1. **Overview**

In this Homework, you will:

1. Come up with a project idea for your web application (that meets project requirements).
2. Learn the basics about databases: designing tables and relationships, primary keys, foreign keys, database constraints, and SQL SELECT statements that can join data from more than one table.
3. Use MySQL Workbench (an open source GUI for MySql, an open source Database Management System) to design a database that supports your web application.

Before starting on this Homework, read the following help documents and tutorials that are associated with this Homework. I will be lecturing on concepts that are presented in these documents and you are responsible for this material.

- How to install MySQLWorkbench
- How to connect to your Temple database (how to discover your database credentials, etc)
- MySQL Workbench Tutorial 1: How to create database tables (with primary keys), enter data, and write single table select statements
- MySQL Workbench Tutorial 2 (continuation of previous tutorial): How to reverse engineer a database, create database relationships (with foreign keys), and how to write select statements that join data from more than one database table

2. **Come Up With Web Application Topic**

Think up a creative idea for a web application project that you will be implementing for the rest of the semester. Your web application functionality will depend on the design of your database and the database must meet certain criteria so that your database works with the Homeworks throughout the rest of the semester. Your Homeworks will culminate in your own individual web application (by the end of the semester).
Here are the criteria for your data model. Each student’s database must have four tables: a user table, a role table, "another table", and an associative table that implements a "many to many" relationship between the user table and the "other" table. While the prescribed database “shape” does limit the choices that a student can select for web application functionality, there are a lot of options that fit the requirements, as you can see below.

### Possible Web Application Ideas

<table>
<thead>
<tr>
<th>What is the topic of the web app?</th>
<th>&quot;User table&quot;</th>
<th>&quot;Other table&quot;</th>
<th>&quot;Associative Table&quot;</th>
<th>Why is this a many-to-many relationship (between the user table and the &quot;other table&quot;)?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Music Concerts</strong></td>
<td>music band</td>
<td>concert hall</td>
<td>Concert: music band schedules a concert at a concert hall</td>
<td>A band can play at many different concert halls. A concert hall can host many different bands.</td>
</tr>
<tr>
<td><strong>Wrestling Tournament Registration</strong></td>
<td>wrestler</td>
<td>tournament</td>
<td>Registration: wrestler registers for a tournament</td>
<td>A wrestler can register for many tournaments. A tournament has many wrestlers registered.</td>
</tr>
<tr>
<td><strong>Home Improvement Quotes</strong></td>
<td>home owner</td>
<td>contractor</td>
<td>Quote: home owner requests quote from a contractor</td>
<td>A home owner can request quotes from several contractors. A contractor can provide quotes for several home owners.</td>
</tr>
<tr>
<td><strong>course registration system</strong></td>
<td>student</td>
<td>course</td>
<td>Registration: student registers for a course</td>
<td>A student can take many courses. A course has many students enrolled.</td>
</tr>
<tr>
<td><strong>Library Borrowing System</strong></td>
<td>library patron</td>
<td>book</td>
<td>Borrow: library patron borrows a book</td>
<td>A library patron can borrow many books. A book can be (over time) borrowed by many library patrons.</td>
</tr>
<tr>
<td><strong>Ecommerce</strong></td>
<td>customer</td>
<td>product</td>
<td>Purchase: customer purchases a product</td>
<td>A customer can purchase many products. A product can be purchased by many customers.</td>
</tr>
<tr>
<td><strong>Sports Blog</strong></td>
<td>sports fan</td>
<td>sports team</td>
<td>Comment: sports fan comments on a sports team</td>
<td>A fan can comment on many teams. A team can be commented on by many fans.</td>
</tr>
<tr>
<td><strong>eCookbook</strong></td>
<td>cook</td>
<td>food item (e.g., brownie)</td>
<td>Recipe: cook contributes a recipe for a food item</td>
<td>A cook can contribute recipes for many food items. A food item can have many different recipes.</td>
</tr>
<tr>
<td><strong>Employee Benefit System</strong></td>
<td>employee</td>
<td>benefit</td>
<td>Selected Benefit: employee selects a benefit</td>
<td>An employee can select many benefits. A benefit can be selected by many employees.</td>
</tr>
<tr>
<td><strong>Travel Blog</strong></td>
<td>traveler</td>
<td>country</td>
<td>Trip: traveler shares experiences about a trip to a country</td>
<td>A traveler can visit many countries. A country can be visited by many travelers.</td>
</tr>
</tbody>
</table>
So that you have a better understanding of your web application will do by the end of the semester, here is a list of the things that you will have done to your web application, once you have completed all the Homeworks.

1. Home page with navigation bar and content that describes what your web application has to offer.
2. List of users (joined with user role table), ability to insert (register), edit, and delete users.
3. List of “other” records, ability to insert, edit, and delete.
4. List of associative records, joined with user data and “other” data.
   ➔ Ability to search for records associated with just a particular user and/or just a particular “other” record and/or attributes from associative table (e.g., date range).
5. Log on and log off functionality. Restricted access to a page within your web site.
6. (You will also have a “blog” page, but this is more related to Homework and project grading, than it is a page you would see on a real web site.)

If you would like to see some examples of web applications that have been implemented in previous semesters, visit my web site (google “sallyk temple”) and click on the “Teaching” tab. Look for links the web applications that have been done by students in previous semesters.

3. Requirements for Project Proposal

Once you have come up with a topic for your web application (such as those listed on the previous page), **answer the following questions about your proposed web application.** These answers will go into a document that you will submit for this Homework. Most of these questions have been already answered for the project suggestions on the previous page, so please compare your answers with those – to make sure that you fully understand the basic concept of a “many to many” database relationship.

1. What is the title of your web application?
2. Who will be users of your web application?
   • What attributes will you store for each of your users?
3. What “other” table will be part of your database?
   • What attributes will be stored for each record of this table?
4. What information will be stored in your associative table (the table that implements a "many to many" relationship between your "other table" and your user table)?
5. Explain how there is a many to many relationship between your user table and your “other” table.
6. What functionality will your web application provide? Your application will provide the following at a very "mechanical" level, but functionally, how will users benefit from your web app?
   • Users will be able to register, log on, and log off.
   • Users will be able to insert (edit and delete) "other" records (whatever you decided to keep in your "other" table).
   • Users will be able to insert (edit/delete) associative records.
7. Write 1-2 paragraphs of “marketing material” that attempts to entice people to become registered users and/or viewers of your web site. This will become the content area of your home page. In your paragraphs, do not explain to me what your web site will be able to do, write text that will entice users to visit your site.
4. Requirements for Data Model

Using MySqlWorkbench, create a database that consists of 4 related database tables (as specified below).

**IMPORTANT: Do not use any SQL KEYWORDS** as table names or field names. Google "SQL keywords" to see what to avoid, but you definitely cannot use these: user, role, password, grant, date... You will get Homework deductions, but more importantly, you will have problems as you attempt to implement your web application.

1. A **user table** and **user role table**. I will be giving you sample code during the semester and my code will expect your table to have the exact design as shown below (table name, field names, field types and other designations under the checkboxes like PK primary key, NN not null, etc). Using the MySql Data modelling tool, create a foreign key from web_user.user_role_id to user_role.user_role_id. Note: your web application will not allow users to modify the user_role table – you (as developer/designer) will enter the records into this table using MySql workbench (in next week’s homework).

![User Table Diagram](image)

![User Role Table Diagram](image)

2. An **"other" table** named according to what you will store in it (can’t be named "other") and including these fields:
   - auto-increment primary key (to uniquely identify a particular "other" record),
   - descriptive character field (must be unique – add database constraint),
   - **At least two more fields** (you choose), **one of which must be a null-able non-character field** (e.g., date, decimal, or integer). Null-able means that it is OK for the user to not put something into that field (optional for the user).
     - Decimal is a good choice for money type fields.
     - Pick Date over DateTime or else you will get an unintelligible real number that stores milliseconds form the beginning of time.
3. An **associative table** that implements a "many to many" relationship between your user table and your "other" table. This table shall be named according to what you will store in it (can't be named "associative"). If you think of your user table as the subject of a sentence and the "other" table as the object of the sentence, then your associative table describes the verb within the sentence. Attributes might be something like "number of items purchased", "when purchased", "discount amount", etc.
   - auto-increment primary key (to uniquely identify a particular associative record),
   - foreign key that references the user table,
   - foreign key that references your "other" table,
   - At least two more fields (you choose), **one of which is a null-able non-character field** (e.g., date, decimal, or integer). Null-able means that it is OK for the user to not put something into that field (optional for the user).

4. **Naming conventions:**
   - Every table shall have a **PK** that is named: `tableName_id`
   - All FKs shall be named `tableName_id` (referring to the `tableName` that they are pointing to)

5. **Somewhere in your user table and/or your “associative” table** shall be the following:
   - at least one optional Date type field, (e.g., you didn’t click “Not Null” in the Table Design Screen).
   - at least one optional Decimal (usually a money amount, don’t click “Not Null”) or an optional integer.
   - at least one LongText.
5. Examples of Data Model and Table Design Screen Captures

Here is an example of what a data model looks like. Recently we have been having some trouble with being able to reverse engineer data models from our databases (due to the crazy large number of student databases that we have). So, if you are not able to reverse engineer a data model from your database, just draw one and take a picture of it.

This is a screen capture of "table design" (right click on table name and select "Alter Table").

This is a screen capture of "foreign keys" within table design (click on "Foreign Keys Tab: from "Alter Table").
6. Project Proposal Submission Requirements

Attach (to Canvas) a word or rtf document (named with your last name in it please) that contains:

- The (red) questions about your proposed web application (copied from section 2 of this document) and your answers to these questions.

- For each of the four tables you designed: a screen capture of its table design and (if it has a foreign key) a screen capture of the foreign key tab of its table design.

   To get a screen capture, you can do an Alt-PrtSc (copies active window into the clipboard), paste into MsPaint (or any other image editor), then copy just the part needed into the document.

- A screen capture of your data model (created in MySQL Workbench) that meets the requirements of this document. If you can’t reverse engineer a data model using MySQL Workbench, just draw a data model and take a picture of it and include that image in the word document (or attach the image file separately).

7. Homework Grading

- **Originality of Project and Data Model**: As in most Homework assignments, points will be deducted if your answer is too similar to the sample or sample(s) provided. In this Homework, I gave a lot of project ideas, so it’s OK if your web application proposal is similar to one of them.

- **Check list**: We will check that your data model meets all of the requirements listed in section 3 “Data Model”. This is very important, since all of your future Homeworks depend on your database design. You had to have three null-able non-character fields (1 a date, 1 an integer or decimal.

- **Consistency**: In this particular assignment, your data model has to be able to support the functionality that you say your web application will offer and this functionality has to align with the “eventual functionality of your web application” (as described above).

- **Professionalism**: When you answer the questions about your proposed web application, especially the last question which asks for “marketing material”, we are looking for quality that would be acceptable by a “real company” that might be paying you to create their web site.

- **Timeliness**: To avoid late penalty, be sure to submit your work into blackboard by the due date and hand in a print out of the word document (in lecture).
8. Suggested Approach

1. Install MySQL Workbench on your laptop or visit a CIS dept lab PC. Connect to your database and practice with MySQL Workbench while you read the two tutorials that are associated with this Homework. You need to understand basic database concepts before you attempt data modeling.

2. After reading the "Possible Project Ideas" (above), come up with your project idea, draw your data model on paper and make sure it meets all the requirements - especially the requirements about the null-able non-character fields, etc. Then answer the questions (the red ones in this document).

3. When you are ready to use MySQL Workbench to create your data model, use the two tutorials (from this Homework) for help. Create all of your tables, then reverse engineer a data model from your tables. If you added your foreign keys correctly, you should see the foreign key relationships shown on the data model. If you are not able to reverse engineer a data model from your database, just draw the data model and take a picture and submit that file for your data model.