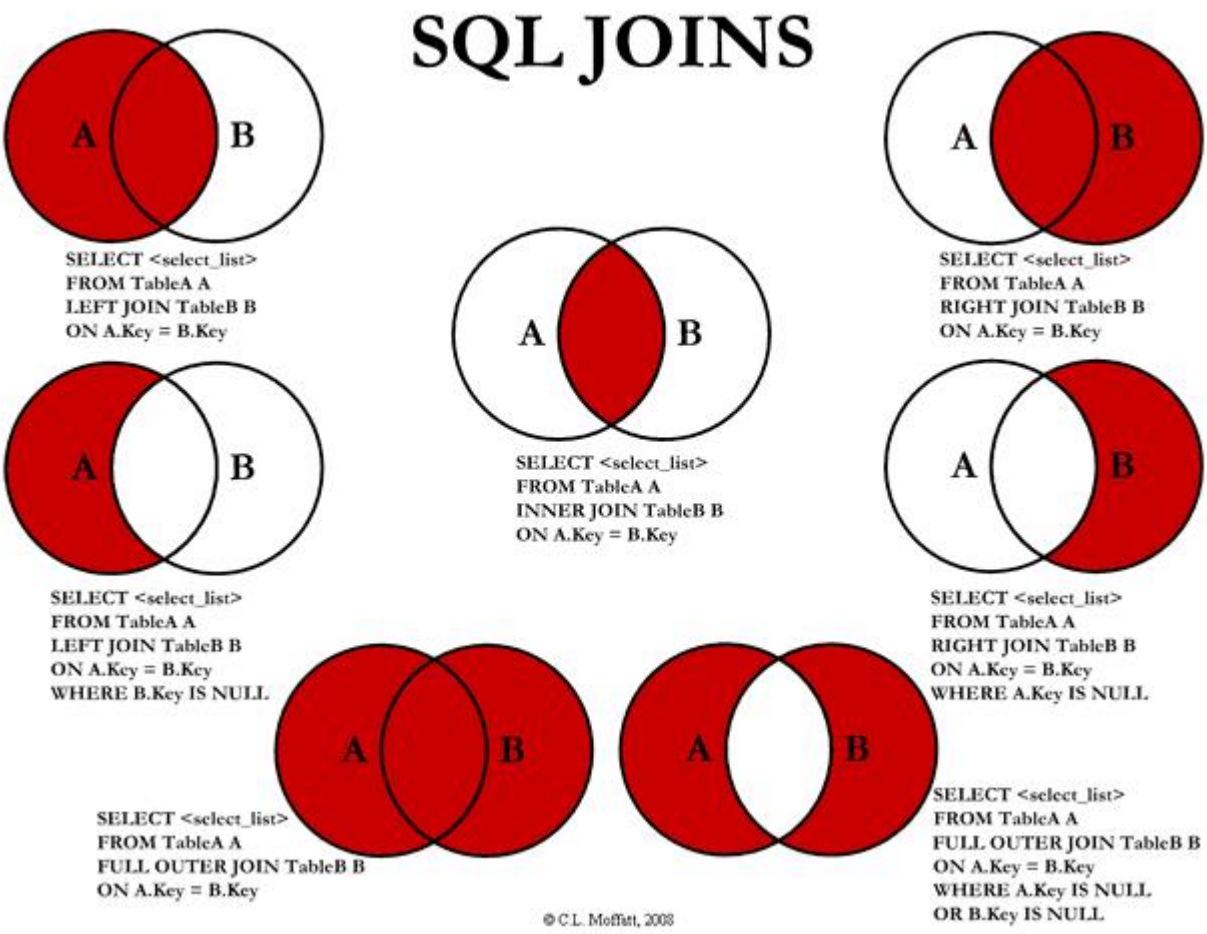
데이터베이스

JOIN



두 개 이상의 테이블을 서로 묶어서 하나의 결과 집합으로 만들어 내는 것
종류 : INNER JOIN, OUTER JOIN, CROSS JOIN, SELF JOIN

<기본키와 외래키 성질의 이해>
1. 기본키(Primary Key : PK) : 공백(Null)이 올 수 없다, 중복될 수 없다.
2. 외래키(Foreign Key : FK) : 기본키를 참조, 기본키에 있는 값만 사용해야 함, 중복될 수 있다.

테이블을 만들때에 데이터 최소화 시켜서 테이블을 만듬
중복과 공간 낭비를 피하고 데이터의 무결성을 위해서 여러개의 테이블로 분리하여 저장
분리된 테이블들은 서로 관계를 가짐
1대 다 관계가 보편적

테이블이 개체(Entity)
E-R 모델
RDBMS 관계형 데이터베이스

JOIN 할 때 기준점을 잡아야 함
JOIN = INNER JOIN

<INNER JOIN 연습하기>
SELECT * from usertbl;
select * from buytbl;
select * from buytbl
inner join usertbl
on buytbl.userID = usertbl.userID;
userID를 공통으로 잡음 중복되는 값이 많음

select userID, prodName, addr, mobile1, mobile2 from buytbl
inner join usertbl
on buytbl.userID = usertbl.userID;
=> 에러 걸림(buytbl userID인지 usertbl userID인지 모르기때문)

select buytbl.userID, buytbl.prodName, usertbl.addr, usertbl.mobile1, usertbl.mobile2 from buytbl
inner join usertbl
on buytbl.userID = usertbl.userID;
=> 공통된 필드 userID 기준으로, 필요한 것만 나옴

as A => A라는 별명만들기

select B.userID, B.prodName, U.addr, U.mobile1, U.mobile2 from buytbl as B
inner join usertbl as U
on B.userID = U.userID;

INNER JOIN : 교집합(공통된 필드가 있어야 함, 공통된 필드 기준으로 조인)
일반적으로는 왼쪽 있는 내용을 기준으로 함
join으로 데이터를 추적해서 찾아야 함.

기본키는 1개 외래키는 여러개여도 상관 없음.

OUTER JOIN : 오라클에서는 있음 SQL에서는 LEFT OUTER JOIN, RIGHT OUTER JOIN으로 함.
LEFT OUTER JOIN
RIGHT OUTER JOIN

select * from usertbl;
select U.userID, U.name, B.prodName, U.addr from usertbl as U
left outer join buytbl as B

```
on u.userID = B.userID;

select * from usertbl;
select U.userID, U.name, B.prodName, U.addr from usertbl as U
left outer join buytbl as B
on u.userID = B.userID
where B.prodName = '';
-> 아무것도 안 나옴

select * from usertbl;
select U.userID, U.name, B.prodName, U.addr from usertbl as U
left outer join buytbl as B
on u.userID = B.userID
where B.prodName is null;
-> prodName이 Null인 것을 찾아줌(한 번이라도 구매하지 않은 사람)

select * from usertbl;
select U.userID, U.name, B.prodName, U.addr from usertbl as U
left outer join buytbl as B
on u.userID = B.userID
where B.prodName is not null;
-> prodName Null이 아닌 것을 찾아줌(한 번이라도 구매한 사람)

<기본키와 외래키 만들기>
create table stdtbl
( stdName varchar(10) not null primary key,
addr char(4) not null
);
create table clubtbl
( clubName varchar(10) not null primary key,
roomNo char(4) not null
);
create table stdclubtbl
( num int auto_increment not null primary key,
stdName varchar(10) not null,
clubName varchar(10) not null,
foreign key(stdName) references stdtbl(stdName),
foreign key(clubName) references clubtbl(clubName)
);

references : 참조
```

```
insert into stdtbl values ('김범수', '경남'),('성시경', '서울'), ('조용필', '경기'), ('은지원', '경북'), ('바비킴', '서울');
insert into clubtbl values ('수영', '101호'),('바둑', '102호'), ('축구', '103호'), ('봉사', '104호');
insert into stdclubtbl values
(null, '김범수', '바둑'), (null,'김범수', '축구'), (null,'조용필', '축구'),
(null,'은지원', '축구'), (null,'은지원', '봉사'), (null,'바비킴', '봉사') ;


select * from stdclubtbl;
select * from stdtbl;
select * from clubtbl;

-- [Q1] 김범수가 가입한 동아리 이름은??
select stdName, clubName from stdclubtbl where stdName = '김범수';

-- [Q2] 학생의 이름, 지역, 가입한 동아리를 출력하는 쿼리문을 작성하세요.

select S.stdName, SC.clubName, S.addr from stdtbl as S
inner join stdclubtbl as SC
on S.stdName = SC.stdName;
-- [Q3] 학생이름, 동아리, 동아리방 번호를 출력하는 쿼리문을 작성하세요.
select * from stdclubtbl;
select * from stdtbl;
select * from clubtbl;
select SC.stdName, SC.clubName, C.roomNo from stdclubtbl as SC
inner join clubtbl as C
on SC.clubName = C.clubName;

-- [Q4] 학생이름, 주소, 동아리명, 방번호를 출력하는 쿼리문을 작성하세요.
select S.stdName, S.addr, SC.clubName, C.roomNo from stdtbl S
inner join stdclubtbl SC
on S.stdName = SC.stdName
inner join clubtbl C
on SC.clubName = C.clubName;

as 생략 해도 됨

outer join : join하여서 제외된 부분까지 나타낼 때 사용
outer join의 활용
Select U.userID, U.name, B.prodName, U.addr from usertbl as U
```

```
left outer join buytbl as B
on u.userID = B.userID;

cross join
예제 데이터 만들 때 사용 빈번하게 쓰는 것은 아님

self join
-- ------------------------------------------------
create table empTbl
( emp char(3), manager char(3), empTel varchar(8) );
insert into empTbl values('나사장', null, '0000');
insert into empTbl values('김재무', '나사장', '2222');
insert into empTbl values('김부장', '김재무', '2222-1');
insert into empTbl values('이부장', '김재무', '2222-2');
insert into empTbl values('우대리', '이부장', '2222-2-1');
insert into empTbl values('지사원', '이부장', '2222-2-2');
insert into empTbl values('이영업', '나사장', '1111');
insert into empTbl values('한과장', '이영업', '1111-1');
insert into empTbl values('최정보', '나사장', '3333');
insert into empTbl values('윤차장', '최정보', '3333-1');
insert into empTbl values('이주임', '윤차장', '3333-1-1');

-- 이부장의 상사의 구내 전화번호는?
select * from empTbl where emp = '이부장';
select emp, empTel from empTbl where emp = '김재무';

select A.emp as '부하직원', B.emp as '직속상관', B.empTel as '직속상관연락처' from empTbl A
inner join empTbl B
on A.manager = B.emp
where A.emp = '이부장';

조직도 같은 때 self join 사용
inner join으로 사용

select A.emp as '부하직원', B.emp as '직속상관', B.empTel as '직속상관연락처' from empTbl A
join empTbl B
on A.manager = B.emp
where A.emp = '이부장';
```

-- union, union all, not in, in
<union>
union은 합집합, **두 테이블의 필드의 개수가 같아야 한다**.
두 필드의 자료형이 호환되어야 한다.

```
select * from stdtbl
union select * from clubtbl;

select stdName, addr from stdtbl
union select stdName, clubName from stdclubtbl;

select stdName, addr from stdtbl
union select num, clubName from stdclubtbl;
```

<union all>
```
select * from stdtbl union all select * from clubtbl;
```
합쳐서 있는 것을 모두 보여줌.(중복값 포함)

<union과 union all의 차이점>
union은 중복값 제외하여 합쳐서 보여줌

<not in과 in의 차이>
```
select name, concat(mobile1, mobile2) as '전화번호' from usertbl
where name not in (select name from usertbl where mobile1 is null);
```
concat : 문자열을 결합시켜줌
concat (문자명 or 필드명) 넣어주면 됨

```
select concat(mobile1, '-', left(mobile2,4), '-', right(mobile2,4)) as '전화번호'
from usertbl;
```
=> 010-0000-0000 형태로 나옴

left(문자열, 자릿수)
right(문자열, 자릿수)

not in : 전체결과에서 중복되는 부분을 제외(서브쿼리 제외)한 나머지 결과를 낼 때 사용
서브쿼리 : 하나의 SQL 문에 포함되어 있는 또 다른 SQL 문을 말합니다.
```
select name from usertbl where mobile1 is null;
```

in : 전체에서 결과 값에 서브쿼리 조건만 나타냄
※ 서브쿼리 자주 사용함(있는 값에서 또 값을 찾아내기 위해서 사용)