Relational Databases

Charles Severance

Unless otherwise noted, the content of this course material is licensed under a Creative Commons Attribution 3.0 License.
http://creativecommons.org/licenses/by/3.0/.

Copyright 2009-2012, Charles Severance

SQLite Browser

Suggest using version 1.3

http://www.pythonlearn.com/sqlite/

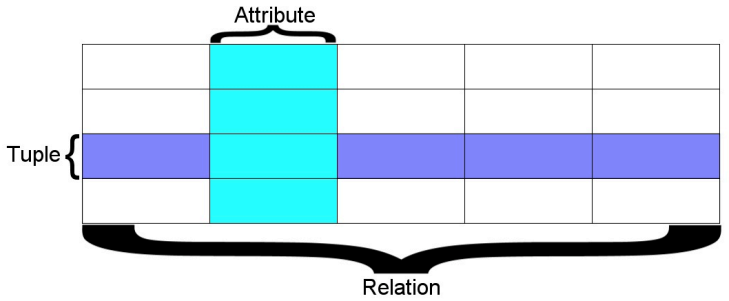
Relational Databases

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.

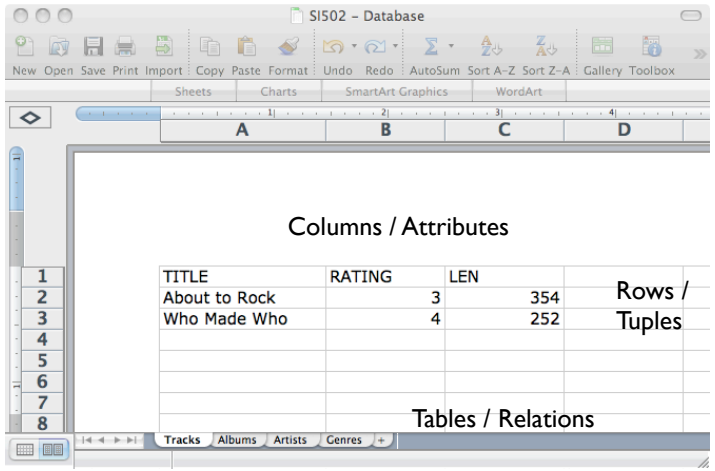
http://en.wikipedia.org/wiki/Relational_database

Terminology

- Database - Contains many tables
- Relation (or table) - contains tuples and attributes
- Tuple (or row) - is a set of fields it 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



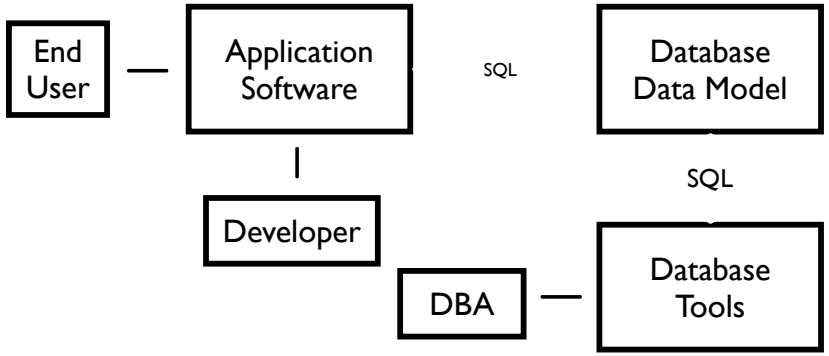
A relation is defined as a set of tuples that have the same attributes. A tuple usually represents an object and information about that object. Objects are typically physical objects or concepts. A relation is usually described as a table, which is organized into rows and columns. All the data referenced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



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, co-ordinate and implement security measures to safeguard the database.

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

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, ...

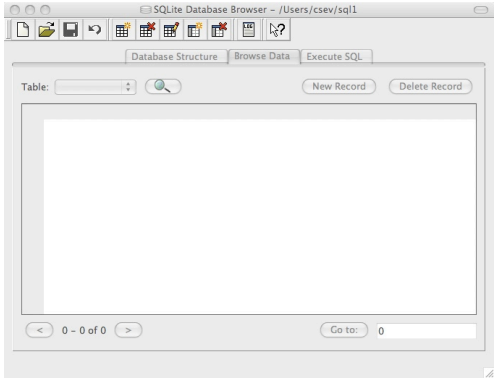
SQLite Database Browser

- SQLite is a very popular database - it is free and fast and small
- We have a program to manipulate SQLite databases
  - http://sqlitebrowser.sourceforge.net/
- SQLite is embedded in Python and a number of other languages

SQLite is in lots of software...

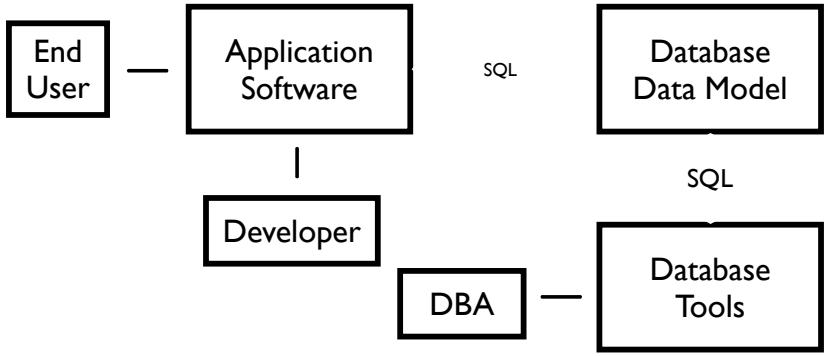
symbian python PHILIPS skype Microsoft McAfee Adobe php Google TOSHIBA Sun microsystems

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



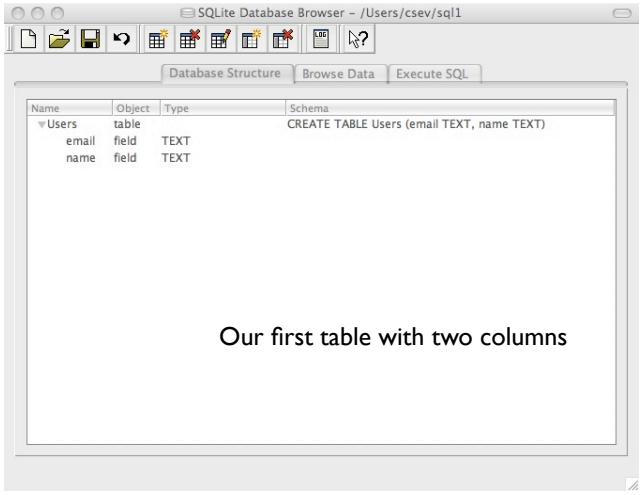
- http://sqlitebrowser.sourceforge.net/

Application Structure

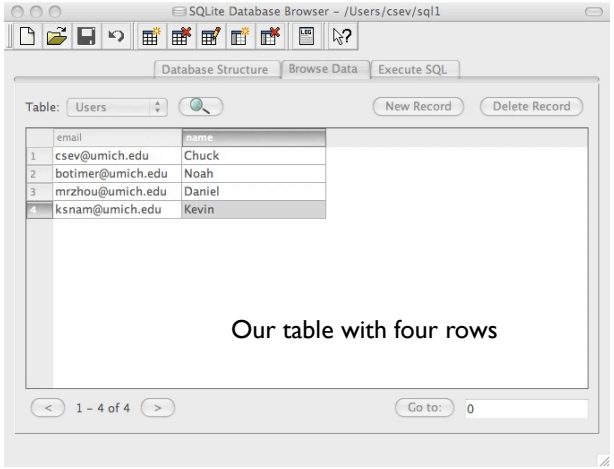


Start Simple - A Single Table

- Lets make a table of People - with a Name and an E-Mail



Our first table with two columns



Our table with four rows

SQL

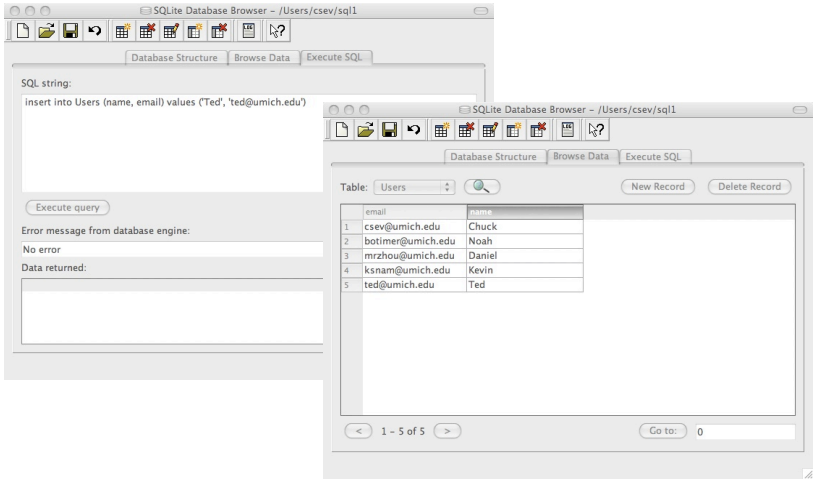
- Structured Query Language is the language we use to issue commands to the database
  - Create a table
  - Retieve some data
  - Insert data
  - Delete data

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

SQL Insert

- The Insert statement inserts a row into a table

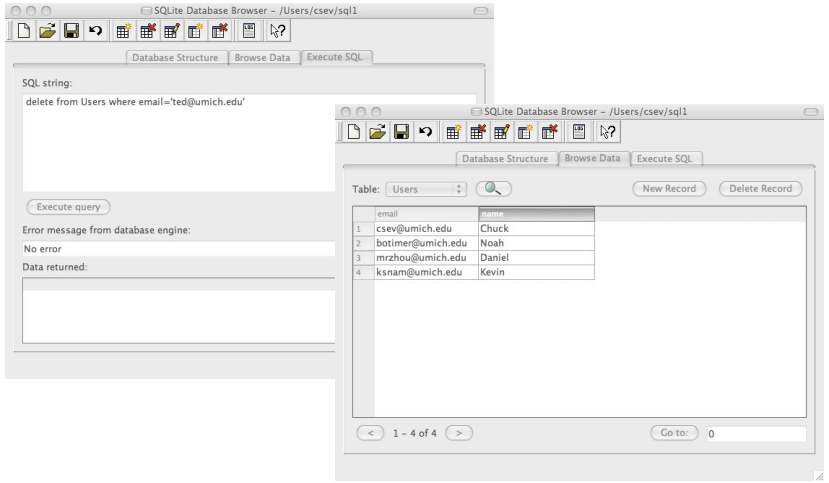
insert into Users (name, email) values ('Ted', 'ted@umich.edu')



SQL Delete

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

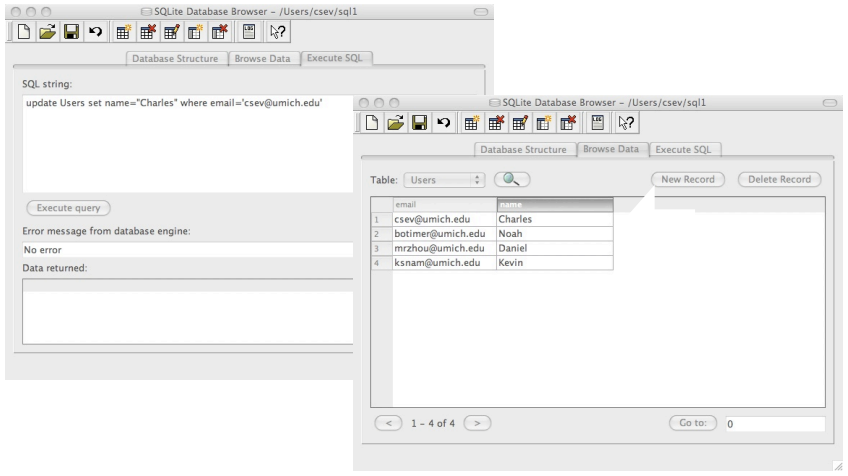
delete from Users where email='ted@umich.edu'



SQL: Update

- Allows the updating of a field with a where clause

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

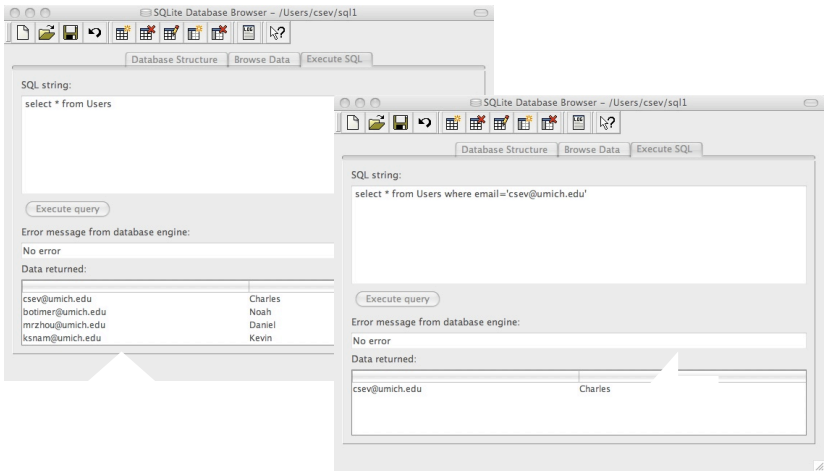


Retrieving Records: Select

- 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'

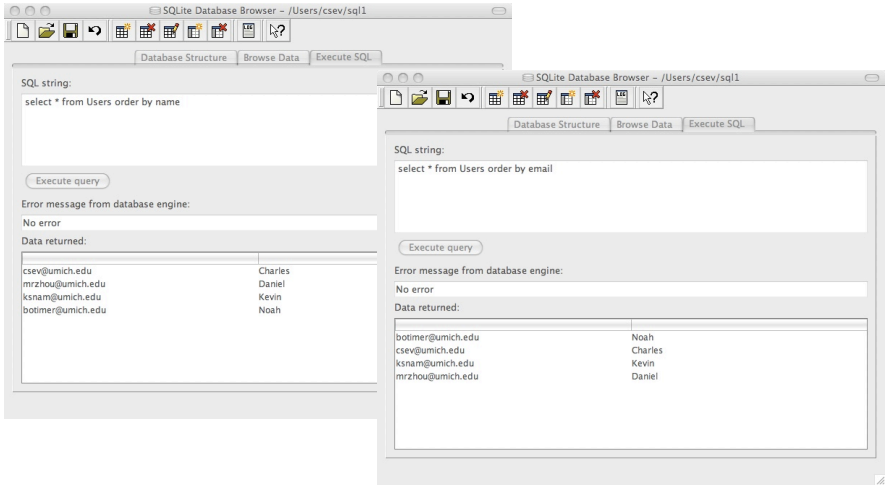


Sorting with ORDER BY

- 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



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)

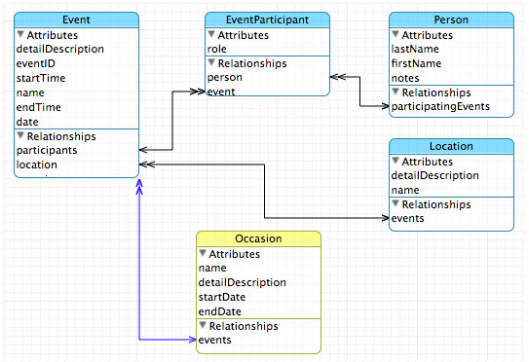
- Tables pretty much look like big fast programmable spreadsheet with rows, columns, and commands
- The power comes when we have more than one table and we can exploit the relationships between the tables

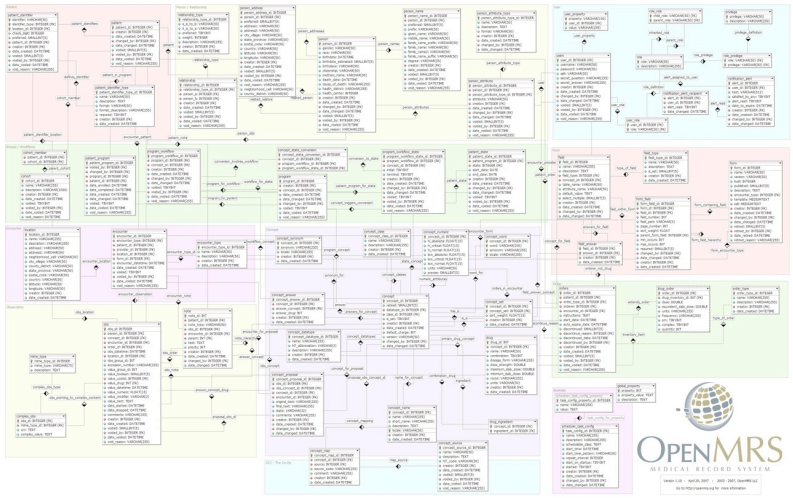
Complex Data Models and Relationships

http://en.wikipedia.org/wiki/Relational_model

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...





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

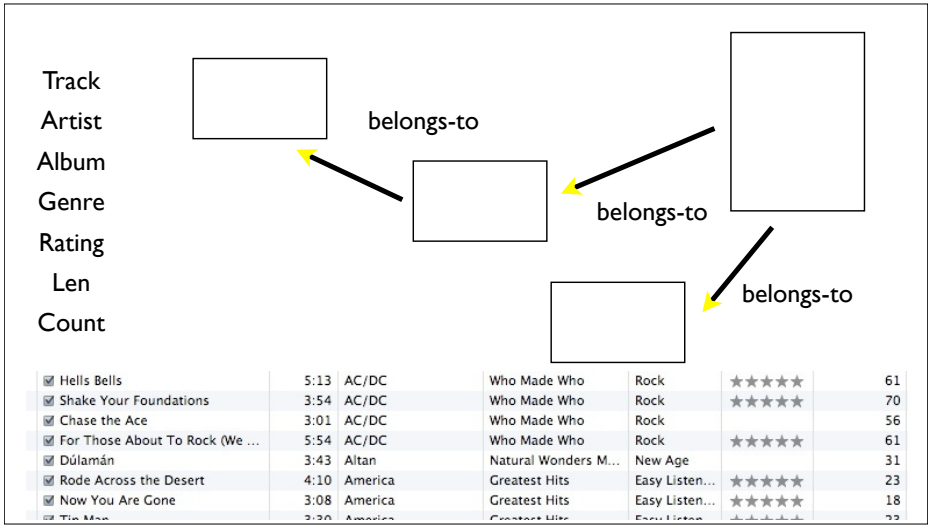
Track	Len	Artist	Album	Genre	Rating	Count
Hells Bells	5:13	AC/DC	Who Made Who	Rock	★★★★★	61
Shake Your Foundations	3:54	AC/DC	Who Made Who	Rock	★★★★★	70
Chase the Ace	3:01	AC/DC	Who Made Who	Rock		56
For Those About To Rock (We …	5:54	AC/DC	Who Made Who	Rock	★★★★★	61
Dúlamán	3:43	Altan	Natural Wonders 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

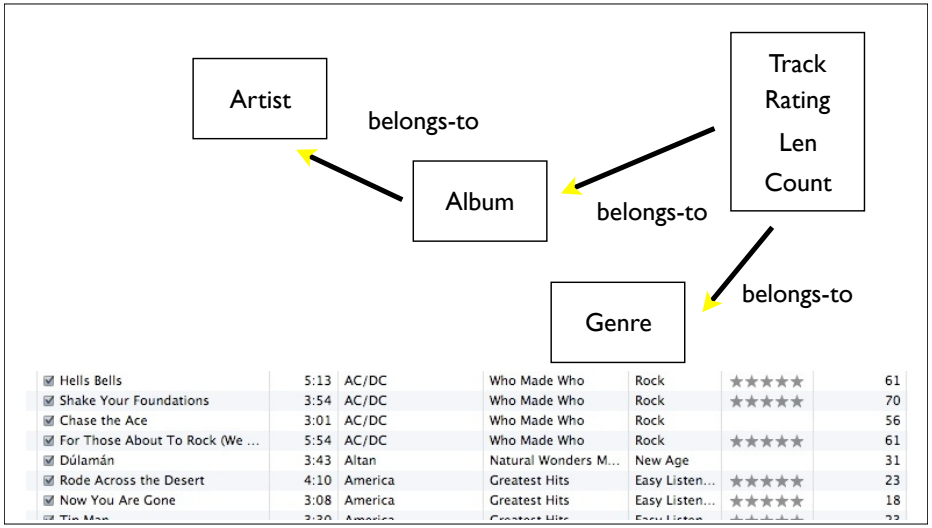
For each "piece of info"...

- 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

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





Representing Relationships in a Database

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?

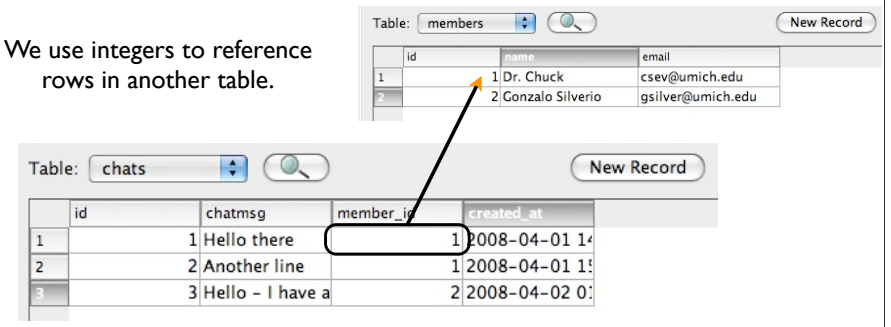
Database Normalization (3NF)

- 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

Integer Reference Pattern

We use integers to reference rows in another table.

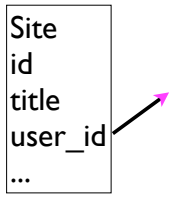


Keys

Finding our way around....

Three Kinds of Keys

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



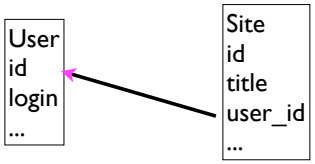
Primary Key Rules

- 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

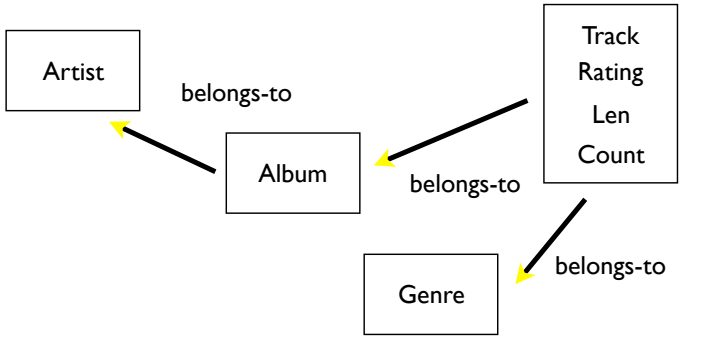
User
id
login
password
name
email
created_at
modified_at
login_at

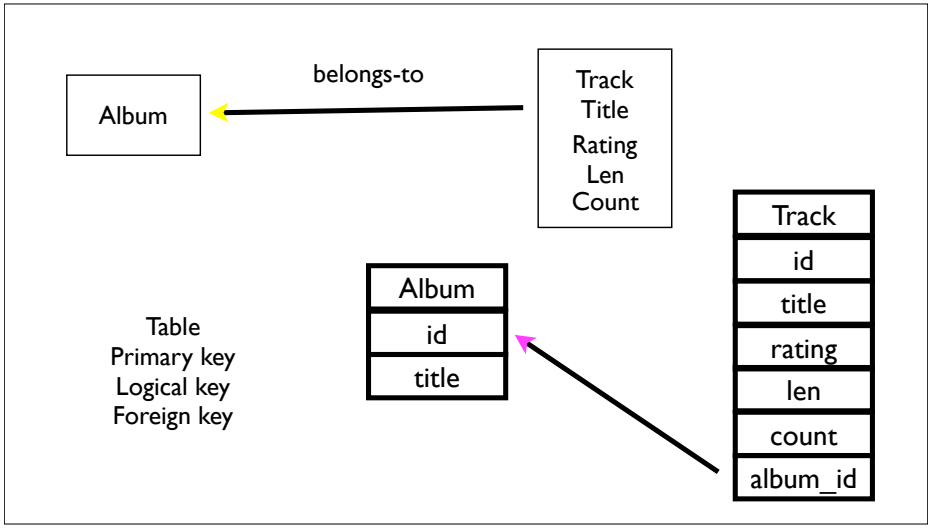
Foreign Keys

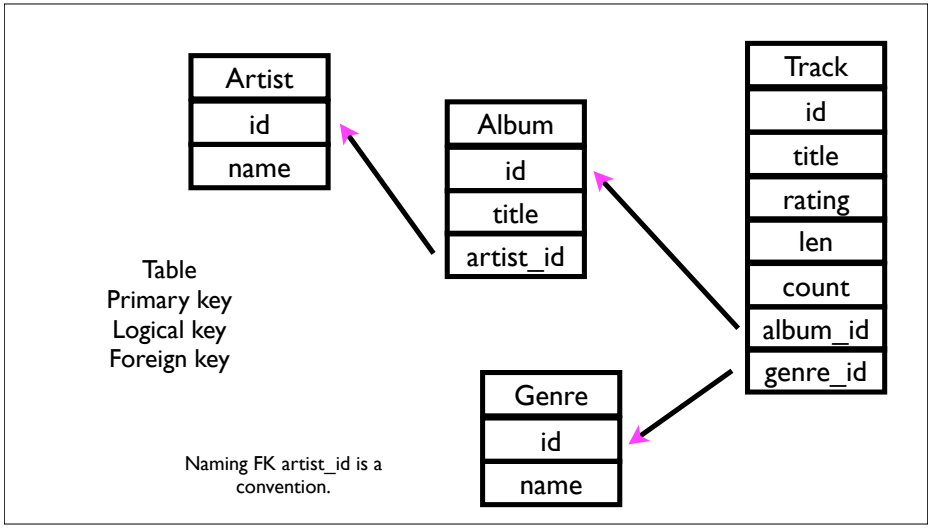
- A foreign key is when a table has a column that contains a key which points the primary key of another table.
- When all primary keys are integers, then all foreign keys are integers - this is good - very good
- If you use strings as foreign keys - you show yourself to be an uncultured swine

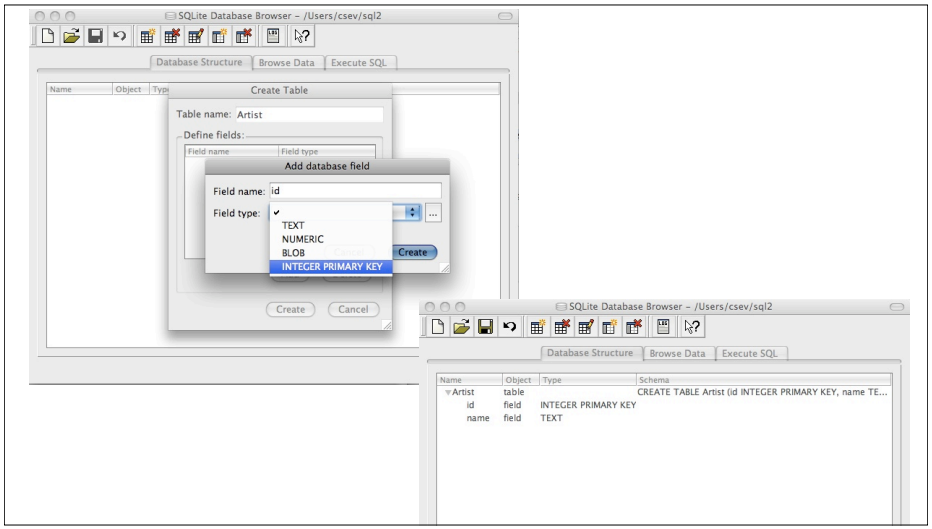


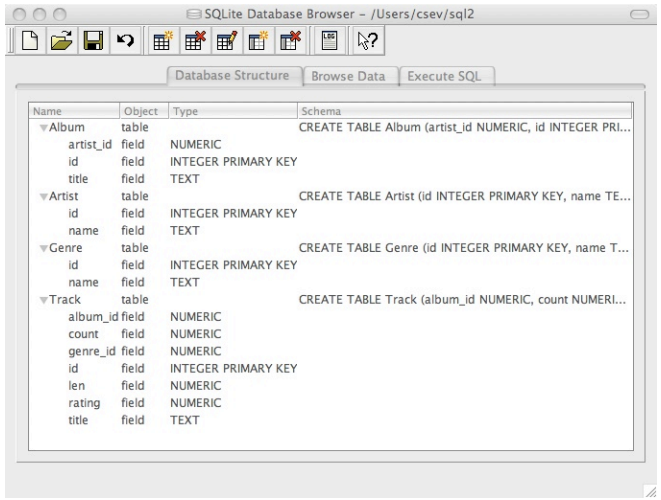
Relationship Building (in tables)

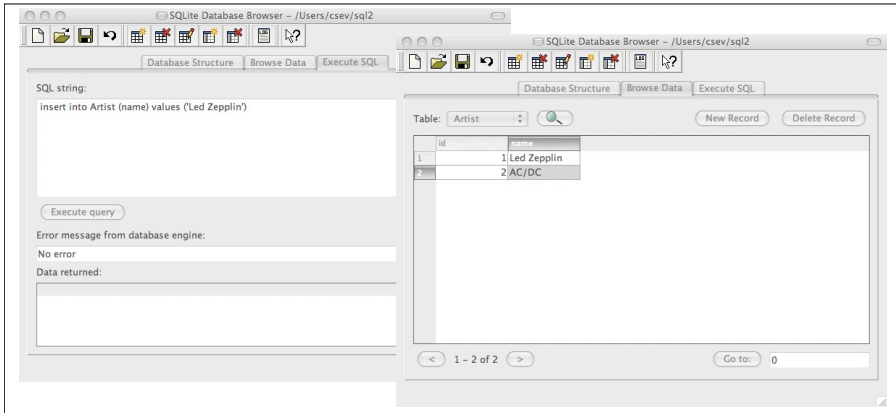



Album
belongs-to
Track
Title
Rating
Len
Count
Track
id
title
rating
len
count
album_id
Album
id
title
Table
Primary key
Logical key
Foreign key

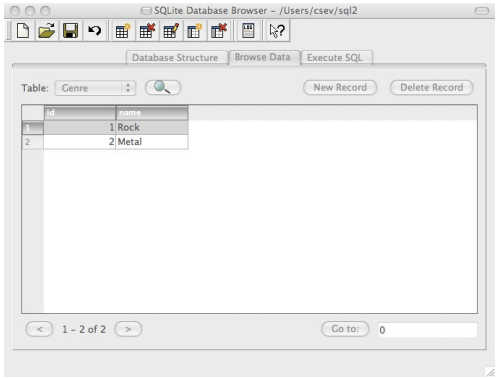

Artist
id
name
Album
id
title
artist_id
Track
id
title
rating
len
count
album_id
genre_id
Genre
id
name
Table
Primary key
Logical key
Foreign key
Naming FK artist_id is a convention.


SQLite Database Browser – /Users/csev/sql2
Database Structure
Browse Data
Execute SQL
Create Table
Table name: Artist
Define fields:
Add database field
Field name: id
Field type:
TEXT
NUMERIC
BLOB
INTEGER PRIMARY KEY
Create
Cancel
SQLite Database Browser – /Users/csev/sql2
Artist table CREATE TABLE Artist (id INTEGER PRIMARY KEY, name TE...
id field INTEGER PRIMARY KEY
name field TEXT

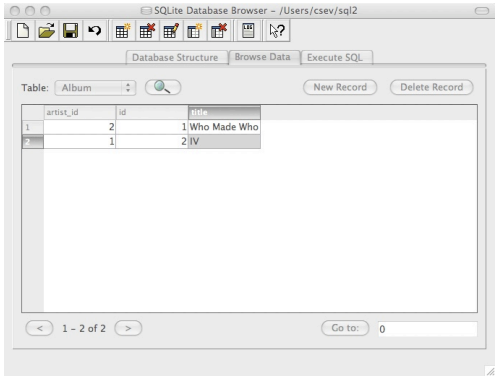

SQLite Database Browser – /Users/csev/sql2
Database Structure
Browse Data
Execute SQL
Name Object Type Schema
Album table CREATE TABLE Album (artist_id NUMERIC, id INTEGER PRI...
artist_id field NUMERIC
id field INTEGER PRIMARY KEY
title field TEXT
Artist table CREATE TABLE Artist (id INTEGER PRIMARY KEY, name TE...
id field INTEGER PRIMARY KEY
name field TEXT
Genre table CREATE TABLE Genre (id INTEGER PRIMARY KEY, name T...
id field INTEGER PRIMARY KEY
name field TEXT
Track table CREATE TABLE Track (album_id NUMERIC, count NUMERI...
album_id field NUMERIC
count field NUMERIC
genre_id field NUMERIC
id field INTEGER PRIMARY KEY
len field NUMERIC
rating field NUMERIC
title field TEXT



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

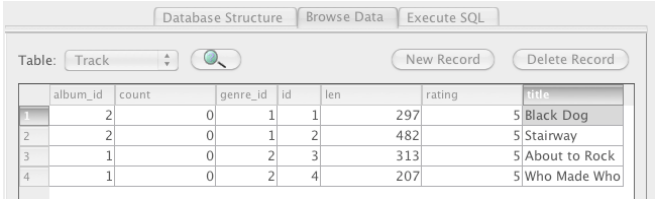


insert into Genre (name) values ('Rock')
insert into Genre (name) values ('Metal')

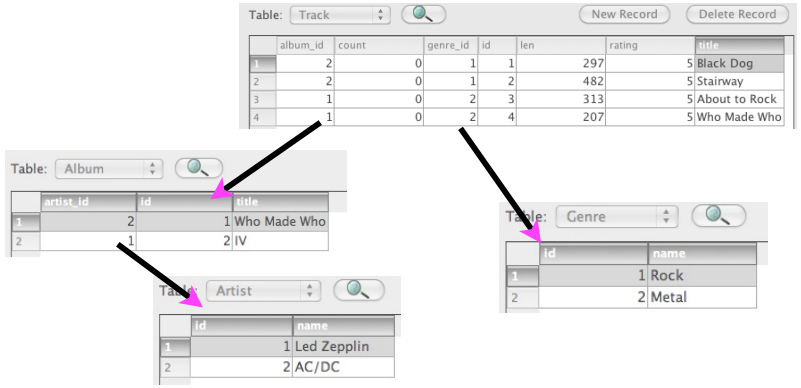


insert into Album (title, artist_id) values ('Who Made Who', 2)
insert into Album (title, artist_id) values ('IV', 1)

```
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)
```



We have relationships!



Using Join Across Tables

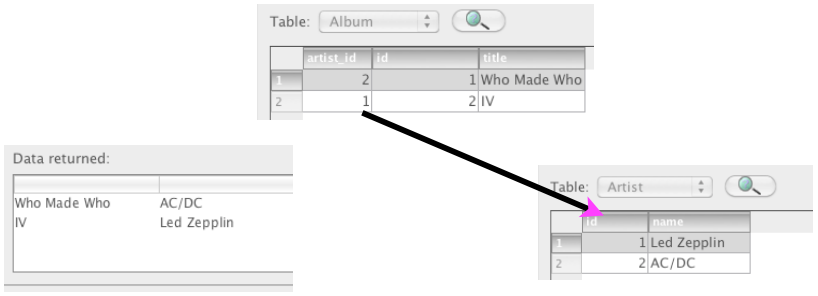
http://en.wikipedia.org/wiki/Join_(SQL)

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



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

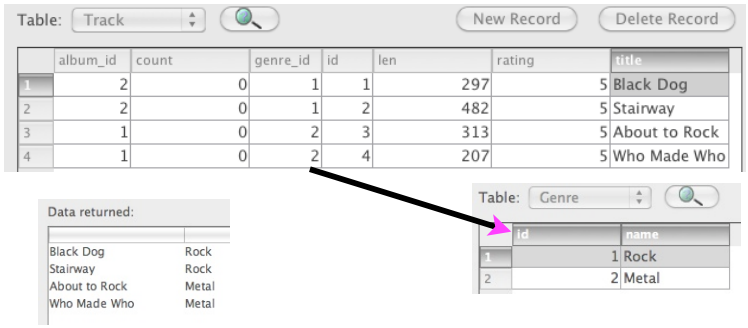
What we want to see

The tables which hold the data

How the tables are linked

Album.title	Album.artist_id	Artist.id	Artist.name
Who Made Who	2	2	AC/DC
IV	1	1	Led Zepplin

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



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

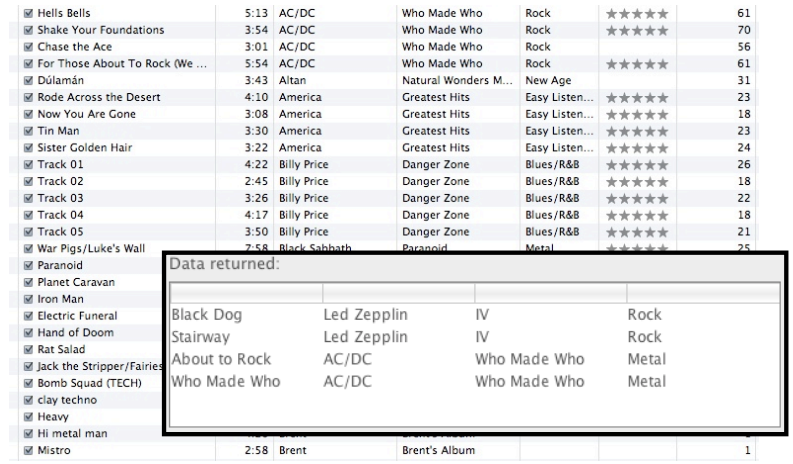
Data returned:

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

What we want to see

The tables which hold the data

How the tables are linked



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 SI572 - 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