Final Project (26% of total grade)

Purpose: In this final project, you will design, build, and document a MySQL database system. Your project will cover various database-related concepts and practices, including views, triggers, stored procedures, functions, normalization, data types, keys, and constraints. This project is worth 26% of your final grade.

Overview

Proposal - Part 1 (6%)

Due Week 12
Deliverables:

A. Written Proposal Document, as a shared Google Doc, not coded
   a. Real World Scenario
   b. Problems & Features
   c. Architecture Description
      i. Database design, including database tools

Presentation / Documents - Part 2 (5% / 15%)

Due Week 14
Deliverables:

A. 5-minute presentation in class
B. SQL file to demo your database and features
C. Presentation deck, as PDF
D. Supporting materials document, as PDF

Part 1: Proposal (6%)

Purpose: Your assignment is to propose a database architecture, (which you will later build, and present to the class). You will be creating a real world scenario that will use a database to solve real world problems with data. You will be the individual managing this database.

Name your file: HTTP5126-FinalProjectProposal-*LastNameFirstName*, replace *LastNameFirstName* with your name as displayed in Blackboard.

Submission: Make a copy of the proposal template provided. Follow the instructions in both this document and the proposal template. When completed, save your document as a PDF file, name your file as above. Upload and submit to Blackboard (Assessments > Final Project > Proposal).

Submission Requirements:

- Make a Copy of the Template:
  - Open the provided Google Docs template.
  - In the top menu, click “File” > “Make a copy” to create your own version.
  - Rename the document to the file name above
- Share Your Document:
  - Click the “Share” button in the top-right corner.
  - Enter my email address: mbebis.edu@gmail.com
  - Set the access level to “Editor.”
- Submit the Link:
  - Once shared, copy the link to your document and submit it to the Final Project Proposal submission box on Blackboard.

Important: All work must be completed directly inside the Google Document. I will be using a tool to review the document’s edit history, so make sure all changes and contributions are recorded within the document itself. Copying and pasting from other sources may affect your evaluation.

Proposal Overview (24 marks)

Real World Scenario (2 marks)

Imagine yourself as the person responsible for managing a database. Describe the real-world context by explaining the environment and your role in handling the data. Your scenario should feel realistic and relevant to the type of database you're designing.

- Examples:
  - Managing ticket sales and showtimes for a movie theater.
  - Organizing customer and product information at a pet store.
  - Tracking personal events and hobbies for you and your friends.

Identifying Problems (2 marks; 1 per problem)

Describe two ways people using your database might want to interact with it. Focus on a specific user group — for example, customers, managers, or employees. What real-world problems are these users trying to solve in your scenario? For each problem:

- Explain the problem from the user's perspective.
- Identify the tables that would be relevant to solving it.

Designing Features (4 marks; 2 per feature)

For each problem you identified, describe a feature your database could offer to solve it using SQL. Each feature should be a clear solution to one problem. If a problem needs two distinct solutions, you may have combined two problems into one. Similarly, avoid proposing two features that achieve the same outcome in different ways.

- These are examples of problems that need features to solve them:
  - Movie Theater: When the list of movies changes (e.g., a new movie is added or an old one is removed), how can the database stay up-to-date without disrupting existing data?
  - Sales Tracking: After a sale is made, how can the database accurately track the transaction and update relevant information, like inventory or revenue?
  - Data Entry: When a friend wants to add new data, how can you simplify the process to avoid errors and ensure consistency?

Architecture Description (16 marks)

Outline the proposed architecture of your database. These should relate to your scenario, problems, and features.

Include the justification for the database architecture you created. In other words, state why each table, view, trigger, and function/procedure are necessary for the premise/features you have identified.

Give your database a name (1 mark)

Database Schema (Entity Relationship Diagram) (9 marks)

- Create an ERD of your database including:
  - Table names (1 mark)
  - Column names (1 mark)
  - Data types (1 mark)
  - Keys (1 mark)
  - Constraints (1 mark)
  - Relationships between tables (1 mark)
- Normalization (1 mark)
- Justification of table architecture (2 marks)

Database tools (6 marks)

- Implement additional tools to support your database's functionality:
  - At least one view (1 mark)
  - At least one trigger (1 mark)
  - At least one function or procedure (1 mark)
  - Justification of tool architecture — explain how each tool supports the identified problems or features (3 marks; 1 per tool)

Notes

- The steps above do not need to be completed in order, they should inform each other and make sense when combined.
- Feel free to reuse the ideas from your lab 6/7, use a similar idea to your Pet Project from your JavaScript Frontend course, or come up with something completely new.
- If you decide to reuse your Lab 6/7 tables, you must ensure the scenario and problems make sense with the database you created previously. You will most likely need to alter those tables by adding to them or by adding new tables to appropriately solve the problems you define.
- Essentially, do not just squeeze a scenario, problems, and features into an old database, you are free to alter that schema so all the pieces described above make sense together.

Example Scenario

Imagine you are managing the data for an animal sanctuary. The sanctuary has a range of operations that could benefit from well-designed database features. Here are some potential problems that your database could help solve:

- **Animal Management:** When a new animal joins or leaves the sanctuary, how can you ensure all relevant tables are updated efficiently?
- **Donations Tracking:** Donors can earmark their contributions for specific types of animals. How can you track and view the total donated funds per animal type?
- **Exhibit Scheduling:** Some animals' sleep schedules determine when their exhibits are open. How can you generate an exhibit schedule based on this information?
- **Feeding Schedule:** Animals have unique dietary requirements that dictate their feeding times. How can you create a feeding schedule that reflects these needs?
- **Visitor Calendar:** The sanctuary is open to the public only during certain hours and closes on specific holidays. How can you generate a calendar showing public opening times?
- **Employee Scheduling:** Employees work different shifts, but must not exceed 88 hours over any two-week period. How can you create a schedule that accounts for these constraints while ensuring proper coverage across shifts?

Part 2: Presentation & Supporting Documents

Your final assignment will be a five-minute presentation to the class, with supporting materials. Since you have 5 minutes, please be prepared with all materials to present when it is your turn.

Name your zip folder: HTTP5126-FinalProjectProposal-*LastNameFirstName*.zip, replace *LastNameFirstName* with your name as displayed in Blackboard.

Submission: You will be submitting a ZIP file with all the requested documents outlined near the bottom of this document, under the header **Deliverables**.

Presentation (5%)

You will describe your scenario, database, and, assuming you got it up-and-running, demonstrate your features. You can also present additional materials for context, such as a wireframe of a user-interface that would interact with the database. It is equally valid to present your failures as it is to present your successes.

Your presentation grade will be based on:

- Use of visual aids
- Presentation structure & flow
- Communication of content
- Preparedness
- Adherence to time limit (5 minutes)

These presentations will be given in class on the last week of classes (Week 14). Here is the suggested breakdown of the slides for the 5-minute presentation:

1. Overview of Problem-statements - To give context, succinctly explain the scenario you have created and the 2 problems you are solving for. (1 minute / 1-3 slides)
2. Database entity diagram - Show and explain your tables and the relationships between those tables using a fully normalized entity relationship diagram. (1 minute / 1-3 slides)
3. Code - Explain and demo your features using code. 2-3 minutes

Supporting Documents (15%)

Your supporting documents will include the following:

1. The PowerPoint slides as a pdf (2%)
2. Your SQL demo code in a file (8%)
3. A Final Report (5%)

Slide Deck
Your slide deck grade will be included in the Supporting Documents section. It will mostly be considered for its effectiveness of a visual aid like its design, content, and structure.

SQL demo code
The SQL code will consider the following:
1. Working code. (2%)
2. Code containing all tables, queries and/or procedure calls required to demo your features. (2%)
3. Database architecture, does the structure of tables and relationships between tables match the designed schema from the proposal (if changes are made to design indicate them in the Report). (2%)
4. Comments explaining the features code created to solve the problems. Also include comments when needed for any complex code. (1%)
5. Easy to read / well-formatted code. (1%)

*Errors in code will not be debugged. If you are submitting code with errors you must explain the issues and why they were unable to be remedied in the final report.

Final Report
The Report will be considered regarding the following:
1. Using your Proposal document, make any changes given in feedback. (1%)
2. Adding to the proposal document, create a new section titled ‘Retrospective’ *(less than 500 words)* and include the following:
    a. An explanation of any changes made from the proposal. If no changes were made, indicate this as well. (1%)
    b. Any challenges encountered during the design and implementation phases. How were those challenges addressed and resolved (or not resolved)? (1%)
        i. Has your code actually solved the real-world problems indicated? If so, how? If not, what prevented you from doing so?
    c. Future plans for this database. What are the next steps you would take to build on or improve your design/implementation? Are there any new features you would want your database to include? (1%)

*Please clearly indicate each section with a header.

Your submission will also be considered for if your code has actually solved the real-world problems indicated. If they are not solved this should be explained in the challenges faced section.

Deliverables

Follow these submission requirements *exactly.*

Upload and submit to Blackboard (Assessments > Final Project > Supporting Documents). Submit the following:

- SQL demo code
- Presentation deck as a PDF document (max 5 slides)
- Link to Final Report Google document

Your submission to blackboard should be a zipped folder with the requested files, including the file name format. When unzipped the folder structure should look like below.

- **HTTP5126-FinalProject-*LastNameFirstName*.zip**
  ↪ **Script-*LastNameFirstName*.sql**
  ↪ **Deck-*LastNameFirstName*.pdf**

Note: When zipping the folder, select all the files individually, then zip. Do not zip the folder directly as this will result in a nested folder.

In the submission comments include the link to your Final Report Document. This document should be created and shared in the same way the proposal was created and shared. Using a Google doc to write the report and shared with me as an editor. See submission instructions from Part 1.

- **Final Report Document Link**

Submission Deductions

- Incorrectly named zip folder [-0.5]
- Incorrect folder structure when unzipped [-0.5]
- Incorrect name of any file [-0.25]
- Report containing more than 500 words [-0.25]
- Report containing more than 600 words [-1]

Conventions Overview

Convention Deductions [-0.25% per]

- General
  - Full words not abbreviations and acronyms
  - Avoid redundancy, do not prefix names with the name of their parent
  - Names should be meaningful and self-explanatory
  - db/table/column name should reflect their real world purpose
  - Names should be lowercase since SQL keywords are UPPERCASE
  - snake_case: underscore_in_place_of_spaces
- Databases
  - Singular name that describes information held in db
- Tables
  - Names should be nouns, 1 or 2 words
  - Table names may be singular OR plural *BUT BE CONSISTENT
- Columns
  - Names should be 1 or 2 words and singular
- Views
  - Names should try not to exceed 4 words (No deduction unless extremely long)
  - Names should describe what the views purpose is
  - When possible include the names of the tables used in the view in the view name
- Triggers
  - Prefix triggers with trg_
  - Name typically in action then table name format (trg_<action>_<table>)
    - Trg_insert_after_movie
  - Examples of trigger actions are:
    - INSERT AFTER
    - UPDATE BEFORE
    - etc.
- Functions
  - Prefix functions with fn_
  - Function name typically in verb then noun format (fn_<verb>_<noun>)
    - fn_get_title
- Procedures
  - Prefix procedures with usp_
  - Procedure name typically in verb then noun format (usp_<verb>_<noun>)
    - usp_get_name

Proposal Overview (24 marks)	**/ 24**
Real World Scenario (2 marks)	**/ 2**
Problems & Features (6 marks)	**/ 6**
Problem 1	/ 1
Solution / Feature 1	/ 2
Problem 2	/ 1
Solution / Feature 2	/ 2
Architecture Description (16 marks)	**/ 16**
Database name	/ 1
Schema (9 marks)	**/ 9**
Table names (1 mark)	/ 1
Column names (1 mark)	/ 1
Data types (1 mark)	/ 1
Keys (1 mark)	/ 1
Constraints (1 mark)	/ 1
Relationships between tables (1 mark)	/ 1
Normalization (1 mark)	/ 1
Justification of table architecture (2 marks)	/ 2
Database tools (6 marks)	**/ 6**
At least 1 view (1 mark)	/ 1
At least 1 trigger (1 mark)	/ 1
At least 1 function OR 1 procedure (1 mark)	/ 1
Justification of tool architecture (3 marks; 1 per tool)	/ 3