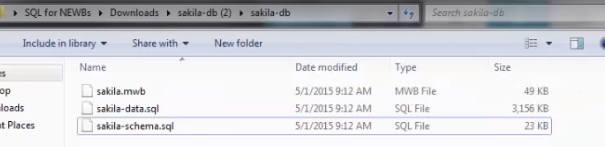
**Notes for SQL for newbs**

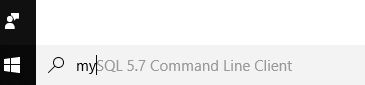
First thing we did was to install mysql and a code editor.

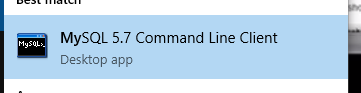
The next thing was to download the resources file from the site in the transcript/resource section.

I unzipped this file and then install these files from the command line strait into the database

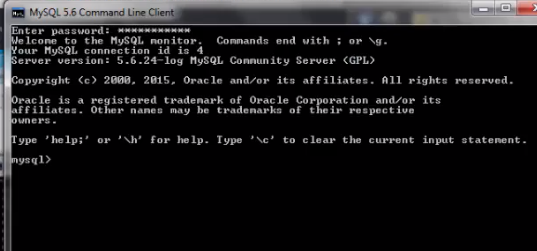
Open the folder 

Then we go to our windows button and search mysql command line client





Then the window pops up and you enter your mysql password

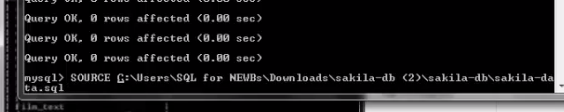


Then you install the SCHEMA FILE FIRST by writing SOURCE and click and drag the file in the command line window which will copy the file path. (remove any quotes surrounding the file path and making sure there is a space after source)and then press enter



Now we want to fill those empty tables with actual data so

After that has loaded do the same for the data



And press enter

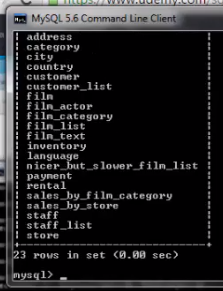
Then to make sure everything went ok wright USE sakila (the name of the database)



To see if there are actual table we can wright

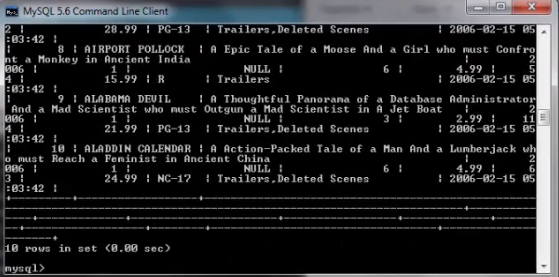
Show tables; and it should show all the tables that are in our database



we can actually wright our sequel queries in the command line client. Example show all (show \*) from film and limit it to 10



And that will show us the first ten records

+

But this is hard to read and unattractive so we will download a gooey and install heidisql

this seems like the program like my MySQL Workbench

Next we did a simple query originally written in the code editor

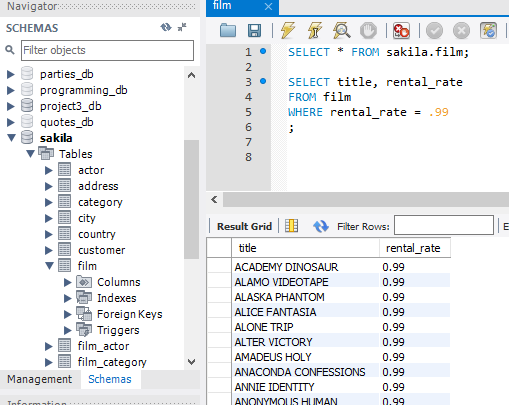
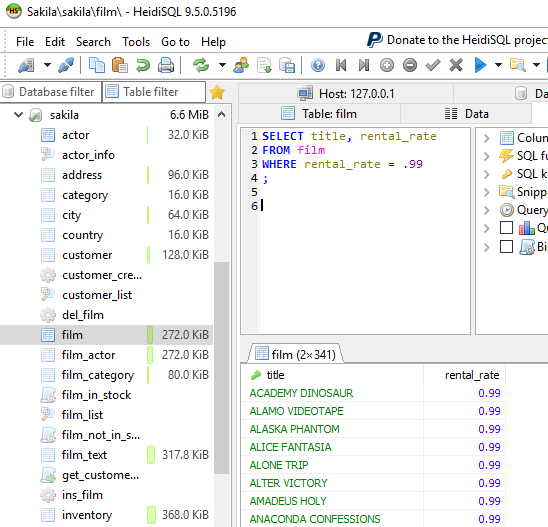
**SELECT** title, rental\_rate ……..

**FROM** film

**WHERE** rental\_rate **=** .99

;

I ran this both in Heidi and Workbench



Did first “brain buster”

*-- to wright a querry find all customer names first and last,*

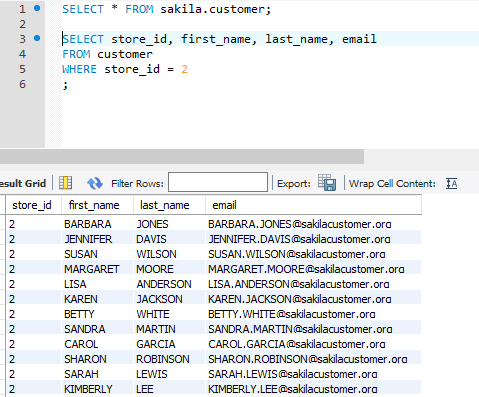
*-- their email addressees and where their a customer of*

*-- for only thoes who are a customer of store number two*

**SELECT** store\_id, first\_name, last\_name, email

**FROM** customer

**WHERE** store\_id **=** 2;

easy

SELECT FROM and WHERE is the skeleton template to making a simple query. Select is what columns you want to be displayed. From is from what table you are referencing. And Where is a secondary parameter you constrain on what columns you have selected.

To count do SELECT COUNT(title) and FROM film; ((everything can be put in lowercase but capitalizing things like select and count are minor conventions for easy reading))

**SELECT** COUNT(title)

**FROM** film

Where title is the column name. A lot of times you may just be counting ids. In mysql or Heidi there is a little cheat in the window that tells you how many rows there are and how many are affected. This query actually gives you the count.

Run it 

Along with count there is GROUP BY is a function that groups the result-set by one or more columns.

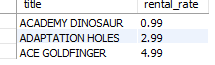
If we did

**SELECT** title, rental\_rate

**FROM** film

**GROUP BY** rental\_rate ;

We would get only



But if we also use count with group by

**SELECT** COUNT(title), rental\_rate

**FROM** film

**GROUP BY** rental\_rate ;



A sneaky trick is also to use a number instead of a title

**GROUP BY** rental\_rate VS

**GROUP BY** 2

BRAINBUSTER

Which rating do we have the most films in?

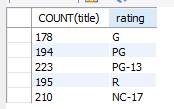
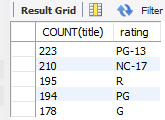
Which rating is the most prevalent in each store?

#1.

**SELECT** COUNT(title), rating

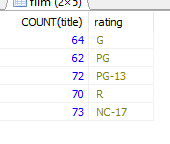
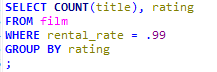
**FROM** film

**GROUP BY** rating ;

  PG-13 Has the most films

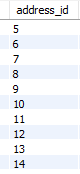
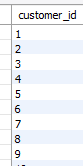
In the given answer they actually had it by COUNT(film\_id) just in case different movies have the same titles

#2. There are 70 films that have a R rating and are $.99. true or false (TRUE)



Now we are going to Connect tables

If we look in the customer table in the sakila database we can see a customer id and an address id

however, the address id doesn’t say anything about it so if we then look in a table that would have the address information. which is the address table

So lets wright a query to find out the customer id

First we wright out our skeleton and put in the two tables we want to use customer and address

**FROM** customer, **address**

Next we select the columns we want used from each table but we have to use dot notation so we know which table we are talking about when we select a column.

**SELECT** customer.customer\_id, customer.first\_name, customer.last\_name, **address**.**address**

But If we ran this by itself it would for each address its giving us every person which is NOT what we want (361 thousand something rows) so it didn’t actually connect the tables .

So what we need to do is connect these two tables so that the information we are getting is specific to a certain customer. So we add our WHERE clause

We want the customer address id equal to the address address id

**WHERE** customer.address\_id **=** **address**.address\_id

And so now these tables are connected so that when we want a customer id its going to give us an address that correlates to the address id for that customer



So now each of these users has their unique address

So now we can send them as much junk mail as we want.

BRAIN BUSTER #3

Want the list of every film (film name), the category of that film(comedy or romcom) and then the language of that film

(what I originally had that DID NOT WORK)

**SELECT** film.film\_id, film.title, catagory.name, **language**.**name**

**FROM** film, **language**, film\_category, category

**WHERE** film.film\_id **=** film\_category.film\_id, **language**.language\_id **=** film.language\_id

;

(what they had)

**SELECT** film.film\_id, film.title, category.name, **language**.**name**

**FROM** film, **language**, film\_category, category

**WHERE** film.film\_id **=** film\_category.film\_id

**AND** **language**.language\_id **=** film.language\_id

**AND** category.category\_id **=** film\_category.category\_id

;