


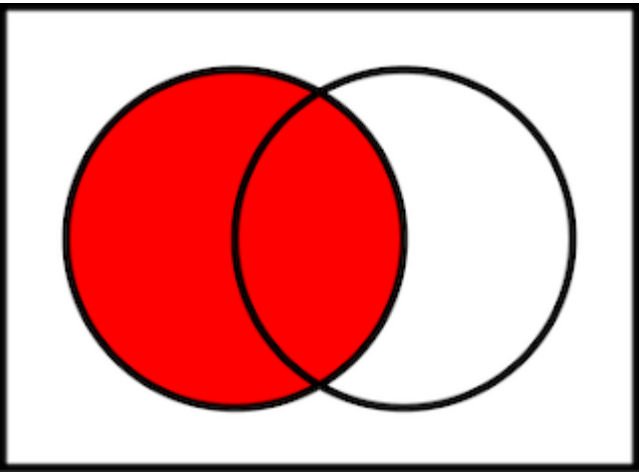
Sub-Query (Outer Join)

In SQL, a join is when rows are selected from multiple tables and joined together on common columns. Thinking about your tables like a Venn Diagram is a great analogy for this.

Let's say you want to query all Accounts and also include their related Opportunities.

```
SELECT Id, Name, (SELECT Id, Name, Amount, CloseDate FROM Opportunities)
FROM Account
```

SOQL_OuterJoin.sql hosted with  by GitHub [view raw](#)




This is an example of an Outer Join because it selects *all* Accounts and will include their related Opportunities via the sub-query. If an Account does not have an Opportunity, it will still be returned here.

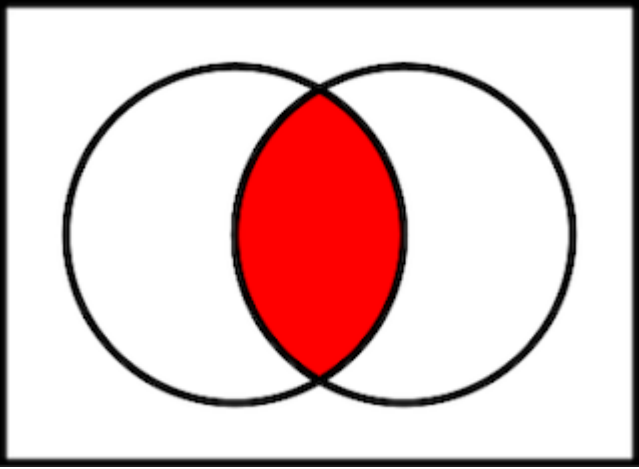
An Outer Join is similar to selecting one whole circle in a Venn Diagram, including the area that overlaps with the other circle.

Sub-Query (Inner Join)

An inner join can seem a little more complex. This is when you need to select the area that intersects between the circles in a Venn Diagram. A more concrete example would be to select all Accounts and their related Opportunities, *only when an Account has a related Opportunity*.

```
SELECT Id, Name, (SELECT Id, Name, Amount, CloseDate FROM Opportunities)
FROM Account
WHERE Id IN (SELECT AccountId FROM Opportunity)
```

SOQL_InnerJoin.sql hosted with  by GitHub [view raw](#)



The addition of line 3 is the only difference between this query and the previous example. The WHERE clause has a sub-query in it which will filter for Accounts whose Id is IN a list of AccountIds selected from Opportunity. This creates out inner join because we now only return Accounts *with* Opportunities.

Why we Don't Use the JOIN Keyword in SOQL

Each SOQL query must select from a single primary table. This is different from SQL where you can select from Table A and Table B, then join them together. I suspect this is not possible in SOQL because it keeps your queries fast and less prone to hitting [limits](#).

It's not uncommon to have a wild SQL query select from many tables, perform complex transformations and output a nicely joined result - all taking 10 minutes to run. Unfortunately SOQL adheres to certain [limits](#) that would prevent you from doing something really complex like this.

While Sub-Queries will solve most of your JOIN use cases in SOQL, you may still find yourself exporting the data and performing transformations outside of Salesforce. It's also not uncommon to toss the data into another database and run SQL on it there instead.