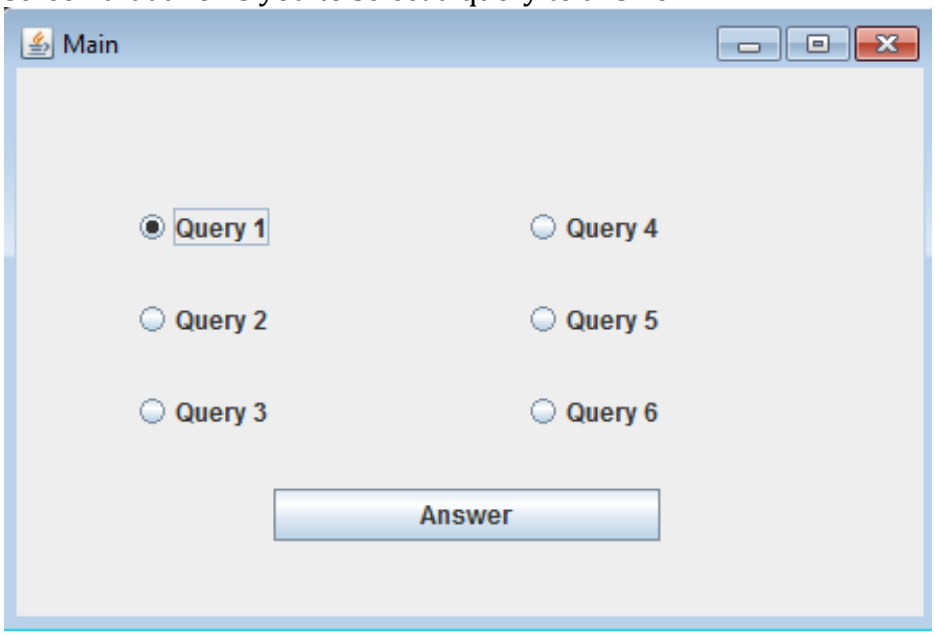
SQL Assessed Coursework

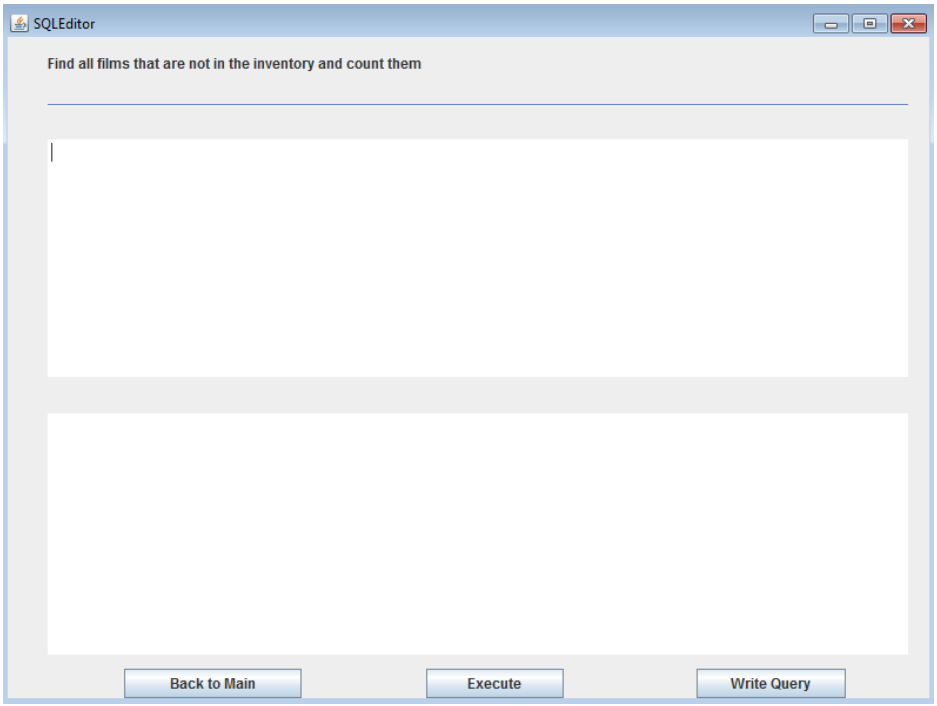
For this exercise you are given a portable JAVA application with an embedded SQLite database. The application comes in the form of a JAR file, which is portable and should run on any platform. In the unlikely case that you face any problems running the JAR file on your machine, you can run it on the lab machines where it works fine. The JAR file is runnable and the application should launch by double clicking it. Otherwise, launch a terminal and execute *java -jar Database.jar* in the directory where the JAR file is. The JAR file can be found here:

https://imperiallondon-my.sharepoint.com/:f:/g/personal/ge14_ic_ac_uk/Er6SkcZEp9RMjgAUxM3DzPMB37jDKJDkCMOICkujb1nFjA?e=YWiwqm

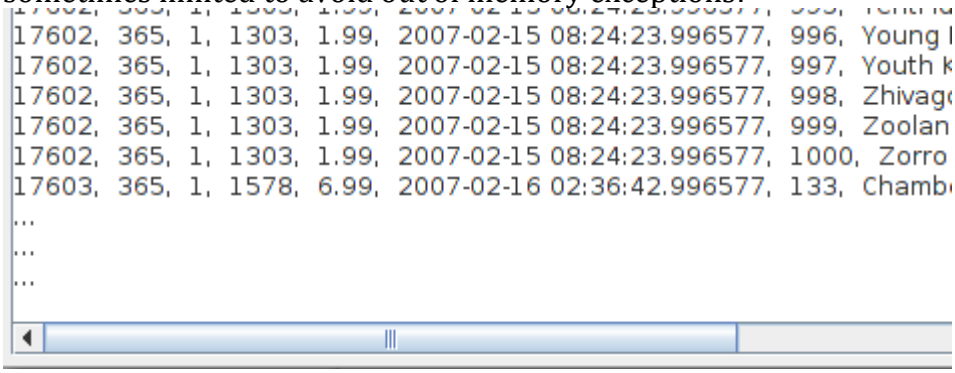
When you first launch the application you should be able to see the following screen that allows you to select a query to answer:



Clicking answer should launch the following screen:



The query that should be answered is written on the top of the screen. Write your query in the first white text box and hit Execute. The "Write Query" button writes your query in a file "Qx.txt" where x is the number of the query. This button is only enabled when your query is correct. Moreover, the result set is sometimes limited to avoid out of memory exceptions:



None of your submitted queries should exceed this limit.

Task: You are to write queries on a DVD rental database. The schema is listed on the next few pages (and can also be inspected on the database).
Result: You need to submit the resulting queries. Please submit a text file for each query. The files are generated when you click "Write Query" and are named Q1.txt, Q2.txt, Q3.txt, Q4.txt, Q5.txt and Q6.txt.

Data Description

The database you will use contains information about DVD rentals (as you used to do before Netflix). The tables contain information about customers, stores and the films customers rent.

Database Schema

- actor – stores actor data including first name and last name.

- film – stores films data such as title, release year, length, rating, etc.
- film_actor – stores the relationships between films and actors.
- category – stores film's categories data.
- film_category- stores the relationships between films and categories.
- store – contains the stores data including manager staff and address.
- inventory – stores inventory data.
- rental – stores rental data.
- payment – stores customer's payments.
- staff – stores staff data.
- customer – stores customer data.
- address – stores address data for staff and customers
- city – stores the city names.
- country – stores the country names.

Queries

a. Find all films that are not in the inventory and count them.
b. Count the number of transactions each staff has been processing and find the staff member (id) with the biggest number of transactions and also the staff member with the biggest sum of the transaction value.
c. Find all stores with more than 300 customers. Report the ID of the store.
d. Find all customers who spent more than 200. Report the ID of the customer as well as the sum spent.
e. Find the films whose rental rate is higher than the average rental rate. Use a subquery and count the number of films.
f. Find films that have return date between 2005-05-29 and 2005-05-30 and report the movie titles. Use a subquery.

Sample Answers

a.

count(film_id)
42

b.

most_transactions	**biggest_sum**
2	2

c.

store_id
1

d.

customer_id	**total**
148	211.55
526	208.58

e.

count(film_id)
659

f.

Title
Swarm Gold
Whale Bikini
Microcosmos Paradise
Escape Metropolis
Borrowers Bedazzled

where the result set is more than 5 rows, only 5 rows are provided

Any problems with the application should be reported on Piazza, so we can act accordingly