M259 SQL LAB Demo 1
Created by Joshua Dickinson (Updated by Yoon Chung Han (2014), Mohit Hingorani (2015))

Querying the Data
First we need to install a MySQL client so that we can communicate with the server on which the database is stored: MySQL Workbench

Accessing the database
Host: tango.mat.ucsb.edu
User: mat259
Password: V1sual1zat1on
Database: (optional)
Port: 3306

Forming a Query
Before moving further, it is recommended to check the basic concepts of SQL and step-by-step SQL Tutorials from W3school.com (http://www.w3schools.com/sql/default.asp)

Queries are like commands that allow us to access a small subset of the database based on a set of criteria. The entire dataset has something like 60 million entries so we don't want to look at all of that all at once.

Here's an example of a simple query:
```
select * from inraw where title = "Life of Pi";
```

In English this means:
"show me all the data about entries from the "inraw" set where the title is "Life of Pi".

Now we only want to see DVDs:
```
select * from inraw where title = " Life of Pi " and itemtype = "acdvd";
```

In English:
"show me all the data about entries from the "inraw" set where the title is "Life of Pi" and the item is a DVD (adult section).

Various MySQL syntax:
```
select * from inraw where title like "%vampire%";
```

In English:
"show me all the data about entries from the "inraw" set where the title has the word "vampire" somewhere inside of it.

The "%" symbols are a regular expression which stands for any number of characters or spaces, so in the example above "vampires" matches and so would "emo vampires" but not "vampiring".

```
select * from inraw where title like "%vampire%" or subj like "%vampire%";
```

In English:
"show me all the data about entries from the "inraw" set where the title or any of the subjects has the word

“vampire” somewhere inside it.

Parenthesis allow us to string together arbitrarily long sets of criteria:

```
select * from inraw where (title like "%vampire%" or subj like "%vampire%") AND title like "%blood%" AND (itemtype = "acbk" or itemtype = "acdvd");
```

In English:
“show me all the data about entries from the “inraw” set where the title or any of the subjects has the word “vampire” somewhere inside it AND make sure that the title has “blood” somewhere in it AND make sure that the item type is either an adult book or an adult dvd.”

Narrowing a Query
So in the previous examples we’re looking at the data from the entire year database which includes activities from 2005 to 2014. If we want to narrow the time frame that we’re searching, we need to add a time criteria into the search.

```
select * from inraw where (month(cout)>1) AND (month(cout)<5) AND title like "%vampire%";
```

In English:
“show me all the data about entries from the “inraw” set that were checked out from February through April AND make sure that the title has “vampire” in it.”

Now let’s narrow the data that the query returns, since we want to focus on dates and ignore everything else. We’re going to ask it to return only the cout, cin, and title.

```
select cout,cin,title from inraw where (month(cout)>1) AND (month(cout)<5) AND title like "% vampire %";
```

In English:
“show me the checkout date and time, the return date and time, and the title of entries from the “inraw” set that were checked out from February through April AND make sure that the title has “vampire” in it.”

Since now we’re specifying exactly what we want back, we can also include bits of code in our requests. Here’s an example of how to find out how many days something was checked out for.

```
select cout,cin,title,TIMESTAMPDIFF(DAY,cout,cin) from inraw where (month(cout)>1) AND (month(cout)<5) AND title like "%vampire%";
```

Where we say “HOUR” we could also use any MySQL unit type, such as MICROSECOND, SECOND, MONTH, etc. Here’s a list options: http://dev.mysql.com/doc/refman/5.5/en/date-and-time-functions.html#function_date-add

Organizing the Results
Maybe we want to see all items that were checked out on a specific day, ordered chronologically by the day they were returned.

```
select * from inraw where date(cout) = '2012-02-29' order by cin;
```

In English:
“show me all the data about entries from the “inraw” set that were checked out on February 29, 2012 and arrange them in order of the day they were returned”

Or if we want to do this in descending order, we can add the descriptor DESC to the end of the query. The default option is ascending, or ASC.

```
select * from inraw where date(cout) = '2012-02-29' order by cin DESC;
```

We can also have hierarchical ordering, for instance putting something in order of date and then alphabetically for items that are returned on the same day. Additional ordering criteria are separated by commas.

```
select date(cin),title from inraw where date(cout) = '2011-01-10' order by date(cin) DESC, title;
```

Converting Natural Language (like English) into MySQL Queries

In English:
"show me the title, itemtype, and cout DateTime of all items from the "inraw" set that were checked out on February 20, 2011 with either "samurai" or "ninja" or "sword" or "fighting" in the title and arrange them reverse alphabetically by title."

solution:

```
select title, itemtype, cout from inraw where date(cout) = '2011-02-20' and (title like "%samurai%" or title like "%ninja%" or title like "%sword%" or title like "%fighting%") order by title DESC;
```

In English:
"show me the title, itemtype, and duration of checkout period in days of all items from the "inraw" set that were checked out on February 20, 2011 and were kept longer than a month and arrange them by the length of the checkout."

solution:

```
select title, itemtype, TIMESTAMPDIFF(DAY,cout,cin) from inraw where date(cout) = '2011-02-20' and TIMESTAMPDIFF(DAY,cout,cin) > 30 order by TIMESTAMPDIFF(DAY,cout,cin) ASC;
```

In English:
"show me everything about items from the "inraw" set that are the itemtype "slides" or "microfilm"." (Look at the list of item types)

lame solution:

```
select * from inraw where (itemtype = "acslide" or itemtype = "arslide" or itemtype = "ahmfm" or itemtype = "armfm" or itemtype = "dhmfmnp" or itemtype = "drmfper");
```

awesome solution:

```
select * from inraw where (itemtype like "%slide%" or itemtype like "%mf%");
```

More Advanced Queries

```
select floor(deweyClass/10)*10 as dewey,
SUM(CASE WHEN year(cin) = '2005' THEN 1 ELSE 0 END) as yr2005,
SUM(CASE WHEN year(cin) = '2006' THEN 1 ELSE 0 END) as yr2006,
SUM(CASE WHEN year(cin) = '2007' THEN 1 ELSE 0 END) as yr2007,
SUM(CASE WHEN year(cin) = '2008' THEN 1 ELSE 0 END) as yr2008,
SUM(CASE WHEN year(cin) = '2009' THEN 1 ELSE 0 END) as yr2009,
SUM(CASE WHEN year(cin) = '2010' THEN 1 ELSE 0 END) as yr2010,
SUM(CASE WHEN year(cin) = '2011' THEN 1 ELSE 0 END) as yr2011,
SUM(CASE WHEN year(cin) = '2012' THEN 1 ELSE 0 END) as yr2012,
SUM(CASE WHEN year(cin) = '2013' THEN 1 ELSE 0 END) as yr2013
```

```
from inraw
where deweyClass is not null
group by floor(deweyClass/10)*10;
```

In English:
"show me how many items are checked out in years 2005, 2006, 2007, 2008, 2009, and 2010 from the inraw set where the dewey class isn't NULL and then separate these into dewey decimal groups of 10."

Using COUNT function:

```
select COUNT(*),COUNT(IF(title = "catch 22", 1, NULL)) from inraw;
```

In English:
"Show me the total count and the number of things that have the title "catch 22" from the inraw set."

Now that you understand how to form different types of queries, try to explore the dataset in order to find interesting patterns. Think about what you want to search for in English and then translate it into code like we've seen. This will allow you to keep track of complicated searches and will also help you to explain your process to the rest of the group as we move forward.