Types of Joins

This lesson discusses the different types of joins as specified by ANSI SQL.

Types of Joins

In the previous section we worked with a single table to learn the basics of SQL. However, relational databases define relationships between tables and often queries require gleaning information from two or more tables. Joins allow us to combine rows from multiple tables using columns common between them. In fact, the relations defined amongst tables is what makes relational databases, relational.

To drive the concepts home, we'll work with two tables that have one column common between them. The two tables are shown below:

Movie Table

MovieID	MovieName
1	Star Wars
2	Sholay
3	The Italian Job

Cinema Table

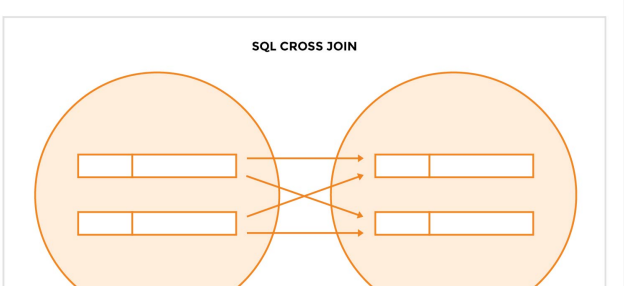
MovieID	Cinema Name	RunFor Days
2	Naz Cinema	101
5	Apollo Theater	45

The ANSI SQL standard defines five types of joins that we'll discuss.

Cross Join

We'll start with the cross join, which is also known as the cartesian product. In this case, we pick the first row of Table A and match it with every row of Table B. Next, we pick the second row of Table A and match it with every row of Table B. There's no condition specified which is tested to determine if a row from Table A should be *joined* with a row from Table B. The resulting table for a cross join of the two example tables will be as follows:

MovieID	MovieName	MovieID	CinemaName	RunForDays
1	Star Wars	2	Naz Cinema	101
1	Star Wars	5	Apollo Theater	45
2	Sholay	2	Naz Cinema	101
2	Sholay	5	Apollo Theater	45
3	The Italian Job	2	Naz Cinema	101
3	The Italian Job	5	Apollo Theater	45

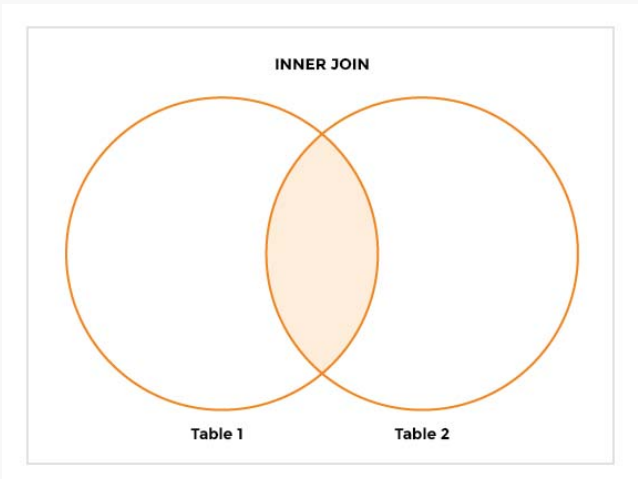


Inner Join

In case of an inner join, a condition, or multiple conditions, are tested to determine if a row from Table A should be joined with a row from Table B. This condition is called the **join predicate**. In the case of our example, the two tables share the movie iD column as the common value between them. The movie iD column establishes a relation between the two tables. Using the common column, we can determine if a given movie was screened on any of the theaters we have in our database and if so, then for how many days.

MovieID	MovieName	MovieID	CinemaName	RunForDays
2	Sholay	2	Naz Cinema	101

In set theory, the Venn diagram representation for an inner join is as follows:

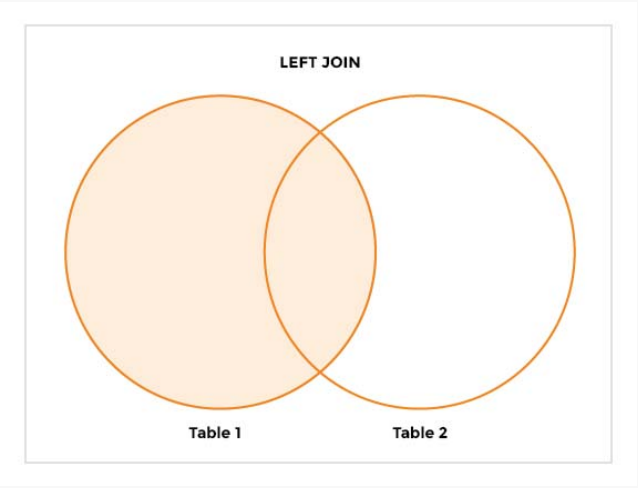


Left Outer Join

In the case of left join, the result set consists of rows that match the join predicate and also rows from the table specified on the left of the left join clause that doesn't match the join predicate. Null is inserted for columns from Table B and for rows from table A that didn't satisfy the join predicate. Said a different way, all rows from the left are always included

in the result set and rows from the right are only included if they match the join predicate.

MovieID	MovieName	MovieID	CinemaName	RunForDays
1	Star Wars	**NULL**	**NULL**	**NULL**
2	Sholay	2	Naz Cinema	101
3	The Italian Job	**NULL**	**NULL**	**NULL**

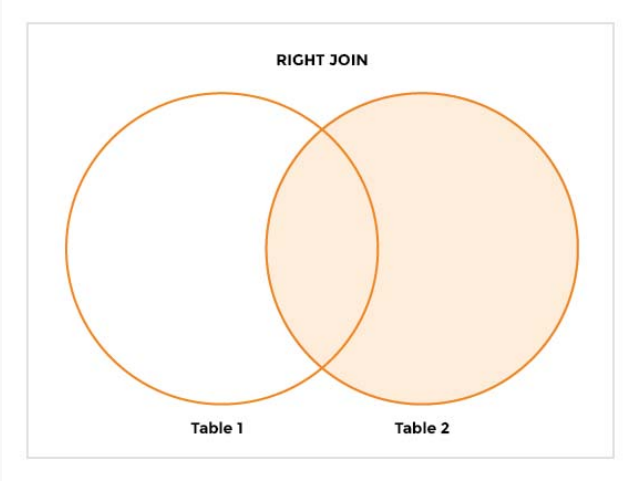


Right Outer Join

The right join is the reverse of the left join. In this case, all rows from the right table are always included in the result set and only those rows from the left table make it to the result set that satisfies the join condition. With left and outer joins, we specify which side of the join is allowed to have a row in the result when the join predicate isn't satisfied.

MovieID	MovieName	MovieID	CinemaName	RunForDays
2	Sholay	2	Naz Cinema	101

NULL	**NULL**	5	Apollo Theater	45

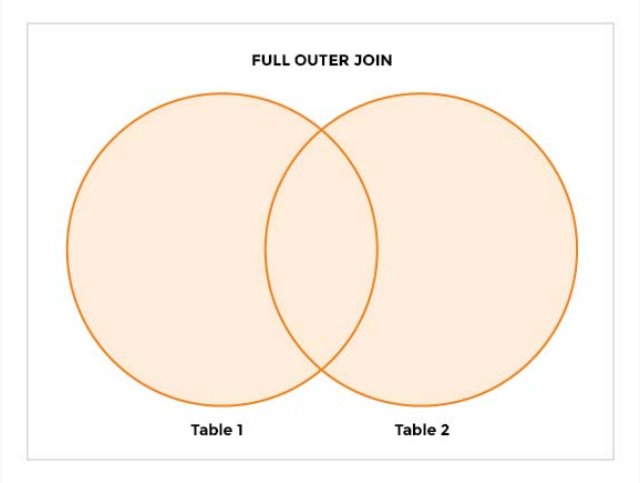


Full Outer Join

In the case of a full join, rows from both the tables are included in the result set. Rows that evaluate true for the join predicate are only included once. Rows that don't match the predicate have NULL inserted for columns belonging to the other table. Note that MySQL doesn't support a full join.

MovieID	MovieName	MovieID	CinemaName	RunForDays
1	Star Wars	**NULL**	**NULL**	**NULL**
2	Sholay	2	Naz Cinema	101
3	The Italian Job	**NULL**	**NULL**	**NULL**
NULL	**NULL**	5	Apollo Theater	45

In set theory, the Venn diagram representation for a full join is as follows:



Self Join

A self join is the result set when a table is joined to itself. If we create a self join of the movie table based on the movie iD the result will be as follows:

MovieID	MovieName	MovieID	MovieName
1	Star Wars	1	Star Wars
2	Sholay	2	Sholay
3	The Italian Job	3	The Italian Job