LMS Rental Project - Phase 3:

This is the final part of the LMS Project. In this phase, you need to start with the LMS DB from Phase 2. If your database has any extra or altered data besides phase 2, you will lose points! For this phase, you will have to implement some more complicated queries, create a user interface.

Task 1: Execute the following queries on the LMS database tables:

Query 1: [5 points]
Add an extra column 'Late' to the Book_Loan table. Values will be 0-for non-late retuns, and 1-for late returns. Then update the 'Late' column with '1' for all records that they have a return date later than the due date and with '0' for those were returned on time.

Query 2: [5 points]
Add an extra column 'LateFee' to the Library_Branch table, decide late fee per day for each branch and update that column.

Query 3: [10 points]
Create a **view** vBookLoanInfo that retrieves all information per book loan. The view should have the following attributes:

- Card_No,
- Borrower Name
- Date_Out,
- Due_Date,
- Returned_date
- Total Days of book loaned out *as 'TotalDays'– you need to change weeks to days*
- Book Title
- Number of days returned late *– if returned before or on due_date place zero*
- Branch ID
- Total Late Fee Balance *'LateFeeBalance' – If the book was not retuned late than fee = '0'*

Submit:

1. Type the appropriate SQL commands in your submission report file.
2. Type the select the RENTAL table command and paste the "Result Grid" screenshot. (3) Provide the action output response for all the above actions.
3. Type the "**create view**" command text in your submission report file. Use the above names for your attributes' names. Do NOT change any name.
4. Type the "**select view**" command and paste the "result grid" screenshot after the 'Create view' SQL code.
5. Finally, type the "action output response", how many row(s) returned.

Task 2: Create a GUI for the LMS database:

LMS Application: **[20 points]**
Create a simple and friendly GUI interface that would be able to perform the following tasks. You may use JAVA programming using JDBC, or C/C++/C# programming with ODBC/Oracle or Python programming or PHP/MySQL or other programming languages to develop a GUI interface. For each query, you need to provide some info to the user about the query purpose. The user will have to type or select the query's input parameters and post the question to your program. The program needs to return all result's rows.

Requirements:

1. User checks out a book, add it to Book_Loan, the number of copies needs to be updated **via trigger** in the Book_Copies table. Show the output of the updated Book_Copies. **[10 points]**
2. Add information about a new Borrower. Do not provide the CardNo in your query. Output the card number as if you are giving a new library card. Submit your editable SQL query that your code executes. **[3 points]**
3. Add a new Book with publisher (use can use a publisher that already exists) and author information to all 5 branches with 5 copies for each branch. Submit your editable SQL query that your code executes. **[5 points]**
4. Given a book title list the number of copies loaned out per branch. **[5 points]**
5. Given any due date range list the Book_Loans that were returned late and how many days they were late. Submit your editable SQL queries that your code executes. **[8 points]**
6. The fifth requirement is to return the **view's results** by applying the following criteria:
    a. List for every borrower the ID, name, and if there is any lateFee balance. The user has the right to search either by a borrower ID, name, part of the name, or to run the query with no filters/criteria. The amount needs to be in US dollars. For borrowers with zero (0) or NULL balance, you need to return zero dollars ($0.00). Make sure that your query returns meaningful attribute names. In the case that the user decides not to provide any filters, order the results based on the balance amount. Make sure that you return all records. Submit your editable SQL query that your code executes. **[10 points]**
    b. List book information in the view. The user must search with borrowerID and any of the following search items: book id, books title, part of book title, or to run the query with no filters/criteria. The late fee amount needs to be in US dollars. The late fee price amount needs to have two decimals as well as the dollar '$' sign. For books that they do not have any late fee amount, you need to substitute the NULL value with a 'Non-Applicable' text. Make sure that your query returns meaningful attribute names. In the case that the user decides not to provide any filters, order the results based on the highest late fee remaining. Submit your editable SQL query that your code executes. **[10 points]**

Note: You can test your code by adding a few new customers, books, book_loans, or by changing some book loan due dates; but make sure before you submit your phase 3 answers and before your demo presentation that your database has only the phase 2 data. Failing to do so will result in losing points.

Please make sure to write out team contributions at the end of the document! This will be *5 points* of the overall grade.

Task 3: Demo Presentation

Demo Presentations will be during demo day which is on Dec 4th (last day of classes). **[15 points]**

Canvas Submission:
Create a folder with your code, named "Code" Inside the "Code" folder, create a Readme.docx file with step-by-step instructions and screenshots on how to install and run your program from scratch. Also, place a database backup (self-contained file), named as LMS.sql inside your "Code" folder,

Create your submission file.
In the front-page, type a title for your project submission and specify your name, and your team member's name.
On the second page, include your honor code. Failing to do so will cost **[10 points].**
Next, you need to follow the guidelines from Task 1 and submit your work.
Edit your submission file and append (after the Task 1) tasks queries from Task 2, screenshots from your GUI for each task, along with some explanation or a meaningful label.

Save your submission file as "Report.docx" or "Report.pdf". Make sure that all commands are editable!

Create a folder named as "teammate1lastname_teammate2lastname" and place inside the "Code" folder and your "Report.docx" file.

Zip your folder and submit it on Canvas any time before the deadline.

HONOR CODE

I pledge, on my honor, to uphold UT Arlington's tradition of academic integrity, a tradition that values hard work and honest effort in the pursuit of academic excellence.

I promise that I will submit only work that I personally create or that I contribute to group collaborations, and I will appropriately reference any work from other sources. I will follow the highest standards of integrity and uphold the spirit of the Honor Code.

Students are required to not share any of the project related documents and solution with others in any way or form even after the completion of the project. Students may, however, show their projects to interviewers.

Late penalty: -5 points per hour