BUSA 326 – Spring 2017 - Project 5

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:

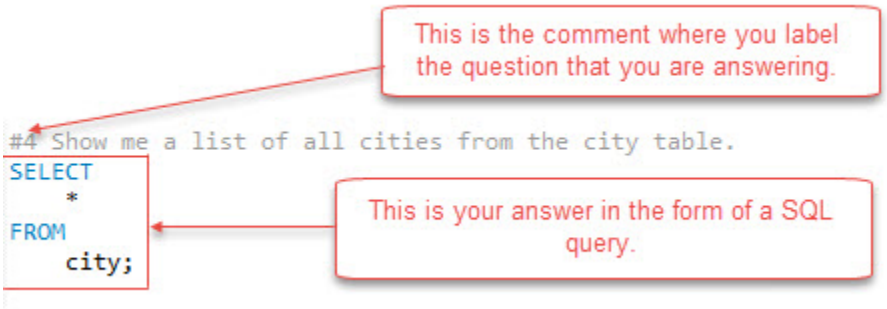
1. Create a query to demonstrate use of the *EXISTS* keyword. List all the actors (*ID, First Name, Last Name)* from the actors table whose actor ID exists in the *film_actor* table. Only list those that have a *film ID* greater than 400. You must construct your statement like it was demonstrated to you in the tutorial for Chapter 18.
2. Create a query to demonstrate the use of *NOT IN.* List all columns from the category table where the name equals *Comedy* and name is not in *Drama* or *Action.*
3. Create a query to demonstrate the use of *NOT IN* and *AND.* List all columns from the category table where the name is not in *Comedy* or *Action* and category ID is 3 or *2.*
4. Create a query to demonstrate the use of a *CASE* expression. List the *film ID* and *title* from the film table. Include a third column named *RatingText. RatingText* should display *Parental Guidance Suggested* when the

rating on the table is *PG, General Audiences* when *G*, and *Restricted* when R. Otherwise, *RatingText* should display *Nomatch* if it does not match any of this criteria. Only show the rows where the *film ID* is greater than 950.

5. Create a query to demonstrate the use of a *CASE* expression in a *WHERE* clause. List all customers who are *Inactive.* Inactive is signified by a *0* in the *active* column of the customer table.

Grading:

- Each number above will be weighted 10 points for a total of 50.
- 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 5 using MySQL and turn it in to the Project 5 Dropbox as a .sql file named "*YourLastName*Proj5.sql" by the due date stated in the Course Schedule