

# Relational Databases and SQLite

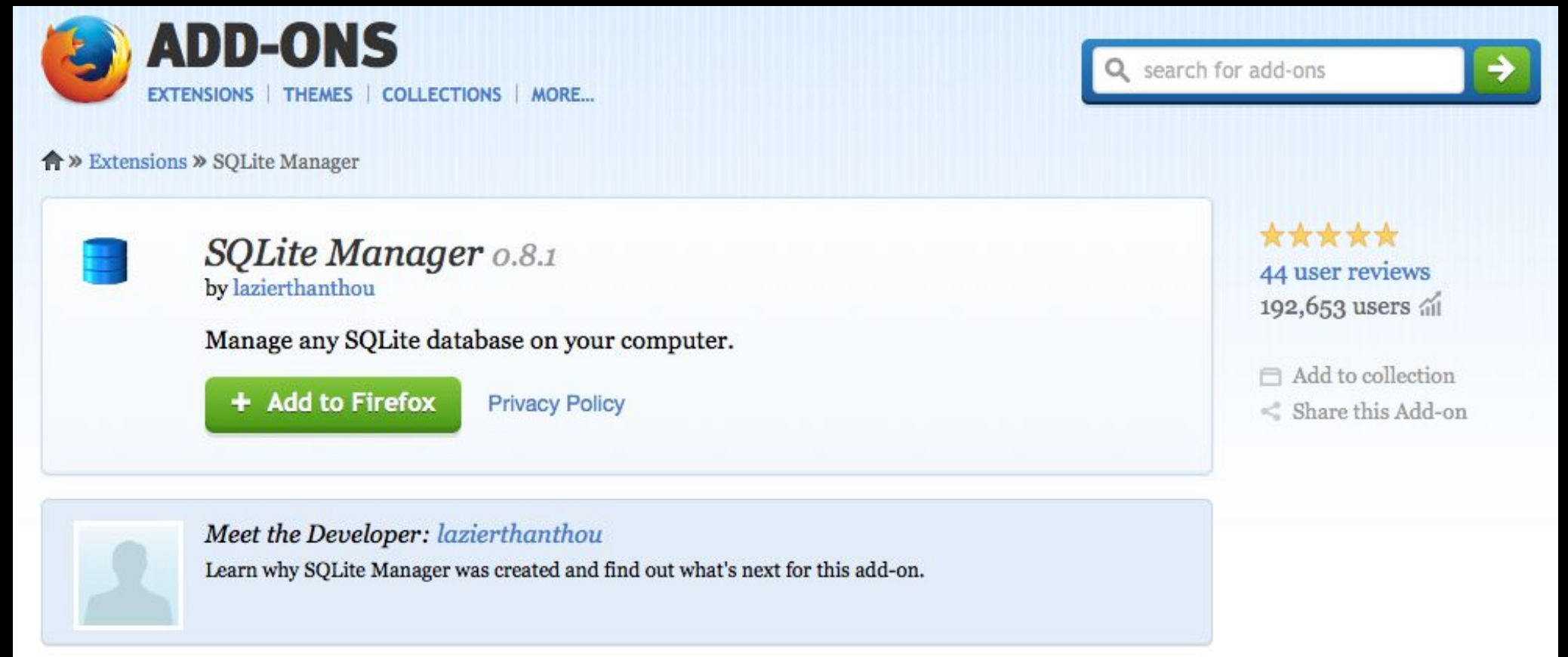
Charles Severance



Python for Informatics: Exploring Information  
[www.pythonlearn.com](http://www.pythonlearn.com)



# SQLite Manager for Firefox




The screenshot shows the Firefox Add-ons page for the SQLite Manager extension. At the top, the Firefox logo is next to the 'ADD-ONS' header, with links for 'EXTENSIONS', 'THEMES', 'COLLECTIONS', and 'MORE...'. A search bar on the right contains the text 'search for add-ons'. Below the header, a breadcrumb trail reads 'Home » Extensions » SQLite Manager'. The main content area features the SQLite Manager extension card, which includes a database icon, the title 'SQLite Manager 0.8.1' by 'lazierthanthou', a description 'Manage any SQLite database on your computer.', and a green '+ Add to Firefox' button. To the right of the card, it shows a 5-star rating, '44 user reviews', and '192,653 users'. Below the card, there is a section 'Meet the Developer: lazierthanthou' with a link to learn more. On the far right, there are links to 'Add to collection' and 'Share this Add-on'.

**ADD-ONS**  
EXTENSIONS | THEMES | COLLECTIONS | MORE...

search for add-ons

Home » Extensions » SQLite Manager


 **SQLite Manager 0.8.1**  
by [lazierthanthou](#)

Manage any SQLite database on your computer.

[+ Add to Firefox](#) [Privacy Policy](#)

★★★★★  
44 user reviews  
192,653 users

[Add to collection](#)  
[Share this Add-on](#)

 **Meet the Developer: [lazierthanthou](#)**  
Learn why SQLite Manager was created and find out what's next for this add-on.

<https://addons.mozilla.org/en-US/firefox/addon/sqlite-manager/>

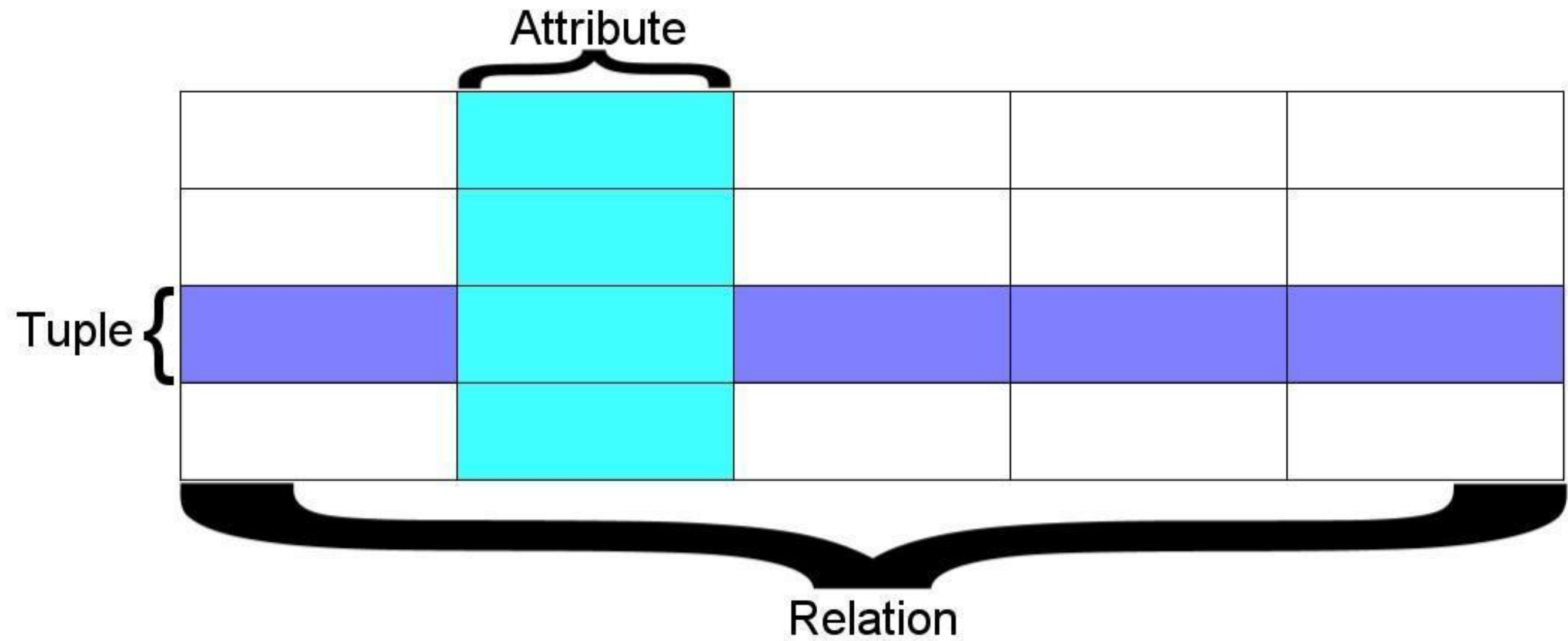
# Relational Databases

Relational databases model data by storing rows and columns in tables. The power of the relational database lies in its ability to efficiently retrieve data from those tables and in particular where there are multiple tables and the relationships between those tables involved in the query.

[http://en.wikipedia.org/wiki/Relational\\_database](http://en.wikipedia.org/wiki/Relational_database)

# Terminology

- **Database** - contains many tables
- **Relation (or table)** - contains tuples and attributes
- **Tuple (or row)** - a set of fields that generally represents an “object” like a person or a music track
- **Attribute (also column or field)** - one of possibly many elements of data corresponding to the object represented by the row



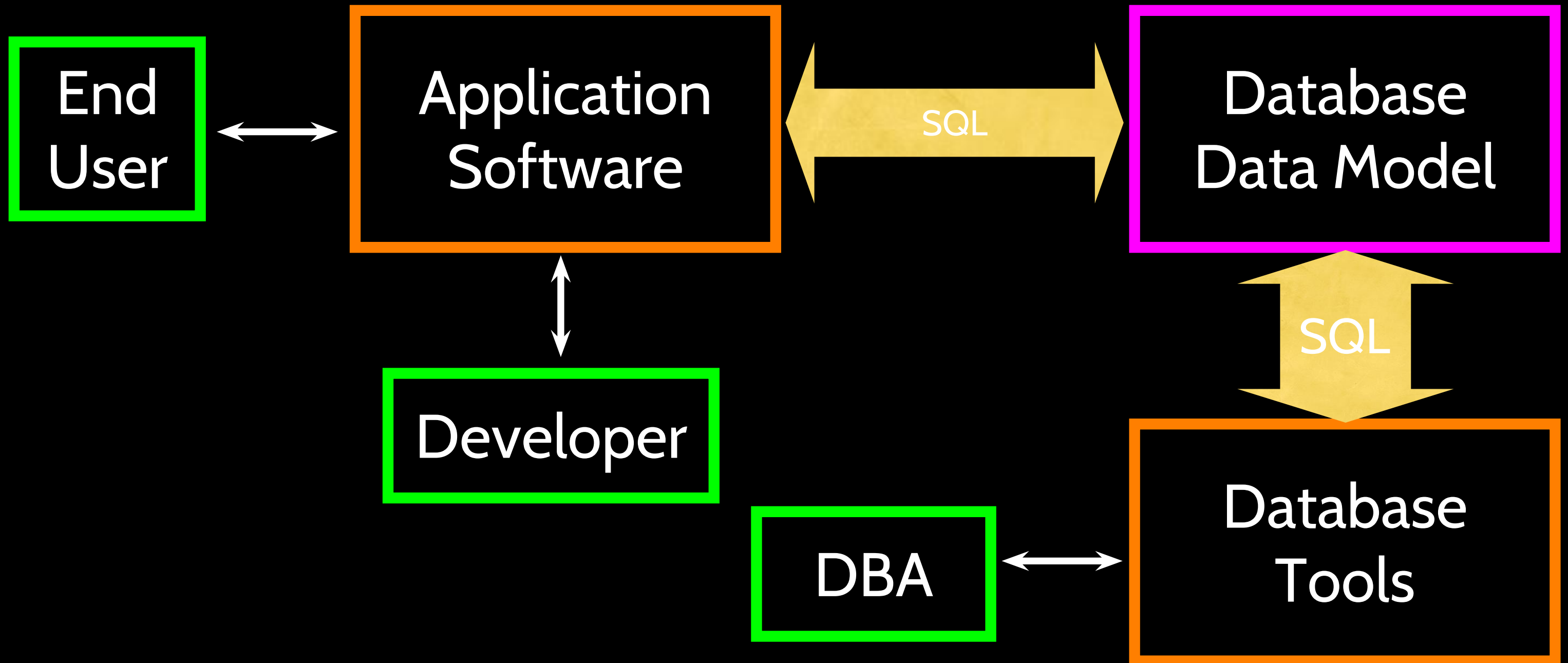
A relation is defined as a set of tuples that have the same attributes. A tuple usually represents an object and information about that object. Objects are typically physical objects or concepts. A relation is usually described as a table, which is organized into rows and columns. All the data referenced by an attribute are in the same domain and conform to the same constraints. (Wikipedia)

SI502 - Database				
New	Open	Save	Print	Import
Copy	Paste	Format	Undo	Redo
AutoSum	Sort A-Z	Sort Z-A	Gallery	Toolbox
Sheets	Charts	SmartArt Graphics	WordArt	
1	2	3	4	
A	B	C	D	
Columns / Attributes				
1	TITLE	RATING	LEN	Rows / Tuples
2	About to Rock	3	354	
3	Who Made Who	4	252	
4				
5				
6				
7				
8				
Tables / Relations				
Tracks Albums Artists Genres				

# Two Roles in Large Projects

- **Application Developer** – Builds the logic for the application, the look and feel of the application – monitors the application for problems
- **Database Administrator** – Monitors and adjusts the database as the program runs in production
- Often both people participate in the building of the “Data model”

# Application Structure





# Database Administrator (dba)

A database administrator (DBA) is a person responsible for the design, implementation, maintenance, and repair of an organization's database. The role includes the development and design of database strategies, monitoring and improving database performance and capacity, and planning for future expansion requirements. They may also plan, coordinate, and implement security measures to safeguard the database.

[http://en.wikipedia.org/wiki/Database\\_administrator](http://en.wikipedia.org/wiki/Database_administrator)

# Database Model

A database model or database schema is the structure or format of a database, described in a formal language supported by the database management system. In other words, a “database model” is the application of a data model when used in conjunction with a database management system.

[http://en.wikipedia.org/wiki/Database\\_model](http://en.wikipedia.org/wiki/Database_model)

# SQL

- **Structured Query Language** is the language we use to issue commands to the database
  - Create a table
  - Retrieve some data
  - Insert data
  - Delete data

<http://en.wikipedia.org/wiki/SQL>

# Common Database Systems

- Three Major Database Management Systems in wide use
  - **Oracle** - Large, commercial, enterprise-scale, very very tweakable
  - **MySQL** - Simpler but very fast and scalable - commercial open source
  - **SqlServer** - Very nice - from Microsoft (also Access)
- Many other smaller projects, free and open source
  - HSQL, **SQLite**, Postgress, ...

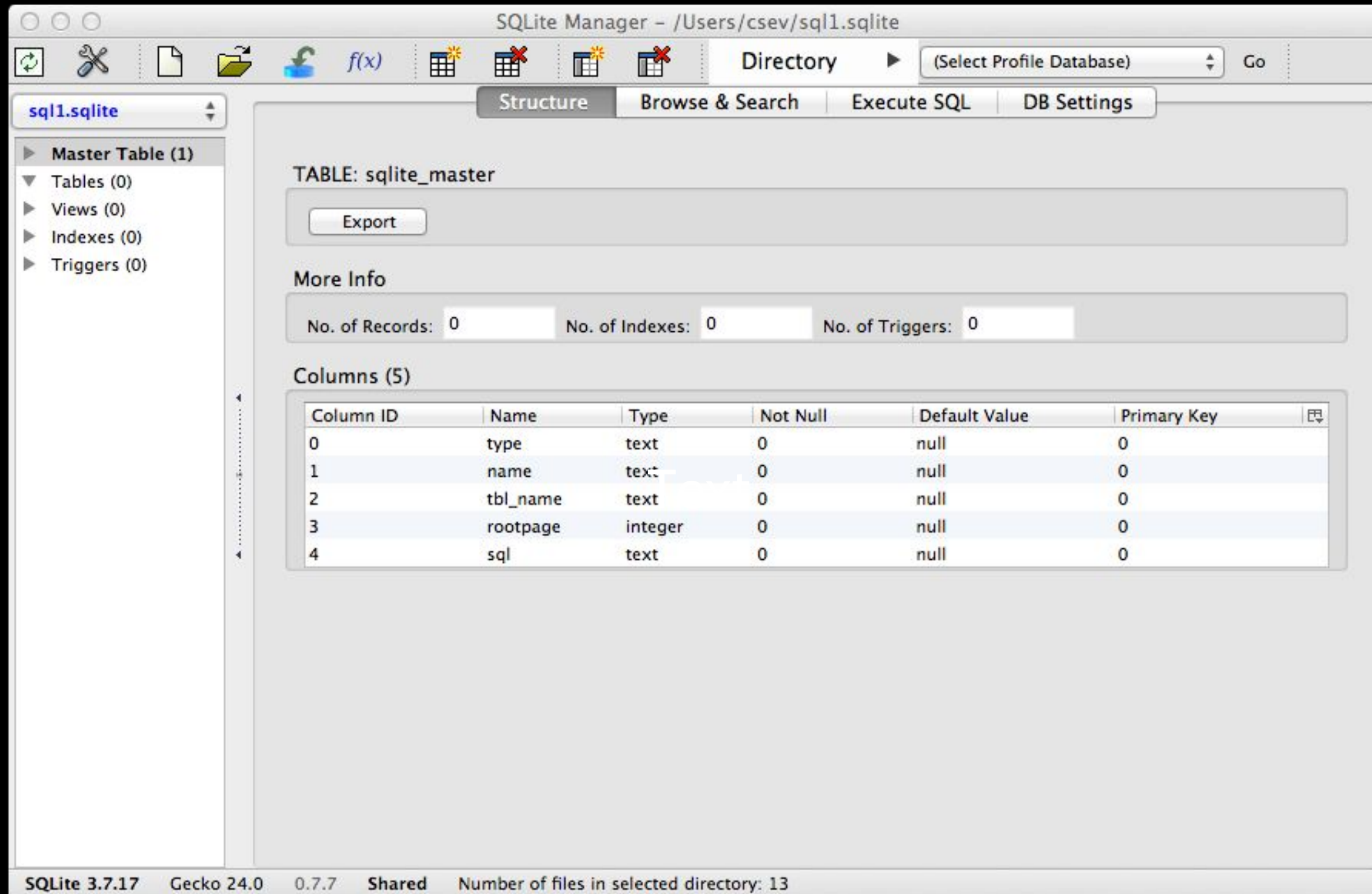
# SQLite Database Manager

- SQLite is a very popular database - it is free and fast and small
- We have a Firefox plugin to manipulate SQLite databases
  - <https://addons.mozilla.org/en-US/firefox/addon/sqlite-manager/>
- SQLite is embedded in Python and a number of other languages

# SQLite is in lots of software...

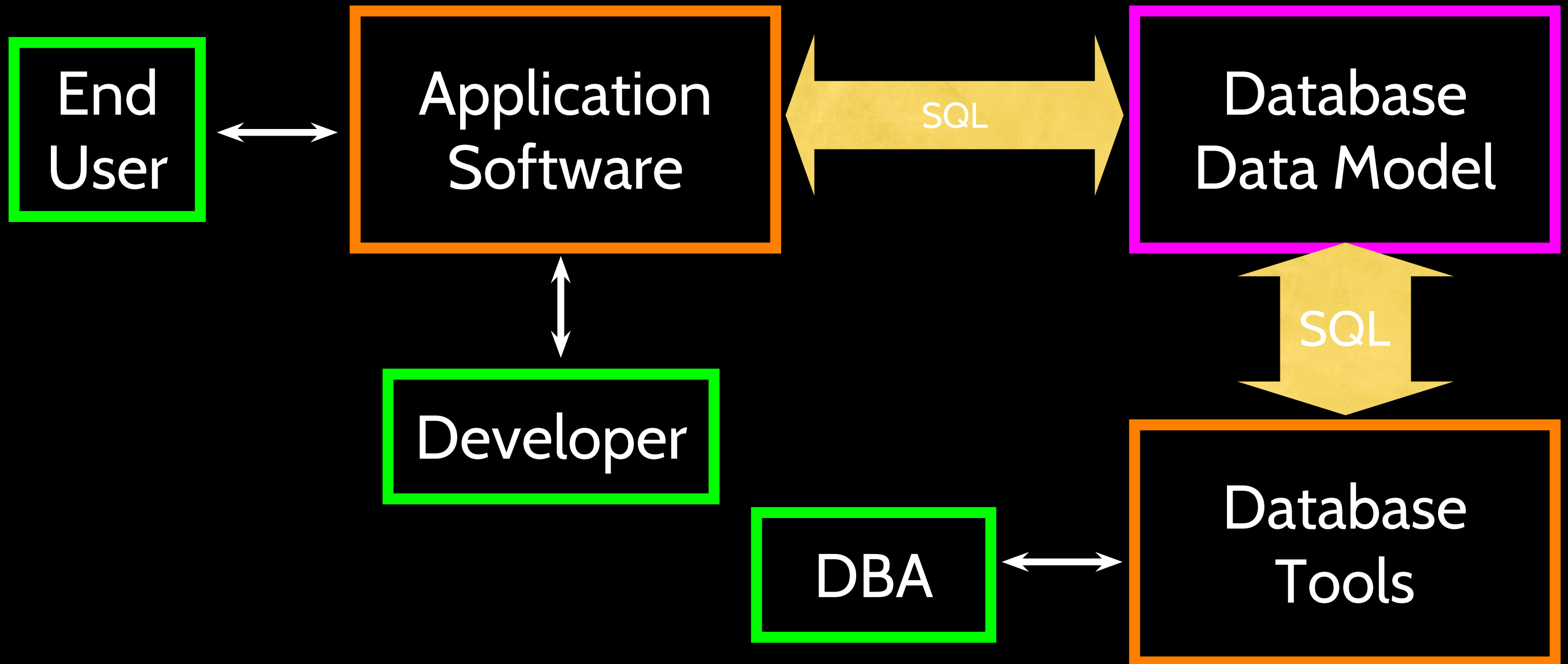
The Symbian logo, featuring the word "symbian" in a lowercase, sans-serif font with a small orange dot above the 'i'.The Python logo, consisting of two interlocking snakes (one blue, one yellow) followed by the word "python" in a lowercase, sans-serif font with a trademark symbol.The Skype logo, featuring the word "skype" in a lowercase, sans-serif font with a trademark symbol, set against a blue, cloud-like background.The Microsoft logo, featuring the word "Microsoft" in a bold, sans-serif font with a trademark symbol.The McAfee logo, featuring the word "McAfee" in a bold, sans-serif font with a trademark symbol.The Adobe logo, featuring a red "A" with a white outline and the word "Adobe" in a sans-serif font below it.The PHP logo, featuring the letters "php" in a lowercase, sans-serif font inside a blue oval.The Google logo, featuring the word "Google" in its multi-colored, sans-serif font with a trademark symbol.The Toshiba logo, featuring the word "TOSHIBA" in a bold, red, sans-serif font.The Sun Microsystems logo, featuring a stylized blue "Sun" logo with the word "Sun" in a script font and "microsystems" in a sans-serif font below it.

<http://www.sqlite.org/famous.html>



<https://addons.mozilla.org/en-US/firefox/addon/sqlite-manager/>

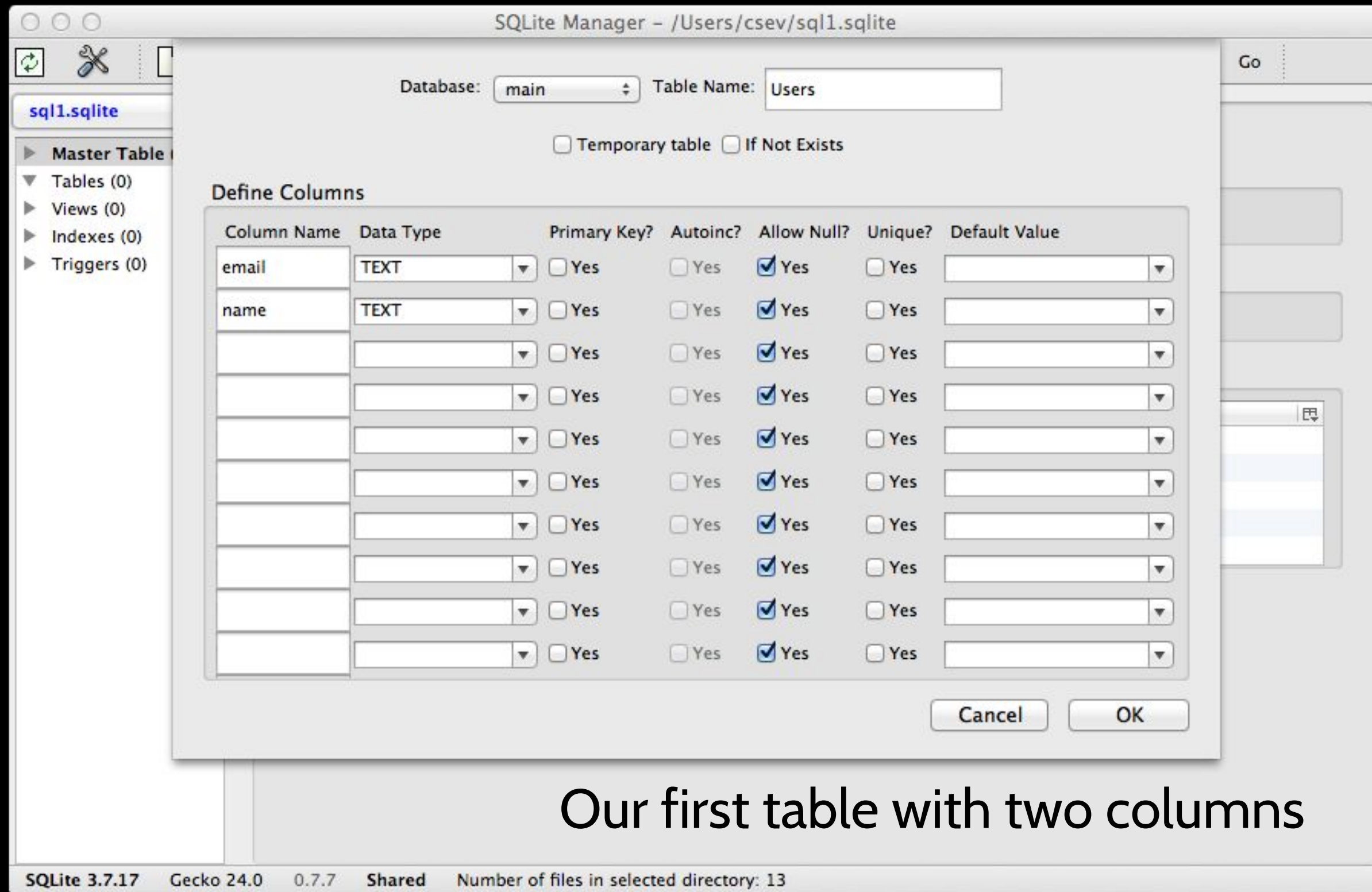
# Application Structure





# Start Simple - A Single Table

- Lets make a table of People - with a Name and an Email using the “wizard” user interface...



SQLite Manager - /Users/csev/sql1.sqlite

Directory (Select Profile Database) Go

Structure Browse & Search Execute SQL DB Settings

sql1.sqlite

- Master Table (1)
- Tables (1)
  - Users
- Views (0)
- Indexes (0)
- Triggers (0)

TABLE Users Search Show All Add Duplicate Edit Delete

rowid	email	name
1	csev@umich.edu	Chuck
2	olga@umich.edu	Olga
3	gab@umich.edu	Gab
4	gaurav@umich.edu	Gaurav

<< < 1 to 4 of 4 > >>

SQLite 3.7.17 Gecko 24.0 0.7.7 Shared Number of files in selected directory: 13 ET: 0 ms

## Our table with four rows

# SQL

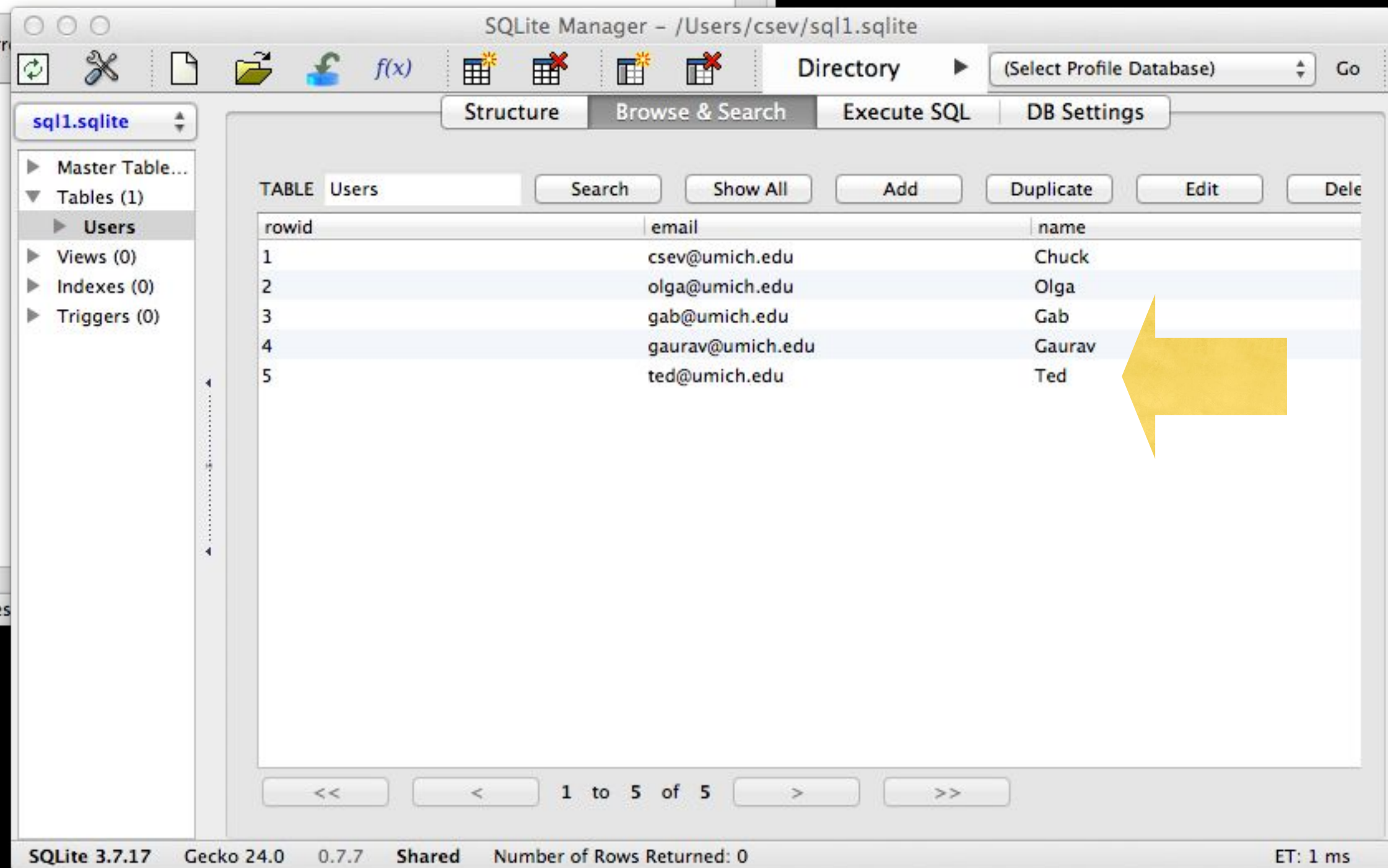
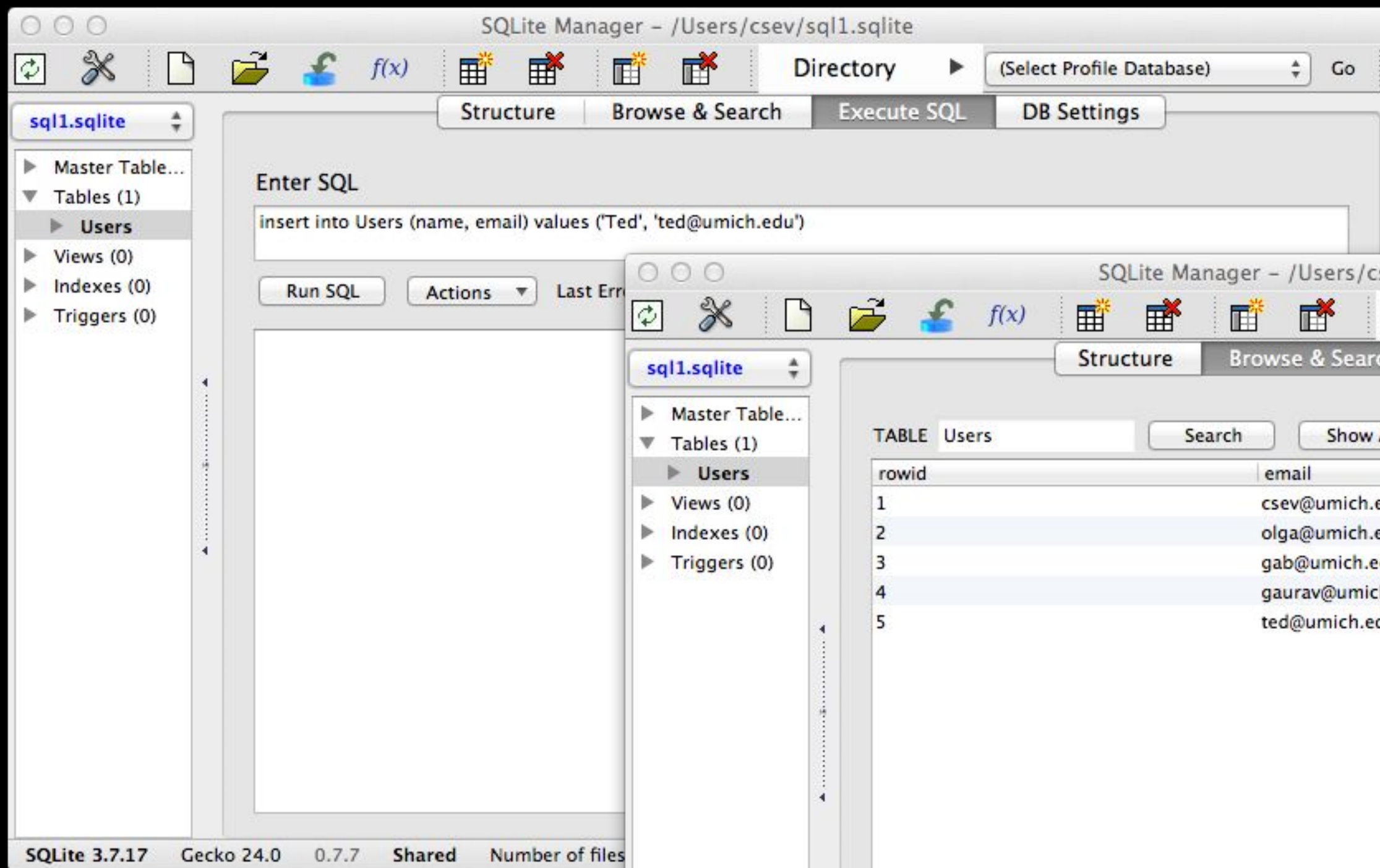
- **Structured Query Language** is the language we use to issue commands to the database
  - Create a table
  - Retrieve some data
  - Insert data
  - Delete data

<http://en.wikipedia.org/wiki/SQL>

# SQL Insert

- The Insert statement inserts a row into a table

```
INSERT INTO Users (name, email) VALUES ('Ted', 'ted@umich.edu')
```



# SQL Delete

- Deletes a row in a table based on a selection criteria

```
DELETE FROM Users WHERE email='ted@umich.edu'
```



SQLite Manager - /Users/csev/sql1.sqlite

Directory (Select Profile Database) Go

Structure Browse & Search Execute SQL DB Settings

sql1.sqlite

- Master Table...
- Tables (1)
  - Users
- Views (0)
- Indexes (0)
- Triggers (0)

Enter SQL

delete from Users where email='ted@umich.edu'

Run SQL Actions Last Error: not an error

SQLite 3.7.17 Gecko 24.0 0.7.7 Shared Number of Rows F

SQLite Manager - /Users/csev/sql1.sqlite

Directory (Select Profile Database) Go

Structure Browse & Search Execute SQL DB Settings

TABLE Users Search Show All Add Duplicate Edit Dele

rowid	email	name
1	csev@umich.edu	Chuck
2	olga@umich.edu	Olga
3	gab@umich.edu	Gab
4	gaurav@umich.edu	Gaurav

<< < 1 to 4 of 4 > >>

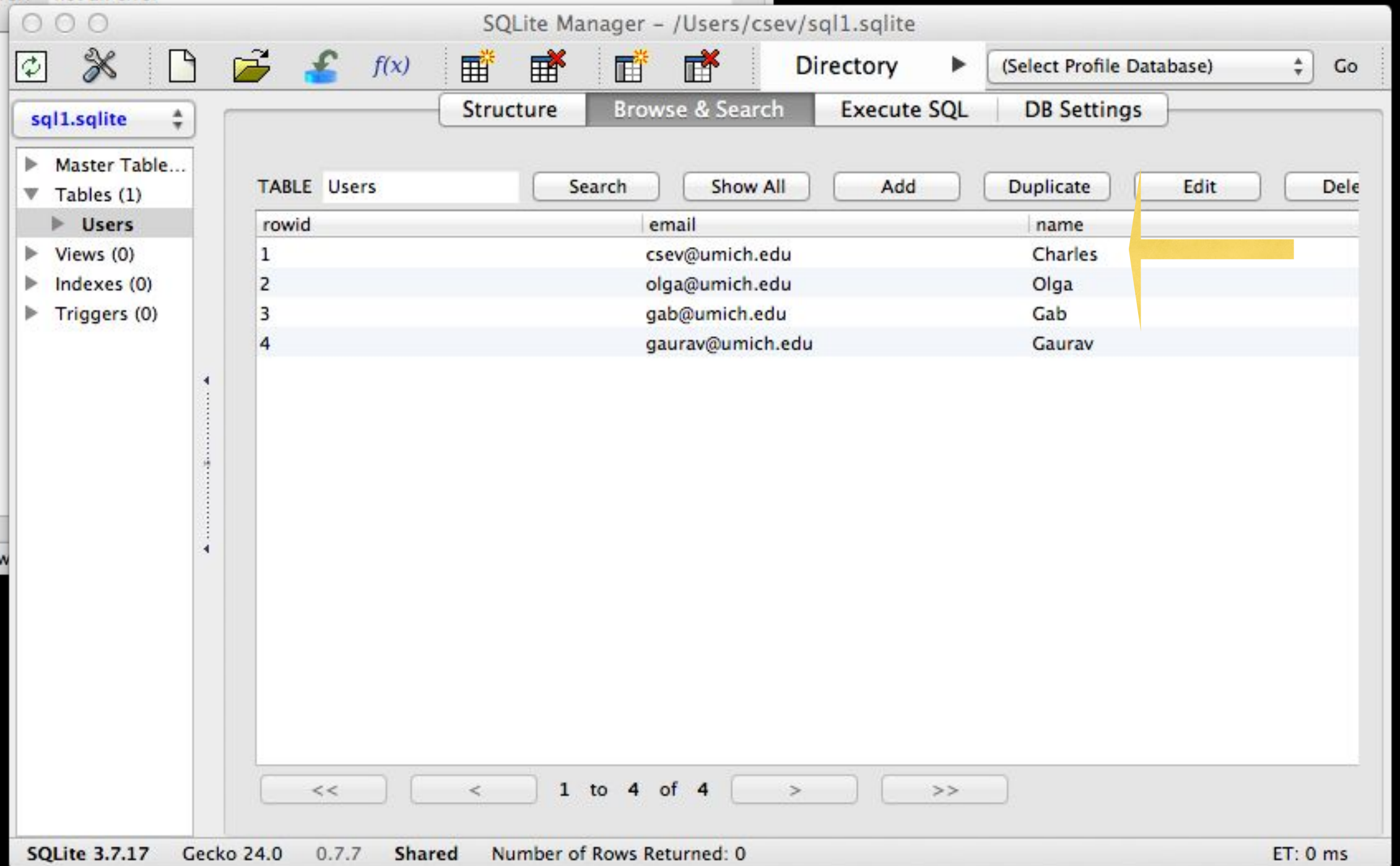
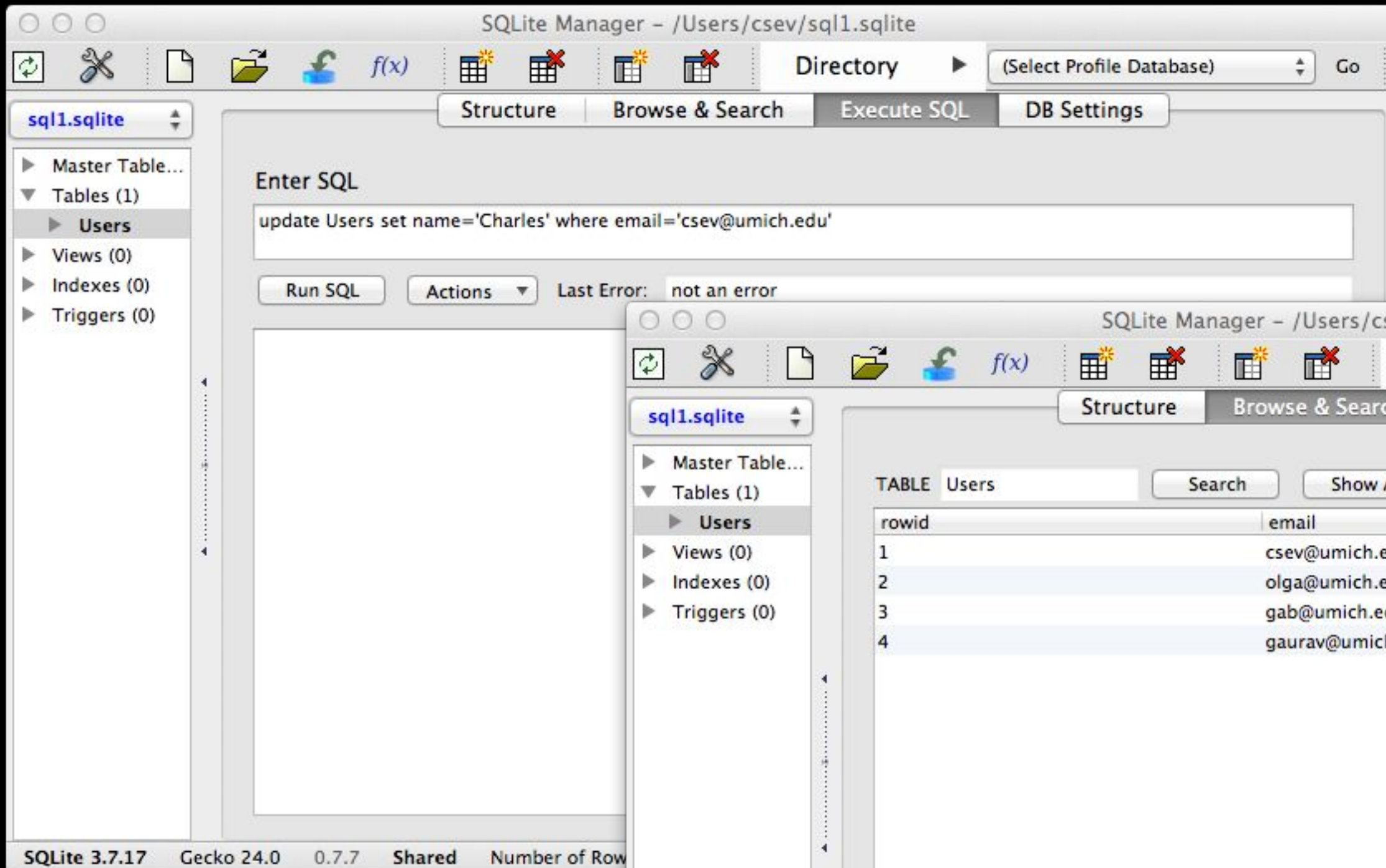
SQLite 3.7.17 Gecko 24.0 0.7.7 Shared Number of Rows Returned: 0 ET: 0 ms



# SQL: Update

- Allows the updating of a field with a where clause

```
UPDATE Users SET name='Charles' WHERE email='csev@umich.edu'
```



# Retrieving Records: Select

- The select statement retrieves a group of records - you can either retrieve all the records or a subset of the records with a WHERE clause

```
SELECT * FROM Users
```

```
SELECT * FROM Users WHERE email='csev@umich.edu'
```

SQLite Manager - /Users/csev/sql1.sqlite

Directory (Select Profile Database) Go

Structure Browse & Search Execute SQL DB Settings

sql1.sqlite

- Master Table...
- Tables (1)
  - Users
- Views (0)
- Indexes (0)
- Triggers (0)

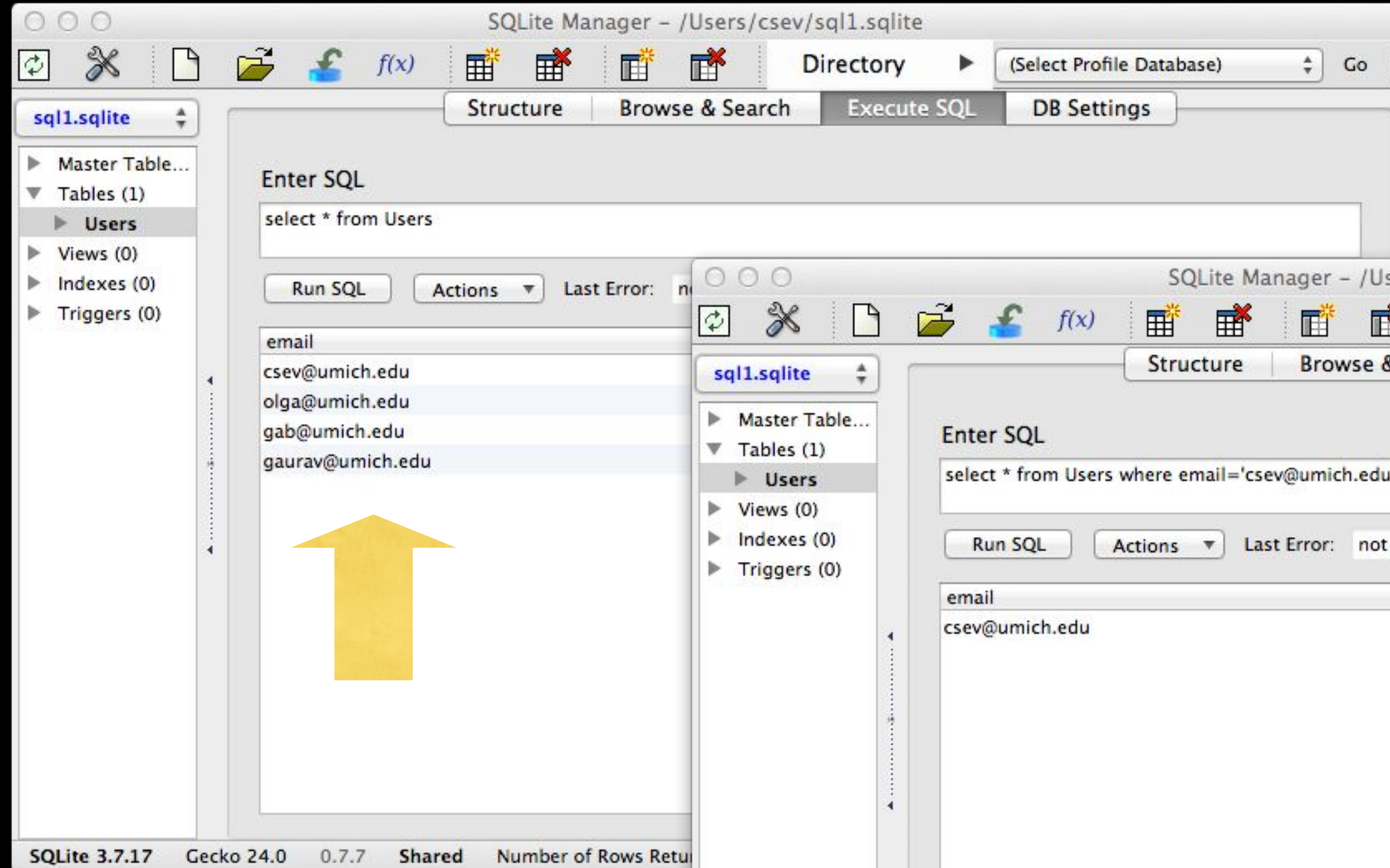
Enter SQL

select \* from Users

Run SQL Actions Last Error: not an error

email
csev@umich.edu
olga@umich.edu
gab@umich.edu
gaurav@umich.edu

SQLite 3.7.17 Gecko 24.0 0.7.7 Shared Number of Rows Returned: 4



SQLite Manager - /Users/csev/sql1.sqlite

Directory (Select Profile Database) Go

Structure Browse & Search Execute SQL DB Settings

sql1.sqlite

- Master Table...
- Tables (1)
  - Users
- Views (0)
- Indexes (0)
- Triggers (0)

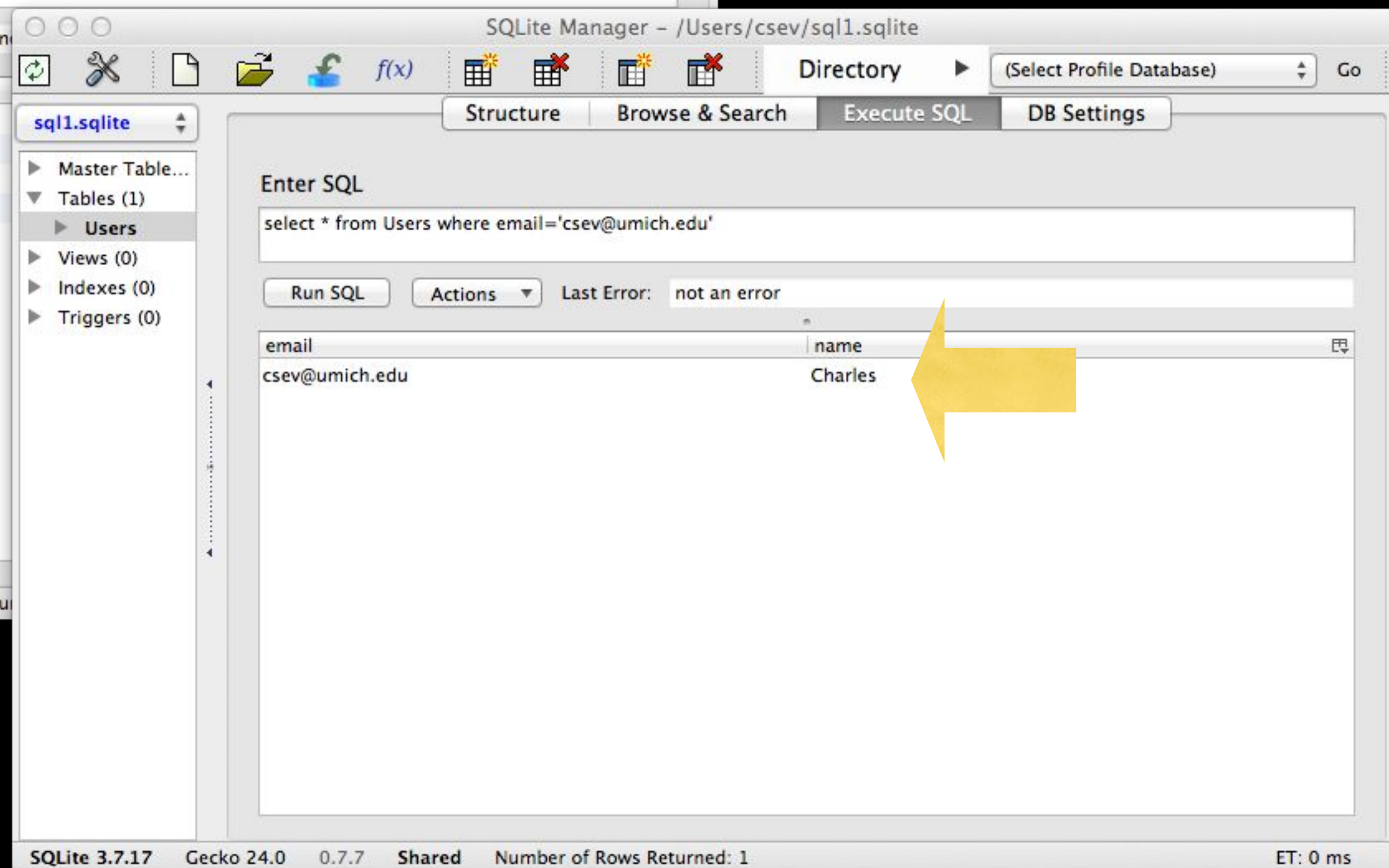
Enter SQL

select \* from Users where email='csev@umich.edu'

Run SQL Actions Last Error: not an error

email	name
csev@umich.edu	Charles

SQLite 3.7.17 Gecko 24.0 0.7.7 Shared Number of Rows Returned: 1 ET: 0 ms



# Sorting with ORDER BY

- You can add an **ORDER BY** clause to **SELECT** statements to get the results sorted in ascending or descending order

```
SELECT * FROM Users ORDER BY email
```

```
SELECT * FROM Users ORDER BY name
```



SQLite Manager - /Users/csev/sql1.sqlite

Directory (Select Profile Database) Go

Structure Browse & Search Execute SQL DB Settings

sql1.sqlite

- Master Table...
- Tables (1)
  - Users
- Views (0)
- Indexes (0)
- Triggers (0)

Enter SQL

select \* from Users order by email

Run SQL Actions Last Error: not an error

email	name
csev@umich.edu	Charles
gab@umich.edu	Gab
gaurav@umich.edu	Gaurav
olga@umich.edu	Olga

SQLite 3.7.17 Gecko 24.0 0.7.7 Shared Number of Rows Returned: 4 ET: 0 ms

# SQL Summary

```
insert into Users (name, email) values ('Ted', 'ted@umich.edu')
```

```
delete from Users where email='ted@umich.edu'
```

```
update Users set name="Charles" where email='csev@umich.edu'
```

```
select * from Users
```

```
select * from Users where email='csev@umich.edu'
```

```
select * from Users order by email
```

# This is not too exciting (so far)

- Tables pretty much look like big fast programmable spreadsheets with rows, columns, and commands
- The power comes when we have more than one table and we can exploit the relationships between the tables

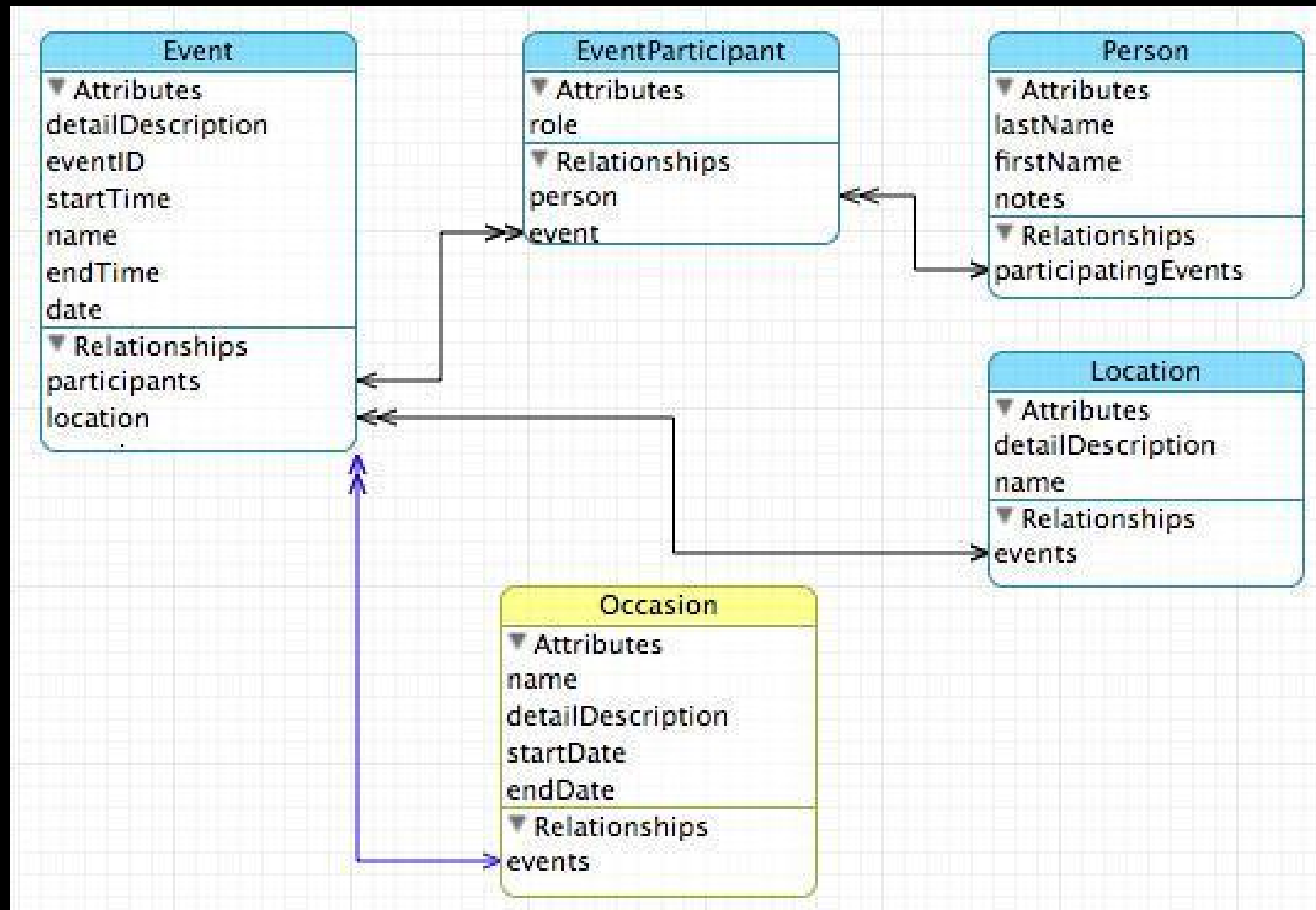


# Complex Data Models and Relationships

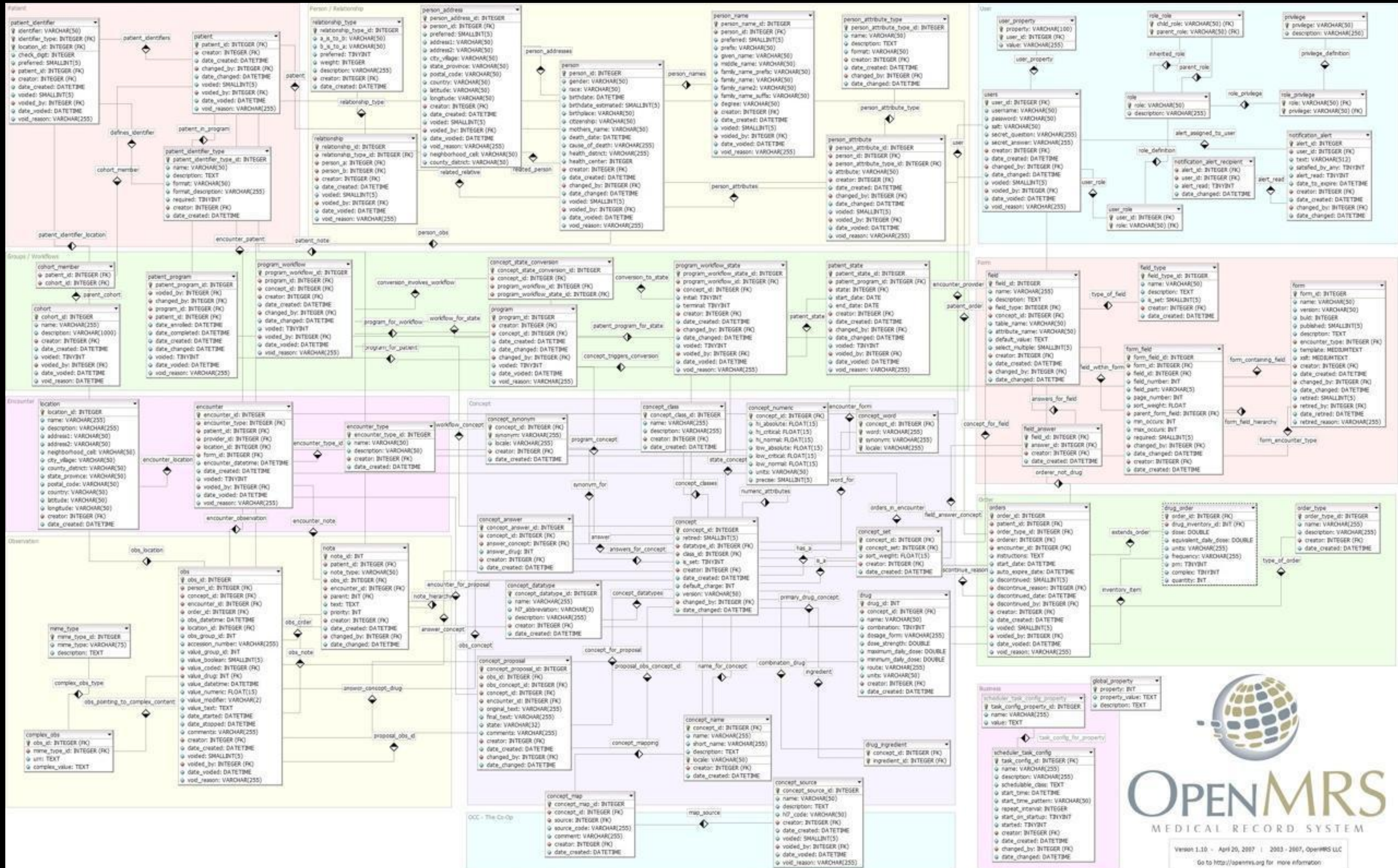
[http://en.wikipedia.org/wiki/Relational\\_model](http://en.wikipedia.org/wiki/Relational_model)

# Database Design

- Database design is an **art form** of its own with particular skills and experience
- Our goal is to avoid the really bad mistakes and design clean and easily understood databases
- Others may performance tune things later
- Database design starts with a picture...







**OPENMRS**  
MEDICAL RECORD SYSTEM

Version 1.10 - April 20, 2007 | 2003 - 2007, OpenMRS LLC  
Go to <http://openmrs.org> for more information

# Building a Data Model

- Drawing a picture of the data objects for our application and then figuring out how to represent the objects and their relationships
- Basic Rule: Don't put the same string data in twice - use a relationship instead
- When there is one thing in the “real world” there should be one copy of that thing in the database



Track	Len	Artist	Album	Genre	Rating	Count
-------	-----	--------	-------	-------	--------	-------

✓ Hells Bells	5:13	AC/DC	Who Made Who	Rock	★★★★★	61
✓ Shake Your Foundations	3:54	AC/DC	Who Made Who	Rock	★★★★★	70
✓ Chase the Ace	3:01	AC/DC	Who Made Who	Rock		56
✓ For Those About To Rock (We ...	5:54	AC/DC	Who Made Who	Rock	★★★★★	61
✓ Dúlamán	3:43	Altan	Natural Wonders M...	New Age		31
✓ Rode Across the Desert	4:10	America	Greatest Hits	Easy Listen...	★★★★★	23
✓ Now You Are Gone	3:08	America	Greatest Hits	Easy Listen...	★★★★★	18
✓ Tin Man	3:30	America	Greatest Hits	Easy Listen...	★★★★★	23
✓ Sister Golden Hair	3:22	America	Greatest Hits	Easy Listen...	★★★★★	24
✓ Track 01	4:22	Billy Price	Danger Zone	Blues/R&B	★★★★★	26
✓ Track 02	2:45	Billy Price	Danger Zone	Blues/R&B	★★★★★	18
✓ Track 03	3:26	Billy Price	Danger Zone	Blues/R&B	★★★★★	22
✓ Track 04	4:17	Billy Price	Danger Zone	Blues/R&B	★★★★★	18
✓ Track 05	3:50	Billy Price	Danger Zone	Blues/R&B	★★★★★	21
✓ War Pigs/Luke's Wall	7:58	Black Sabbath	Paranoid	Metal	★★★★★	25
✓ Paranoid	2:53	Black Sabbath	Paranoid	Metal	★★★★★	22
✓ Planet Caravan	4:35	Black Sabbath	Paranoid	Metal	★★★★★	25
✓ Iron Man	5:59	Black Sabbath	Paranoid	Metal	★★★★★	26
✓ Electric Funeral	4:53	Black Sabbath	Paranoid	Metal	★★★★★	22
✓ Hand of Doom	7:10	Black Sabbath	Paranoid	Metal	★★★★★	23
✓ Rat Salad	2:30	Black Sabbath	Paranoid	Metal	★★★★★	31
✓ Jack the Stripper/Fairies Wear ...	6:14	Black Sabbath	Paranoid	Metal	★★★★★	24
✓ Bomb Squad (TECH)	3:28	Brent	Brent's Album			1
✓ clay techno	4:36	Brent	Brent's Album			2
✓ Heavy	3:08	Brent	Brent's Album			1
✓ Hi metal man	4:20	Brent	Brent's Album			1
✓ Mistro	2:58	Brent	Brent's Album			1

# For each “piece of info”...

- Is the column an object or an attribute of another object?
- Once we define objects, we need to define the relationships between objects.

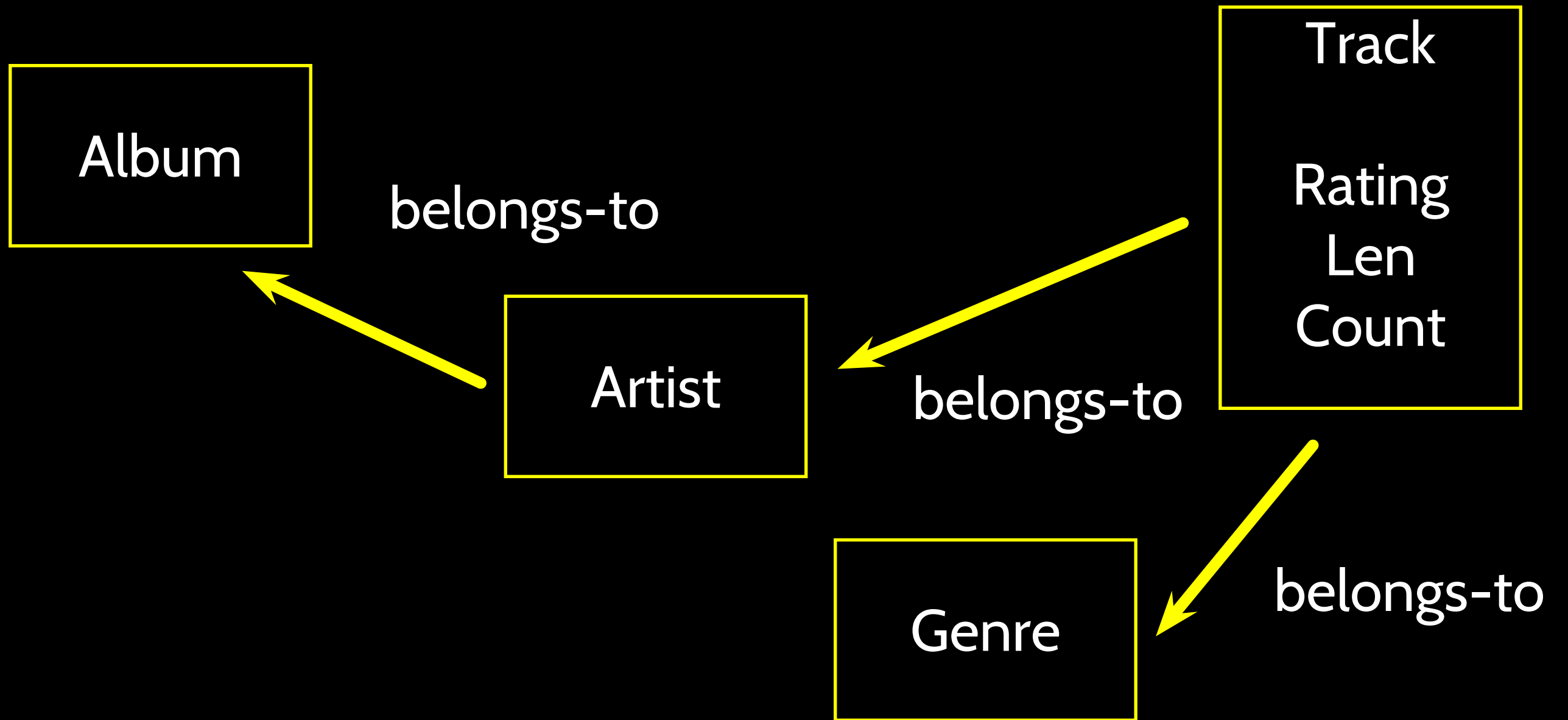
Len Album  
Genre

Artist Rating

Track Count

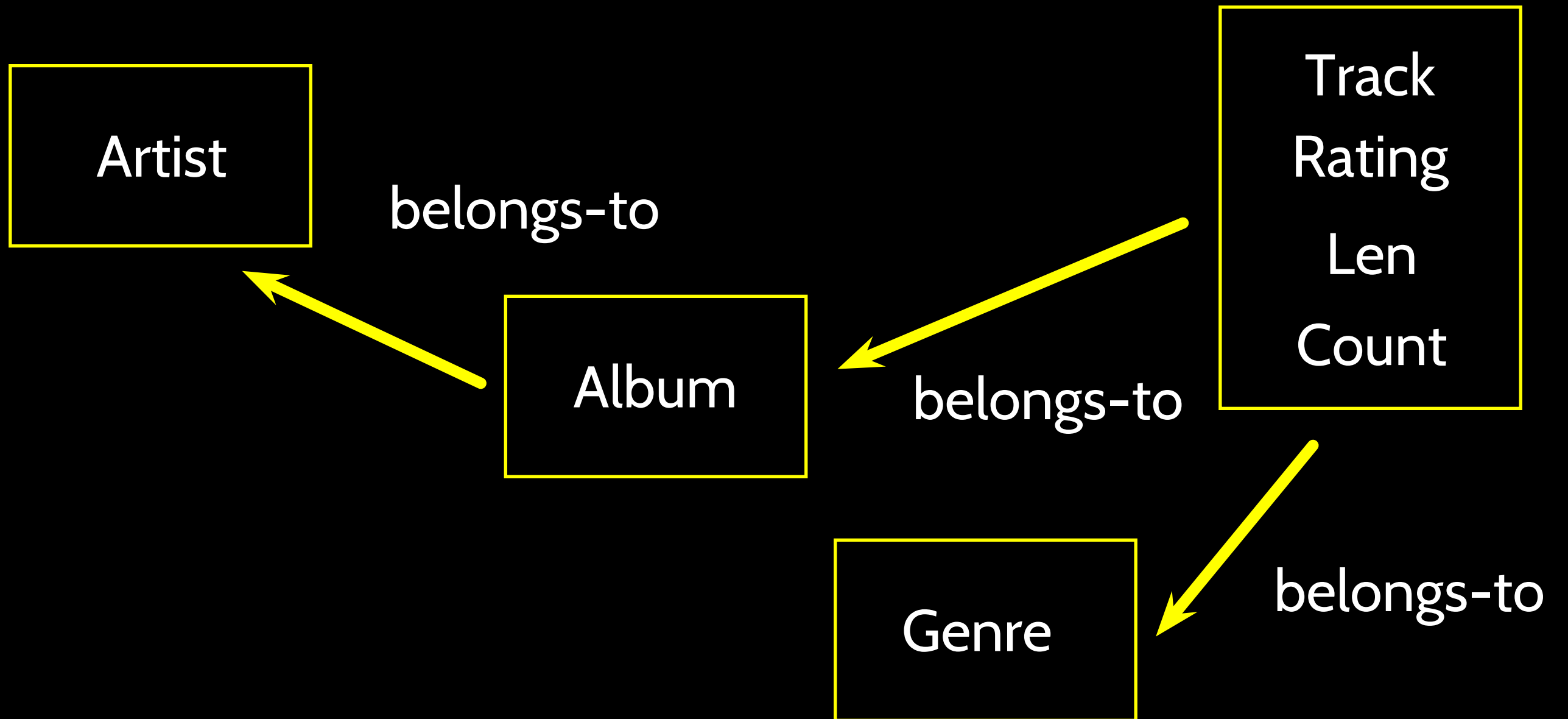
<input checked="" type="checkbox"/> Hells Bells	5:13	AC/DC	Who Made Who	Rock	★★★★★	61
<input checked="" type="checkbox"/> Shake Your Foundations	3:54	AC/DC	Who Made Who	Rock	★★★★★	70
<input checked="" type="checkbox"/> Chase the Ace	3:01	AC/DC	Who Made Who	Rock		56
<input checked="" type="checkbox"/> For Those About To Rock (We ...	5:54	AC/DC	Who Made Who	Rock	★★★★★	61
<input checked="" type="checkbox"/> Dúlamán	3:43	Altan	Natural Wonders M...	New Age		31
<input checked="" type="checkbox"/> Rode Across the Desert	4:10	America	Greatest Hits	Easy Listen...	★★★★★	23
<input checked="" type="checkbox"/> Now You Are Gone	3:08	America	Greatest Hits	Easy Listen...	★★★★★	18
<input checked="" type="checkbox"/> Tip Man	3:20	America	Greatest Hits	Easy Listen...	★★★★★	22

Track  
Artist  
Album  
Genre  
Rating  
Len  
Count



<input checked="" type="checkbox"/> Hells Bells	5:13	AC/DC	Who Made Who	Rock	★★★★★	61
<input checked="" type="checkbox"/> Shake Your Foundations	3:54	AC/DC	Who Made Who	Rock	★★★★★	70
<input checked="" type="checkbox"/> Chase the Ace	3:01	AC/DC	Who Made Who	Rock		56
<input checked="" type="checkbox"/> For Those About To Rock (We ...	5:54	AC/DC	Who Made Who	Rock	★★★★★	61
<input checked="" type="checkbox"/> Dúlamán	3:43	Altan	Natural Wonders M...	New Age		31
<input checked="" type="checkbox"/> Rode Across the Desert	4:10	America	Greatest Hits	Easy Listen...	★★★★★	23
<input checked="" type="checkbox"/> Now You Are Gone	3:08	America	Greatest Hits	Easy Listen...	★★★★★	18
<input checked="" type="checkbox"/> Tip Man	3:20	America	Greatest Hits	Easy Listen...	★★★★★	22





<input checked="" type="checkbox"/> Hells Bells	5:13	AC/DC	Who Made Who	Rock	★★★★★	61
<input checked="" type="checkbox"/> Shake Your Foundations	3:54	AC/DC	Who Made Who	Rock	★★★★★	70
<input checked="" type="checkbox"/> Chase the Ace	3:01	AC/DC	Who Made Who	Rock		56
<input checked="" type="checkbox"/> For Those About To Rock (We ...	5:54	AC/DC	Who Made Who	Rock	★★★★★	61
<input checked="" type="checkbox"/> Dúlamán	3:43	Altan	Natural Wonders M...	New Age		31
<input checked="" type="checkbox"/> Rode Across the Desert	4:10	America	Greatest Hits	Easy Listen...	★★★★★	23
<input checked="" type="checkbox"/> Now You Are Gone	3:08	America	Greatest Hits	Easy Listen...	★★★★★	18
<input checked="" type="checkbox"/> Tip Man	3:20	America	Greatest Hits	Easy Listen...	★★★★★	22

# Representing Relationships in a Database

<input checked="" type="checkbox"/> Hells Bells	5:13	AC/DC	Who Made Who	Rock	★★★★★	61
<input checked="" type="checkbox"/> Shake Your Foundations	3:54	AC/DC	Who Made Who	Rock	★★★★★	70
<input checked="" type="checkbox"/> Chase the Ace	3:01	AC/DC	Who Made Who	Rock		56
<input checked="" type="checkbox"/> For Those About To Rock (We ...)	5:54	AC/DC	Who Made Who	Rock	★★★★★	61
<input checked="" type="checkbox"/> Dúlamán	3:43	Altan	Natural Wonders M...	New Age		31
<input checked="" type="checkbox"/> Rode Across the Desert	4:10	America	Greatest Hits	Easy Listen...	★★★★★	23
<input checked="" type="checkbox"/> Now You Are Gone	3:08	America	Greatest Hits	Easy Listen...	★★★★★	18
<input checked="" type="checkbox"/> Tip Man	3:20	America	Greatest Hits	Easy Listen...	★★★★★	22

We want to keep track of which band is the “creator” of each music track...

What album does this song “belong to”??

Which album is this song related to?

# Database Normalization (3NF)

- There is \*tons\* of database theory - way too much to understand without excessive predicate calculus
  - Do not replicate data - reference data - point at data
  - Use integers for keys and for references
  - Add a special “key” column to each table which we will make references to. By convention, many programmers call this column “id”

[http://en.wikipedia.org/wiki/Database\\_normalization](http://en.wikipedia.org/wiki/Database_normalization)

# Integer Reference Pattern

We use integers to reference rows in another table

Use 'control + ;' to execute the query.	
id	
1	Led Zeppelin
2	AC/DC

Artist

id	artist_id	title
1	2	Who Made Who
2	1	IV

Album

# Keys

Finding our way around....

# Three Kinds of Keys

- **Primary key** - generally an integer auto-increment field
- **Logical key** - What the outside world uses for lookup
- **Foreign key** - generally an integer key pointing to a row in another table





# Primary Key Rules

## Best practices

- Never use your **logical key** as the **primary key**
- **Logical keys** can and do change, albeit slowly
- **Relationships** that are based on matching string fields are far less efficient than integers performance-wise

User

id

login

password

name

email

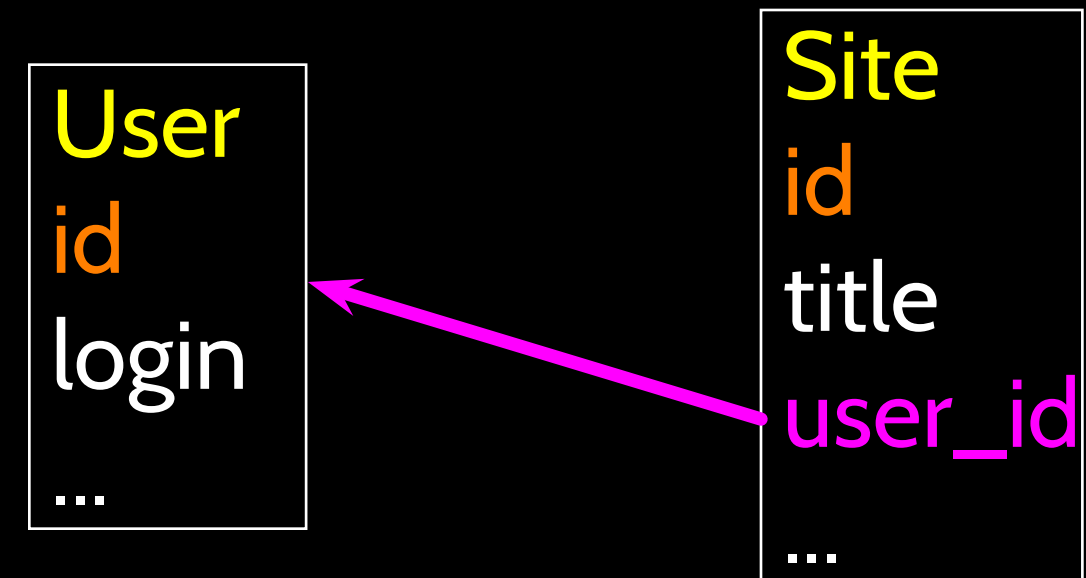
created\_at

modified\_at

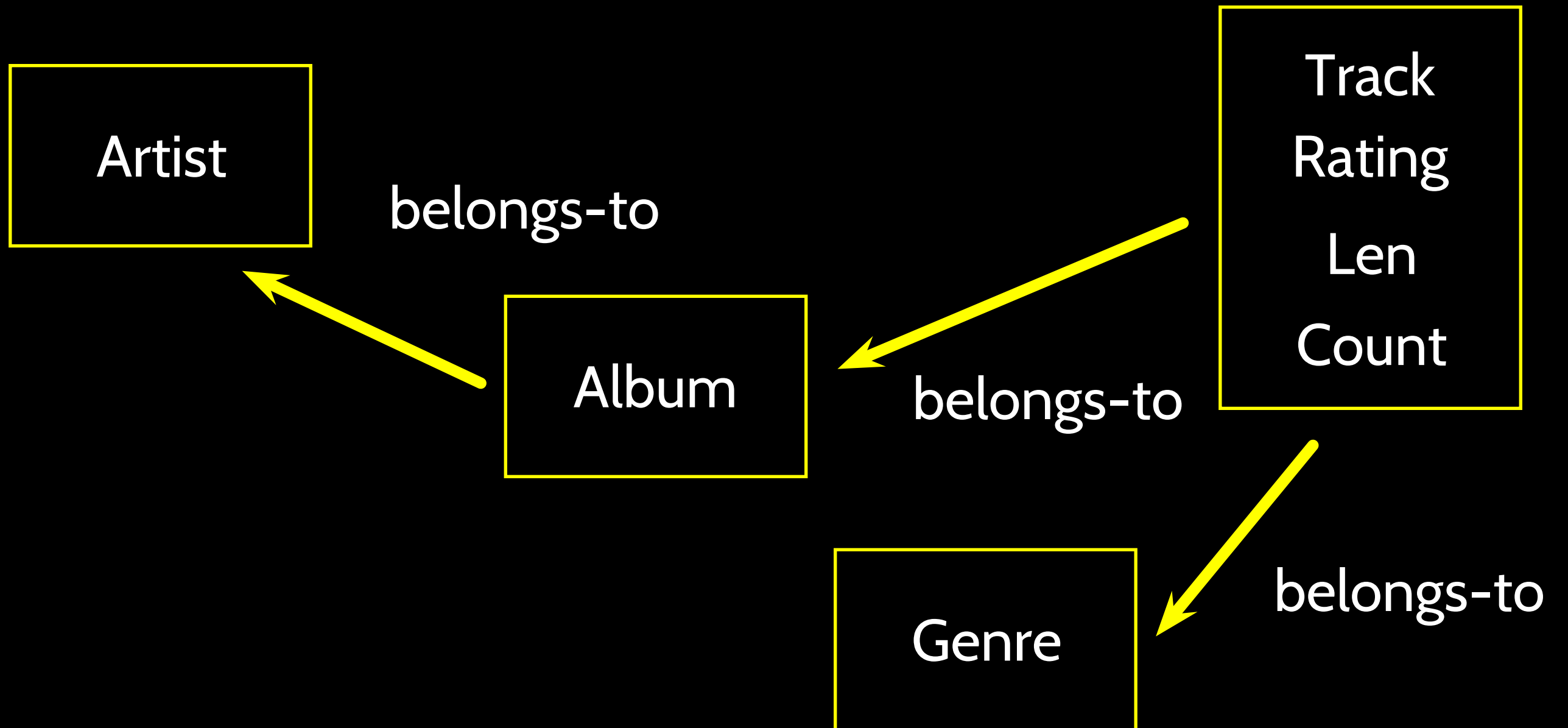
login\_at

# Foreign Keys

- A **foreign key** is when a table has a column that contains a key which points to the **primary key** of another table.
- When all primary keys are integers, then all foreign keys are integers - this is good - very good
- If you use strings as foreign keys - you show yourself to be an uncultured swine



# Relationship Building (in tables)



<input checked="" type="checkbox"/> Hells Bells	5:13	AC/DC	Who Made Who	Rock	★★★★★	61
<input checked="" type="checkbox"/> Shake Your Foundations	3:54	AC/DC	Who Made Who	Rock	★★★★★	70
<input checked="" type="checkbox"/> Chase the Ace	3:01	AC/DC	Who Made Who	Rock		56
<input checked="" type="checkbox"/> For Those About To Rock (We ...	5:54	AC/DC	Who Made Who	Rock	★★★★★	61
<input checked="" type="checkbox"/> Dúlamán	3:43	Altan	Natural Wonders M...	New Age		31
<input checked="" type="checkbox"/> Rode Across the Desert	4:10	America	Greatest Hits	Easy Listen...	★★★★★	23
<input checked="" type="checkbox"/> Now You Are Gone	3:08	America	Greatest Hits	Easy Listen...	★★★★★	18
<input checked="" type="checkbox"/> Tip Man	3:20	America	Greatest Hits	Easy Listen...	★★★★★	22



Table  
Primary key  
Logical key  
Foreign key

Album
id
title

Track
id
title
rating
len
count
album_id

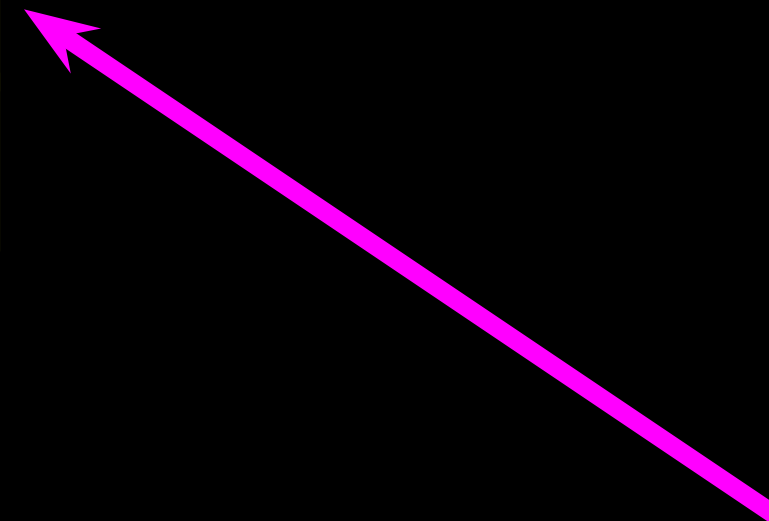
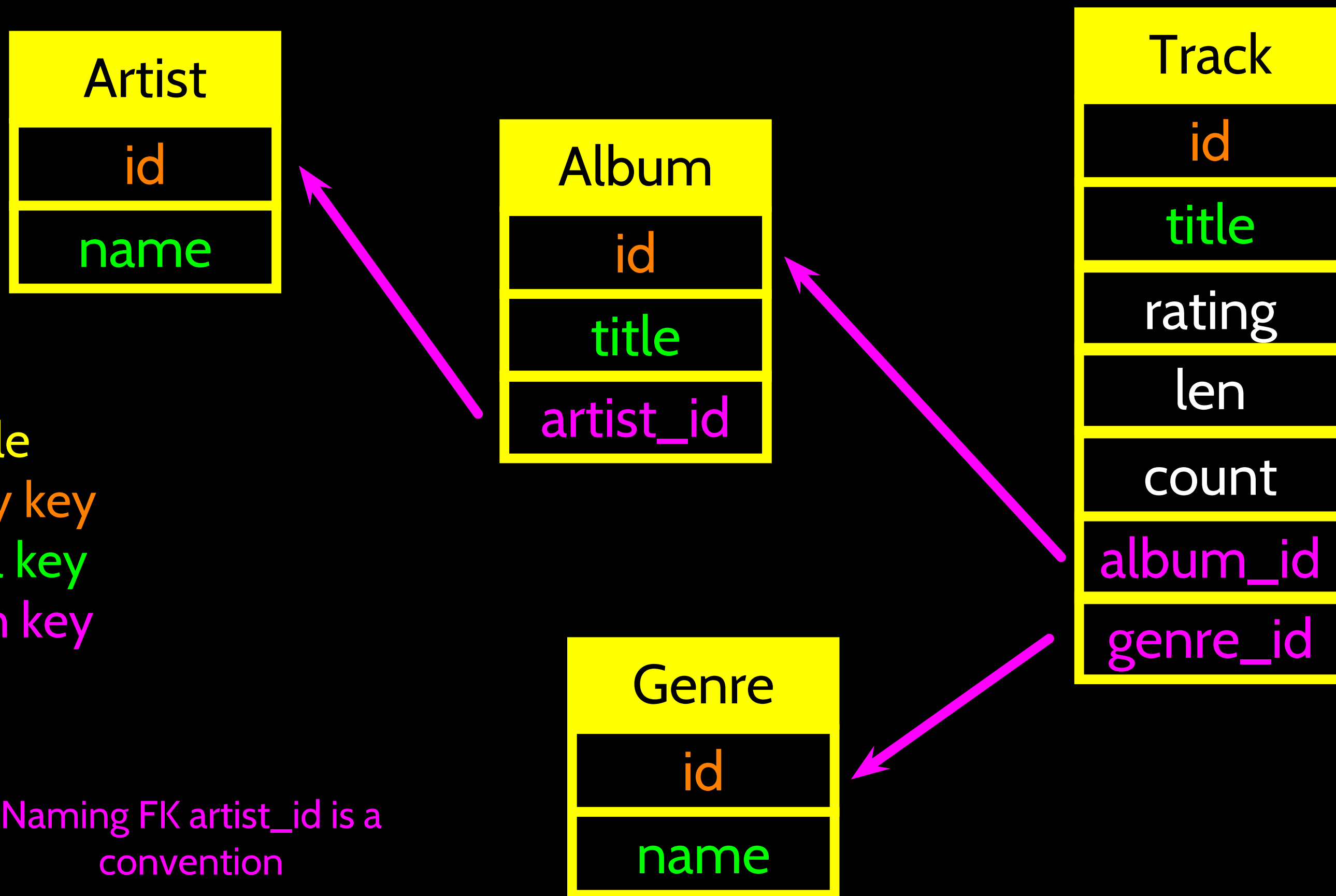


Table  
Primary key  
Logical key  
Foreign key



Naming FK artist\_id is a convention

Database: main

Table Name: Artist

☐ Temporary table ☐ If Not Exists

## Define Columns

Column Name	Data Type	Primary Key?	Autoinc?	Allow Null?	Unique?	Default Value
id	INTEGER	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	
name	TEXT	<input type="checkbox"/> Yes	<input type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> Yes	
		<input type="checkbox"/> Yes	<input type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> Yes	
		<input type="checkbox"/> Yes	<input type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> Yes	
		<input type="checkbox"/> Yes	<input type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> Yes	
		<input type="checkbox"/> Yes	<input type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> Yes	
		<input type="checkbox"/> Yes	<input type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> Yes	
		<input type="checkbox"/> Yes	<input type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> Yes	
		<input type="checkbox"/> Yes	<input type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> Yes	
		<input type="checkbox"/> Yes	<input type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> Yes	

Cancel

OK



SQLite Manager - /Users/csev/sql1.sqlite

Directory (Select Profile Database) Go

Structure Browse & Search Execute SQL DB Settings

sql1.sqlite

- Master Table (1)
- Tables (5)
  - Album
  - Artist
  - Genre
  - Track
  - sqlite\_sequence
- Views (0)
- Indexes (4)
- Triggers (0)

TABLE: Genre

Drop Empty Rename Reindex Copy Export

Create statement

CREATE TABLE "Genre" ("id" INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL UNIQUE , "name" TEXT)

More Info

No. of Records: 0 No. of Indexes: 1 No. of Triggers: 0

Columns (2)

Column ID	Name	Type	Not Null	Default Value	Primary Key
0	id	INTEGER	1	null	1
1	name	TEXT	0	null	0

Name

Type

Not Null

Default

Add Column

SQLite 3.7.17 Gecko 24.0 0.7.7 Shared Number of Rows Returned: 0 ET: 0 ms

SQLite Manager - /Users/csev/sql1.sqlite

sql1.sqlite

▶ Master Table (1)

▼ Tables (5)

▶ Album

▶ Artist

▶ Genre

▶ Track

▶ sqlite\_sequence

▶ Views (0)

▶ Indexes (4)

▶ Triggers (0)

Structure

Browse & Search

Execute SQL

DB Settings

Directory

(Select Profile Database)

Go

TABLE: Album

Drop

Empty

Rename

Reindex

Copy

Export

Create statement

CREATE TABLE "Album" ("id" INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL UNIQUE , artist\_id INTEGER,"title" TEXT)

More Info

No. of Records: 0

No. of Indexes: 1

No. of Triggers: 0

Columns (3)

Column ID	Name	Type	Not Null	Default Value	Primary Key
0	id	INTEGER	1	null	1
1	artist_id	INTEGER	0	null	0
2	title	TEXT	0	null	0

Name

Type

Not Null

Default

Add Column

SQLite 3.7.17   Gecko 24.0   0.7.7   Shared   Number of Rows Returned: 0   ET: 0 ms

SQLite Manager - /Users/csev/sql1.sqlite

Refresh

Cut

New

Open

Save

Print

Find

Find & Replace

Undo

Redo

Close

Close All

Quit

Directory

(Select Profile Database)

Go

sql1.sqlite

Master Table (1)

Tables (5)

Album

Artist

Genre

Track

sqlite\_sequence

Views (0)

Indexes (4)

Triggers (0)

Structure

Browse & Search

Execute SQL

DB Settings

TABLE: Track

Drop

Empty

Rename

Reindex

Copy

Export

Create statement

CREATE TABLE "Track" ("id" INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL UNIQUE , album\_id INTEGER, genre\_id INTEGER, len INTEGER, rating INTEGER, "title" TEXT, "count" INTEGER)

More Info

No. of Records: 0No. of Indexes: 1No. of Triggers: 0

Columns (7)

Column ID	Name	Type	Not Null	Default Value	Primary Key
0	id	INTEGER	1	null	1
1	album_id	INTEGER	0	null	0
2	genre_id	INTEGER	0	null	0
3	len	INTEGER	0	null	0
4	rating	INTEGER	0	null	0
5	title	TEXT	0	null	0
6	count	INTEGER	0	null	0

Name

Type

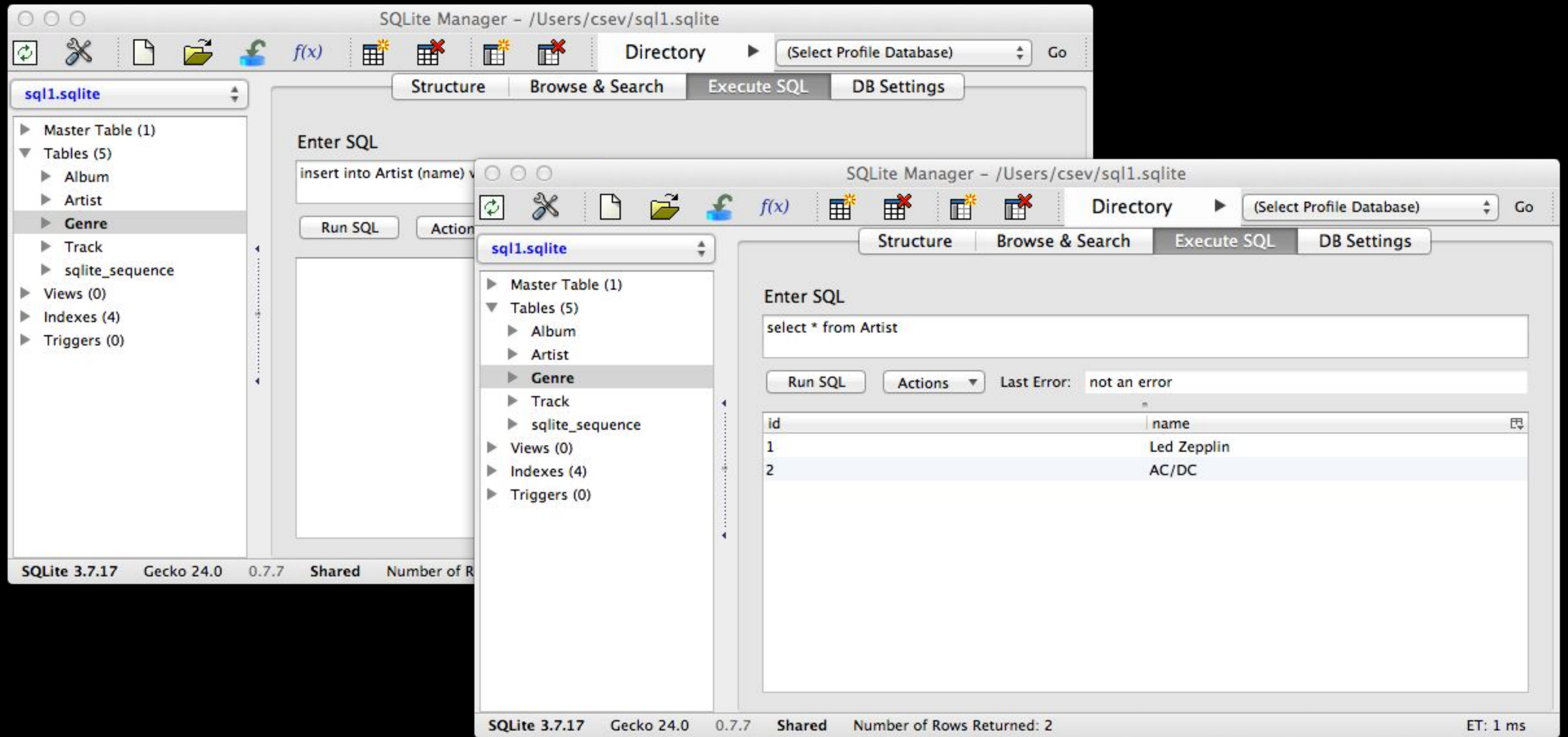
Not Null

Default

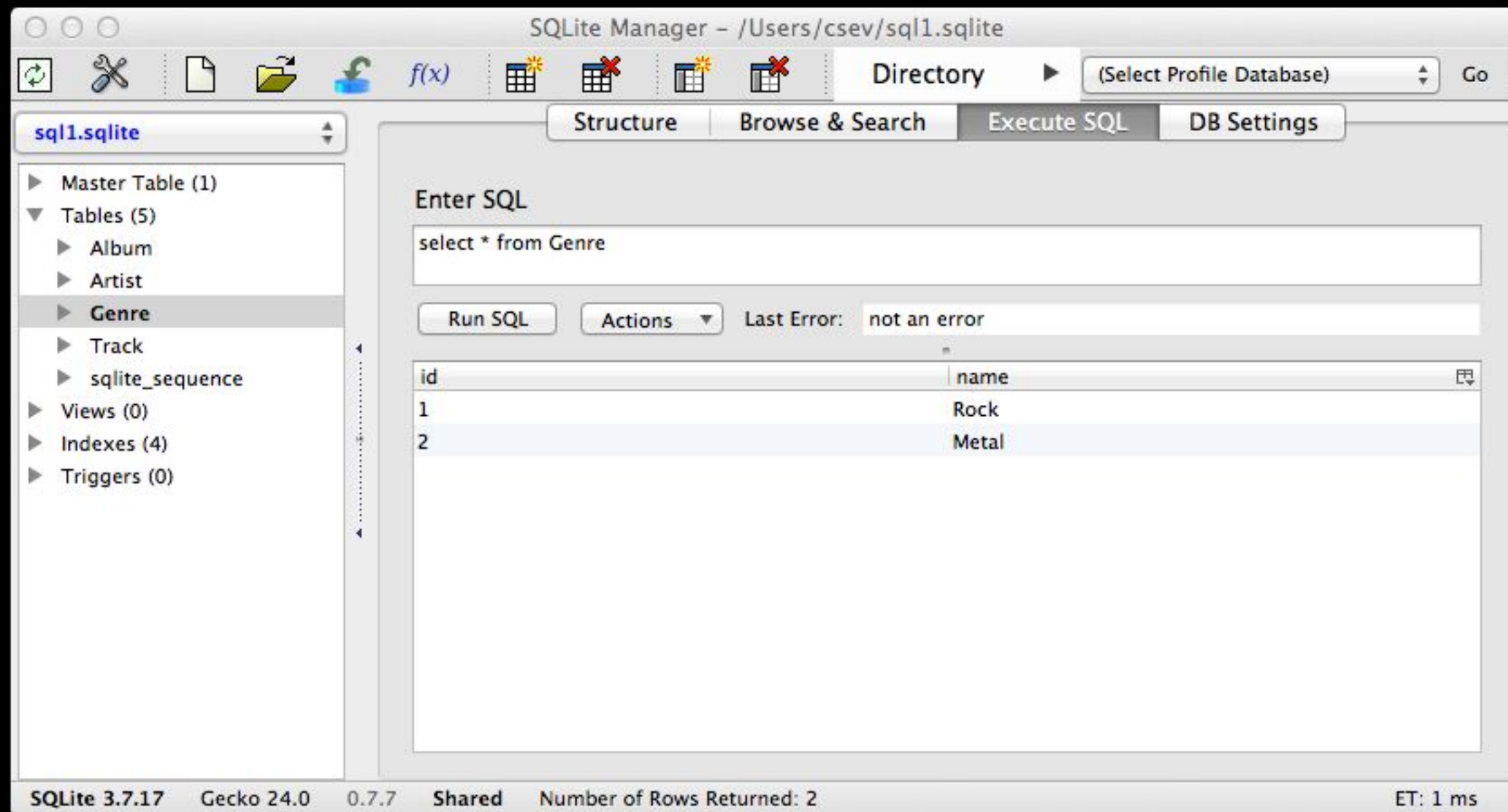
Add Column

SQLite 3.7.17Gecko 24.00.7.7SharedNumber of Rows Returned: 0ET: 0 ms





insert into Artist (name) values ('Led Zeppelin')  
insert into Artist (name) values ('AC/DC')



insert into Genre (name) values ('Rock')  
insert into Genre (name) values ('Metal')

SQLite Manager - /Users/csev/sql1.sqlite

Directory (Select Profile Database) Go

Structure Browse & Search Execute SQL DB Settings

sql1.sqlite

- Master Table (1)
- Tables (5)
  - Album
  - Artist
  - Genre
  - Track
  - sqlite\_sequence
- Views (0)
- Indexes (4)
- Triggers (0)

Enter SQL

```
select * from Album;
```

Run SQL Actions Last Error: not an error

id	artist_id	title
1	2	Who Made Who
2	1	IV

SQLite 3.7.17 Gecko 24.0 0.7.7 Shared Number of Rows Returned: 2 ET: 0 ms

insert into Album (title, artist\_id) values ('Who Made Who', 2)  
insert into Album (title, artist\_id) values ('IV', 1)

```
insert into Track (title, rating, len, count, album_id, genre_id)
values ('Black Dog', 5, 297, 0, 2, 1)
insert into Track (title, rating, len, count, album_id, genre_id)
values ('Stairway', 5, 482, 0, 2, 1)
insert into Track (title, rating, len, count, album_id, genre_id)
values ('About to Rock', 5, 313, 0, 1, 2)
insert into Track (title, rating, len, count, album_id, genre_id)
values ('Who Made Who', 5, 207, 0, 1, 2)
```

id	album_id	genre_id	len	rating	title	count
1	2	1	297	5	Black Dog	0
2	2	1	482	5	Stairway	0
3	1	2	313	5	About to Rock	0
4	1	2	207	5	Who Made W...	0



# We have relationships!

id	album_id	genre_id	len	rating	title	count
1	2	1	297	5	Black Dog	0
2	2	1	482	5	Stairway	0
3	1	2	313	5	About to Rock	0
4	1	2	207	5	Who Made W...	0

Track

id	artist_id	title
1	2	Who Made Who
2	1	IV

Album

id	name
1	Led Zeppelin
2	AC/DC

Artist

id	name
1	Rock
2	Metal

Genre

# Using Join Across Tables

[http://en.wikipedia.org/wiki/Join\\_\(SQL\)](http://en.wikipedia.org/wiki/Join_(SQL))

# Relational Power

- By removing the replicated data and replacing it with references to a single copy of each bit of data we build a “web” of information that the relational database can read through very quickly - even for very large amounts of data
- Often when you want some data it comes from a number of tables linked by these foreign keys

# The JOIN Operation

- The JOIN operation **links across several tables** as part of a select operation
- You must tell the JOIN **how to use the keys** that make the connection between the tables using an **ON clause**

id	artist_id	title
1	2	Who Made Who
2	1	IV

title	name
Who Made Who	AC/DC
IV	Led Zeppelin

id	
1	Led Zeppelin
2	AC/DC

Use 'control + ;' to execute the query.

`select Album.title, Artist.name from Album join Artist on Album.artist_id = Artist.id`

What we want  
to see

The tables that  
hold the data

How the tables  
are linked

id	artist_id	title
1	2	Who Made Who
2	1	IV

id	
1	Led Zeppelin
2	AC/DC

Use 'control + ;' to execute the query.

Album.title      Album.artist\_id      Artist.id      Artist.name

title	artist_id	id	name
Who Made Who	2	2	AC/DC
IV	1	1	Led Zeppelin

```
select Album.title, Album.artist_id, Artist.id, Artist.name
from Album join Artist on Album.artist_id = Artist.id
```

id	album_id	genre_id	len	rating	title	count
1	2	1	297	5	Black Dog	0
2	2	1	482	5	Stairway	0
3	1	2	313	5	About to Rock	0
4	1	2	207	5	Who Made W...	0

title	name
Black Dog	Rock
Stairway	Rock
About to Rock	Metal
Who Made Who	Metal

id	name
1	Rock
2	Metal

`select Track.title, Genre.name from Track join Genre on Track.genre_id = Genre.id`

What we want  
to see

The tables which  
hold the data

How the tables  
are linked



# It can get complex...

```
select Track.title, Artist.name, Album.title, Genre.name
from Track join Genre join Album join Artist on Track.
genre_id = Genre.id and Track.album_id = Album.id and
Album.artist_id = Artist.id
```

What we want  
to see

The tables which  
hold the data

How the tables  
are linked

title	name	title	name
Black Dog	Led Zeppelin	IV	Rock
Stairway	Led Zeppelin	IV	Rock
About to Rock	AC/DC	Who Made Who	Metal
Who Made Who	AC/DC	Who Made Who	Metal

✓ Hells Bells	5:13	AC/DC	Who Made Who	Rock	★★★★★	61
✓ Shake Your Foundations	3:54	AC/DC	Who Made Who	Rock	★★★★★	70
✓ Chase the Ace	3:01	AC/DC	Who Made Who	Rock		56
✓ For Those About To Rock (We ...	5:54	AC/DC	Who Made Who	Rock	★★★★★	61
✓ Dúlamán	3:43	Altan	Natural Wonders M...	New Age		31
✓ Rode Across the Desert	4:10	America	Greatest Hits	Easy Listen...	★★★★★	23
✓ Now You Are Gone	3:08	America	Greatest Hits	Easy Listen...	★★★★★	18
✓ Tin Man	3:30	America	Greatest Hits	Easy Listen...	★★★★★	23
✓ Sister Golden Hair	3:22	America	Greatest Hits	Easy Listen...	★★★★★	24
✓ Track 01	4:22	Billy Price	Danger Zone	Blues/R&B	★★★★★	26
✓ Track 02	2:45	Billy Price	Danger Zone	Blues/R&B	★★★★★	18
✓ Track 03	3:26	Billy Price	Danger Zone	Blues/R&B	★★★★★	22
✓ Track 04	4:17	Billy Price	Danger Zone	Blues/R&B	★★★★★	18
✓ Track 05	3:50	Billy Price	Danger Zone	Blues/R&B	★★★★★	21
✓ War Pigs/Luke's Wall	7:58	Black Sabbath	Paranoid	Metal	★★★★★	25
✓ Paranoid	2:53	Black Sabbath	Paranoid	Metal	★★★★★	22
✓ Planet Caravan						
✓ Iron Man						
✓ Electric Funeral						
✓ Hand of Doom						
✓ Rat Salad						
✓ Jack the Stripper/Fai						
✓ Bomb Squad (TECH)						
✓ clay techno						
✓ Heavy						
✓ Hi metal man	4:20	Brent	Brent's Album			1
✓ Mistro	2:58	Brent	Brent's Album			1

title	name	title	name
Black Dog	Led Zepplin	IV	Rock
Stairway	Led Zepplin	IV	Rock
About to Rock	AC/DC	Who Made Who	Metal
Who Made Who	AC/DC	Who Made Who	Metal

# Complexity Enables Speed

- Complexity makes speed possible and allows you to get very fast results as the data size grows
- By normalizing the data and linking it with integer keys, the overall amount of data which the relational database must scan is far lower than if the data were simply flattened out
- It might seem like a tradeoff - spend some time designing your database so it continues to be fast when your application is a success

# Additional SQL Topics

- **Indexes** improve access performance for things like string fields
- **Constraints** on data - (cannot be NULL, etc..)
- **Transactions** - allow SQL operations to be grouped and done as a unit
- **See SI664 - Database Design (All Semesters)**

# Summary

- Relational databases allow us to **scale** to very large amounts of data
- The key is to have **one copy of any data** element and use relations and joins to link the data to multiple places
- This greatly **reduces the amount of data which much be scanned** when doing complex operations across large amounts of data
- Database and SQL design is a bit of an **art form**



# Acknowledgements / Contributions



These slides are Copyright 2010- Charles R. Severance ([www.dr-chuck.com](http://www.dr-chuck.com)) of the University of Michigan School of Information and [open.umich.edu](http://open.umich.edu) and made available under a Creative Commons Attribution 4.0 License. Please maintain this last slide in all copies of the document to comply with the attribution requirements of the license. If you make a change, feel free to add your name and organization to the list of contributors on this page as you republish the materials.

Initial Development: Charles Severance, University of Michigan School of Information

... Insert new Contributors here

...