Creating Tables

Correction: foreign key syntax was incorrect!

Basic SELECT

Let's get an author:

```
SELECT author_id, author_name FROM author WHERE author_id = 1;
```

What does this do?

SELECT — gets data from the database
columns — what columns (attributes) do we want? (relational theory: project)
FROM table — from where do we want it?
WHERE — what rows do we want? (relational theory: select)
The query really runs inside-out:

Scan tables (FROM)
Filter down to rows we want (WHERE)
Pick the columns
The column selection in SELECT is basically done last.

Another note: SQL is case-insensitive (generally)

We can rename columns:

```
SELECT author_id AS id, author_name AS name
FROM author
WHERE author_id = 1;
```

More fun with SELECT

There is a lot SQL can do! (look at the SQLite docs)

We can order, and limit:

```
SELECT *
FROM article
ORDER BY title
```

```
LIMIT 5
```

sorts by title (alphabetically)
only displays the first 5 results
you can use order and limit on their own, I just don't want a few thousand results in my window.
LIMIT is not standard SQL; it is supported by SQLite, MySQL, and PostgreSQL. SQL Server and Oracle have their own way of doing it, and standard SQL has yet another (supported by recent versions of PostgreSQL). It's super-useful, though, so we'll keep doing it.
We can also skip, to display the next 5:

```
SELECT *
FROM article
ORDER BY title
LIMIT 5 OFFSET 5
```

Combining Tables

If we can just query one table, that's kinda lame. We can join tables, too, to take advantage of these foreign keys and relationships.

```
SELECT author_name, inst_name
FROM author
JOIN institution ON inst_id = current_inst_id
LIMIT 10;
```

What it does:

Pulls data from both tables
Matches up rows using the specified criteria
Returns matched rows
Simple example with people and things.

Semantically equivalent to:

```
SELECT author_name, inst_name
FROM author, institution
WHERE inst_id = current_inst_id
LIMIT 10;
```

FROM builds the cartesian product: every possible combination. WHERE then limits that to the ones that match.

I prefer the JOIN syntax, because it is easier to read, and keeps joining conditions separate from selection conditions.

We can join more than one table!

```
SELECT paper_id, author_name, inst_name, title
FROM paper
JOIN authorship USING (paper_id)
JOIN author USING (author_id)
-- add in institution using the `inst_id` column
JOIN institution USING (inst_id)
WHERE paper_id = 1
ORDER BY position
```