

JOINS

Joined Tables

- A joined table is a table derived from two other (real or derived) tables according to the rules of the particular join type.
- Inner, outer, and cross-joins are available.
- The general syntax of a joined table is:

`T1 join_type T2 [join_condition]`

Joined Tables

- Parentheses can be used around JOIN clauses to control the join order.
- In the absence of parentheses, JOIN clauses nest left-to-right.

CROSS JOIN

T1 CROSS JOIN T2

- For every possible combination of rows from T1 and T2 (i.e., a Cartesian product), the joined table will contain a row consisting of all columns in T1 followed by all columns in T2.
- If the tables have N and M rows respectively, the joined table will have $N * M$ rows.

CROSS JOIN

- FROM T1 CROSS JOIN T2 is equivalent to
- FROM T1 INNER JOIN T2 ON TRUE
- It is also equivalent to FROM T1, T2

CROSS JOIN

- `SELECT T1.ID, T2.ID FROM T1 CROSS JOIN T2`
- `SELECT T1.ID, T2.ID FROM T1 INNER JOIN T2
ON TRUE`
- `SELECT T1.ID, T2.ID FROM T1, T2`

num		name
-----+		
1		a
2		b
3		c

num		value
-----+		
1		xxx
3		yyy
5		zzz

```
SELECT * FROM t1 CROSS JOIN t2;
```

num	name	num	value
1	a	1	xxx
1	a	3	yyy
1	a	5	zzz
2	b	1	xxx
2	b	3	yyy
2	b	5	zzz
3	c	1	xxx
3	c	3	yyy
3	c	5	zzz

Qualified joins

- `T1 { [INNER] | { LEFT | RIGHT | FULL } [OUTER] } JOIN T2 ON boolean_expression`
- `T1 { [INNER] | { LEFT | RIGHT | FULL } [OUTER] } JOIN T2 USING (join column list)`
- `T1 NATURAL { [INNER] | { LEFT | RIGHT | FULL } [OUTER] } JOIN T2`

Qualified joins

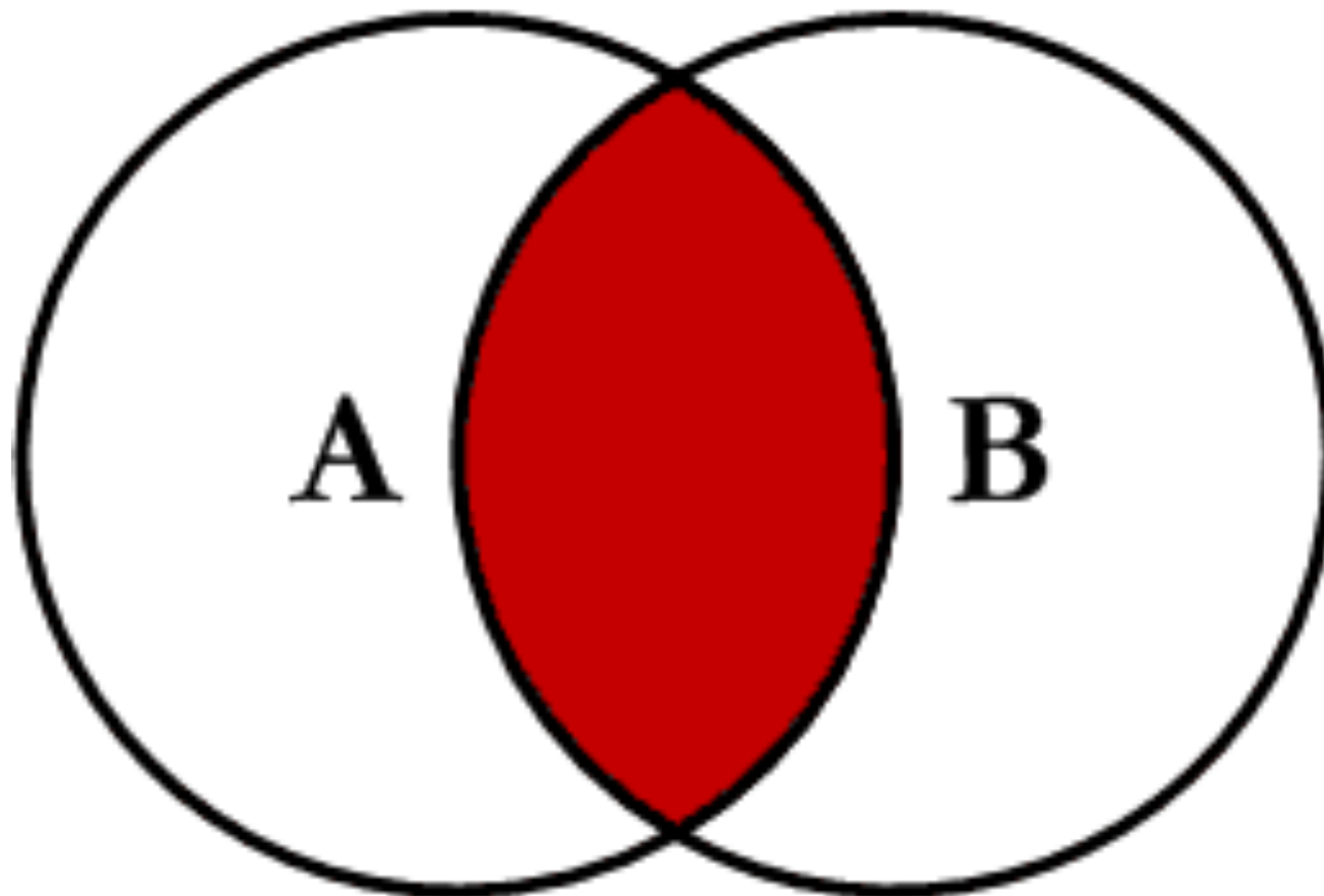
- The words `INNER` and `OUTER` are optional in all forms.
- `INNER` is the default; `LEFT`, `RIGHT`, and `FULL` imply an outer join.
- The join condition is specified in the `ON` or `USING` clause, or implicitly by the word `NATURAL`.
- The join condition determines which rows from the two source tables are considered to “match”

INNER JOIN

- For each row $R1$ of $T1$, the joined table has a row for each row in $T2$ that satisfies the join condition with $R1$.

INNER JOIN

```
SELECT * FROM A INNER JOIN B ON A.ID = B.ID
```



```
SELECT * FROM t1 INNER JOIN t2 ON t1.num = t2.num;
```

num	name	num	value
1	a	1	xxx
3	c	3	yyy

```
SELECT * FROM t1 INNER JOIN t2 USING (num);
```

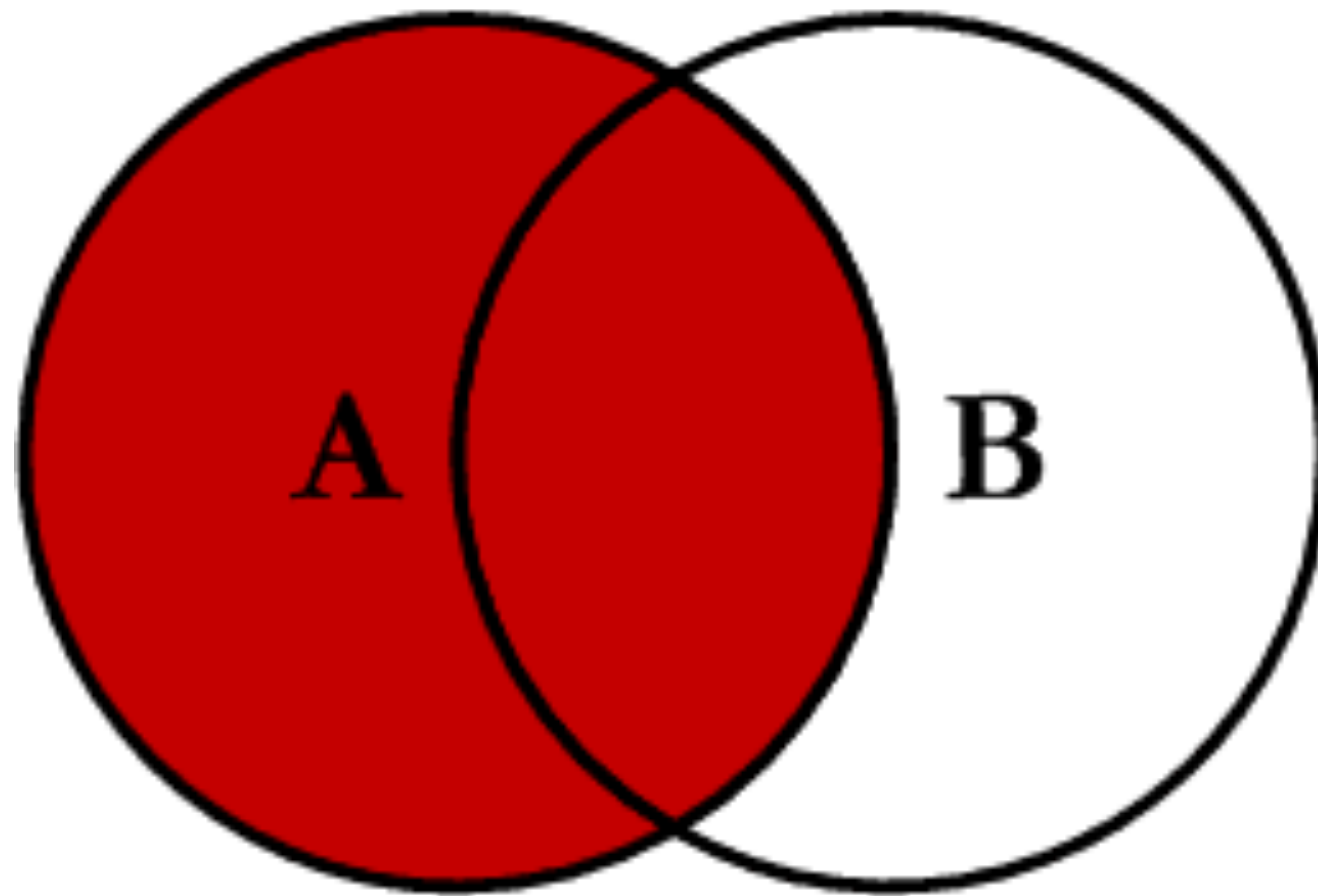
num		name		value
1		a		xxx
3		c		yyy

```
SELECT * FROM t1 NATURAL INNER JOIN t2;
```

num		name		value
-----+-----+-----				
1		a		xxx
3		c		yyy

LEFT JOIN

```
SELECT * FROM A LEFT JOIN B ON A.ID = B.ID
```



```
SELECT * FROM A LEFT OUTER JOIN B ON A.ID = B.ID
```



```
SELECT * FROM t1 LEFT JOIN t2 ON t1.num = t2.num;
```

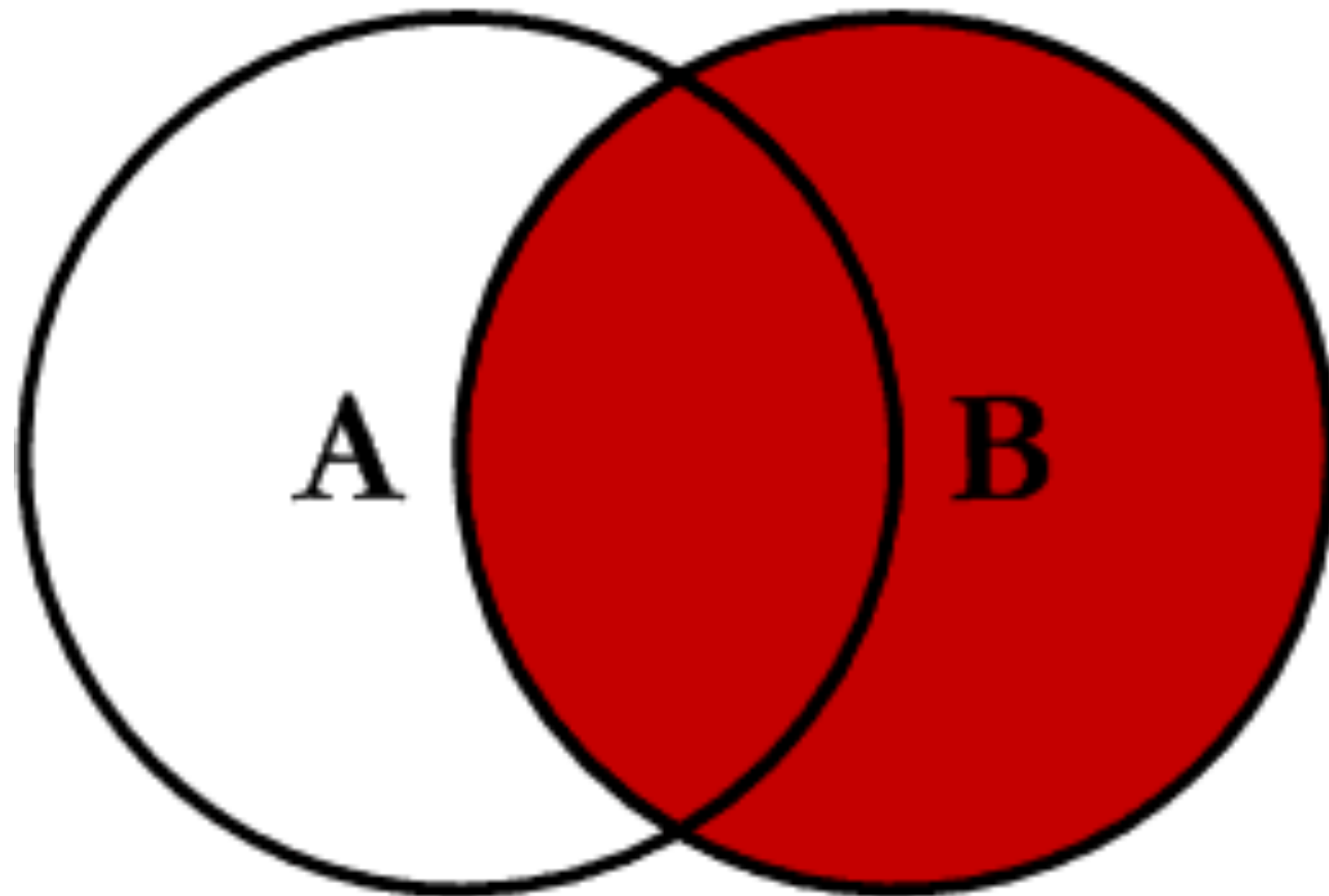
num	name	num	value
1	a	1	xxx
2	b		
3	c	3	yyy

```
SELECT * FROM t1 LEFT JOIN t2 USING (num);
```

num		name		value
1		a		xxx
2		b		
3		c		yyy

RIGHT JOIN

```
SELECT * FROM A RIGHT JOIN B ON A.ID = B.ID
```



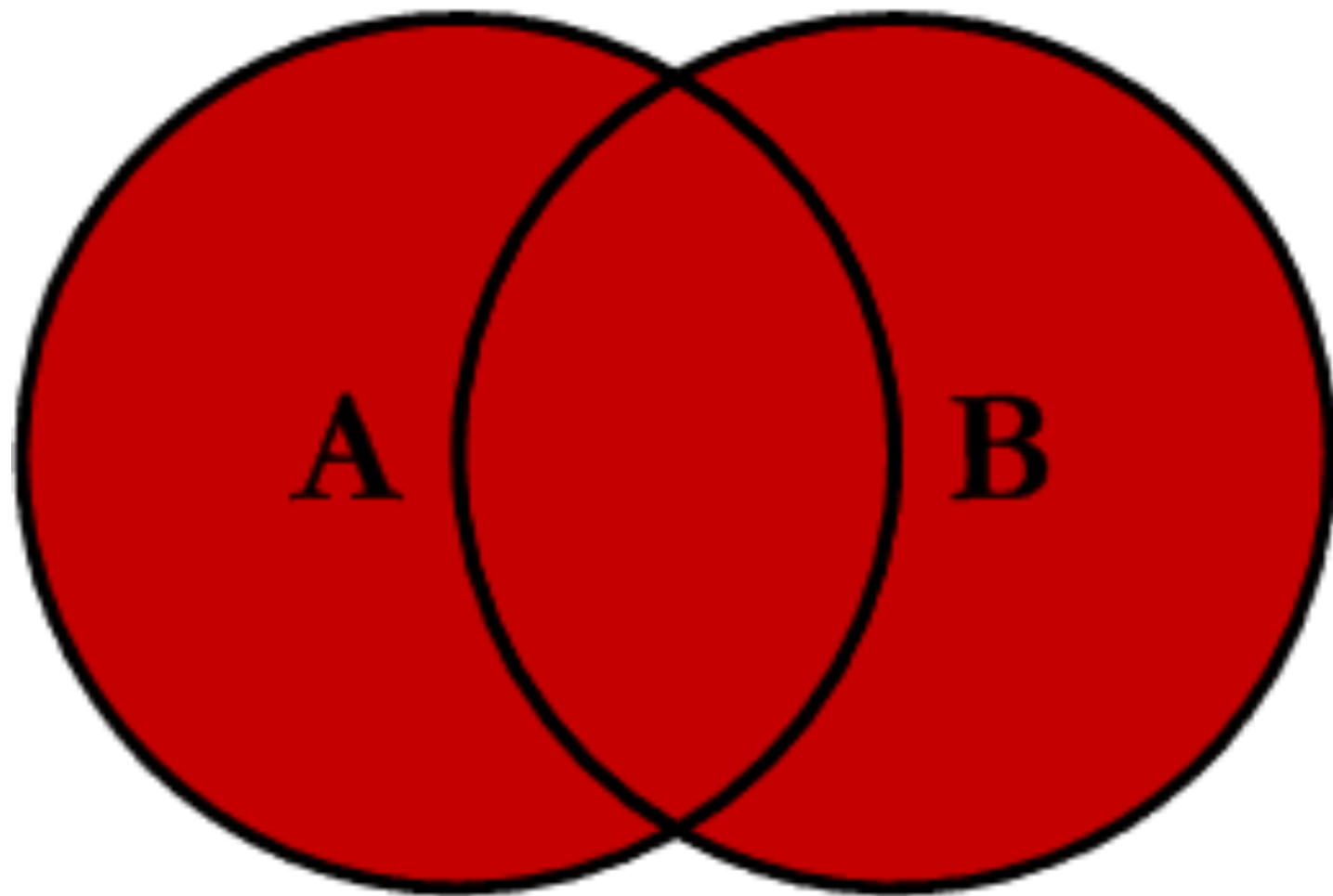
```
SELECT * FROM A RIGHT OUTER JOIN B ON A.ID = B.ID
```

```
SELECT * FROM t1 RIGHT JOIN t2 ON t1.num = t2.num;
```

num	name	num	value
1	a	1	xxx
3	c	3	yyy
		5	zzz

FULL JOIN

```
SELECT * FROM A FULL JOIN B ON A.ID = B.ID
```



```
SELECT * FROM A FULL OUTER JOIN B ON A.ID = B.ID
```

```
SELECT * FROM t1 FULL JOIN t2 ON t1.num = t2.num;
```

num	name	num	value
1	a	1	xxx
2	b		
3	c	3	yyy
		5	zzz

JOIN

- The join condition specified with `ON` can also contain conditions that do not relate directly to the join.
- This can prove useful for some queries but needs to be thought out carefully.

```
SELECT * FROM t1 LEFT JOIN t2 ON t1.num = t2.num  
AND t2.value = 'xxx';
```

num	name	num	value
1	a	1	xxx
2	b		
3	c		

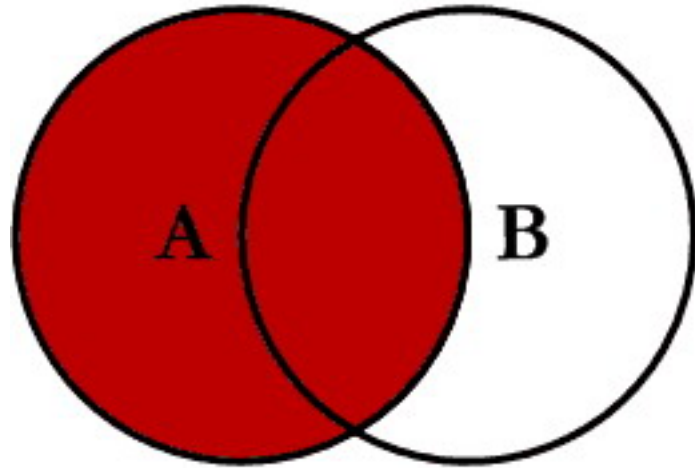

```
SELECT * FROM t1 LEFT JOIN t2 ON t1.num = t2.num
WHERE t2.value = 'xxx';
```

num	name	num	value
1	a	1	xxx

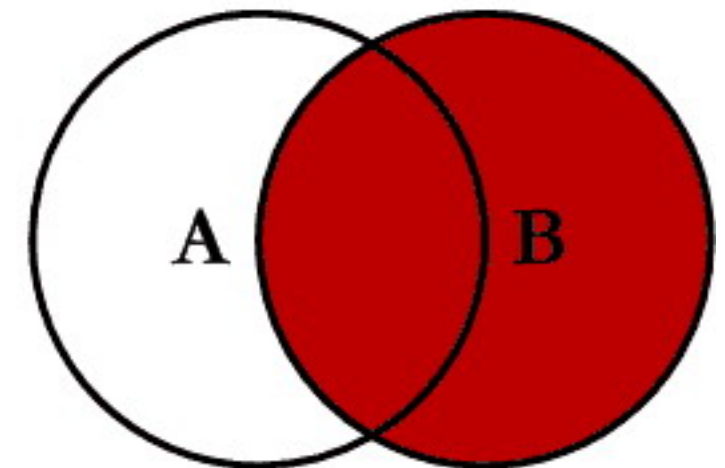
JOIN

- Notice that placing the restriction in the WHERE clause produces a different result
- This is because a restriction placed in the ON clause is processed before the join, while a restriction placed in the WHERE clause is processed after the join.
- That does not matter with inner joins, but it matters a lot with outer joins.

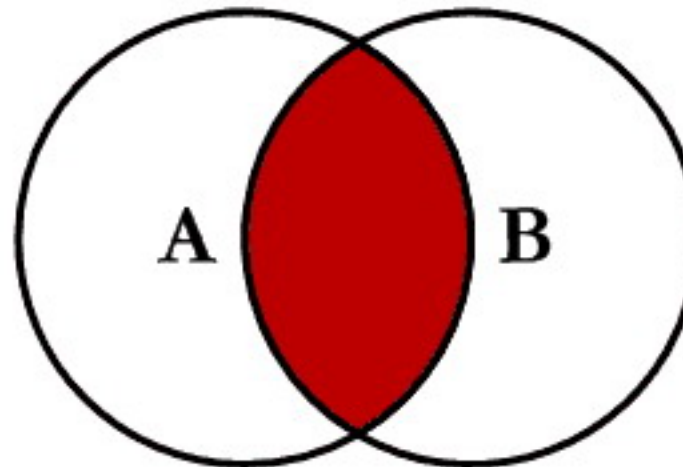
SQL JOINS



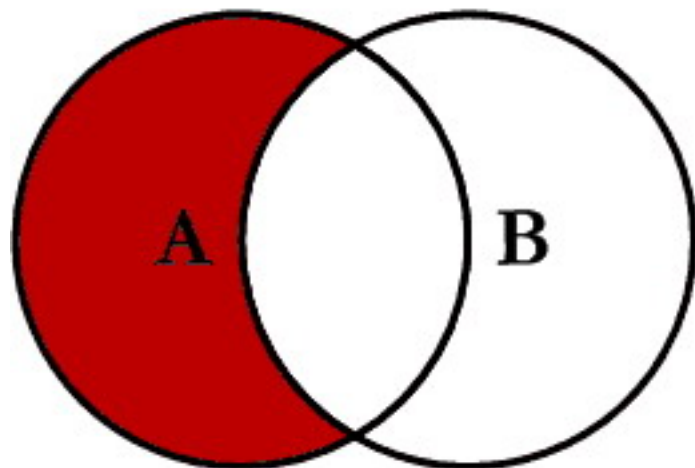
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



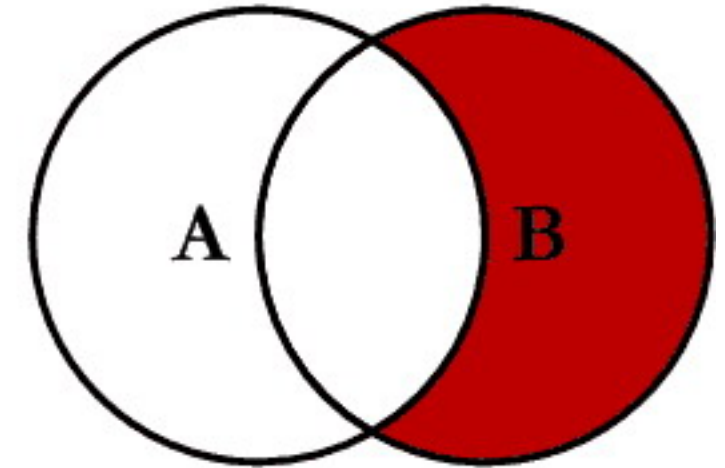
```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



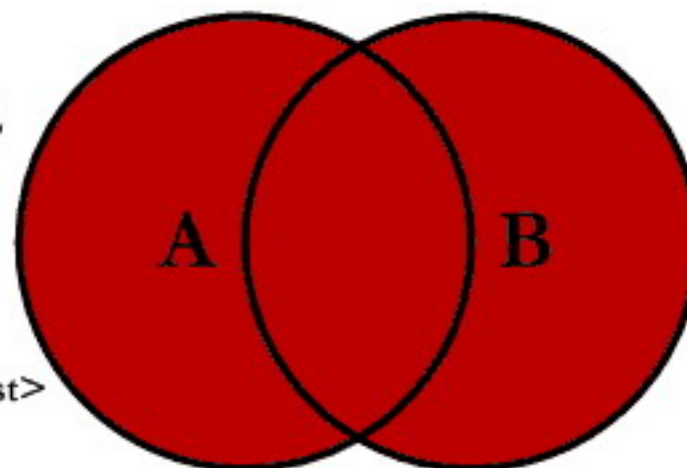
```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



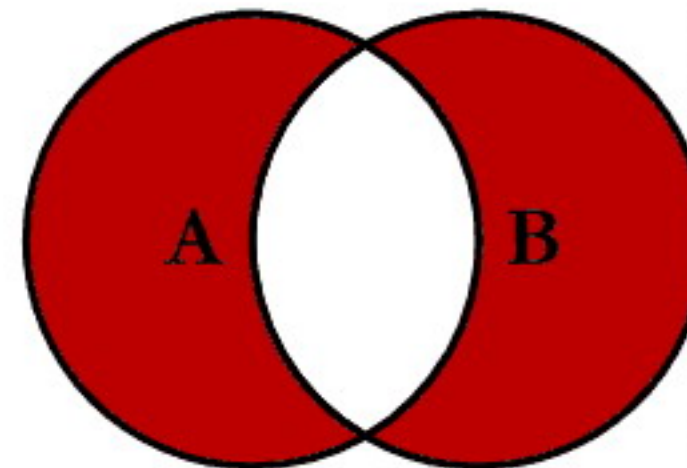
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```



```
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
```

Questions?