Introduction to SQL with Postgres.

Objectives:

By the end of this chapter, you should be able to:

- Define what a relational database and SQL are
- Have Postgres installed on your machine
- Use basic psql syntax to list essential information about tables and databases

Welcome to the SQL curriculum! We will be learning how to use SQL to communicate with a database and store information permanently. Let's get started by learning a few essential definitions.

Definitions

1. **Database** - a collection of records that can easily be updated, accessed and managed. Databases are used to capture and analyze data on a more permanent basis.

2. **Relatonal Database** - a type of database that is structured so that relationships can be established among stored information.

3. **SQL** - Structured Query Language is the standardized language for communicating with and managing data in relational database. The acronym is pronounced like the word "sequel".

4. **RDBMS** - A Relational DataBase Management System is a database management system based on a "relational" model. This model was actually developed before the SQL language and is the basis for SQL and systems like MySQL, Postgres, Oracle, IBM DB2, Microsoft SQL Server and many more.

5. **PostgreSQL** - PostgreSQL or "postgres" is an RDBMS that is open source (free for everyone to use and contribute too!). Postgres powers some of the largest companies in the world.

6. **Schema** - the organization of data inside of a database. The database schema represents the collection and association of tables in a database.

7. **Table** - A series of columns and rows which store data inside of a database. An example of a table is "users" or "customers".

8. **Column** - A portion of a table which has a specific category and data type. If we had a table called "users", we could create columns for "username", "password", which would both be a variable amount of characters or text. Postgres has quite a few data types, which we will see later

9. **Row / Record** - Each row in a table represents a record stored. In our "users" table, we may have a row that looks like 1, "elie", "secret". Where 1 represents a unique id, "elie" represents the value of the "username" and "secret" represents the value of the "password".

10. **psql** - a command line program, which can be used to enter PostgreSQL queries directly, or executed from a file.

Installing Postgres

First install PostgreSQL with homebrew.

```
brew install postgres
```

Start postgres

```
postgres -D /usr/local/var/postgres
```

Open up a new terminal tab (command + t).

Create a test database:

```
createdb test
```

(Optional) The commands below configure PostgreSQL to start automatically:

```
mkdir -p ~/Library/LaunchAgents
ln -sfv /usr/local/opt/postgresql/*.plist ~/Library/LaunchAgents
launchctl load -w ~/Library/LaunchAgents/homebrew.mxcl.postgresql.plist
```

You should be able to type in `psql test` and see a new shell, connecting to the `test` database.

To exit out of `psql` type in `\q`.

PSQL Syntax

Let's examine some useful postgres commands you will be using in `psql`:

- `\du` - lists users
- `\dt` - lists tables
- `\d+ table_name` - list details about the table name
- `\l` - lists databases
- `\c NAME_OF_DB` - connect to a database

New databases can be created in two ways:

1. In `psql` type `CREATE DATABASE name_of_db;`
2. In the `terminal` type `createdb name_of_db`

Existing databases can be removed in two ways:

1. In `psql` type `DROP DATABASE name_of_db;` - make sure you are not connected to that database or the command will not work
2. In the `terminal` type `dropdb name_of_db`

Syntax Gotchas

1. The most important thing with SQL syntax is to end your statements with a SEMI-COLON `;`. SQL will not understand when you have finished your statement unless it sees that.
2. You also MUST make sure to put all text strings in **single** quotes `'`, *not* double quotes. SQL views double quotes as a name of a table and single quotes as a string.