Homework: Project Proposal & Data Model

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. Learn how to use MySQL Workbench (an open source GUI for MySQL, an open source Database Management System) to create a data model of a database that will support your web application.

2. Before You Start

Before starting on this Homework, read the following tutorials that are associated with this Homework.

- How to install MySQL Workbench
- How to connect to your Temple database (shows you how to find your database credentials, etc)
- MySQL Workbench Tutorial: How to create database tables and relationships (primary keys, foreign keys), enter data, and write SELECT statements
- MySQL Workbench Data Model Tutorial (continuation of previous tutorial): How to create a data model, add tables and relationships to the model, save and print the model, forward engineer a SQL database creation script from the data model.

3. 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>&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>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>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>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>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>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>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>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>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>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>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 your web application will be able to do, once you have completed all the Homeworks.

1. Display a home page with content that describes what your web application has to offer.
2. List data from tables of your database:
   - web_user joined with user_role,
   - other,
   - associative joined with web_user and other.
3. Log on and log off. Restrict certain functionality only to logged in users.
4. Insert/edit/delete data from web_user, other, and associative.

You will also have a "blog" page, but this does not add to your web app functionality – it is there to enhance your Homeworks and assist with grading.

4. Requirements for Your Data Model

Once you have come up with a topic for your web application (such as those listed on the previous page), think about the database tables you will need to support your web application. Each student will have 4 tables:

- user_role (exactly as prescribed below)
- web_user
- YOU DECIDE what to have for your “other” table
- YOU DECIDE what ot have for your associative table

**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. Because I give sample code thoughout the semester, you must design tables **web_user and user_role** exactly as shown.
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 – click on “UQ” do database adds uniqueness constraint),
   - a Long character field that holds a fully qualified URL to an image (this will enable our web site to show pictures associated with the data).
   - **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).
     - Select “decimal” data type for dollar amount fields.
     - Select Date over DateTime so you get just year/month/day, no milliseconds.

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. **Somewhere in your user table and/or your "associative" table** shall be the following:
   - at least one optional Date type field, ("optional" means you didn't click “Not Null” in the Table Design Screen. Optional means the database will accept an inserted record that has “null” for that value).

   For the purpose of your homework assignments, **this date field must be something that a user would type in – NOT something that your web app should automatically fill in**. For example, if you were to have a field named “date_inserted”, there would be no need to ask the user to type that in – the application should just pre-populate that automatically.

   - at least one optional Decimal (usually a money amount) or an optional integer.

5. **No redundancy.** In a well designed relational database, you should not have any data redundancy. You should not store anything that can be calculated. Examples:
   - If your database store someone’s birthday, you would not also store their age – because you can calculate a person’s age from their birthday.
   - If your database stores the value of a stock item and it stores quantity (how many items are in stock), you would not also store the value of that inventory (item value times quantity).

6. **Naming conventions:**
   - Every table shall have a **Primary Key** that is named: tableName_id.
     - Example: web_user.web_user_id
   - All **Foreign Keys** shall be named the same as the Primary Key they reference.
     - Example: user_role.web_user_id
5. Requirements for Your 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?** (which you will eventually put on your home page)
2. **Who will be users of your web application?**
   - Even though all student databases will have tables web_user and user_role designed exactly as prescribed (so that my sample code will work for you), you can still say that your users are customers or sports fans or whatever.
3. **What functionality will your web application provide?** At a “mechanical level”, your application will allow the user to add/edit/delete data from three tables, but functionally, how will users benefit from your web app?
4. **Explain how there is a many to many relationship between your user table and your “other” table.**
5. **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.

6. Project Proposal Submission Requirements

Attach the following to the Canvas Homework assignment:

1. **A document** that contains the following. To help us with grading, please name the document with your last name in it.
   - The (red) questions about your proposed web application (copied from an earlier section 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. If you would paste in the whole screen capture, the text in the large image would be too small to be legible.

2. **The data model file (.mwb)** that you saved from MySQL Workbench of the database you designed that meets the requirements in this document.
   - Note: there is a bug trying to open up a data model file from within MySQL Workbench, but you can open up your data model if you close out MySQL Workbench and just double click on the .mwb file from File Explorer.

3. **A PDF of your data model** (File – Print to File from within the MySQL Workbench data modeling tool).
7. Homework Grading

- **Originality of Project and Data Model:** In most Homework assignments, points are deducted if your answer is too similar to the 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 this document. This is very important, since all of your future Homeworks depend on your database design.
  
a. Do **web_user** and **user_role** table designs exactly match what was specified in this document? (You need this so that sample code will work for you throughout the semester.)

b. Does your “**other**” table have an auto-increment PK, a unique (character) descriptive field, a field for an image URL, and >= 2 fields including >=1 nullable non-character field (date, integer, or decimal)?

c. Does your **associative table** have an auto-increment PK, two foreign keys (one pointing to **web_user** and one pointing to your “**other**” table), and >= 2 fields including >=1 nullable non-character field (date, integer, or decimal)?

d. Somewhere in your “**other**” table and/or associative table, do you have at least one nullable date field and at least one nullable integer or decimal field?

e. Does your database design support your proposed web application functionality? Have you come up with functionality and a data model that “make sense”, that someone user would want to utilize?

f. Did you name each primary key field **[tableName]_id** (following database naming conventions)?

g. Did you name each foreign key field the same as the primary key it references (following database naming conventions)?

- **Consistency:** For this assignment, your data model has to be able to support the functionality that you say your web application will offer and this functionality must align with the “eventual functionality of your web application” (that the users will be able to insert/update/delete/show data from the 3 tables as described above).

- **Professionalism:** Try to make your answers to the questions posed in this assignment (especially the last question which asks for “marketing material”) be realistic - acceptable by a small “real company” who might be paying you to create their web site.

- **Timeliness:** To avoid late penalty, be sure to submit your work into Canvas by the due date.
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, make sure you have done the two MySQL WB two tutorials (from this Homework – the regular tutorial followed by the data modelling tutorial). For this homework, you can actually do it 100% in the data modelling tool and get all the screen captures from that.
9. Examples of Data Model and Table Design Screen Captures

I wrote a MySQL Workbench Data Modeling tutorial that explains how to create a data model (and print to PDF).

Here is an example of a data model (see below). This particular data model example would NOT be OK for your first homework since your data model needs to have web_user and user_role designed exactly as prescribed. The other two tables are up to you to name and design, within the requirements stated in this document.

In this example, venu would be the “other” table and “concert” would be the associative table. Make sure you use non-identifying relationships (dotted relationship lines, foreign keys have pink diamond icon, not key icons). Make sure that the “many” side of the relationship line (“crows feet”) is attached to the associative table, not to web_user or “other”.

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").