## **SQL 2 (Multiple Tables)**

https://github.com/Techtonica/curriculum/blob/master/databases/sql-2.md

## Independent practice

In SQLite on your own machine, you are going to create the tables for a microblogging platform (an app similar to Twitter). Your database should be able to store user information and posts by specific users. One post must *belong to*exactly one user. One user can have many posts. Later, we'll add the ability for users to follow each other, but not now.

1. With pencil/pen and paper, write out the data for a database with the following spec. Fill in the fields with fake data that you make up. Be sure to link posts to a certain existing user!

Here's a sample schema:

```
- a table named: `users`
- which has a text field named: `name`
- and a text field named: `email`
- a table named: `posts`
- which has an integer field named: `user_id`
- and a text field named: `content`
```

2. Enter the SQLite command line program using the terminal commad sqlite3 twitter.sqlite3. (This will save your database to a file called twitter.sqlite3). Once there, use SQL statements to create two tables according to the spec for the two tables in step 1.

```
CREATE TABLES users(name TEXT, email TEXT);

CREATE TABLES posts(user_id INTEGER, email TEXT);
```

- 3. Once you have your two tables set up, compare your table set-up with another apprentice's.
- 4. Add sample data to the tables yourself (make up some users and posts).

```
INSERT INTO users(name,email)
VALUES ("Nina","nina@email");

INSERT INTO posts(user_id,content,date)
VALUES(1,"segundo post nathalia",2015);

INSERT INTO posts(user_id,content,date)
VALUES(3,"primeiro post nina",2015);
```

- 5. Try writing queries that get data such as:
- All the tweets by a given user

```
1 SELECT content
2 FROM posts
3 WHERE user_id = 1;
```

• The 10 most recent tweets by any user

```
SELECT content
FROM posts
ORDER BY date;
```

• Use a join to get a user's info along with their tweets

```
1  select * from users
2  inner join posts
3  on users.user_id = posts.user_id
4  where users.name="Nathalia";
5  
6  select *
7  from users
8  inner join posts
9  on users.user_id = posts.user_id;
```

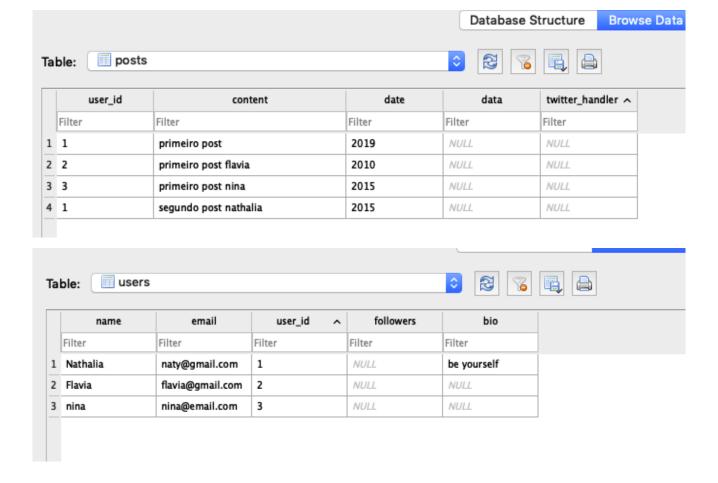
- Make up your own!
- 6. Now let's perform a database migration. We will add a new column to an existing table in our database. Add a text field named: bio to your existing users table!

```
ALTER TABLE users
ADD COLUMN bio TEXT;
```

7. Add some sample data to your new bio fields, but not for every user (leave some of their bios blank).

```
1  UPDATE users
2  SET bio = 'be yourself'
3  WHERE id = 1;
```

- 8. Try writing queries that get data such as:
- Just the bios of all users in the databases
- A list of users that have no bio
- Just the names of those users that have no bio
- Make up your own!



## **Extra Challenge**

If you complete the above, we'll move on to creating a join table.

- 1. Add a new table named follows that has an int field follower\_id and an int field followed\_id. Draw a picture for this table.
- 2. Write some queries that get data such as:
  - All users that user 2 is following
  - Hint: in follows, look for follower\_id 2, and get all of the user IDs that are in those rows' followed\_id fields. Then get the users by that set of user IDs.
  - All users that follow user 2 (users that have 2 as the followed\_id)
  - Hint: in follows, look for followed\_id 2, and get all of the user IDs that are in those rows' follower id fields. Then get those users.
  - Find which user has the most followers