**Week 4 Day 1 Notes**

* Today’s lesson is all about saving our data from a program like last week’s CRM assignment into a database.

1. **What is a relational database?**

* It means if you have a couple different tables, they can relate to one another.
* SQL is specifically designed to work with relational databases. A relational database is a data store that is capable of representing relationships between different sets of data. For example, a database may have a table of customers and a table of addresses. A single customer may have more than one address and a single address may have more than one customer. A relational database has facilities for describing these relationships and SQL was designed to make these features accessible and flexible. In a relational database information is said to be presented in rows and columns. A row of data is analogous to a single data record. In fact, those of us who have been using databases since before SQL, sometimes still refer to rows as records.
* Relational databases will be covered more in a future lecture.

1. **What is a database table?**A table is a collection of related data held in a structured format within a database. It consists of columns and rows. In relational databases and flat file databases, a table is a set of data elements (values) using a model of vertical columns (identifiable by name) and horizontal rows, the cell being the unit where a row and column intersect.
2. **What is an ORM?***(bullet points for 3 are mostly repeating same thing in differnt ways)*

* Object-relational mapping in computer science is a programming technique for converting data between incompatible type systems using object-oriented programming languages. This creates, in effect, a "virtual object database" that can be used from within the programming language (such as Ruby). There are both free and commercial packages available that perform object-relational mapping.
* (^above point stated differently) [Object-Relational Mapping](http://en.wikipedia.org/wiki/Object-relational_mapping) (ORM) is a technique that lets you query and manipulate data from a database using an object-oriented paradigm ( When talking about ORM, most people are referring to a library that implements the Object-Relational Mapping technique, hence the phrase "an ORM". (<https://stackoverflow.com/questions/1279613/what-is-an-orm-and-where-can-i-learn-more-about-it>)
* An ORM library (such as ActiveRecord) is a completely ordinary library written in your language of choice (such as Ruby) that encapsulates the code needed to manipulate the data, so you don't use SQL anymore; you interact directly with an object in the same language you're using.
* An ORM is essentially a tool that's used to manage and convert database data from an object-oriented programming perspective. So in other worlds, database tables can be treated as classes and their fields can be treated as properties. And this ultimately eliminates the need to write a lot of tedious code.
* [https://en.wikipedia.org/wiki/List\_of\_object-relational\_mapping\_software#Ruby](https://en.wikipedia.org/wiki/List_of_object-relational_mapping_software" \l "Ruby)

List of ORM software for many different object oriented languages.

1. **Intro to SQL and SQLite**

* Its called *lite* because it isn’t a full-on server, its more for messing around on your local machine rather than hosting data for the internet.
* When running SQLite3 in terminal you can type *.help* for a list of commands.
* .*tables* command lets you see all the tables inside your database.
* *.database* lets you see all the databases you have in the folder.
* CREATE TABLE films(id , title, rating); ---creates a table called films. The semicolon is necessary to let you know the command has finished, its like putting a period at the end of a sentence. Otherwise SQL will keep waiting for you to finish the command.
* (id, title, rating); all have a datatype, you need to specify what types of data go in each column (string, integer, etc.).
* (id INTEGER PRIMARY KEY, title TEXT, rating INTEGER);  
  Integer primary key lets you know this is the most important column that can be used as an identifier. Changes set the data types.
* In specifying integers, floats, or both, you have to look up the data type names for each database system (SQLite3 has its own that may be different from others).
* Not actually required to put CREATE TABLE in all caps, its just a best practice.
* INSERT INTO films(id INTEGER PRIMARY KEY, title TEXT, rating INTEGER); *---fills out a row in the database*.
* CTRL + C *---forces quit if you get into an error*
* SELECT \* from films; *---(the \* selects all columns, if you wanted to select one column you would put in its name in place of the \*)*
* To update an item that is inside films table...   
  UPDATE films SET title=”Scarface”; *---(putting this in would update every single film entry)*

UPDATE films SET title=”Scarface” WHERE id=1; *---(this selects a specific item to update) (note that its not like ruby where it would be ==, in this particular use case in Sql3 one = means assignment)*

* One should not put your database into a version control system like github. Database info changes too much, plus its usually private info you don’t want public.
* DELETE FROM films WHERE id=2; *---(deletes row, if you put in an id that doesn’t exist by accident, it won’t do anything, note if you didn’t specify a specifc thing to delete it would delete everything so be careful)*
* DROP TABLE films; *---(looks for a table called films and deletes it, so deletes all data from table and the table itself)*
* ALTER TABLE films ADD COLUMN title TEXT; *---(adds a column)*
* SELECT \* FROM films WHERE title=”Star Wars”; *---(asks to select only films out of entire database that read as ONLY “Star Wars”, it is not searching to see if it has Star Wars somewhere in the title and returning all that do, see blue item blue to how to search for any title containing Star Wars...)*
* **SELECT \* FROM films WHERE title LIKE “%Star Wars%”**

***---(selects any title that contains Star Wars. % sign is saying anything can come before or after Star Wars in the titles... Note you have to use the double “ and not the single ‘.***

* SELECT \* FROM films WHERE id > 1;  *---(selects all films with an id greater than one)*

(need to go back to hour and ten minute mark to review what he said)

1. **Using MiniRecord** **& Active\_record Gems**

* gem "activerecord", “=4.2.9”

gem "mini\_record" *---(specifies gem we are going to use in our Gemfile??? Will have to do this again in today’s assignments. require\_relative Gemfile in the film.rb)*

* gem library – activerecord: *Databases on Rails. Build a persistent domain model by mapping database tables to Ruby classes. Strong conventions for associations, validations, aggregations, migrations, and testing come baked-in.*
* Gem library – mini\_record: *With it you can add the ability to create columns outside the default schema, directly in your model in a similar way that you just know in others projects like DataMapper or MongoMapper.*
* *Can look up documentation for each gem if you want more info on them.*
* But putting in bundle install in terminal, we install these gems.
* *class Film < ActiveRecord::Base (lets class Film inherit the automatic mapping from gem ActiveRecord?)*
* *Film.auto\_upgrade! --(Put at end of class, it takes care of effecting any changes to the underlying structure of the tables or columns.)*
* *Don’t mess around with the gem code too much. Its very rare that you will actually need to alter the gem code.*
* [*https://github.com/explore*](https://github.com/explore) *Is a good place to find useful Ruby gems.*
* *Once you set up your files properly, you can create a film in either irb or sqlite3, and the changes you make will be reflected in the other.*
* *Just by convention, the name of your ruby class should be singular, but the name of your table should be plural. Film.rb and films.sqlite3 respectively.*

Notes below copy and pasted from the page for assignment 01 - CRM: Saving your Data.

* A database management system (DBMS) is the software that allows us to interact with the database, kind of the way that the Excel program allows us to interact with Excel files. There are many different database management systems and we'll see several over the course of the program.
* There are many different variations on databases but the most widely used type is called a relational database. Relational databases organize data into tables, each table being made up of columns and rows.
* You can think of the structure of the database like objects in Ruby. Every table in a database represents one type of object (like a Ruby class), the columns in a table represent the attributes of that object and each row represents an individual instance of an object.
* SQL stands for Structured Query Language, its pronounced sequel. It is a standard language for interacting with relational databases, it is just one type of database. It's used to define the structure of the database (i.e. create tables and define the columns on a table) as well as to retrieve data stored in the database (i.e. fetch all of the orders from the last month).
* All great programmers are lazy in the best way possible: they want to get more results with less effort. This is where ORM (Object-Relational Mapping) becomes important. An ORM tool, in this case ActiveRecord, allows us to represent our SQLite database tables with Ruby objects, or in layman's terms, write Ruby code that gets translated into SQL to talk to Databases.

[Relational Databases For Dummies](http://code.tutsplus.com/tutorials/relational-databases-for-dummies--net-30244)

[Ruby SQLite3 Tutorials: Connecting to the Database](http://zetcode.com/db/sqliteruby/connect/)

[Ruby SQLite3 Tutorials: Queries](http://zetcode.com/db/sqliteruby/queries/)

[Today's Slides](http://bitmakerlabs.s3.amazonaws.com/slides/wdi-october-2015/Databases and SQL.pdf)

[Active Record](https://github.com/rails/rails/tree/master/activerecord)

[Mini Record](https://github.com/DAddYE/mini_record)