# SQL Unit 10
# Inserting, Updating, and Deleting Data

Ryan Nixon

## Back to CRUD

- As a student pointed out, a more appropriate expansion of CRUD is "Create, Read, Update, Delete"
- Today will cover Create, Update & Delete at the row level

## Before We Begin

- The syntax additions I’m about to cover are very flexible. I’m only covering the common use cases
- For your SQL project you will need to understand and use INSERT statements. UPDATES and DELETES will not be required but *will* be on the test
- Be sure in your lab time to work with all three statements so that you are familiar with them for the test

## Inserts

- Inserts will add new rows to your tables
- They support two forms of data entry, direct (good for line-by-line entry) and query (good for copying existing data)

## Inserts

- `INSERT INTO table (col1, col2)`
  ```
      VALUES (val1, val2);
  ```
- OR
- `INSERT INTO table (col1, col2)`
  ```
      SELECT col1, col2 FROM table2;
  ```

## Inserts

-
  ```
  INSERT INTO table (col1, col2)
      VALUES (val1, val2);
  ```
- The first parentheses determine the columns that you will be providing data for
- These are optional, but if they are excluded the VALUES section must contain all columns in the table
- It is a common practice to exclude the ‘id’ column which will allow AUTO_INCREMENT to assign an id

## Inserts

-
  ```
  INSERT INTO table (col1, col2)
        VALUES (val1, val2);
  ```
- The second parentheses are for the data that will be inserted. All rules apply such as quotations around strings
- The order of the values must match the order provided in the first parentheses, or if they have been excluded the order of the column definitions for the table
- If you are required to provide a value but do not have one, NULL or DEFAULT may sometimes work as placeholders. Your mileage will vary according to the database

## Inserts

-
  ```
  INSERT INTO table (col1, col2)
      SELECT col1, col2 FROM table2;
  ```
- Instead of parentheses you may provide a SELECT statement that returns the same number of columns as required
- This is similar to the syntax for creating views or temporary tables
- Using SELECT will also support multiple rows being inserted, and is great for copying data

## Inserts

-
  ```
  INSERT INTO person (first, last, gender_id)
       VALUES('Ryan', 'Nixon', 'M');
  ```
- OR

-
  ```
  INSERT INTO person_faculty (person_id,
  department_id)
       SELECT person.id, department.id
       FROM person
       LEFT JOIN department
       ON department.name = 'Engineering Department'
       WHERE first = 'Ryan' AND last = 'Nixon';
  ```

# Inserts

- Note the exclusion of the Primary Key on the examples. If you do not specify columns, then the first example would become the following:

-
  ```
  INSERT INTO person
        VALUES(NULL, 'Ryan', '', 'Nixon', 'M', NULL, NULL);
  ```

- This provides NULL for the Primary Key (triggering auto_increment) and NULL for all other fields that are empty such as dob and address_id.

## Updates

- Updates will change existing data within table rows that match a boolean condition
- It is just as simple as INSERT to use, but be careful how you use it!

# Updates

-
  ```
  UPDATE table SET
        col1 = ‘Value’,
        col2 = 1
  WHERE col1 = ‘OldValue’;
  ```

# Updates

-
  ```
  UPDATE table SET
      col1 = ‘Value’,
      col2 = 1
  WHERE col1 = ‘OldValue’;
  ```
- You can name each field that you would like to change the information for followed by the value it will change to
- Multiple changes are supported, separated by commas

# Updates

-
  ```
  UPDATE table SET
      col1 = ‘Value’,
      col2 = 1
  WHERE col1 = ‘OldValue’;
  ```
- The WHERE condition is *extremely important*. It tells the database which rows to update
- Leaving out the WHERE will cause the update to affect all rows in the table
- When updating a single row, it is common to set the WHERE clause to filter by the primary key

# Updates

-
  ```
  UPDATE person SET
      first = ‘Shawn’,
      middle = ‘Ryan’
  WHERE first = ‘Ryan’ AND last = ‘Nixon’;
  ```
- OR

-
  ```
  UPDATE person SET
      first = ‘Shawn’,
      middle = ‘Ryan’
  WHERE id = 30752649;
  ```

## Deletes

- Just like they sound, deletes remove entire rows from a table
- Just like DROP TABLE, when something is deleted it is gone. There is no undelete

## Deletes

- `DELETE FROM table WHERE col1 = ‘Value’;`

## Deletes

- `DELETE FROM table` **`WHERE col1 = ‘Value’`**`;`
- The WHERE condition tells the database what rows to delete. Neglecting it will remove *all* rows from your table

## Deletes

-
  ```
  DELETE FROM person
      WHERE middle = ‘Ryan’ AND last = ‘Nixon’;
  ```
- OR

-
  ```
  DELETE FROM person
      WHERE id = 30752649;
  ```

## A Few Notes

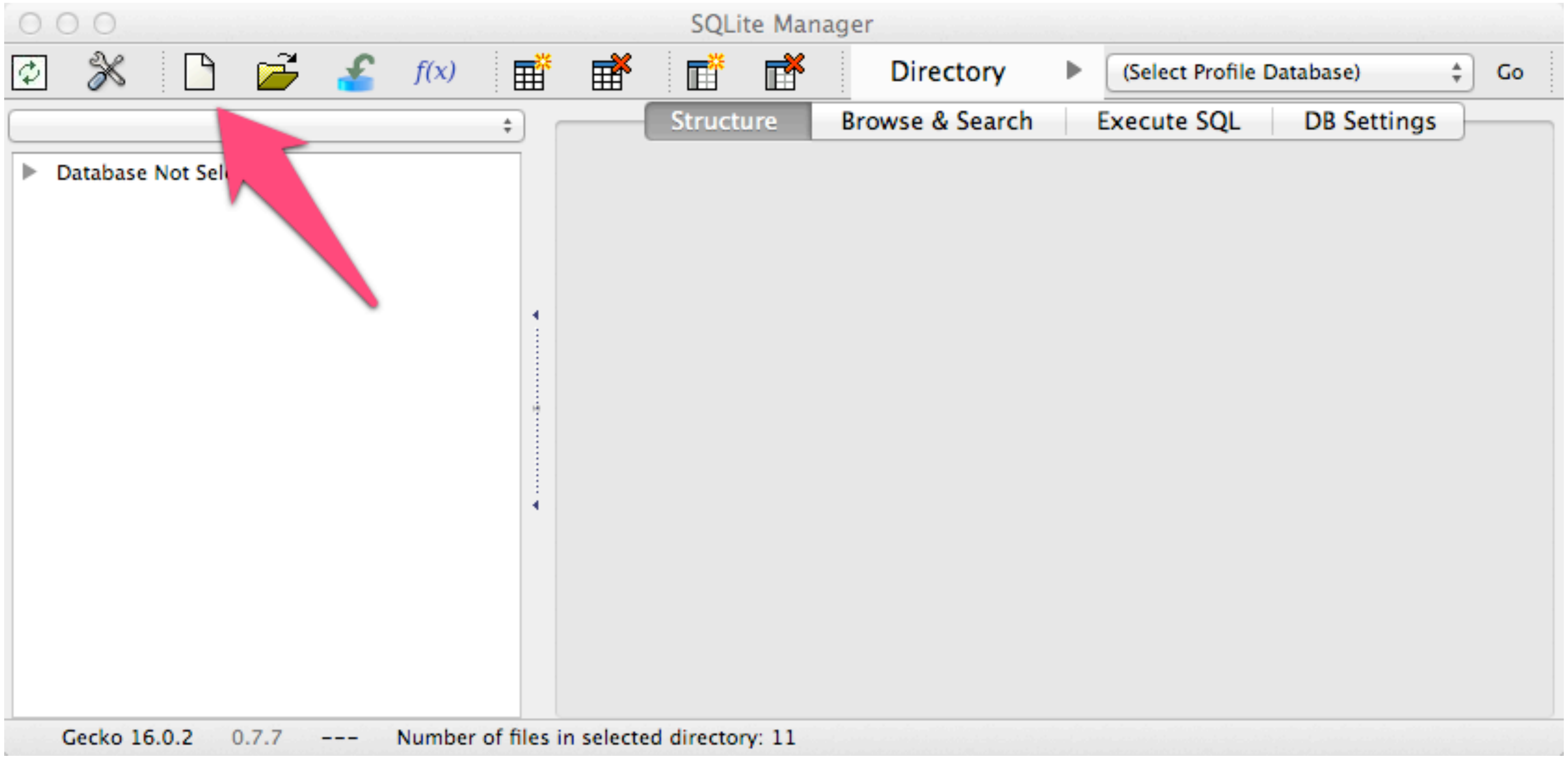
- If the WHERE condition for an UPDATE or DELETE produces no rows, then that query will still succeed but will not change anything
- If an INSERT or UPDATE changes table so that a field violates one of the constraints (Foreign Key, Not Null) then the command will fail, often loudly. SQLite does not seem to always fail loudly so be cognizant of how many times you run a query

## If Time Allows...

Some examples with the class database

# Creating a Database in Sqlite

- Click on the ‘New’ button, then give your database a name and save it to a folder



# Creating a Database in Sqlite

- Open the 'DB Settings' tab

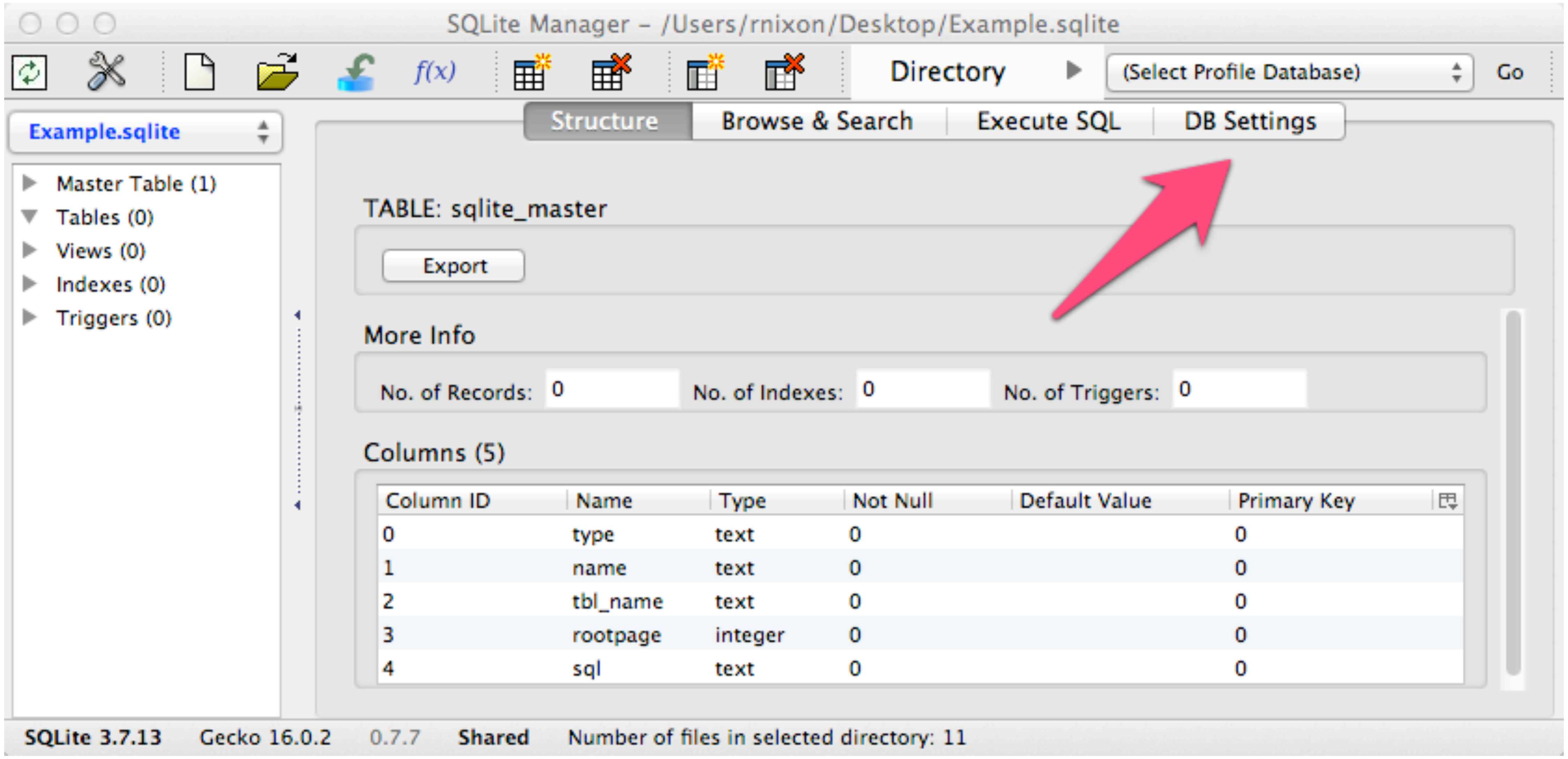


TABLE: sqlite_master

More Info

No. of Records: 0 No. of Indexes: 0 No. of Triggers: 0

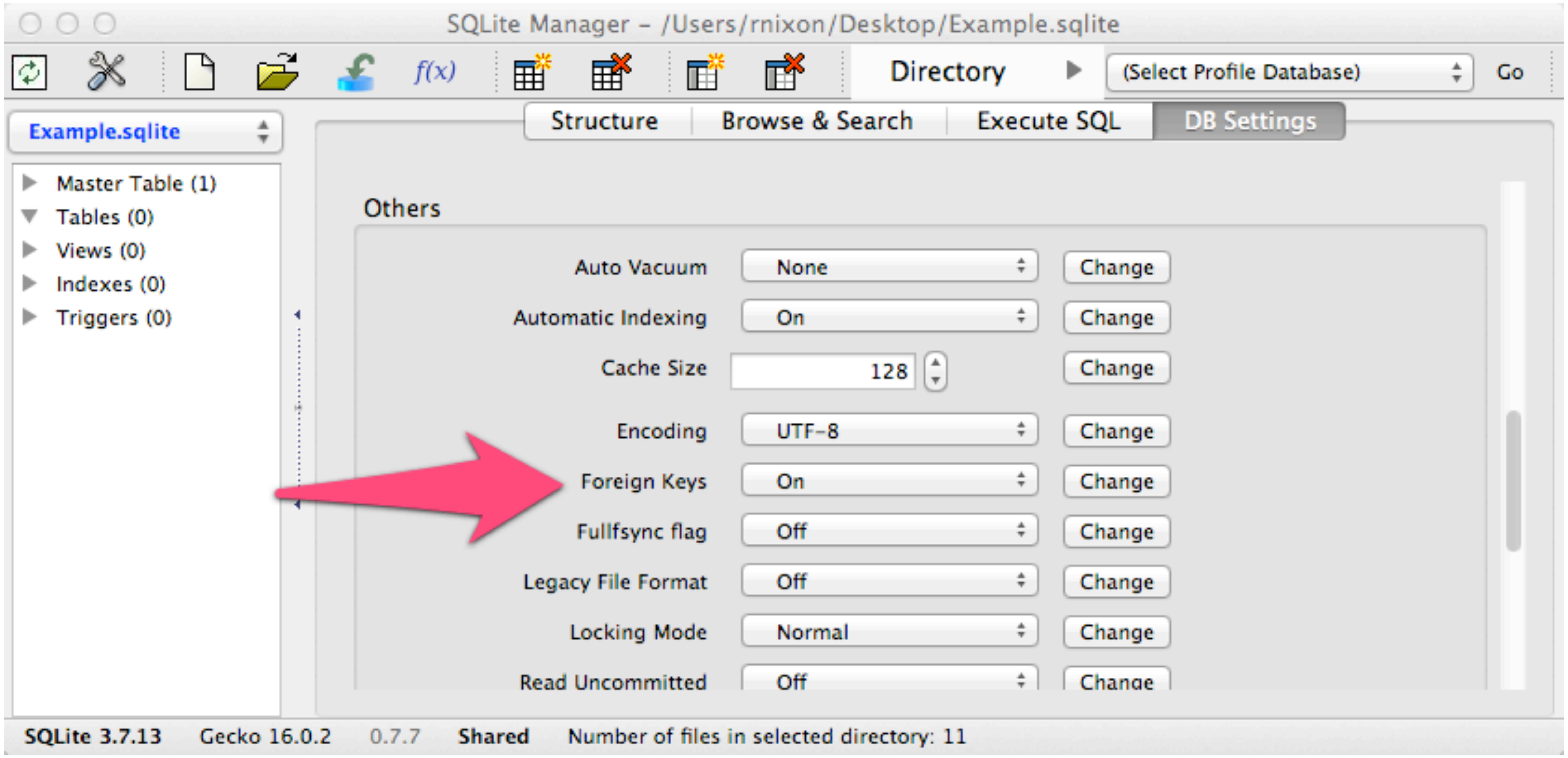
Columns (5)

| Column ID | Name | Type | Not Null | Default Value | Primary Key |
|---|---|---|---|---|---|
| 0 | type | text | 0 | | 0 |
| 1 | name | text | 0 | | 0 |
| 2 | tbl_name | text | 0 | | 0 |
| 3 | rootpage | integer | 0 | | 0 |
| 4 | sql | text | 0 | | 0 |

SQLite 3.7.13 Gecko 16.0.2 0.7.7 Shared Number of files in selected directory: 11

# Creating a Database in Sqlite

- Turn on Foreign Keys by changing the 'Foreign Keys' value to 'On' and clicking Change



Others

| | | |
|---|---|---|
| Auto Vacuum | None | Change |
| Automatic Indexing | On | Change |
| Cache Size | 128 | Change |
| Encoding | UTF-8 | Change |
| Foreign Keys | On | Change |
| Fullfsync flag | Off | Change |
| Legacy File Format | Off | Change |
| Locking Mode | Normal | Change |
| Read Uncommitted | Off | Change |

SQLite 3.7.13 Gecko 16.0.2 0.7.7 Shared Number of files in selected directory: 11

# Reminders

- No more assignments! Be sure to practice all CRUD statements on the example database and your project databases
- Lab Wednesday in SSB 172