Database Systems Project

This is your final project and will be worth 10% of your class grade. Use everything you have learned to build a database from scratch!

Description:
It's time to put everything together from this class and build your own database. This project can be divided into three distinct phases:

Phase #1 Schema: Design and populate a database.
Phase #2 Query: Create useful SQL queries.
Phase #3 Python: Write a Python front-end program.

Grading Scale
Each phase is divided into levels for grading purposes.

Grade	Levels working and completed
90-100	Do Schema Level 2, Query Level 2, and Python Level 1
80-90	Do Schema Level 2, Query Level 2 **or** Do Schema Level 1, Query Level 2, Python Level 1
70-80	Do Schema Level 1, Query Level 1, Python Level 1
60-70	Do Schema Level 1, Query Level 1
0-60	Less working and completed than level 60-70. Nothing turned in will earn a zero grade.

Phase #1 Schema:
This is the portion of the project where you get to choose what goes into your database, as well as how the data relates. You will not have to generate data from scratch unless you want to do so. Here are two ideas you can use to create and populate your unique database:

1. Make up a fake business! What do you sell? What customers do you track? How many things do you have in stock? How much do they cost? Use https://www.mockaroo.com/ or find a similar website to look at and generate the fake data you can create to create your fake business.
2. Use real data and create a database designed to answer questions about that data. Use https://www.kaggle.com/datasets to grab interesting real data such as the "Witch Trials Dataset."

3. Use an LLM to create fake data for you! (LLM can only be used to create data, not create a schema.)

Creating the schema involves creating the ER diagram, “forward-engineering” the ER diagram into a database schema, and populating each individual table with data. Draw an ER diagram (you must submit your ER diagram and database schema with your final submission) and create tables.

Note 1: Before you forward-engineer your database, make sure the database name starts with a lowercase letter. (Few Years ago, a student ran into an obscure bug when using an uppercase letter.)

Note 2: Your database must be sufficiently different from the example databases we used/designed in class. This means movie rental, university, and car business.

Level	**Tasks Working and Completed**
Level 1	Tables are populated with at least 200 data rows in total (across all tables). You have at least 4 tables in your database. All tables should relate in some way.
Level 2	Tables are populated with at least 400 data rows in total (across all tables). You have at least 8 tables in your database. All tables should relate in some way.

Phase #2 Query:
Queries can be divided into queries that read the database and queries that modify the database. Modification can be harder due to foreign key constraints.

Once Phase #1 is completed, create a .sql query file in Workbench/Sandbox and complete queries according to the levels below. Note that ***each query should have a comment above it in the .sql file explaining (1) what it is supposed to be doing, (2) if it works or doesn’t work, and (3) a sample of what it returns (screenshot).***

Level	**Tasks Working and Completed**
Level 1	You successfully write and execute 4 useful “read-only” SQL queries on your database. 1. One query must be a basic SELECT/FROM/WHERE query 2. One query must answer a question that needs to join 2 or more tables 3. One query must have a subquery 4. One query must have an aggregate function and GROUP BY clause
Level 2	In addition to level 1, you successfully write and execute 4 more “modification” SQL queries on your database.

	1. One query must add a record to one or more tables. 2. One query must delete a record from one or more tables. 3. You have foreign key ON UPDATE constraints that make sense on a table/tables and write an SQL query to demonstrate how one of them works. 4. You have foreign key ON DELETE constraints that make sense on a table/tables and write an SQL query to demonstrate how one of them works.

Phase #3 Python:
Create a Python script with separate functions for all SQL queries you have written. Each function must have a docstring describing what it does and take the db connection as input (similar to the Sakila python example). At least half of the functions must accept user input via input() to change parameters in the SQL queries. Use the sakila_starter.py example as a starting point.

Level	**Tasks Working and Completed**
Level 1	You create python functions to run your Level #1 and Level #2 queries. At least two functions of the "read-only" queries and two functions of the "modification" queries must accept user input() to change parameters of the SQL queries dynamically.

Final Deliverables:
Navigate to our class on D2L and find the link for Final Project. There are four deliverables you should zip up and submit:

1. **Export dump file** of both database and contents (phase #1)
    a. Follow these directions and make sure you select the box to "Export to Self-Contained File": https://dev.mysql.com/doc/workbench/en/wb-admin-export-import-management.html
    b. Must submit your ER diagrams/Database Schema
2. **Queries file** (name it my_queries.sql). Be sure it contains comments above the queries stating (1) what it is supposed to be doing, (2) if it works or doesn't work, and (3) a sample of what it returns (screenshots). (phase #2)

3. **Python file** (named front-end.py) that connects to your database and runs your queries (phase #3). Attach a video (screen recording) of your outputs.

4. **Project log file** (named README.txt) stating:

a. A description of the type of data in your database and from where you got the data (or else, where you generated the fake data.)
b. How to execute your files.
c. The grade you expect to get and why based on the grading criteria
d. Challenges you faced