**CSCI 4125/5125 Course Project**

**Data Models and Database Systems**

**Fall 2023**

**Course Project**

**Phase 5: Introductory SQL (9/22)**

**Due: Sunday, 10/1 @ 11:59pm**

**Reading:** Silberschatz Chapters 3.1 – 3.4, 3.6 – 3.9, 4.3 – 4.5

**Submission Guidelines:**

1. This assignment is worth 100 points for all students.

2. It is your responsibility to make sure all files are readable and submitted on time.

**Submission:**

Task1. Submit a single .sql file containing your queries and a single .txt file containing your output. Make sure to label queries (as comments) and output with the appropriate problem number.

Task2. Submit your Java source code and a screenshot of the output after running the code.

**Task 1. SQL Retrieval Queries [42 points]**

**Submit**: your answers in a single .sql file named and the outputs from each of your queries in a .txt file.

--1. [4 pts] Find all products (their names) with a price greater than $100.

SELECT P\_NAME FROM PRODUCT WHERE P\_PRICE >100;

--2. [4 pts] Find all Customers who were not referred by another customer.

SELECT C\_ID,C\_NAME FROM CUSTOMER

WHERE C\_REFERRERID IS NULL;

--3. [4 pts] Find the average rating for all reviews.

SELECT AVG(R\_RATING) FROM REVIEW;

--4. [4 pts] Find all the “Ticket” from product and sort the output by the price (any direction). Hint: remember the LIKE operator.

SELECT P\_NAME,P\_PRICE FROM PRODUCT

WHERE P\_NAME LIKE '%Ticket%'

ORDER BY P\_PRICE DESC;

--5. [4 pts] Find the minimum and maximum date of births amongst all customer. Use only a single query that returns a single row (ex. 01-JAN-70, 01-JAN-99).

SELECT MIN(C\_DATEOFBIRTH)AS MINDOB,MAX(C\_DATEOFBIRTH)AS MAXDOB

FROM CUSTOMER;

--6. [4 pts] Find all reviews that contain the word "Great". Make sure to remove case sensitivity Hint: remember the LIKE operator.

SELECT \* FROM REVIEW

WHERE UPPER(R\_TEXT) LIKE '%GREAT%';

--7. [4 pts] Find the number of reviews for P05 with a rating between 3 and 5.

SELECT COUNT(\*) FROM REVIEW

WHERE R\_PRODUCTID='P05' AND R\_RATING BETWEEN 3 AND 5;

--8. [7 pts] Find the name of the product with the highest price. Do not hardcode any prices or other values ‚Äì you must use SQL without assuming you know the current database snapshot.

SELECT P\_NAME FROM PRODUCT

WHERE P\_PRICE = (SELECT MAX(P\_PRICE) FROM PRODUCT);

--9. [7 pts] Find the product names with a price greater than the average overall price of products + 25% (i.e., greater than 1.25 \* average price). Do not hardcode any prices or other values ‚Äì you must use SQL without assuming you know the current database snapshot.

SELECT P\_NAME FROM PRODUCT

WHERE P\_PRICE > 1.25\*(SELECT AVG(P\_PRICE) FROM PRODUCT);

**Task 2. Query Processing [13 points]**

One of benefits of SQL is the program-data independence supported by the relational model (or the DBMS). However, to fully understand the power of SQL, it is helpful to know how the equivalent could be implemented in the flat file model. In this task, you will write a Java method, not SQL, that is the flat file model solution for Query #8 in Task 1. Your method should include the following:

* Read the relevant .csv file(s) given in Phase 4. You can hardcode the file name(s).
* Process the input to compute the query result. When you process the input, you will now have to hardcode the column positions.
* Print the results to the console. As a sanity check, your results should be the same as in Task 1.
* None of your code will use SQL – Java only.
* *Hint*: We have used some useful methods in previous phases of the project, e.g., trim(), split(), etc.
* *Hint*: It may be useful to store results in a data structure, such as a HashMap, that you can later print after you after done all of the processing.

**Submit:** A single Java file containing a method (you can use more than one) besides main() that solves only Query #8 from Task 1 and a screenshot of the output printed to the console.