SQL

To create snippets, use snip it tool or  [carbon.sh.now](https://carbon.now.sh/" \t "_blank)

Screenshot query and results separately

What I learned in this module:

✅ SELECT \* FROM table – Retrieves all columns from a specified table.

✅ LIMIT-Restricts the number of rows returned in a query.

✅ AS– Renames a column or table for readability.

✅ WHERE– Filters records based on a condition.

✅ COUNT– Returns the number of rows that match a condition.

✅ GROUP BY– Groups rows with the same values into summary rows.

✅ ORDER BY– Sorts the result set in ascending or descending order.

✅ MIN / MAX– Returns the smallest or largest value in a column.

✅ SUM– Adds up all values in a specified column.

✅ AVG– Calculates the average value of a column.

✅ AND / OR / NOT– Combines multiple conditions in a query.

When to use SQL:

When data sizes are extra large

The data could already be stored in SQL

Simplicity/detailed: simple to learn can do a lot with it

 It uses relational algebra and tuple calculus to speed up common data maneuvers.

SQL writes the complex code for you so you can focus more on the interpretation and

presentation

SQL is open-sourced/free to use

SQL is “battle tested” for 30 yrs so we know it’s a good tool

Data =recorded information

Database stores data

Can read, write, modify, or query a database

Server=biggest, container of multiple databases

Database=medium size, inside server

Tables: small, can have multiple tables in database, made up of columns and rows (like one Excel worksheet)

Errors: “Out of memory” means the result is too big to show in browser

-Fix by limiting

[**CSVFiddle.io**](https://csvfiddle.io/) to practice SQL queries

Save via “Share Workspace” copy url and save it Word then can go back to it

SELECT \* FROM table\_name; (table name is generic, put whatever table analyzed)

**SELECT** means out of all the data, I want something returned

**FROM** means this is the table that has my data

**\*** means return all columns

If you don’t want all columns, can list specific columns col1, col2

If there are spaces in the names use quotations “column 3”

i.e. SELECT borrower, “Due to IDA” FROM “banking\_data”;

no quotes on borrower since no spaces

spacing doesn’t matter in SQL

must have ; at end of query to tell SQL the statement is over

can all be lowercase but the SQL commands in all caps are easier to read and typical

**LIMIT** allows you to narrow the number of rows you want returned

It is at end of existing query and followed by an integer

This command works for MySQL and PostgreSQL

Alias =different name= **AS** function

Shortens the typing time by giving a column a new shorthand name for the rest of query

i.e. instead of col1 we could use c1

SELECT col1 AS c1 FROM table\_name;

**WHERE** selects all the data where a certain condition is met/want filtered data

Simplest conditional statement is equals statement. this keeps all data from table in a column is equal to a defined value

i.e. state column in data, use =, and specify value

SELECT \* FROM table\_name WHERE col1 = “red”;

Only keeps rows that are red

SELECT \* FROM table\_name WHERE col2 =47

Only keeps rows that are equal to 47

If the value you want is numeric, you can use =, > or<

Extensions of WHERE

**AND** narrows down results by requiring each condition to be true

**OR** It allows for a broader selection of results by including records that satisfy

either condition, or both

**NOT** includes all records where the condition is not true i.e. WHERE NOT

**Quotation marks:**

Double quotations for table or column names

Single quotation marks for the data you’re looking for

**COUNT**

Counts rows/returns one number

Goes after SELECT and before FROM (same with all aggregate functions Max, Min, AVG)

Column name goes after COUNT in **parenthesis**

i.e. SELECT COUNT (\*) FROM table\_name;

or SELECT COUNT (column\_name) FROM table\_name;

**GROUP BY**

Goes after table name

Do aggregate math but split the rows by a category

Column name you want to group goes after GROUP BY

i.e. SELECT color, COUNT(\*) FROM table\_name GROUP BY color;

**MIN** returns

Numeric: smallest value from a column

Text: alphabetically first value

Date/time: earliest date/time

**MAX** returns

Numeric: largest value from a column

Text: alphabetically last value

Date/time: latest date/time

**SUM** adds multiple rows together into single number/integer

Same location as other aggregate functions-remember use parentheses

**AVG** or average is numberical mean of a column

Same location as aggregations

**ORDER BY**

Sorts the column returned in descending DESC/ascending ASC order

SQL defaults to ordering in ascending order

Works with text and numbers

Can order by multiple columns, SQL prioritizes first column given/first sorted

https://csvfiddle.io/#

https://csvfiddle.io/#