7_15_2022_-Oracle

select 한다는 것 - 데이터를 골라 불러오고 싶다.

select ename, dname, loc

ename은 emp에 있고 dname, loc은 dept에 있다

from emp, dept

emp,와 dept에서 불러온다.

이렇게만 불러올 시 두 조합의 불러올 수 있는 모든 정보가 출력된다.

```
where emp.deptno = dept.deptno
```

조건을 걸어준다 - emp에 있는 deptno와 = dept에 있는 deptno는 같다(row를 추리는 역할)

```
and sal>= 2000;
```

조건을 and 으로 구분해서 더 달아줄 수 있다.

ansi join

같은 결과를 보기 좋게 바꾸는 문법

```
select ename, dname, loc
from emp
inner join dept on emp.deptno=dept.deptno
```

join 할 값이 같다면

```
select ename, dname, loc
```

```
from emp
```

```
inner join dept using(deptno);
```

결과

```
ENAME      DNAME                          LOC
JONES      RESEARCH                     DALLAS
BLAKE      SALES                           CHICAGO
CLARK      ACCOUNTING               NEW YORK
KING       ACCOUNTING                 NEW YORK
FORD       RESEARCH                       DALLAS
```

**********가계부를 만들어 보면 좋다

self join

```
1  select e1.ename, e1.mgr, e2.empno, e2.ename
```

같은 목록 안에서 불러와야함

```
2  from emp e1, emp e2
```

같은 목록을 두번 부르기 위해 이름을 다르게 정해줌

```
3  where e1.mgr = e2.empno
```

같은 정보의 mgr값과 empno같다.

```
1  select e1.ename, e1.mgr, e2.empno, e2.ename
2  from emp e1, emp e2
3* where e1.mgr = e2.empno(+)
```

(+) 는 outer join 남는 값을 보여준다.

/=ansi join

```
1  select e1.ename, e1.mgr, e2.empno, e2.ename
2  from emp e1
3* left outer join emp e2 on e1.mgr=e2.empno
```

결과

```
ENAME             MGR      EMPNO ENAME

FORD             7566       7566 JONES
JAMES            7698       7698 BLAKE
TURNER           7698       7698 BLAKE
MARTIN           7698       7698 BLAKE
WARD             7698       7698 BLAKE
ALLEN            7698       7698 BLAKE
MILLER           7782       7782 CLARK
CLARK            7839       7839 KING
BLAKE            7839       7839 KING
JONES            7839       7839 KING
SMITH            7902       7902 FORD
KING
```

두가지 목록을 불러오기

```
select ename, dname, grade
2  from emp
3  join dept using(deptno)
4  join salgrade on sal between losal and hisal;
```

서브 쿼리

하나의 sql 문장절에 포함된 또 다른 select 문장, 따라서 두번 질의를 해야 얻을 수 있는 결과를 한번의 질의로 해결이 가능하게 하는 쿼리. 정렬을 두번해야할 때도 있고 구체적으로 정리해야 할 때 사용.

—main-query 또는 outer-query

—sub-query 또는 inner-query 두 쌍이 같은 의미

괄호로 반드시 묵어줘야함

서브쿼리는 메인쿼리의 다음 부분에 위치할 수 있음

select, delete, update 문의 from, where 절

select 문의 having 절

insert 문의 into 절

update 문의 set 절

단일행 서브 쿼리

```
SQL> select dname
2  from dept
3  where deptno = (select deptno from emp where ename='SMITH');
```

SMITH와 같은 deptno를 가지고 있는 정보

Allen과 같은 부서에서 근무하는 사원의 이름과 부서의 번호 출력

```
1  select ename,deptno
2  from emp
3* where deptno  =(select deptno from emp where ename='ALLEN')
```

ALLEN과 동일한 직책을 가진 사원의 사번과 이름 직책을 출력

```
1  select empno, ename, job
2  from emp
3* where job = (select job from emp where ename='ALLEN')
```

ALLEN의 급여와 동일하거나 더 많이 받는 사원의 이름과 급여를 출력

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

DALLAS에서 근무하는 사원의 이름 , 부서번호를 출력

```
SQL> select ename, deptno
2  from emp
3  where deptno=(select deptno from dept where='DALLAS');
```

SALES 부서에서 근무하는 모든 사원의 이름과 급여를 출력

```
SQL> select ename, sal
  2  from emp
  3  where deptno = (select deptno from dept where dname='SALES');
```

자신의 직속 상관이 KING인 사원의 이름과 급여를 출력

```
SQL> select ename, sal
  2  from emp
  3  where mgr=(select empno from emp where ename='KING');
```

다중행

값이 여러개인데 비교하고 싶으면 in을 사용하면 된다.

-급여를 3000 이상 받는 사원이 소속된 부서와 동일한 부서에서 근무하는 사원들의 이름과 급여 부서번호를 출력

```
1  select ename, sal,deptno
2  from emp
3* where deptno in(select deptno from emp where sal>=3000)
```

-IN연산자를 이용하여 부서별로 가장 급여를 많이 받는 사원의 사원번호, 급여, 부서번호를 출력해 보세요

```
1  select empno,sal,deptno,ename
2  from emp
3* where sal in(select  max(sal) from emp group by deptno)
```

-직책이 MANAGER인 사원이 속한 부서의 부서번호와 부서명과 부서의 위치 출력

```
SQL> select deptno, dname, loc
2  from dept
3  where deptno in(select deptno from emp where job='MANAGER');
```

-30번 부서의 사원들 중에서 급여를 가장 많이 받는 사원보다 더 많은 급여를 받는 사원의 이름과 급여 출력

단일행

```
SQL> select ename, sal
2  from emp
3  where sal>(select max(sal) from emp where  deptno=30);
```

다중행

```
SQL> select ename,sal
2  from emp
3  where sal> all(select sal from emp where deptno=30);
```

-직책이 'SALESMAN' 보다 급여를 많이 받는 사원들의 이름과 급여를 출력 (ANY 연산자 이용)

```
SQL> select ename, sal
2  from emp
3  where sal > any(select sal from emp where job='SALESMAN');
```

-부서번호가 30번인 사원들의 급여중 최저 급여보다 높은 급여를 받는 사원의 이름, 급여를 출력

```
단일행
  SQL> select ename,sal
  2  from emp
  3  where sal> (select min(sal) from emp where deptno=30);
복수행
  SQL> select ename, sal
  2  from emp
  3  where sal> any(select sal from emp where deptno=30);
```

-직책이 'SALESMAN' 사원의 최소 급여보다 많이 받는 사원들의 이름과 급여, 직책을 출력하되 'SALESMAN' 은 출력안함 (ANY 연산자를 사용)

```
SQL> select ename, sal, job
2  from emp
3  where sal > any(select sal from emp where job='SALESMAN')
4  and job!='SALESMAN';
```

-SMITH 와 동일한 직책을 가진 사원의 이름과 직책을 출력

```
SQL> select ename, job
2  from emp
3  where job =(select job from emp where ename='SMITH');
```

-직책이 'SALESMAN' 인 사원이 받는 급여들의 최대 급여보다 많이 받는 사원들의 이름과 급여를 출력하되 부서번호가 20번인 사원은 제외 (ALL 연산자 이용)

```
1  select ename, sal
2  from emp
3  where sal>all(select sal from emp where job='SALESMAN')
4* and deptno !=20
```

-직책이 'SALESMAN' 인 사원이 받는 급여들의 최소 급여보다 많이 받는 사원들의 이름과 급여를 출력하되 부서번호가 20번인 사원은 제외(ANY 연산자 이용)

```
SQL> select ename, sal
2  from emp
3  where sal > any(select min(sal) from emp where job='SALESMAN')
4  and deptno !=20;
```

특정 row의 값을 다른 값들과 함께 출력하려면 (이전글 - 본문 - 다음글)

```
1  select empno, ename, sal,
2  lead(sal,1,0)over (order by sal asc)as lead_sal,
3  lag(sal,1,0) over (order by sal asc)as lag_sal
4  from emp
5* order by sal asc
```

```
EMPNO ENAME             SAL   LEAD_SAL    LAG_SAL
 7369 SMITH             800        950          0
 7900 JAMES             950       1250        800
 7521 WARD             1250       1250        950
 7654 MARTIN           1250       1300       1250
 7934 MILLER           1300       1500       1250
 7844 TURNER           1500       1600       1300
 7499 ALLEN            1600       2450       1500
 7782 CLARK            2450       2850       1600
 7698 BLAKE            2850       2975       2450
 7566 JONES            2975       3000       2850
 7902 FORD             3000       5000       2975
 7839 KING             5000                  3000
```

oracle은 웹서버에서도 사용할 수 있다.

localhost:8080/apex 쉽게 작업 가능

DML(Data Multipulatio Language)

테이블 내의 데이터를 입력 수정 삭제

- insert into 테이블명(칼럼명1, 칼럼명2,…..)

  values(값1, 값2,….)

  한번에 하나의 행만 입력 할 수 있다.

  insert 절에 명시되는 칼럼의 갯수와 value 절의 갯수는 일치해야 한다.

  모든 칼럼의 내용을 다 저장할때는 칼럼명은 생략 가능하다.
- update 테이블명 set 칼럼명1 = 수정값, 칼럼명2 = 수정값 where whrjswjf
- delete from 테이블명 where 조건절

insert update delete 는 commit 하기 전까지는 임시반영 rollback 으로 복구 할 수 있다.

create table 테이블명(값1,,,값2,,,)

모든 테이블에 값을 입력할때는 그냥 작성해도 된다.

Oracle 데이터베이스

여러 clients가 접속할 수 있게 만듦. 동일한 계정으로 접속할 수 도 있음.

많은 clients가 동일한 계정으로 동일한 table을 수정 한다면?

먼저 수정을 시작한 사람이 같은 내용을 commit 하기 전까지 다른 사용자는 아무것도 수정할 수 없다 locked 먼저 사용자가 commit 하는 순간 갱신된다.

+++++++COMMIT++++++

emp table의 deptno는 dept table 의 deptno를 참조(reference) 중인 것

```
SQL>  create table dept2
  2  (deptno number(2) primary key,
  3  dname varchar2(14),
  4  loc varchar2(13));


SQL> create table emp2
  2  (empno number(4) primary key,
  3  ename varchar2(12),
  4  job varchar2(12),
  5  deptno number(2) references dept2(deptno));
```

안에 정보를 입력하기

```
SQL> insert into dept2
  2  values(10, 'JAZZ','YANGJU');
dept2에 먼저 입력한 뒤

SQL> insert into emp2
  2  (empno, ename, job, deptno)
  3  values(1000, '재즈','player',10);
emp2에 참조
```

desc user_tables 테이블 조회

user_sequences 시퀀스 조회

user_constraints 제약 조건을 조회

R - reference

C - not Null

P - primary key

제약조건의 이름을 지어줄 수 있다.

1. member_num_pk

   멤버테이블 제약조건 넘 프라이머리키

2. member_name_nn

   member 테이블 name 제약조건

3. emp2_empno_pk
4. emp2_deptno_fk

제약 조건의 이름 부여하기

```
create table member(
num number constraint member_num_pk primary key,
name varchar2(12) constraint member_name_nn not null,
addr varchar2(16));
```

varchar 가변

char 고정

unique 중복허용x

```
SQL> create table test(
  2  num number constraint test_num_pk primary key,
  3  name char(10) constraint test_name_u unique,
  4  deptno number(2) constraint test_deptno_fk references dept(deptno));
```

테이블 수준에서도 제약조건을 지정할 수 있음

```
create table test2(
num number,
name char(10),
deptno number(2),
constraint test2_num_pk primary key(num),
constraint test2_name_u unique(name),
constraint test2_deptno_fk foreign key(deptno) references dept(deptno))
```

테이블 삭제(drop)

테이블 생성 (create) - row를 수정할 때는 update

```
create table test(num number);
```

테이블 변경(alter)

```
alter table test add(name varchar2(12));
alter table test add(addr varchar2(20));
```

이미 만들어진 칼럼을 수정(modify)

```
alter table test modify(name varchar2(21));
```

칼럼 명 바꾸기(rename column XXXX to yyyy)

```
alter table test rename column name to myname;
```

칼럼 삭제(drop)

```
alter table test drop(addr);
```

제약조건 추가하기(add(constraint))-table level constraint

```
alter table test add(constraint test_num_pk primary key(num));
alter table test add(constraint test_myname_u unique(myname));
```