

A Visual Explanation of SQL Joins

I thought Ligaya Turmelle's [post on SQL joins](#) was a great primer for novice developers. Since SQL joins *appear* to be set-based, the use of [Venn diagrams](#) to explain them seems, at first blush, to be a natural fit. However, like the commenters to her post, I found that the Venn diagrams didn't quite match the [SQL join syntax](#) reality in my testing.

I love the concept, though, so let's see if we can make it work. Assume we have the following two tables. **Table A** is on the left, and **Table B** is on the right. We'll populate them with four records each.

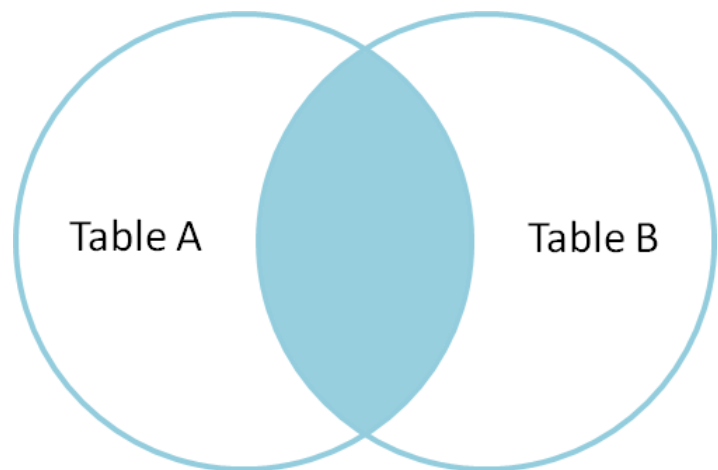
id	name	id	name
1	Pirate	1	Rutabaga
2	Monkey	2	Pirate
3	Ninja	3	Darth Vader
4	Spaghetti	4	Ninja

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.

```
SELECT * FROM TableA
INNER JOIN TableB
ON TableA.name = TableB.name
```

id	name	id	name
1	Pirate	2	Pirate
3	Ninja	4	Ninja

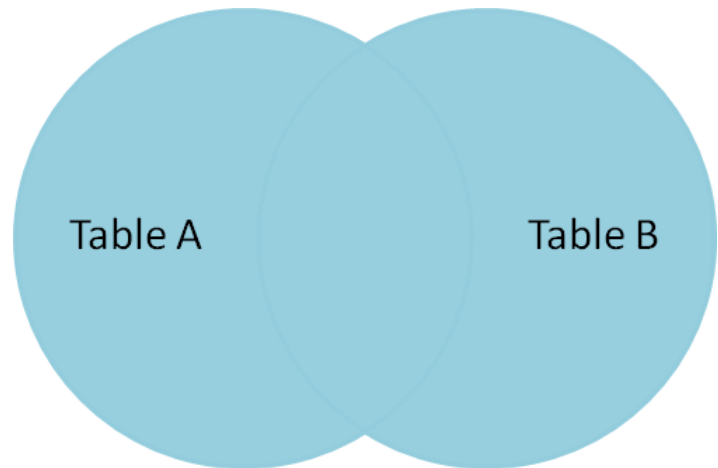
Inner join produces only the set of records that match in both Table A and Table B.



```
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

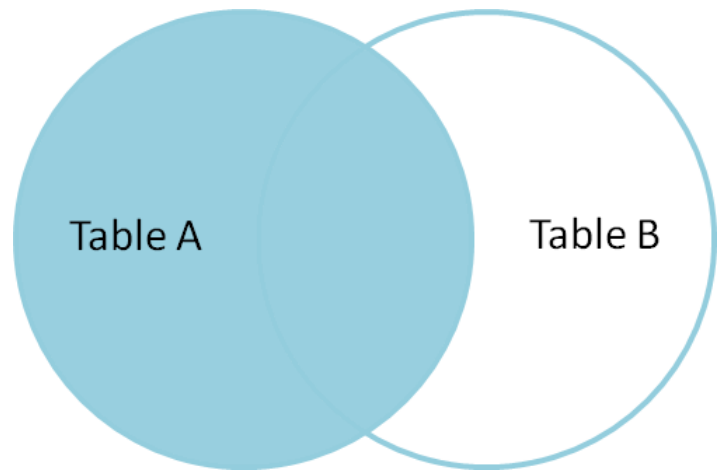
Full outer join produces the set of all records in Table A and Table B, with matching records from both sides where available. If there is no match, the missing side will contain null.



```
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

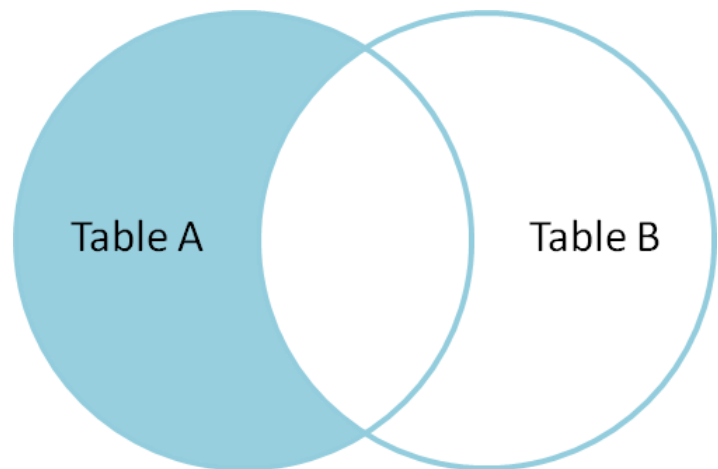
Left outer join produces a complete set of records from Table A, with the matching records (where available) in Table B. If there is no match, the right side will contain null.



```
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

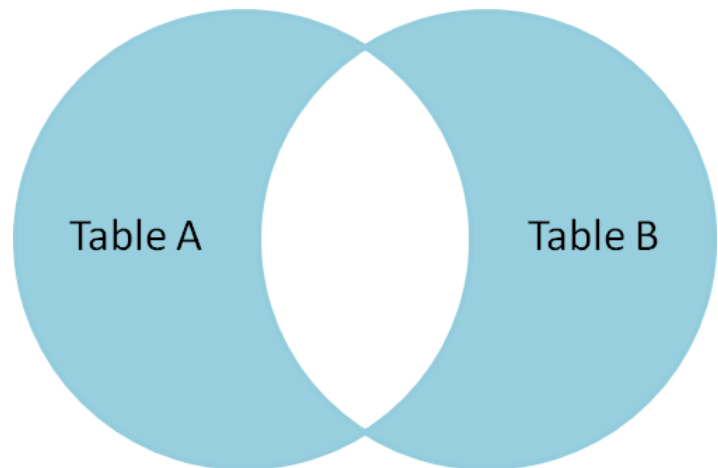
To produce the set of records only in Table A, but not in Table B, we perform the same left outer join, then **exclude the records we don't want from the right side via a where clause.**



```
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 $4 \times 4 = 16$ 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.