Beyond Promise

SQL 학습 소모임 2강

About Join !

신기술 개발팀

김 경 남

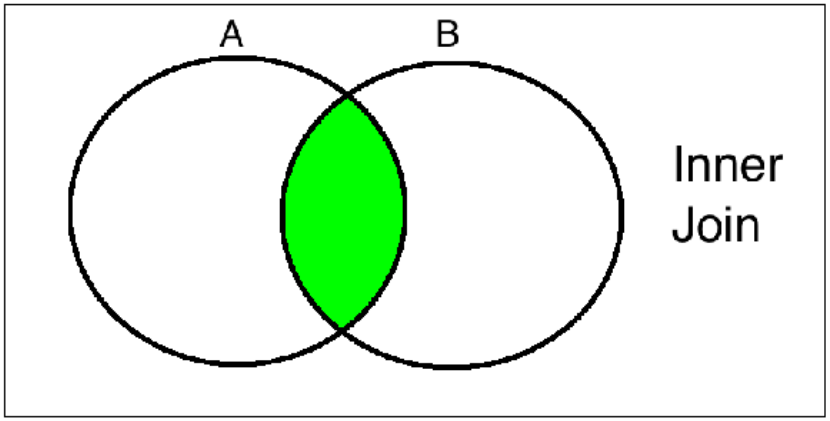
1. Join 의 개념(1)

서로 관계 있는 데이터 집합에서 주어진 조건을 이용하여 데이터를 추출하는 것.

* 관계 : 하나의 데이터집합에서 다른 데이터 집합을 찾아갈 수 있는 방법 (예: Foregin key 관계)

* 데이터 집합 : 데이터가 존재하는 단위. (예: 테이블, 중간 연산 과정의 결과 데이터)

*조건 : 데이터를 추출하기 위한 연산 조건 (예: =)



1. Join 의 개념 (2)

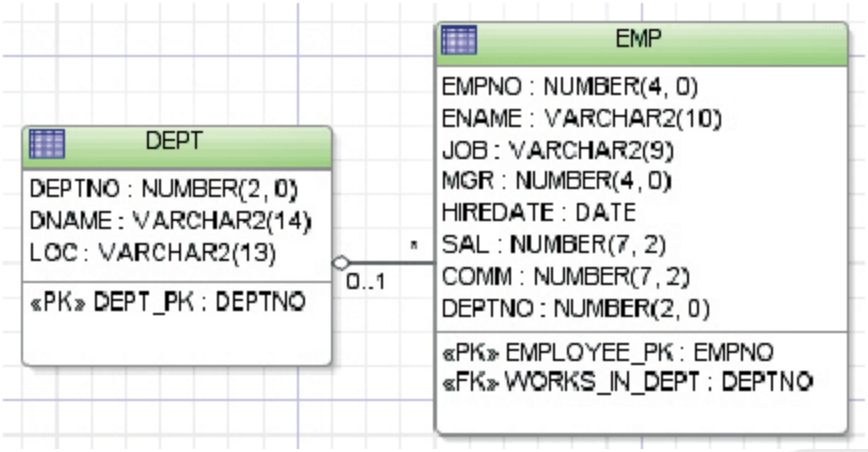
서로 관계 있는 데이터 집합에서 주어진 조건을 이용하여 데이터를 추출해보자

- select e.empno,e.ename,d.dname
  from emp e join dept d on e.deptno = d.deptno

ANSI SQL

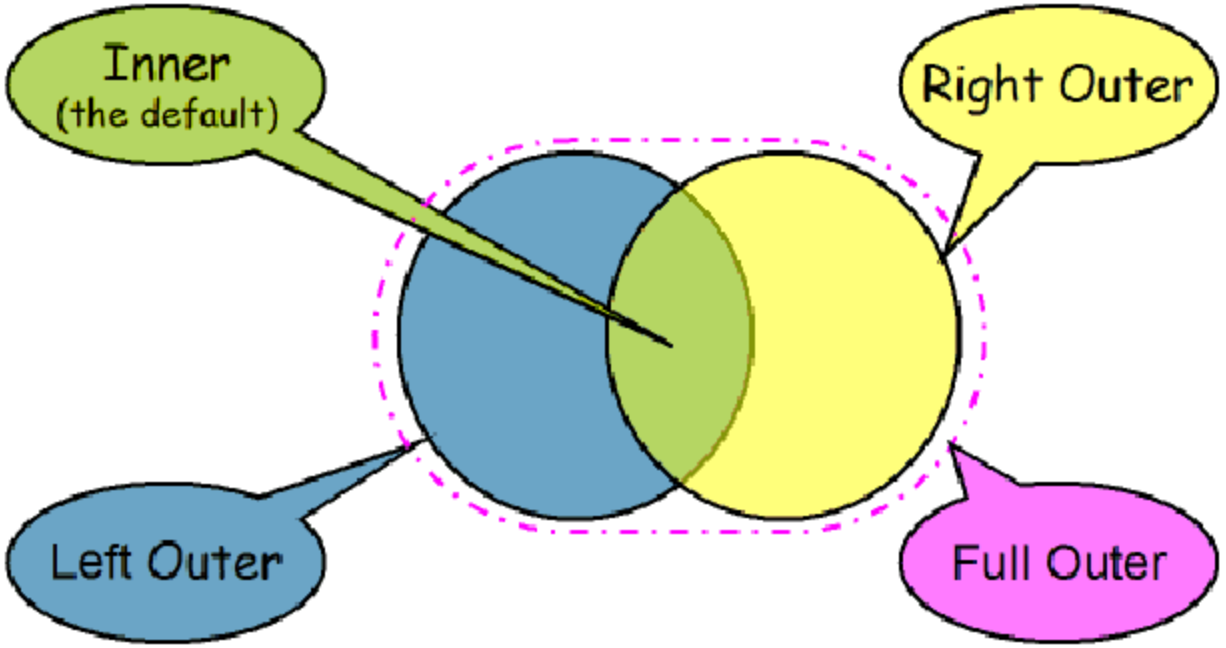
- select e.empno,e.ename,d.dname
  from emp e ,dept d
  where e.deptno = d.deptno

- 관계 : emp 테이블의 deptno 와 dept 테이블의 deptno
- 데이터 집합 : emp, dept
- 조건 : e.deptno = d.deptno



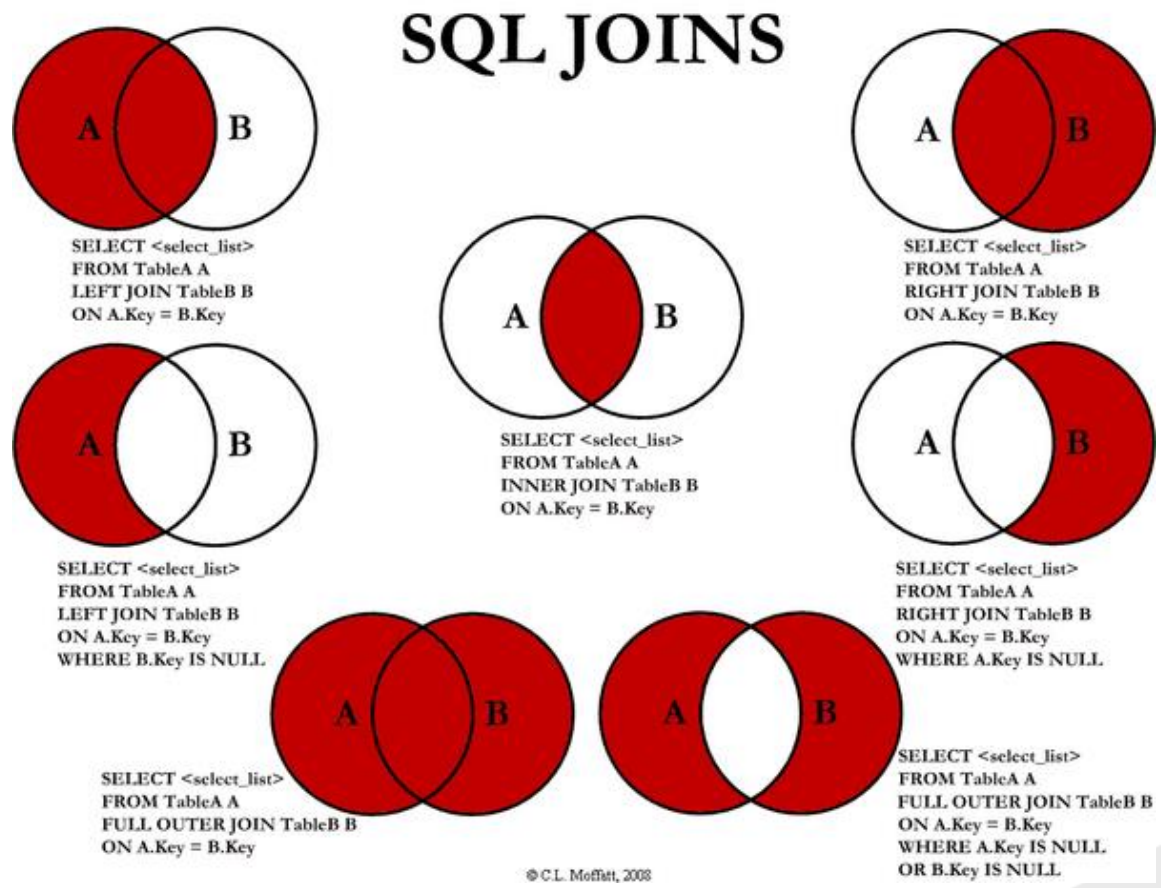
2. Join 종류 (1)

Join 의 종류에는 크게 inner join, outer join 이 존재한다.



2. Join 종류 (2)

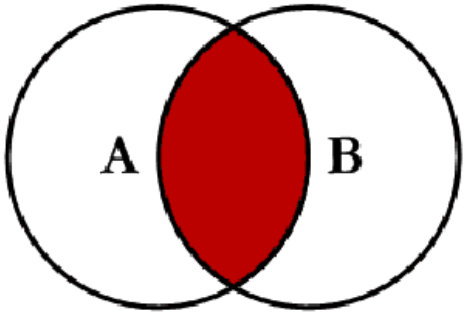
조금 더 상세화 하면...



3. Inner Join (1)

서로 다른 데이터 집합에서 공통되는 부분을 추출하는 Join

Inner JOIN



- SELECT <select_list>
  FROM Table_A A INNER JOIN Table_B B
  ON A.Key = B.Key

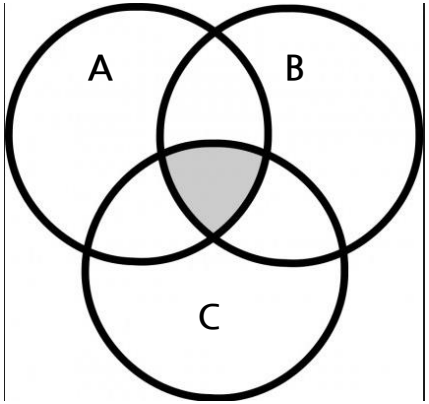
* inner 키워드는 주로 생략.

- SELECT <select_list>
  FROM Table_A A ,Table_B B
  Where A.Key = B.Key

- 위의 두가지 형식으로 전부 사용 가능한다. 위의 sql 문이 ANSI SQL (SQL:2011) 문임.
- Oracle 사용자들은 일반적으로 아래 sql 문 형식에 더 익숙함.
- 두가지 방식을 모두 알아둘 필요가 있음.

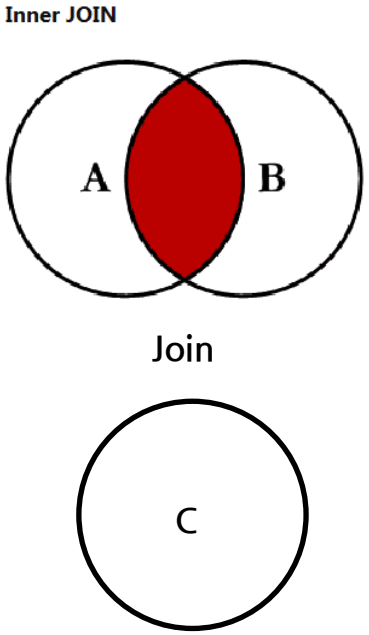
3. Inner Join (2)

조인은 꼭 table 간에만 이루어지는 것은 아니다.



- SELECT <select_list>
  FROM Table_A A JOIN Table_B B
  ON A.Key = B.Key
  JOIN Table_C C
  ON B.Key = C.Key

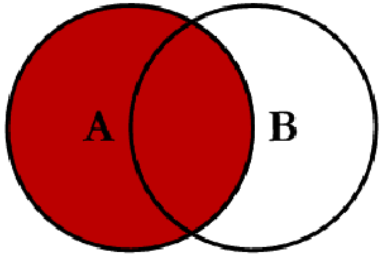
- SELECT <select_list>
  FROM Table_A A ,Table_B B, Table_C
  Where A.Key = B.Key
  and B.Key = C.Key



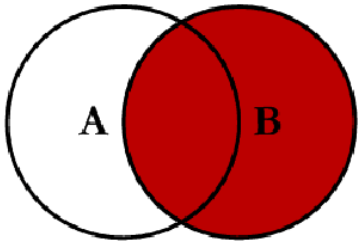
• SQL 은 집합 기반의 언어이다. 위의 예에서 이해해야 하는 것은 Table_A와 Table_B 가 join 조건에 의해서 조인한 후 생기는 중간 데이터 집합과 Table_C 가 다시 join 을 한다는 것이다. Join 의 정의를 설명하면서 “테이블” 이라고 하지 않고 “데이터 집합” 이라고 한것은 이러한 이유에 기인한다.

4. Outer Join (1)

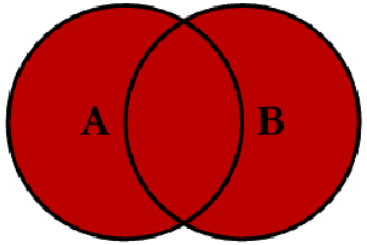
서로 다른 데이터 집합 사이에서 공통되는 부분 이외에도 어느 데이터 집합의 데이터는 모두 추출하고자 할 때 사용.



Left outer



Right outer



Full outer

- SELECT <select_list>
  FROM Table_A A LEFT OUTER JOIN Table_B B
  ON A.Key = B.Key

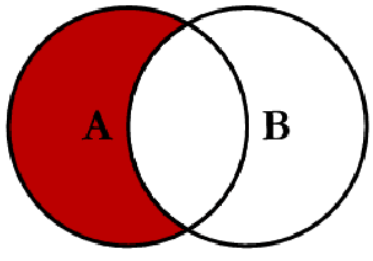
- SELECT <select_list>
  FROM Table_A A ,Table_B B
  Where A.Key = B.Key (+)

- SELECT <select_list>
  FROM Table_A A ,Table_B B
  Where A.Key *= B.Key

• 위의 두가지 형식으로 전부 사용 가능한다. 위의 sql 문이 ANSI SQL (SQL:2011) 문임.
• DBMS 벤더마다 outer join 은 자신들만의 특화된 문법을 사용하기도 함.
• 기준테이블의 데이터는 모두 가져 옴
• 위의 쿼리에서 기준테이블 : Table_A
• 위의 쿼리에서 조인테이블 : Table_B

4. Outer Join (2)

조인 조건을 만족하지 못하는 데이터는 조인 컬럼에 대해서 null 을 반환한다.



차집합 구하기

- SELECT <select_list>
  FROM Table_A A LEFT OUTER JOIN Table_B B
  ON A.Key = B.Key
  WHERE ???

tableA

aid	aname
1	aaa1
2	aaa2
3	aaa3
4	aaa4
5	aaa5

tableB

bid	bname
1	bbb1
2	bbb2
3	bbb3
8	bbb8
9	bbb9

```
select *
from tableA a left outer join tableB b
on a.aid = b.bid
```

aid	aname	bid	bname
1	aaa1	1	bbb1
2	aaa2	2	bbb2
3	aaa3	3	bbb3
4	aaa4	(NULL)	(NULL)
5	aaa5	(NULL)	(NULL)

```
select *
from tableA a left outer join tableB b
on a.aid = b.bid
where b.bid is null
```

aid	aname	bid	bname
4	aaa4	(NULL)	(NULL)
5	aaa5	(NULL)	(NULL)

4. Outer Join (3)

ANSI 표준 SQL 을 사용할 때 Outer join 인 경우 Where 절과 on 절에 오는 조건절의 사용을 주의해야 한다. Inner join은 관계 없다.

tableA

aid	aname
1	aaa1
2	aaa2
3	aaa3
4	aaa4
5	aaa5

tableB

bid	bname
1	bbb1
2	bbb2
3	bbb3
8	bbb8
9	bbb9

1)

```
select *
from tableA a left outer join tableB b
on a.aid = b.bid
where b.bid = 2
```

aid	aname	bid	bname
2	aaa2	2	bbb2

2)

```
select *
from tableA a join tableB b
on a.aid = b.bid
where b.bid = 2
```

aid	aname	bid	bname
2	aaa2	2	bbb2

- ✓ 1)번 쿼리와 2)번 쿼리는 모두 에러 없이 실행되고 결과 집합을 리턴한다.
- ✓ 1) 번 쿼리의 결과는 원래 원하던 의도의 쿼리가 아니다. 1) 번 쿼리는 결과적으로 비용이 많이 드는 outer join 을 사용할 필요가 없다. 또는 결과 집합이 원하는 결과가 아닌데도 모르고 넘어가는 경우일 수도 있다.
- ✓ 후자의 경우는 결국 논리적 오류이며 나중에 이상한 데이터가 생기는 원인이 된다.

4. Outer Join (4)

Join 테이블의 조건은 on 절에 기술한다. Where 절에 기술하면 outer join 이 성립하지 않는다.

tableA

aid	aname
1	aaa1
2	aaa2
3	aaa3
4	aaa4
5	aaa5

tableB

bid	bname
1	bbb1
2	bbb2
3	bbb3
8	bbb8
9	bbb9

```
select *
from tableA a left outer join tableB b
on a.aid = b.bid and b.bid = 2
```

aid	aname	bid	bname
1	aaa1	(NULL)	(NULL)
2	aaa2	2	bbb2
3	aaa3	(NULL)	(NULL)
4	aaa4	(NULL)	(NULL)
5	aaa5	(NULL)	(NULL)

4. Outer Join (5)

기준 테이블에 오는 조건절의 위치는 ?

tableA

aid	aname
1	aaa1
2	aaa2
3	aaa3
4	aaa4
5	aaa5

tableB

bid	bname
1	bbb1
2	bbb2
3	bbb3
8	bbb8
9	bbb9

```
select *
from tableA a left outer join tableB b
on a.aid = b.bid  and a.aid = 2
```

```
select *
from tableA a left outer join tableB b
on a.aid = b.bid
where  a.aid = 2
```

aid	aname	bid	bname
2	aaa2	2	bbb2

?

aid	aname	bid	bname
1	aaa1	(NULL)	(NULL)
2	aaa2	2	bbb2
3	aaa3	(NULL)	(NULL)
4	aaa4	(NULL)	(NULL)
5	aaa5	(NULL)	(NULL)

4. Outer Join (6)

전체 최종 Row 에 영향을 주는 기준테이블에 대한 조건은 Where 절에 기술

tableA

aid	aname
1	aaa1
2	aaa2
3	aaa3
4	aaa4
5	aaa5

tableB

bid	bname
1	bbb1
2	bbb2
3	bbb3
8	bbb8
9	bbb9

```
select *
from tableA a left outer join tableB b
on a.aid = b.bid
where  a.aid = 2
```

aid	aname	bid	bname
2	aaa2	2	bbb2

```
select * from dept;
```

	DEPTNO	DNAME	LOC
1	10	ACCOUNTING	NEW YORK
2	20	RESEARCH	DALLAS
3	30	SALES	CHICAGO
4	40	OPERATIONS	BOSTON

```
select * from dept_s1;
```

	DEPTNO	PROF
1	10	a10
2	20	a20
3	40	a40

Quiz

1)

```
select a.deptno,a.dname,b.prof
from dept a
left outer join dept_s1 b on a.deptno = b.deptno
where  a.deptno > 30
order by 1;
```

	DEPTNO	DNAME	PROF
1	40	OPERATIONS	a40

2)

```
select a.deptno,a.dname,b.prof
from dept a
left outer join dept_s1 b on a.deptno = b.deptno and a.deptno > 30
order by 1;
```

	DEPTNO	DNAME	PROF
1	10	ACCOUNTING	
2	20	RESEARCH	
3	30	SALES	
4	40	OPERATIONS	a40

3)

```
select a.deptno,a.dname,b.prof
from dept a
left outer join dept_s1 b on a.deptno = b.deptno and b.deptno > 10
order by 1;
```

	DEPTNO	DNAME	PROF
1	10	ACCOUNTING	
2	20	RESEARCH	a20
3	30	SALES	
4	40	OPERATIONS	a40

4)

```
select a.deptno,a.dname,b.prof
from dept a
left outer join dept_s1 b on a.deptno = b.deptno
where b.deptno > 10
order by 1;
```

	DEPTNO	DNAME	PROF
1	20	RESEARCH	a20
2	40	OPERATIONS	a40

4. Outer Join (7) - 종합

1. Outer join 의 종류
   - outer join 종류 : left outer join, right outer join, full outer join
     - ✓ full outer join 은 가능한 사용하지 않음.
   - DBMS 변경에 따른 영향을 최소화할 수 있도록 기존의 "(+)" 형태의 오라클 종속적인 Syntax를 지양하고, ANSI 표준으로 작성 권고.

```
select * from dept;
```

	DEPTNO	DNAME	LOC
1	10	ACCOUNTING	NEW YORK
2	20	RESEARCH	DALLAS
3	30	SALES	CHICAGO
4	40	OPERATIONS	BOSTON

```
select * from dept_s1;
```

	DEPTNO	PROF
1	10	a10
2	20	a20
3	40	a40

예시1) dept 테이블을 기준으로 dept_s1 테이블의 데이터를 조인

<기존 방식>

```
select a.deptno, a.dname,b.prof
 from dept a, dept_s1 b
where a.deptno = b.deptno(+);
```

<표준 방식>

```
select a.deptno, a.dname,b.prof
from dept a left outer  dept_s1 b
        on a.deptno = b.deptno
```

```
select a.deptno,a.dname,b.prof
from dept a
left outer join dept_s1 b on a.deptno = b.deptno
order by 1;
```

	DEPTNO	DNAME	PROF
1	10	ACCOUNTING	a10
2	20	RESEARCH	a20
3	30	SALES	
4	40	OPERATIONS	a40

2. 주의 사항

- 예시2 : 기준 테이블의 조건 위치
  - ➢ 전체 최종 Row 에 영향을 주는 기준 테이블(여기서는 dept 테이블)에 대한 조건은 Where 절에 기술
  - ➢ On 절에 기술되는 조건은 조인에만 영향을 주므로 전체 Row 에 영향을 끼치지 않음.
- 예시3 : 조인 테이블의 조건 위치
  - ➢ 기준 테이블과 조인되는 테이블(여기서는 dept_s1 테이블)의 조건은 on 절에 기술. Where 절에 기술할 경우 outer 조인이 성립되지 않음

<예시2> dept 테이블 조건절 위치에 따른 결과 셋

Case1) 정상 사용

```
select a.deptno,a.dname,b.prof
from dept a
left outer join dept_s1 b on a.deptno = b.deptno
where  a.deptno > 30
order by 1;
```

	DEPTNO	DNAME	PROF
1	40	OPERATIONS	a40

Case2) 잘못된 사용

```
select a.deptno,a.dname,b.prof
from dept a
left outer join dept_s1 b on a.deptno = b.deptno and a.deptno > 30
order by 1;
```

	DEPTNO	DNAME	PROF
1	10	ACCOUNTING	
2	20	RESEARCH	
3	30	SALES	
4	40	OPERATIONS	a40

<예시3> dept_s1 테이블의 조건절 위치에 따른 결과 셋

Case1) 정상 사용

```
select a.deptno,a.dname,b.prof
from dept a
left outer join dept_s1 b on a.deptno = b.deptno and b.deptno > 10
order by 1;
```

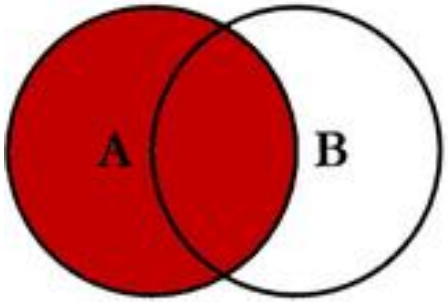
	DEPTNO	DNAME	PROF
1	10	ACCOUNTING	
2	20	RESEARCH	a20
3	30	SALES	
4	40	OPERATIONS	a40

Case2) 잘못된 사용

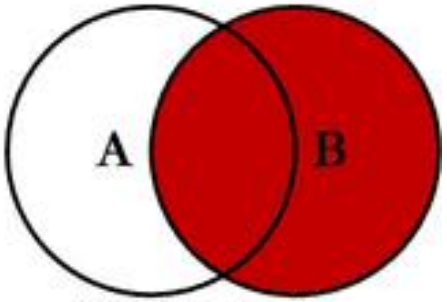
```
select a.deptno,a.dname,b.prof
from dept a
left outer join dept_s1 b on a.deptno = b.deptno
where b.deptno > 10
order by 1;
```

	DEPTNO	DNAME	PROF
1	20	RESEARCH	a20
2	40	OPERATIONS	a40

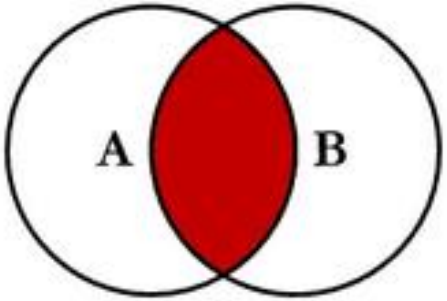
SQL JOINS



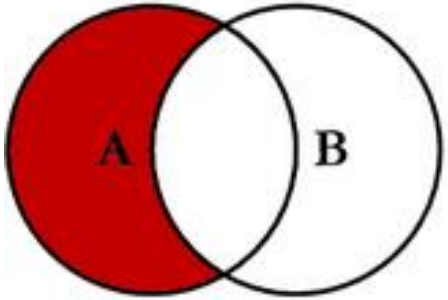

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



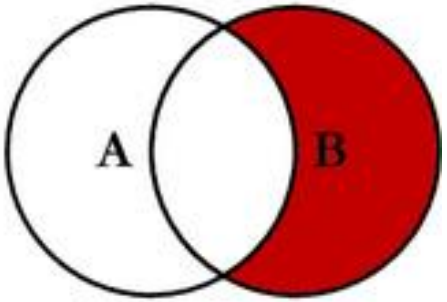

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



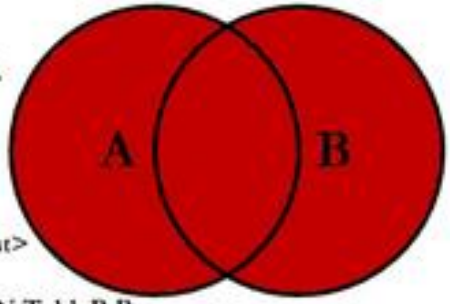

```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



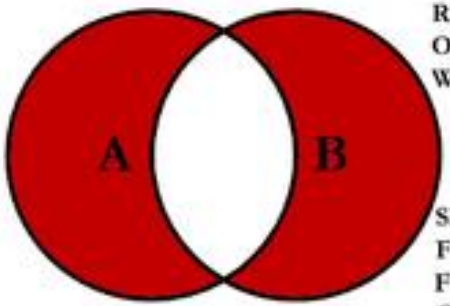

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```