DSC 270 - Data Manipulation
Fall 2023
Database Project

This project is to be done on your own without help from others. The exception is that you can ask the professor for help. You may use your notes, the slides on Moodle, and surf the Internet to learn techniques, but not to look for answers.

Part 1: SQL Queries: 14 queries @ 1 point each marked right or wrong = 70%.

Write SQL queries for the **gravity_bookstore** database. You should test your queries against the database to make sure they work! Submit your answers for Part 1 as a pdf file on Moodle.

A) Assume that you have logged in to mysql. What command will list the databases?
> SHOW DATABASES;

B) Assume that you have logged into mysql and have issued the command **use beverages;** to use the **beverages** database. What command will give the information about the table **xyz** in the **beverages** database?
> DESCRIBE xyz;

For questions C-L, use the **gravity_books** database and the diagram on Moodle for it.

C) Write a query that outputs the **number of records** in the **book_language** table where **language_name** contains the word **English**.
> SELECT COUNT(language_name) FROM book_language WHERE language_name LIKE '%English%';

D) Write a query that outputs the **book_id** of records in the **book_author** table for authors with 8888 < **author_id** < 9000.
> SELECT book_id FROM book_author WHERE author_id BETWEEN 8889 AND 8999;

E) Write a query that outputs the **book_id** of records in the **book_author** table for authors with **author_id** > 900 or **author_id** < 100.
> SELECT book_id FROM book_author WHERE author_id>900 OR author_id<100;

F) Write a query that outputs the average number of pages from books with **publisher_id** 628 in the **book** table.
> SELECT AVG(num_pages) FROM book WHERE publisher_id=628;

G) Write a query that outputs the unique **author_name**s from the **author** table.
> SELECT DISTINCT author_name FROM author;

H) Write a query that outputs the book_id, title, and author_id for each book.

> SELECT book.book_id, book.title, book_author.author_id FROM book JOIN book_author WHERE book.book_id=book_author.book_id;

I) Write a query that outputs the book_id, title, author_id, and author_name for each book.
> SELECT book.book_id, book.title, book_author.author_id, author.author_name FROM book JOIN book_author ON book.book_id = book_author.book_id JOIN author ON book_author.author_id = author.author_id;

J) Write a query that outputs the **customer_id** and **first_name** for the first 10 records in the **customer** table when the table is sorted by **first_name** in reverse alphabetical order (i.e. from z to a).
> SELECT customer_id, first_name FROM customer ORDER BY first_name DESC LIMIT 10;

K) Write a query that outputs the **publisher_id** and maximum **num_pages** of books for each **publisher_id** in the **book** table.
> SELECT publisher_id, MAX(num_pages) FROM book GROUP BY publisher_id;

L) Write a query that outputs the **status_id** and lowest **order_id** for each **status_id** > 3 in the **order_history** table and have the output sorted from highest **status_id** to lowest **status_id**.
> SELECT status_id, MIN(order_id) FROM order_history GROUP BY status_id HAVING (status_id) > 3 ORDER BY status_id DESC;

M) Use the **student_test_db** database for this question. Write a query that puts your last_name, first_name, favorite_color, favorite_number, and birthdate (as a DATE object) into the **student_data** table.
> INSERT INTO student_data VALUES("Shrestha", "Pranjal", "Maroon", 05, DATE("2003-09-09"));

N) Use the **student_test_db** database for this question. Write a query that **UPDATES** the favorite number for the person **with your last_name and first_name** in the **student_data** table to be -999.
> UPDATE student_data SET favorite_number=-999 WHERE last_name="Shrestha" AND first_name="Pranjal";