SQL: Lab 3

1. Using a Noncorrelated Subquery (Postgres):

The **jupiter.order_fact** table contains information about orders that were placed by customers. Create a report that **lists the retail customers whose average retail price exceeds the company average retail sales**.

a. Using Postgres, write a query that displays the **average of Total_Retail_Price** for all retail prices in the table jupiter.order_fact:

- Subset the rows so that only the retails sales are included (Order_type=1)
- You will need to cast the variable Total_Retail_Price using the following:

```
CAST(REPLACE(REPLACE("Total_Retail_Price",'$',''),',','')
AS DOUBLE PRECISION)
```

What's the value for the **average of Total_Retail_Price**? (round your answer to two decimal places)

b. Write a query that displays Customer_ID, AVG(Total_Retail_Price) for those **customers whose average retail price exceeds the company average retail price**. The query should do the following:

- Display the values for Customer_ID and the AVG(Total_Retail_Price). Name the second column MeanPrice. Remember that you will need to cast the Total_Retail_Price column.
- Subset the rows so only the retail sales are included (Order_Type=1)
- Include only groups where the customer’s average retail price exceeds the company average.
- Order by descending MeanPrice

What is the value of ***MeanPrice*** in the ***fourth*** observation on the report? (round your answer to two decimal places)

2. Using a Noncorrelated Subquery (Postgres):

Each month, a memo that lists the employees who have birthdates for that month is posted. Create a report for the month of September and list Employee_ID and the first and last names for all employees who have birthdates during the month of September.

You can find **Employee_Name in the jupiter.employee_addresses** table and **Birth_Date in the jupiter.employees** table. Both tables contain the column **Employee_ID**.

a. Create a query that returns a **list of employee IDs** for employees with a **September** birthdates. The query should do the following:
   - Display Employee_ID numbers.
   - Use the jupiter.employees table.
   - Return only employees whose birthdate (Birth_Date) is in the month of **September**.
   - Order by ascending Employee_ID
   - To convert the column Birth_Date to a date column, you will need to use: `TO_DATE("Birth_Date", 'MM/DD/YYYY')`

   What is the value of ***Employee_ID*** in the ***fourth*** observation on the report?

b. Using the query in 2.a. as a noncorrelated subquery, write a query that displays the employee IDs and the Employee_Name. The final query should do the following:
   - Display Employee_ID and Employee_Name
   - Use the jupiter.employee_addresses table
   - Select Employee_ID only for employees who had birthdates in **September**
   - Order the final results by ascending Employee_Name

   What is the value of ***Employee_Name*** in the ***fourth*** observation on the report? (**Use Copy feature on DataGrip to copy-paste the exact value into Moodle**)

3. Queries (SQLite):

Find all movies that are **NOT** one of the following genre categories:

- 'Comedy','Comedy/Drama','Exercise','Fantasy','Foreign','Animation',' Horror','TV Classics','VAR','War'
- Display only the movie name
- Order the report by **descending** movie name

What is the value of **Movie_Name** in the ***17th*** observation on the report? (**Use Copy feature on DataGrip to copy-paste the exact value into Moodle**)

4. Queries (SQLite):

Find the **names of the people** who own the following movies:

- Movie_ID = '20372','8727','31670'
- Note that in the table **people_movies**, the column ID refers to the ID of the table, and person_id refers to the ID of the person. Refer back to the last slide of Class #3 for a diagram of the tables.
- Order the report by **ascending person name**

What is the value of **name** in the ***first*** observation on the report? (**Use Copy feature on DataGrip to copy-paste the exact value into Moodle**)