SQL for Data Science – Week 3:
Subqueries and Joins in SQL

Objectives

- Retrieve data from multiple tables using subqueries.
- Join tables using an Inner Join and table aliases.
- Filter a given data set using set theory by joining tables using Natural, Outer, and Self Joins.
- Assess the risk versus benefit of using a Cross Join or Cartesian Join on a set of data.
- Create an analysis table from multiple queries using the UNION operator.

Subqueries

- Queries embedded into other queries
- Merge data from multiple sources together
- Helps with adding other filtering criteria
- Example: Need to know the region each customer is from who has had an order with freight over 100
    - SELECT CustomerId, CompanyName, Region FROM Customers
      Where customerID in (SELECT customerID FROM Orders WHERE Freight >100);
- Start with inner query then outer most and then join
    - Always performs innermost select portion first

Subquery Best Practices

- No limit to number of subqueries
- Performance slows as deeply nest
- Subquery selects can only retrieve a single column
- INDENT
- www.poorsql.com
    - preformats codes
- Example: Total number of orders placed by every customer
    - SELECT customer_name, customer_state (SELECT COUNT(*) as orders FROM ORDERS WHERE ORDERS.customer_id = Customer.customer_id) as ORDERS FROM customers ORDER BY Customer_name

Joins

- Associate correct records from each table on the fly
- Allow data retrieval from multiple tables in one query

Cartesian (Cross Joins)

- Each row from the first table joins with ALL the rows of another table
- Example: SELECT product_name, unit_price, company_name from suppliers CROSS JOIN products;
    - NOT MATCHING ON ANYTHING
- NOT FREQUENTLY USED

Inner Joins

- INNER JOIN keyword selects records that have matching values in both tables
  - Only the inside of the venn diagram
- Example 1:
  - Select whatever FROM table a inner join Products ON a.supplyID = Products.supplyId
- Example 2:
  - SELECT o.OrderId, c.CompanyName, e.LastName
    FROM ((Orders o INNER JOIN Customers c ON
    o.CustomerId = c.CustomerId)
    INNER JOIN Employees e on o.EmployeeId= e.EmployeeId)

Self Joins

- When table references data in itself

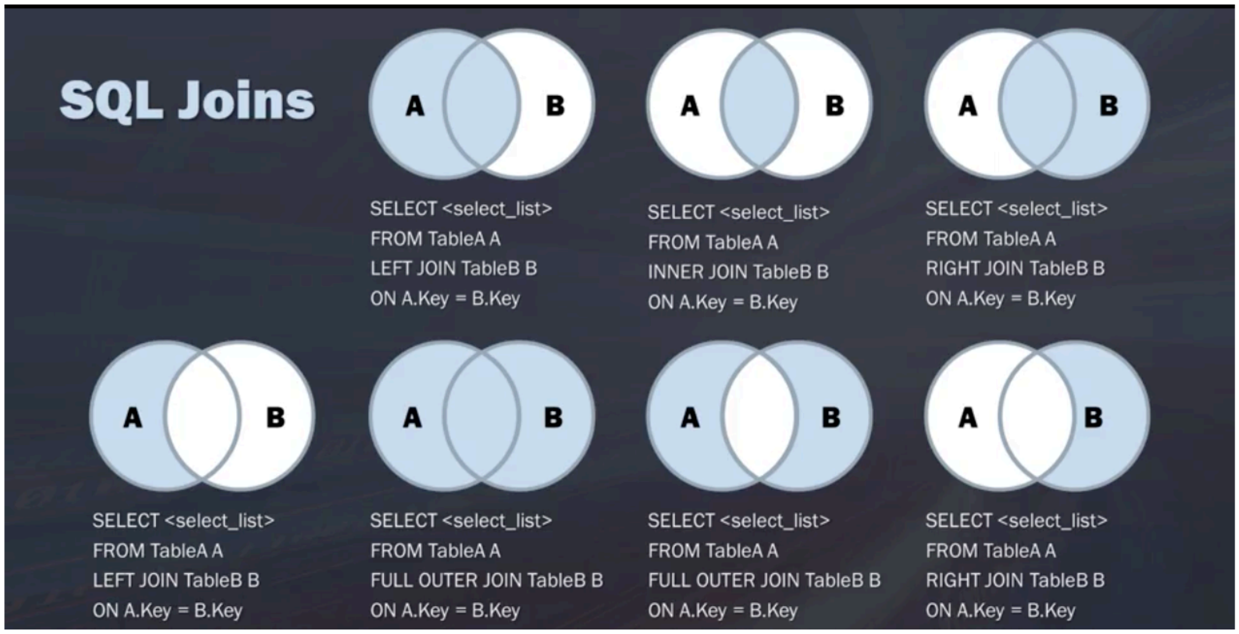
Advanced Joins

- Left
  - Returns all records from the left table and the matched records from the right table
  - Result is NULL from the right side, if there is no match
- Right
  - Returns all records from the right table and the matched records from the left table
  - Result is NULL from the left side, if there is no match
- R or L? Just switch the order
- Full outer
  - Returns all records where there is a match in either table one or there is a match in table two
  - FULL OUTER JOIN keyword returns all the rows from the left table (Customers), and all the rows from the right table (Orders). If there are rows in "Customers" that do not have matches in "Orders", or if there are rows in "Orders" that do not have matches in "Customers", those rows will be listed as well.
  -

Unions

- Used to combine the results set of two or more select statements
- If some customers or suppliers have the same city, each city will only be listed once, because UNION selects only distinct values. Use UNION ALL to also select duplicate values!
- Columns in each select statement must be in the same order
- Data Types must be the same
- Example: Which German Cities have suppliers
  - Select City, Country FROM Customer
    WHERE Country = 'Germany'
    UNION
    SELECT City, Country FROM Suppliers

```
WHERE Country = 'Germany'
ORDER BY City;
```



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

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

```
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
```

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