; --cartesian product (107 X 27 rows)

select a.employee\_id, a.last\_name, a.department\_id, b.department\_name

from employees a , departments b

where a.department\_id = b.department\_id; --equi join ( 106 rows)

natural join : 조인할 두 테이블에서 동일한 이름의 컬럼값이 일치할때 조인 결과를 생성 (동일한 이름의 컬럼앞에 소유자(테이블명, alias)를 선언하지 않습니다.)

select a.employee\_id, a.last\_name, a.department\_id, b.department\_name

from employees a natural join departments b; --error

select a.employee\_id, a.last\_name, department\_id, b.department\_name

from employees a natural join departments b; --- 32 rows

-- a.department\_id = b.department\_id and a.manager\_id = b.manager\_id

join ~using (column name) : 동일한 이름의 속성이 여러개인 테이블을 조인할 때, 하나의 컬럼으로만 equi 방식 조인을 수행

(동일한 이름의 컬럼앞에 소유자(테이블명, alias)를 선언하지 않습니다.)

select a.employee\_id, a.last\_name, department\_id, b.department\_name

from employees a join departments b using (department\_id); --106rows

conn scott/oracle

drop table dept2 purge; --테이블 삭제

desc dept2 --구조 삭제 확인

select \* from dept2; --레코드 삭제 확인

create table dept2 (deptid , dname, loc)

as

select \* from dept; --테이블 구조, 레코드 복제

desc dept2

select \* from dept2;

#두 테이블의 조인 컬럼이름이 다른 경우 : join~on 조건

Q> 사원번호, 사원이름, 소속부서번호, 소속부서이름을 검색

select a.empno, a.ename, a.deptno, b.dname

from emp a join dept2 b on a.deptno = b.deptid;

desc emp

desc salgrade

Q> 사원별로 이름, 급여, 급여등급을 검색 (non-equi join)

select a.ename, a.sal, b.grade

from emp a, salgrade b

where a.sal between b.losal and b.hisal;

select a.ename, a.sal, b.grade

from emp a join salgrade b on a.sal between b.losal and b.hisal;

emp테이블(자기참조관계가 있음) : empno(PK) <-----(참조) mgr(FK)

self join 가능

Q> 사원번호, 사원이름, 관리자번호, 관리자이름 검색

select e.empno, e.ename, e.mgr, m.ename

from emp e , emp m

where e.mgr = m.empno;

select e.empno, e.ename, e.mgr, m.ename

from emp e join emp m on e.mgr = m.empno;

Q> 부서번호, 사원이름, 동료사원이름 검색 (동료사원이름은 같은 부서에 속한 다른 사원이름만 검색 결과로 생성)

select e.deptno, e.ename, c.ename

from emp e, emp c

where e.deptno = c.deptno

and e.ename <> c.ename;

select e.deptno, e.ename, c.ename

from emp e join emp c on e.deptno = c.deptno and e.ename <> c.ename;

select e.deptno, e.ename, c.ename

from emp e join emp c on e.deptno = c.deptno

where e.ename <> c.ename;

select e.empno, e.ename, e.deptno, d.dname

from emp e cross join dept d ; ---(14 X 4 rows)

insert into emp (empno, ename)

values (8000, 'Hong'); --레코드 추가

commit; --저장

select \* from emp; --확인

select a.empno, a.ename, deptno, b.dname

from emp a join dept b using (deptno); ---equi join (8000번 사원누락)

# equi join에서 누락된 조인되지 못한 row를 결과집합으로 가져오려면 outer join을 수행해야 하며, 조인할 레코드가 없는 테이블의 조인조건에 outer 연산자 (+)를 추가합니다.

select a.empno, a.ename, a.deptno, b.dname

from emp a , dept b

where a.deptno = b.deptno(+);

select a.empno, a.ename, a.deptno, b.dname

from emp a left outer join dept b on a.deptno = b.deptno ;

Q> 사원이 없는 부서 정보를 포함해서 부서별 부서번호, 부서이름, 사원번호, 사원이름을 검색

select a.empno, a.ename, deptno, b.dname

from emp a join dept b using (deptno); --equi join (40번 부서정보 없음)

select b.deptno, b.dname, a.empno, a.ename

from emp a , dept b

where a.deptno(+) = b.deptno

order by 1 ;

select b.deptno, b.dname, a.empno, a.ename

from emp a right outer join dept b on a.deptno = b.deptno

order by 1 ;

Q> 사원이 없는 부서 정보를 포함하고, 부서를 배정받지 못한 사원정보를 포함해서 부서별 부서번호, 부서이름, 사원번호, 사원이름을 검색

select b.deptno, b.dname, a.empno, a.ename

from emp a , dept b

where a.deptno(+) = b.deptno(+)

order by 1 ; ---error

select b.deptno, b.dname, a.empno, a.ename

from emp a full outer join dept b on a.deptno = b.deptno

order by 1 ;

#n개의 테이블을 조인할 경우 최소 조인조건은 n-1개 선언해야 합니다.

conn hr/oracle

desc employees

desc departments

desc locations

Q> 사원이름, 부서이름, 부서의 도시명을 검색

select a.last\_name, b.department\_name, c.city

from employees a , departments b , locations c

where a.department\_id = b.department\_id

and b.location\_id = c.location\_id;

select a.last\_name, b.department\_name, c.city

from employees a join departments b

on a.department\_id = b.department\_id

join locations c on b.location\_id = c.location\_id;

조인연습문제1]

conn hr/oracle

select a.location\_id, a.street\_address, a.city, a.state\_province, b.country\_name

from locations a, countries b

where a.country\_id = b.country\_id;

select a.location\_id, a.street\_address, a.city, a.state\_province, b.country\_name

from locations a join countries b on a.country\_id = b.country\_id;

select a.location\_id, a.street\_address, a.city, a.state\_province, b.country\_name

from locations a natural join countries b ;

select a.location\_id, a.street\_address, a.city, a.state\_province, b.country\_name

from locations a join countries b using(country\_id) ;

조인연습문제3]

select a.last\_name, a.job\_id, b.department\_id, b.department\_name

from employees a , departments b , locations c

where a.department\_id = b.department\_id

and b.location\_id = c.location\_id

and c.city = 'Toronto';

select a.last\_name, a.job\_id, b.department\_id, b.department\_name

from employees a join departments b

on a.department\_id = b.department\_id

join locations c on b.location\_id = c.location\_id and c.city = 'Toronto';

select a.last\_name, a.job\_id, b.department\_id, b.department\_name

from employees a join departments b

on a.department\_id = b.department\_id

join locations c on b.location\_id = c.location\_id

where c.city = 'Toronto';

조인연습문제4]

select e.last\_name "Employee", e.employee\_id "Emp#" ,

m.last\_name "Manager", e.manager\_id "Mgr#"

from employees e, employees m

where e.manager\_id = m.employee\_id;

select e.last\_name "Employee", e.employee\_id "Emp#" ,

m.last\_name "Manager", e.manager\_id "Mgr#"

from employees e join employees m

on e.manager\_id = m.employee\_id; ---106rows (회사대표 king사원 누락)

조인연습문제5]

select e.last\_name "Employee", e.employee\_id "Emp#" ,

m.last\_name "Manager", e.manager\_id "Mgr#"

from employees e, employees m

where e.manager\_id = m.employee\_id(+);

select e.last\_name "Employee", e.employee\_id "Emp#" ,

m.last\_name "Manager", e.manager\_id "Mgr#"

from employees e left outer join employees m

on e.manager\_id = m.employee\_id;

조인연습문제6]

select e.department\_id department,

e.last\_name employee,

c.last\_name colleague

from employees e, employees c

where e.department\_id = c.department\_id

and e.last\_name <> c.last\_name

order by 1;

select e.department\_id department,

e.last\_name employee,

c.last\_name colleague

from employees e join employees c

on e.department\_id = c.department\_id

and e.last\_name <> c.last\_name

order by 1;

조인연습문제8] Davies. 사원 입사날짜 이후에 입사한 모든 사원의 이름과 입사날짜를 출력하시오(self join +non-equi join)

select b.last\_name, b.hire\_date

from employees a, employees b

where a.last\_name='Davies' and a.hire\_date < b.hire\_date;

--and b.last\_name <> 'Davies'

조인연습문제 9] 사원들중에서 각사원의 관리자보다 먼저 입사한 사원들을 아래와 같이 출력하시오

select e.last\_name last\_name , e.hire\_date hire\_date,

m.last\_name last\_name, m.hire\_date hire\_date

from employees e, employees m

where e.manager\_id = m.employee\_id

and e.hire\_date < m.hire\_date;

select e.last\_name last\_name , e.hire\_date hire\_date,

m.last\_name last\_name, m.hire\_date hire\_date

from employees e join employees m

on e.manager\_id = m.employee\_id

and e.hire\_date < m.hire\_date;