Special Course in Software Engineering – Autumn 2023

Python programming exercise 4 (MAX: 8 points)

In this exercise, you will create a simple program to carry out **C**reate-**R**ead-**U**pdate-**D**elete (**CRUD**) operations on a database. This is a very common and typical set of operations one can do on any database.

Just like last exercise, you are **ENCOURAGED** to use ***Copilot*** to help you work on this exercise. A quick demonstration was provided during the last exercise. If you feel that was inadequate, then we recommend that you do your own research on the use of ***Copilot*** and then use it in the way you find suitable. And, if you do use **Copilot** in your code, then **mention it in your file**.

Write a Python program for a library management system, where your program must…

- Create a database with three tables called ***Books, Users,*** and ***Reservations***:
    1. ***Books*** table has the columns *BookID, Title, Author, ISBN, Status*
    2. ***Users*** table has the columns *UserID, Name, Email*
    3. ***Reservations*** table has the columns *ReservationID, BookID, UserID, ReservationDate*
- Provide users (**library staff**) the following choices to interact with the abovementioned database:
    1. Add a new book to the database.
        - This addition will be done only to the ***Books*** table
    2. Find a book's detail based on *BookID.*
        - The result will include the book's reservation status, and the user's details someone has reserved the said book.
        - The logic will require accessing all the three tables.
    3. Find a book's reservation status based on the *BookID, Title, UserID,* and *ReservationID.*
        - When accepting input from the user, your program must determine based on the first two letters of the text if it's a *BookID (starts with LB), UserID (starts with LU),* or *ReservationID (starts with LR)*. Otherwise, the text entered is a *Title.*
        - If none of the above conditions are satisfied, then it mostly likely means that the said book does not exist in the database. So, a corresponding message should be displayed to the user.
    4. Find all the books in the database.
        - The result will include details from all the three tables.
    5. Modify / update book details based on its *BookID*
        - If the modification involves updating the reservation status of a book, then changes will be made to all the three tables.
        - Otherwise, modifications are most likely to be just to the ***Books*** table.
    6. Delete a book based on its *BookID.*
        - If a book that is being deleted is also reserved, then the deletion will happen to both the ***Books*** and the ***Reservations*** tables.
    7. Exit

- For retrieving results from the three tables, you need to us **JOIN** keyword in your SQL queries. Following is an example:

Imagine there is a table ***Student*** with columns *StudentID, StudentName,* and *StudentAddress.* And another table called ***Course*** with columns *CourseID, CourseName, StudentID,* and *CourseTeacher.*

So, to find out which student is taking which course, you can use the **JOIN** keyword to construct a query as follows:

```
SELECT
StudentID, StudentName, CourseName
FROM
Student
INNER JOIN Course
ON Students.StudentID = Course.StudentID
```

Using the column *StudentID,* the above query finds commonality across the two tables of ***Student*** and ***Course.*** As long as for every *StudentID* in the ***Student*** table*,* there is a corresponding entry in the ***Course*** table, then the above query will return a result that displays *StudentID*, *StudentName* and the *CourseName*.

Please note that the above example concerns only two tables. In your program, you must handle three tables.

- Commit your exercise solution to your GitHub account
- Enter the URL of your exercise repository in the **"Exercise 4 Submission Box".**

Special Course in Software Engineering – Autumn 2023

The following is **not a mandatory** part of the exercise, but it's for those who want something more challenging. Doing this optional part of the exercise successfully can give you **5 bonus points**.

- Use the ***Alice's Adventures in Wonderland*** text from the **nltk** library's **gutenberg** module to conduct **topic modeling.**
    - **Topic modeling** is a text-classification algorithm that helps you analyze raw text by identifying the topics (themes) embedded within them.
    - You will need to install and use the **genism** package to carry out topic modeling.
- Your program must also visualize the topics generated from the modeling.
    - You will have to install and use the **pyLDAvis** package to do this.

NOTE: We are aware that solutions for the above exercise can be found quite easily online. It is okay to look for ideas and syntax and some help with pieces of your code. However, please try to put those pieces together to build your own code. Avoid using an already available solution.

Special Course in Software Engineering – Autumn 2023