Scenario:

You work for a DVD rental company. Your manager has asked you to pull some data on the company, its inventory, and customers. You will complete this by developing and running SQL queries.

Initialize Your Database (NOTE: If you already did this in Project 1, you do not need to do this step again):

1. Download *RentalCompany.sql* from Doc Sharing.
    a. Run the script to create tables and data. You should not receive errors. There may be a few warnings, but that is okay.
    b. To verify you created all tables, run the following SQL:

    SELECT TABLE_NAME, TABLE_ROWS, TABLE_TYPE, CREATE_TIME FROM `information_schema`.`tables` WHERE `table_schema` = 'rentalcompany' AND TABLE_TYPE = 'BASE TABLE';

    Your results should look like the table below. This will help verify that you properly created the schema/database and tables.

actor	200	BASE TABLE
address	603	BASE TABLE
category	16	BASE TABLE
city	600	BASE TABLE
country	109	BASE TABLE
customer	599	BASE TABLE
film	1000	BASE TABLE
film_actor	5462	BASE TABLE
film_category	1000	BASE TABLE
film_text	1000	BASE TABLE
inventory	4581	BASE TABLE
language	6	BASE TABLE
payment	16086	BASE TABLE
rental	16005	BASE TABLE
staff	2	BASE TABLE
store	2	BASE TABLE

Directions:

Note: *For any UPDATE, INSERT, or DELETE statements that you construct as your answers to the questions below, you MUST provide and label the following components:*

- *Initial validation section with your SELECT statement*
- *Run section with transaction, commit, and rollback statements*
- *Final validation section with you SELECT statement*
- *You must have a semicolon after every statement so I can highlight several statements at once and run them at once.*

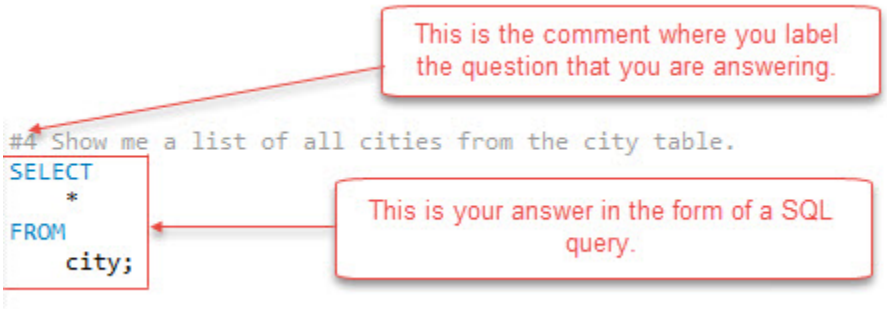
These "sections" were discussed and demonstration in the tutorials for the chapters.
Failure to construcs your answers in this format will result in significant point deductions.

1. Modify *film* table by increasing the *rental duration* by 4 days for films that have the word *Forever* in their *title*.

2. Modify *customer* table by changing the *store ID* to 1 and *active* to 0 for all customers who have *Barnett* as their *last name*. Do not specify first names in your SQL statement.
3. Add one new *customer* and provide data for all columns on the table for the new *customer*. Come up with your own data to enter (ex: enter your name as first and last name).
4. Create a new *actor* record on the actor table for *Kirk James* by copying the record for *Alec Wayne*.
5. Delete *language IDs* from the *language* table that have no *films* assigned to them.

Grading:

- Each number above will be weighted 10 points for a total of 50.
- Each statement must have a semicolon at the end to signify the end of the statement.
- In addition to meeting all the specified criteria under the *Directions* section, you must use the *Beautify* feature in MySQL so your SQL is easily readable.
- Points will be deducted if each question is not clearly labeled within your SQL file. You will do this by commenting out the descriptive line with the pound sign as shown in the *MySQL Expressions* tutorial video. See example below. Your final file will have a comment, then sql statement, then comment, then sql statement, etc.



- To grade your file, I will open your *.sql* file in my version of MySQL and click run. If your SQL does not run, points will be deducted. Next, I will determine if your SQL was written correctly and returned the correct data results.

Complete all of Project 4 using MySQL and turn it in to the Project 4 Dropbox as a .sql file named "*YourLastName*Proj4.sql" by the due date stated in the Course Schedule