HyperionDev

Capstone Project – Databases

Task

Visit our website

Introduction

This capstone project is a milestone in your learning so far. You will be consolidating the knowledge which you have gained and applying it to a real-world situation! It will give you the opportunity to demonstrate your competence in Python and SQL, while creating an application you can add to your portfolio. Understanding a programming language or development technology is a key skill. However, being able to apply your knowledge in order to create software to meet the unique specifications that a client may want is one of the most desirable skills in the industry.

Be creative – you will be tasked with a set of criteria to meet, but the rest is up to you. It is worth spending time and effort to make this a project that you can be proud to show prospective employers.

The task at hand

For this project, you are required to create a program for a bookstore. The program should allow the clerk to enter data about new books into the database, update book information, delete books from the database, and search to check the availability of books.

Remember, any great design must be functional and meet the specifications provided by the user. A software solution that looks good and works but doesn't do what the user wants it to is like creating a bike with square wheels. It may be an interesting exercise, but not a very useful one.

Practical task

You will complete this task in three parts. First, you will create a program to handle book inventory. Then, you will expand your relational database to include an additional table. Finally, you will check if your code adheres to best practices, such as code modularity, data validation, and error handling.

Part 1

1. Create a program called **shelf_track** that can be used by a bookstore clerk. The program should allow the clerk to:

    - add new books to the database,
    - update book information,
    - delete books from the database, and
    - search the database to find a specific book.

2. Create a database called `ebookstore`, and a table called `book`. The table should have the following structure:

id	**title**	**authorID**	**qty**
3001	A Tale of Two Cities	1290	30
3002	Harry Potter and the Philosopher's Stone	8937	40
3003	The Lion, the Witch and the Wardrobe	2356	25
3004	The Lord of the Rings	6380	37
3005	Alice’s Adventures in Wonderland	5620	12

3. Populate the table with the above values. You can also add your own values if you wish.

4. The program should present the user with the following menu:

```
1. Enter book
2. Update book
3. Delete book
4. Search books
0. Exit
```

The program should perform the function that the user selects. In the case of “update book”, the default option should be to update the quantity of that book, but the user should also be able to select title or authorID if a correction is required. The implementation of these functions is left up to you, but a demonstration of the topics we have covered previously should be shown.

Part 2

Next, we will enhance the complexity of the database by introducing an additional table.

1. Create a table called `author` in the `ebookstore` database. The table should have the following structure:

id	**name**	**country**
1290	Charles Dickens	England
8937	J.K. Rowling	England
2356	C.S. Lewis	Ireland
6380	J.R.R. Tolkien	South Africa
5620	Lewis Carroll	England

2. Populate the table with the above values, and any additional ones necessary if you have added additional entries to the `book` table.

3. Add a '`View details of all books`' option to the menu. The menu should now look like this:

```
1. Enter book
2. Update book
3. Delete book
4. Search books
5. View details of all books
0. Exit
```

4. Add functionality to your program to display the book title, author name, and country in a user-friendly way when the '`view details of all books`' option is selected. See the example below: (**Hint**: Use the `zip()` function).

```
Details
--------------------------------------------------

Title: The Tale of Two Cities
Author's Name: Charles Dickens
Author's Country: England
----------------------------------------------------

Title: The Lion, the Witch and the Wardrobe
Author's Name: C.S. Lewis
Author's Country: Ireland
----------------------------------------------------

Title: Alice’s Adventures in Wonderland
Author's Name: Lewis Carroll
Author's Country: England
```

5. When the user selects option 2 (`Update book`), they should enter the book ID they want to update, review the current author's name and country, provide new values for the author's name and/or country, and then save the updated information to the database.

Note that the book table contains the `title` column, while the author table includes the `name` and `country` columns. The `authorID` column in the book table serves as a foreign key, linking to a corresponding author with a matching ID (which is the primary key) in the author table. To retrieve the necessary information from both tables, you should construct a query using an `INNER JOIN`. This will allow you to obtain the relevant columns from both tables simultaneously.

Part 3

Finally, apply best practices for code maintainability by implementing the following improvements:

1. **Code modularity:** Divide the program into smaller, reusable parts for better maintainability and readability. This can be achieved by implementing functions to handle specific functionalities, such as adding books, updating books, deleting books, etc. Additionally, create functions to encapsulate database-related operations, like establishing a database connection and creating tables.

2. **Error handling:** Implement robust error handling to manage unexpected user inputs and database errors. This will involve incorporating validation checks and exception handling and providing user-friendly error messages.

3. **Data validation:** Ensure that user inputs are validated to prevent data inconsistencies and security risks such as **SQL injection** attacks. For example, validate that the book ID and author ID are integers and have exactly four characters.

4. **Close database connections:** Ensure that the connection to the SQLite database is properly closed after completing operations. Use context managers (with statements) to automatically handle closing the connection, preventing database locks and other issues.

5. **Refactoring**: Review the code to identify redundant or inefficient sections and improve them. Use clear and descriptive names that accurately convey the purpose of variables and functions.

6. **Debugging**: Test your program to verify if it works as expected, and ensure that any logical errors are resolved.

7. **Documentation and style**: The program must be well-documented, follow good coding practices, and be styled according to the **relevant style guide** to maintain consistency and readability.

Important: Be sure to upload all files required for the task submission inside your task folder and then click "Request review" on your dashboard.

Share your thoughts

Please take some time to complete this short feedback **form** to help us ensure we provide you with the best possible learning experience.