COMP 3005: Database Management Systems **(Due: Dec. 10, 2021 (11:59 PM))**

COMP 3005 Project (Fall 2021)

Instructor: Ahmed El-Roby

1. Problem Statement

Design and implement an application for an online bookstore (Look Inna Book). This application lets users browse a collection of books that are available in the bookstore. A user can search the bookstore by book name, author name, ISBN, genre, etc.. When a book is selected, information on the author(s), genre, publisher, number of pages, price, etc. can be viewed. A user can select as many books as she likes to be added to the checkout basket. A user needs to be registered in the bookstore to be able to checkout. When checking out, the user inserts billing and shipping information (can be different than those used in registration), and completes the order. The bookstore has the feature of tracking an order via an order number. A user can use this order number to track where the order is currently. Although shipping is carried out by a third-party shipping service, the online bookstore should have the tracking information available for when the user inquires about an order using the order number. Assume all books are shipped from only one warehouse (no multiple order numbers for multiple books shipped from multiple warehouses). The bookstore owners can add new books to their collections, or remove books from their store. They also need to store information on the publishers of books such as name, address, email address, phone number(s), banking account, etc.. The banking account for publishers is used to transfer a percentage of the sales of books published by these publishers. This percentage is variable and changes from one book to another. The owners should have access to reports that show sales vs. expenditures, sales per genres, sales per author, etc.. The application should also be able to automatically place orders for new books if the remaining quantity is less than a given threshold (e.g., 10 books). This is done by sending an email to the publisher of the limited books to order a number of books equal to how many books were sold in the previous month (you do not have to implement the email sending component).

2. Project Report

You need to submit one report file that contains the following sections. You can add other sections, but the following sections (except Bonus Features) **must be** in the report:

2.1. Conceptual Design

This section should explain the conceptual design of the database. That is, the ER-diagram of the database for the bookstore and explanation of all the assumptions made in the diagram regarding cardinalities and participation types. Make sure that the assumptions do not contradict with the problem statement in Section 1.

2.2. Reduction to Relation Schemas

Reduce your ER-diagram into relation schemas and list these in this section.

2.3. Normalization of Relation Schemas

Given the problem statement and your design, write the set of functional dependencies that apply to your database. Show that your relation schemas are either in a good normal form (show tests), or if they are not, show how to decompose them into a good normal form (show decomposition work), then show the testing work to make sure that they are in a good normal form.

2.4. Database Schema Diagram

This section should show the final schema diagram for the database of the bookstore. This diagram should be similar to the schema diagram of the university database that we study in this course.

2.5. Implementation

Feel free to use whichever programming language(s) for your application. Your application can be a web-based application, or a desktop application. In this section, you should describe the architecture of your application. That is, what the modules in your application are and how they interact. You are encouraged to include a diagram of the application's architecture and explain (in text) scenarios of using the application and the workflow of your application. It is mandatory that you use a **relational database** to store all your data (e.g., no key-value stores).

Your application should have two different user interfaces: The first is for the users of the application through which the user can browse and buy books. The second is for the bookstore owners/managers through which they can add/remove books, display reports, etc. (more details about the application's features can be found in Section 1).

Include screenshots of your application's two interfaces in different scenarios (e.g., checking out, displaying a report, etc.).

2.6. Bonus Features (Optional Section)

You are free to add bonus features to your online bookstore. **You will be rewarded bonus marks for these features**. The choice of bonus features is entirely up to you. An example of a bonus feature is approximate search for books. For example, when the user searches for "The Lord of The Rings", the user would find multiple matches "The Lord of The Rings: The Fellowship of the Ring", "The Lord of The Rings: The Two Towers", "The Lord of The Rings: The Return of the King", "Lord of Flies", etc., assuming that these were the titles of the books in the database. Note that this approximate match is not limited to matching prefixes only. Another example of a bonus feature would be to show similar books (e.g., other books by the same author, or other books with the same genre). The latter feature is a good example of using views.

2.7. GitHub Repository

All your source code for your application should be uploaded to a public GitHub repository. The code needs to be well-documented and a decent README file that clearly states the instructions for running your code should be provided. The GitHub repository should also include a directory titled "SQL" that includes all the SQL DDL statements and SQL queries used in your application. You can organize the files in this directory in whichever way you would like. But all the files should have the .sql extension. For example, the directory "SQL" could have four files "DDL.sql", "Functions.sql", "Triggers.sql", and "Queries.sql". The files should be well-documented using comments in SQL (e.g., purpose of trigger, what a query retrieves from the database, etc.). This section should include the url to this GitHub repository.

2.8. Appendix I

This section should include your availability (from 9 am to 5 pm) for a 20 minutes demosntration of your work on December 11th. Please list **at least three time slots**. Please see the instructions in Section 3.

3. Instructions for Submission

Please follow these instructions in your submission:

- You can work on this project individually, or in groups. The maximum number of students in a group is three. If you do the project individually, you will get a 20% bonus. If you do it in groups of two, you will get a 10% bonus.
- Submit your project report as one pdf file. All group members should make a submission on Brightspace.
- Make sure that your GitHub repository is public.
- The due date (December 10th) is a **hard deadline**. No submissions will be accepted after the deadline. It is your responsibility to guarantee there is a version of your report uploaded before the due date. If there is no submission after the due date, you will receive no marks for the project.
- Some groups will be randomly selected to demonstrate their work one day after the due date (i.e., on December 11th). It is your responsibility to make sure that your system is working properly.