Databases #2: Basic SQL

Follow along at sqlfiddle.com
and select postgres

What is SQL?

- SQL is the Structured Query Language
- Code in SQL is often referred to as "queries"
- Queries create tables, insert, retrieve, update, and delete rows
- The language is simple, linear, and typically easy to read

Syntax

- Queries are comprised of SQL keywords and arguments
- Keywords are typically CAPITALIZED, while the custom names of things are typically snake_case
- Keywords can be chained with no additional syntax, just space separations
- Individual keywords and arguments are typically called "clauses"
- Queries MUST end with a semicolon, because this indicates there are no more clauses in the query
- Strings MUST be single quotes, NOT double quotes

Creating a Table

```
CREATE TABLE table_name (
    column_one TYPE CONSTRAINT,
    column_two TYPE(arg),
);
```

- We saw this in our previous slides, but let's cover it again
- Despite being two words, `CREATE TABLE` is one keyword, there's no generic CREATE
- It takes in 2 arguments, a table name and a set of columns in parens
- Columns are comma separated sets of column name, data type, and as many constraints as needed

Inserting Rows

```
INSERT INTO table_name VALUES ('column_one value', 2);
                 /* or */
INSERT INTO table_name (column_two) VALUES (2);
```

- We insert rows one at a time with `INSERT INTO`
- The first argument is the name of the table we're inserting in to
- An optional second argument is which columns we want to insert to
    - Otherwise they're inserted in column order, from left to right
- When we are ready to specify what we're inserting, we use keyword `VALUES`
- The values either insert in column order, or the order specified after table name

Retrieving Rows

```sql
/* All users */
SELECT * FROM users;

/* Only username field for user 1 */
SELECT username FROM users WHERE id = 1;
```

- Retrieving existing rows is done using `SELECT`
- This is a highly modular command that can be extended with many key words
- At its most basic, you specify which fields you want (* for all) from which table
- From there, we use `WHERE` to conditionally only select certain rows
- There's a lot to learn about `SELECT`s, so don't try to learn it all at once

Updating Rows

```
/* Update a particular user's name */
UPDATE users SET first_name = 'Will' WHERE id = 1;

/* Fix misspellings */
UPDATE people SET name = 'Jack' WHERE name = 'Jakc';
```

- Updates are done using the `UPDATE` keyword, which takes in a table name
- We tell it what the new value should be using `SET`, and telling it which fields should change
- We determine which rows to change to the new values using `WHERE`, just like `SELECT`

Deleting Rows

```
/* Delete post 1234 from the database */
DELETE FROM posts WHERE id = 1234;

/* Delete users named "carl" */
DELETE FROM users WHERE name = 'Carl';
```

- The DELETE FROM keyword behaves much like the UPDATE and SELECT ones
- After we specify which table, we need only add a WHERE clause, nothing else
- Deleted rows are gone for good, so make sure you are deleting the right things!
- SELECTing before DELETEing is a good way to double check that you're sure

Exercise: Friends Table

- Let's use what we learned on SQLFiddle.com
- Create a new table called `friends` that has an primary key auto incrementing id, a first name, and a last name
- Add 'Sam', 'Betty', and 'Jeff' to `friends`. Give them different last names.
- Betty and Jeff got married, so update Betty's last name to Jeff's
- Betty and Jeff got all coupley and stopped hanging out with y'all, so remove them from your friends table
  - Try doing this in one query, instead of removing them individually

Do a final select all to make sure you only have Sam left. If you have others in there, make sure you go back and double check each of your steps.

Additional Resources

- Codecademy: Learn SQL
- PSQL Docs: Create Table
- PSQL Docs: Insert
- PSQL Docs: Select
- PSQL Docs: Update
- PSQL Docs: Delete