What is SQL

SQL (pronounced S.Q.L. or Sequel, no one really cares) stands for Structured Querying Language and is a language that is used to query, form, and manipulate databases.

Structure of an SQL environment

Servers
The computer that contains your database or databases. This is typically your highest level of "container" for databases.

Databases
A collection of tables.

Schemas
Schemas are the higest level of organization for your database. They are the most granular level of organization.
and cam be used to organize your data and conatain multiple tables.

tables
A collection of rows & columns.

Different Database Types

I'll be talking about SQL like it's a standardized language and although it technically is, in practice SQL databases you'll most likely be interacting with are going to created and run by major corporations who will sprinkle their own flavor of SQL on top of or in lieu of standard SQL. The commands we'll be focusing in this course are pretty standard and shouldn't be drastically different from that which you'll be using in most other databases. Some of the biggest databases are listed below:
1. MySQL - Open Source - Created by Oracle
2. PostgreSQL - Open Source - Created by PostgreSQL
3. Microsoft SQL Server - Closed Source - Created by Microsoft
4. Oracle - Closed Source - Created by Oracle

SQL Data Types
Depending on the database type that you use, the datatypes that you'll be working with will be different. Here are the basic types and what they might be referred to by in your database.

Text types: Used to store text, can even store numbers as text

1. char
2. varchar
3. text

Numerical types: Used to store numbers

1. int
2. bigint

3. smallint
4. float
5. double
6. decimal
7. real
8. numerical
9. boolean

Date Types: Used to store dates and times

Binary Types: Large files stores using "1's and 0's", hence binary
1. blob
2. tinyblob
3. mediumblob

SQL Commands

selects all columns in the dataset, you can also specify columns like so:

```
SELECT weather, temperature
FROM dataset_1;
```

LIMIT - Can be used to limit the amount of data that is pulled into a query, very useful when first exploring the data.

```
select *
FROM dataset_1 LIMIT 10;
```

DISTINCT - Gets the unique values from a column.

```
SELECT DISTINCT passanger
FROM dataset_1;
```

WHERE - Used to filter data based on a specific condition.

```
SELECT *
FROM dataset_1
WHERE destination = 'Home';
```

order by - Used to sort the data in a specific order.

```
SELECT *
FROM dataset_1
ORDER BY coupon;
```

Aliasing - Used to rename columns in a query.

```
SELECT destination as 'Destination'
FROM dataset_1;
```

Comments - Used to comment out code.

```
SELECT * -- This is a comment
FROM dataset_1;
```

Aggregations - Used to aggregate data.

1. **GROUP BY** - Used to group data by a specific column.

```
SELECT occupation
FROM dataset_1
GROUP BY occupation;
```

2. **AVG** - Used to get the average value of a column.

```
SELECT weather,
AVG(temperature) AS 'avg_temp'
FROM dataset_1
GROUP BY weather;
```

3**. COUNT** - Used to get the count of a column.

```
SELECT weather,
COUNT(temperature) AS 'count_temp'
FROM dataset_1
GROUP BY weather;
```

4. **COUNT (DISTINCT)** - Used to get the count of a column, but only unique values.

```
SELECT weather,
COUNT(DISTINCT temperature) AS 'count_distinct_temp'
FROM dataset_1
GROUP BY weather;
```

5. **SUM** - Used to get the sum of a column.

```
SELECT weather,
SUM(temperature) AS 'sum_temp'
FROM dataset_1
GROUP BY weather;
```

6. **MIN** - Used to get the minimum value of a column.

```
SELECT weather,
MIN(temperature) AS 'min_temp'
FROM dataset_1
```

```
GROUP BY weather;
```

7. **MAX** - Used to get the maximum value of a column.

```
SELECT weather,
MAX(temperature) AS 'max_temp'
FROM dataset_1
GROUP BY weather;
```

8. **HAVING** - Used to filter data based on a specific condition.

```
SELECT occupation
FROM dataset_1
GROUP BY occupation
HAVING occupation = 'Student';
```

Combining Data

Joins and Unions - Oftentimes we'll want to enrich our data by adding more data from other datasets to it. There are two major ways we can combine our datasets together:

1. **Unions** - a union (SQL Union) is the process of stacking two tables on top of one another.

```
SELECT DISTINCT destination
FROM (
SELECT *
FROM dataset_1
UNION
SELECT *
FROM table_to_union);
```

2. **Joins** - a join (SQL Join) is the process of combining two tables together.

1**. inner join** - Used to join two tables on a specific column.

```
SELECT a.destination, a.time, b.part_of_day
FROM dataset_1 a
INNER JOIN table_to_join b
ON a.time  = b.time;
```

2. left join - Used to join two tables on a specific column.
3. right join - Used to join two tables on a specific column.
4. full join - Used to join two tables on a specific column.

Advanced Querying

Subquerying - Used to query data from a specific column.

```
SELECT destination, passanger
```

```
FROM (SELECT * FROM dataset_1 WHERE passanger = 'Alone');
```

Advanced Filtering - Used to filter data based on a specific condition.

```
SELECT *
FROM dataset_1
WHERE weather LIKE 'Sun%';
```

Wildcard Operators - Used to filter data based on a specific condition.

BETWEEN - Used to filter data based on a specific condition.

```
SELECT DISTINCT temperature
FROM dataset_1
WHERE temperature BETWEEN 29 AND 75;
```

IN - Used to filter data based on a specific condition.

```
SELECT occupation
FROM dataset_1
WHERE occupation IN ('Sales & Related', 'Management');
```

Advanced Aggregation –

Window Functions -

OVER and PARTITION BY

```
SELECT
	destination,
	weather,
	AVG(temperature) OVER (PARTITION BY weather) AS 'avg_temp_by_weather'
FROM dataset_1;
```

ROW_NUMBER() - Used to get the row number of a specific row.

```
SELECT
	destination,
	weather,
	ROW_NUMBER() OVER (PARTITION BY weather ORDER BY destination) AS
'avg_temp_by_weather'
FROM dataset_1;
```

RANK() - Used to get the rank of a specific row.

```
SELECT
	destination,
	weather,
	RANK() OVER (PARTITION BY weather ORDER BY destination) AS
'avg_temp_by_weather'
FROM dataset_1;
```

DENSE_RANK() - Used to get the rank of a specific row.

```
SELECT
destination,
weather,
DENSE_RANK() OVER (PARTITION BY weather ORDER BY destination) AS
'avg_temp_by_weather'
FROM dataset_1;
```

NTILE

```
SELECT time, NTILE (50) OVER (ORDER BY time)
FROM dataset_1;
```

LAG - Used to get the value of a specific row.

```
SELECT destination, time, LAG(row_count , 1, '99999') OVER (ORDER BY row_count) AS
'LaggedCount'
FROM dataset_1;
```

LEAD - Used to get the value of a specific row.

```
SELECT destination, time, LEAD(row_count , 1, '99999') OVER (ORDER BY row_count) AS
'LaggedCount'
FROM dataset_1;
```

Manipulating Tables

CREATE - Used to create a new table.

DROP - Used to drop a table.

ALTER - Used to alter a table.

VIEWS - Used to create a view.

Resources data

https://mode.com/sql-tutorial/sql-window-functions/

https://www.educative.io/blog/what-are-database-schemas-examples