SQL JOIN’s

http://www.w3schools.com/sql/sql_join.asp

An SQL JOIN clause is used to combine rows from two or more tables, based on a common field between them.

The most common type of join is: **SQL INNER JOIN (simple join)**. An SQL INNER JOIN return all rows from multiple tables where the join condition is met.

Let's look at a selection from the "Orders" table:

OrderID	CustomerID	OrderDate
10308	2	1996-09-18
10309	37	1996-09-19
10310	77	1996-09-20

Then, have a look at a selection from the "Customers" table:

CustomerID	CustomerName	ContactName	Country
1	Alfreds Futterkiste	Maria Anders	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mexico

Notice that the "CustomerID" column in the "Orders" table refers to the customer in the "Customers" table. The relationship between the two tables above is the "CustomerID" column.

Then, if we run the following SQL statement (that contains an INNER JOIN):

```
SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate
FROM Orders
INNER JOIN Customers
ON Orders.CustomerID=Customers.CustomerID;
```

it will produce something like this:

OrderID	**CustomerName**	**OrderDate**
10308	Ana Trujillo Emparedados y helados	9/18/1996
10365	Antonio Moreno Taquería	11/27/1996
10383	Around the Horn	12/16/1996
10355	Around the Horn	11/15/1996
10278	Berglunds snabbköp	8/12/1996

Table A is on the left, and **Table B** is on the right. We'll populate them with four records each.

ID	Name	ID	Name
1	Pirate	1	Rutabaga
2	Monkey	2	Pirate
3	Ninja	3	Darth Vader
4	Spaghetti	4	Ninja

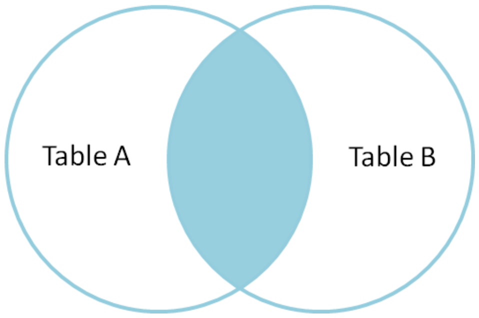
Let's join these tables by the name field in a few different ways and see if we can get a conceptual match to those nifty Venn diagrams.

http://blog.codinghorror.com/a-visual-explanation-of-sql-joins/

This exercise produces the same named columns from 2 tables - to make the output column names more "readable" either change the table definition so the columns names are unique or use the AS

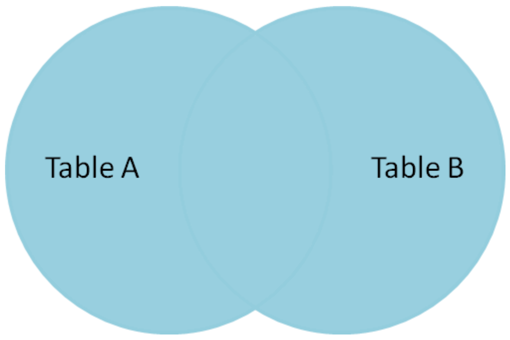
```
SELECT * FROM TableA

INNER JOIN TableB

ON TableA.name = TableB.name

id  name      id   name

--  ----      --   ----

1   Pirate    2    Pirate

3   Ninja     4    Ninja
```

Inner join produces only the set of records that match in both Table A and Table B.



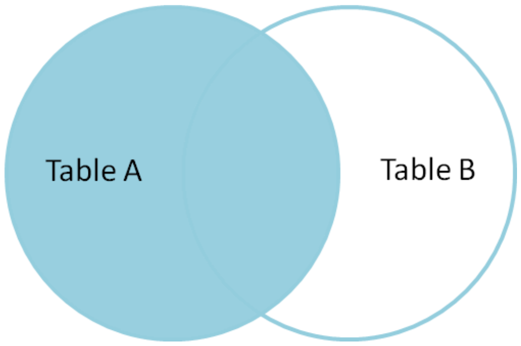
```
SELECT * FROM TableA

FULL OUTER JOIN TableB

ON TableA.name = TableB.name

id    name       id    name

--    ----       --    ----

1     Pirate     2     Pirate

2     Monkey     null  null

3     Ninja      4     Ninja

4     Spaghetti  null  null

null  null       1     Rutabaga

null  null       3     Darth Vader
```

Full outer join produces the set of all records in Table A and Table B, with matching records from both sides where available. If there is no match, the missing side will contain null.



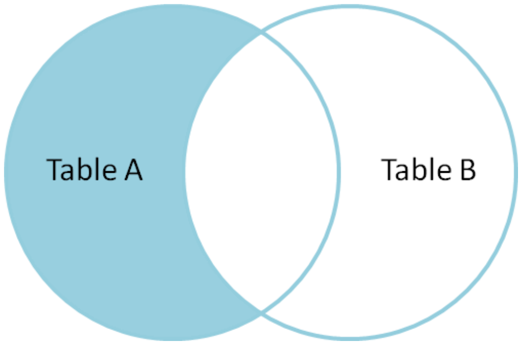
```
SELECT * FROM TableA

LEFT OUTER JOIN TableB

ON TableA.name = TableB.name

id  name      id   name

-- ----     --    ----

1   Pirate     2     Pirate

2   Monkey     null  null

3   Ninja      4     Ninja

4   Spaghetti  null  null
```

Left outer join produces a complete set of records from Table A, with the matching records (where available) in Table B. If there is no match, the right side will contain null.



```
SELECT * FROM TableA

LEFT OUTER JOIN TableB

ON TableA.name = TableB.name

WHERE TableB.id IS null

id  name      id    name

-- ----     --     ----

2   Monkey     null   null

4   Spaghetti  null   null
```

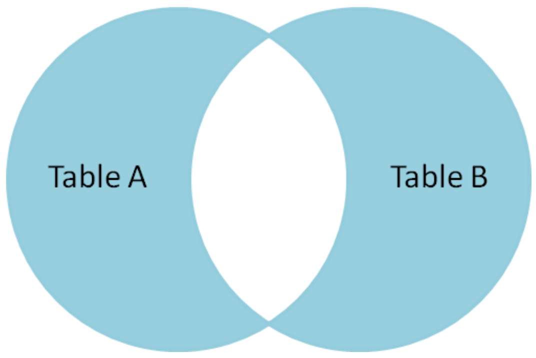
To produce the set of records only in Table A, but not in Table B, we perform the same **left outer join**, then **exclude** the records we don't want **from** the **right** side via a **where clause**.



```
SELECT * FROM TableA

FULL OUTER JOIN TableB

ON TableA.name = TableB.name

WHERE TableA.id IS null

OR TableB.id IS null
```

```
id   name       id   name

--   ----       --   ----

2    Monkey     null  null

4    Spaghetti  null  null

null  null      1    Rutabaga

null  null      3    Darth Vader
```

To produce the set of records unique to Table A and Table B, we perform the same **full outer join**, then **exclude** the records we don't want from **both** sides via a **where clause**.



There's also a cartesian product or cross join, which as far as I can tell, can't be expressed as a Venn diagram:

```
SELECT * FROM TableA

CROSS JOIN TableB
```

This joins "everything to everything", resulting in 4 x 4 = 16 rows, far more than we had in the original sets. If you do the math, you can see why this is a very dangerous join to run against large tables.