Week 3 Concepts Assignment

CS441

*This is exercises based on the textbook ones for Chapters 3 and 4. Note, there are additional steps or requirements that might not be in the book, so be sure to follow this document vs directly following the textbook. For example, in Exercise 3-1 I am having you skip Steps 1-3.*

*Note, all required student files are located in the Week 1 Module of Canvas. Look for the “Student Download files” and add to your htdocs folder. Remember that the weekly code will be copied back and forth from your local Github repository and the htdocs folder.*

*Before starting, make sure to copy the directories “book\_apps” and “ex\_starts” to your local computer in the following directory: C:\*xampp\htdocs\

**Instructions:**

# Exercise 3-1 Use phpMyAdmin with a database

This exercise will give you a chance to use phpMyAdmin to review the databases that are used with this book and to test the types of SQL statements that you will use in your PHP applications.

**If necessary, set a password for the root user** ***(skip this step, we will continue to use root with no password with XAMPP)***

1. Start Apache, MySQL, and phpMyAdmin as shown in figure 3-9. Then, log in as the root user.
2. ~~If you haven’t already set a password for the root user, use phpMyAdmin to do that. If you’re only using MySQL for working with the applications in this book, use a password like “sesame” so it’s easy to remember. Otherwise, use a more secure password.~~
3. ~~Click the “Log out” button (the Exit sign) to log out. This should take you back to the Welcome page. Then, log in as the root user with the new password. That will take you to the Home page.~~

**Run the script for creating the book databases**

1. On the Home page, review the list of databases that are available in the sidebar. Then, click the Databases tab to see the same databases. Note that these databases include the databases that MySQL and phpMyAdmin use to manage their own operations. Are the my\_guitar\_shop1 and my\_guitar\_shop2 databases both available?
2. Use the procedure in figure 3-10 to import and run the script that creates the databases for this book. This will create or recreate the two databases for this book. As a result, the my\_guitar\_shop1 and my\_guitar\_shop2 databases will both be shown in the sidebar and in the Databases tab.

**Review the my\_guitar\_shop1 database**

1. Select the my\_guitar\_shop1 database to display the tables for this database, and click the Browse button to view the data for the products table.

Capture a screenshot of the data for the products table:

1. Click the Structure tab to view the column definitions for the products table. Note that none of the columns allows nulls or provides default values.

**Run SQL statements against the my\_guitar\_shop1 database**

1. Use the SQL tab to run the first query in figure 3-4. Then, run the second query.
2. Run the query in figure 3-5. Then, modify the list price value in the query so it only selects products with a price that’s less than 400, and run the query again.
3. Run the first query in figure 3-6 to add a row to the products table. Then, browse the products table to view the new row. Last, run a DELETE statement to delete the new row.

Capture a screenshot of the results:

1. Continue to experiment until you’re sure that you know how to code the SQL queries that your PHP applications will use.

**Log in as a different user and check that user’s privileges**

1. Log out of phpMyAdmin, and log back in as mgs\_tester with pa55word as the password. This user was created by the SQL script that you ran in step 5 of this exercise.
2. Use the SQL tab to run this SELECT statement:

SELECT \* FROM categories

This statement should be refused because mgs\_tester can only run statements against the products table.

Capture a screenshot of the result:

**Experiment**

1. Continue to experiment until you’re confident that you understand the use of phpMyAdmin and the types of SQL statements that you’ll use in your PHP applications. Are there any other observations you can make about utilizing phpMyAdmin?

# Exercise 4-1 Enhance the Product Manager application

*This exercise has you enhance the Product Manager application by adding a page that lets you add and delete categories.*

**Test the Product Manager application**

1. Start the Chrome browser and run the application in the ex\_starts\ch04\_ex1 directory. To do that, you can use this URL:

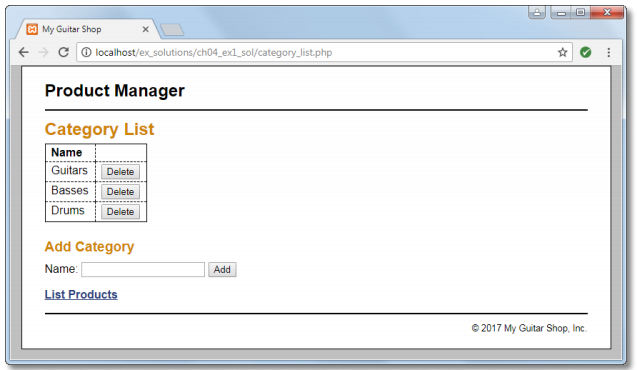
<http://localhost/ex_starts/ch04_ex1/>

This should display the products for the first category in the database named my\_guitar\_shop1.

1. View the products in each of the categories.
2. Add a new product to the database. When you add this product, make sure to enter valid values for a product. Capture a screenshot and paste it below before completing the next sections. Then, delete the product you just added from the database.
3. Click on the List Categories link at the bottom of the page. Note that this link leads to a page that’s under construction. However, the link back to the Product List page does work.

**Add a Category List page**

*In the rest of this exercise, you’ll add a page that looks like the below. Hint: The steps say write code, and then reference an example. Make sure to open up and look at that reference for guidance how to complete the exercise!*



1. Open the category\_list.php file that’s in the directory for this application. It contains some of the headings and a link back to the Product List page.
2. In the category\_list.php file, write the code that creates the category table shown above with all of the category names in the first column and Delete buttons in the second column, similar to how the index.php file works. Then, test that this table is displayed correctly.
3. In the category\_list.php file, write the code that lets the user add a category to the database. This code should consist of a form that accepts the name for a new category followed by a Submit button that displays “Add”, similar to how the add\_product\_form.php file works. Then, test that this form is displayed correctly.
4. Create an add\_category.php file that adds a category to the database and a delete\_category.php file that deletes a category from the database. These files should display the Category List page after they add or delete a category, similar to how the add\_product.php and delete\_product.php files work.
5. Test the application by adding two categories. Then, navigate to the Product List page and note that the list of categories includes the new categories. Next, navigate to the Add Product page and note that the drop-down list includes the new categories.
6. Test the application by deleting the categories that you just added. However, don’t delete any of the existing categories because that will lead to products without categories. If necessary, though, you can restore the database by running the create\_db.sql script again as described in the appendixes.

***Submit this assignment, including screenshots, and any edited file via the Github Repository. Files modified, other than this template are:*** category\_list.php, add\_category.php, delete\_category.php.