Multiple Tables

```
SELECT *
FROM orders
LIMIT 5;

SELECT *
FROM subscriptions
LIMIT 5;

SELECT *
FROM customers
LIMIT 5;
```

JOIN

```
SELECT * FROM orders JOIN customers ON orders.customer_id = customers.customer_id;

SELECT orders.order_id, customers.customer_name FROM orders
JOIN customers ON orders.customer_id = customers.customer_id;
```

Inner Joins

JOIN (often called an *inner join*) our result only includes rows that match our `ON` condition.

```
SELECT * FROM classes
JOIN students ON classes.id = students.class_id;
```

Left Joins

A *left join* will keep all rows from the first table, regardless of whether there is a matching row in the second table.

```
SELECT * FROM table1 LEFT JOIN table2 ON table1.c2 = table2.c2;
```

Suppose we want to know how many users subscribe to the print newspaper, but not to the online.

```
select * from newspaper
```

```
left join online
on newspaper.id = online.id
where online.id is null;
```

Primary Key vs Foreign Key

Primary keys have a few requirements:

None of the values can be `NULL`.

Each value must be unique (i.e., you can't have two customers with the same `customer_id` in the `customers` table).

A table can not have more than one primary key column.

When the primary key for one table appears in a different table, it is called a foreign key

Cross Join

```
SELECT shirts.shirt_color,
  pants.pants_color
FROM shirts
CROSS JOIN pants;
```

- A more common usage of `CROSS JOIN` is when we need to compare each row of a table to a list of values.

Union

Sometimes we just want to stack one dataset on top of the other

```
SELECT * FROM table1 UNION
SELECT * FROM table2;
```

SQL has strict rules for appending data:

Tables must have the same number of columns.

The columns must have the same data types in the same order as the first table.

With

Often times, we want to combine two tables, but one of the tables is the result of another calculation.

```
WITH previous_results AS (
  SELECT ...
  ...
  ...
  ...
)
SELECT *
FROM previous_results
JOIN customers
 ON _____ = _____;
```

Summarize

`JOIN`

will combine rows from different tables if the join condition is true.

`LEFT JOIN`

will return every row in the *left* table, and if the join condition is not met, `NULL` values are used to fill in the columns from the *right* table.

Primary key

is a column that serves a unique identifier for the rows in the table.

Foreign key

is a column that contains the primary key to another table.

`CROSS JOIN`

lets us combine all rows of one table with all rows of another table.

`UNION`

stacks one dataset on top of another.

`WITH`

allows us to define one or more temporary tables that can be used in the final query.

LINK

https://www.codecademy.com/learn/learn-sql/modules/learn-sql-multiple-tables/cheatsheet