Project 1. Auction Database

SWE3003 Database Systems - Fall 2024

Due date: Nov. 13 (Wed) 11:59pm

1 Purpose of the Project

In this project, your task is to analyze the requirements for designing, implementing, documenting, and testing a database system for an online auction site. Building a website involves a client-side development team that typically includes a web designer, who makes decisions on the design aspect of the site, and a front-end developer who implements the user interface using HTML, CSS, JSP, JavaScript, and other technologies.

While the front-end development team focuses on the website's visual design and user interface, the back-end developers create core services that handle business workflow, data modeling, databases, file servers, cloud services, and other crucial components.

It's important to note that front-end development does not necessarily require a deep understanding of fundamental computer science. Instead, it is often considered a creative field that combines elements of art, design, and human science. In our database systems course, we will be primarily focused on back-end development. Therefore, your role in this project will be as a back-end developer.

Unfortunately, your team has not yet hired any front-end developer. Instead, you're provided with a text-based menu-driven interface. You can find the code in in /home/swe3003/auction/ in In-Ui-Ye-Ji cluster. The following is the output of the provided skeleton program.

```
$ java Auction
----< Login menu >
----(1) Login
----(2) Sign up
----(3) Login as Administrator
----(Q) Quit
1
----< User Login >
 ** To go back, enter 'back' in user ID.
     user ID: bnam
     password: changethis
You are successfully logged in.
```

When a user selects one of the choices, its corresponding function is called. For example, if a user selects '1', the program calls `LoginMenu()` function, which asks for an ID and password. After the ID and password are verified, the program outputs the following menus.

```
---< Main menu > :
----(1) Sell Item
----(2) Status of Your Item Listed on Auction
----(3) Buy Item
----(4) Check Status of your Bid
----(5) Check your Account
----(Q) Quit
```

If a user selects 3, the program calls its corresponding function and prints out the next level menu.

```
3
----< Select category > :
    1. Electronics
    2. Books
    3. Home
    4. Clothing
    5. Sporting Goods
    6. Other categories
```

```
    7. Any category
    P. Go Back to Previous Menu
2
----< Select the condition >
   1. New
   2. Like-new
   3. Used (Good)
   4. Used (Acceptable)
   P. Go Back to Previous Menu
2
---- Enter keyword to search the description :
Silberschatz
---- Enter Seller ID to search :
 ** Enter 'any' if you want to see items from any seller.
any
---- Enter date posted (YYYY-MM-DD):
 ** This will search items that have been posted after the designated date.
2023-01-01
 Item ID | Item description | Condition | Seller | Buy-It-Now | Current Bid | highest bidder | Time left | bid close
-------------------------------------------------------------------------------------------------------------
105 | Datbase System Concepts by Silberschatz | Like-new | kyobobook | 35000 | 0 |  | 1 day 11 hrs | 2023-03-23 23:00
253 | Operating System Concepts by Silberschatz | Like-new | youngpoong | 25000 | 100 | john | 0 day 2 hrs | 2023-03-22 14:00
```

When it comes to back-end web application development, Java is one of the most popular programming languages available. Therefore, I recommend that you use Java and JDBC to implement this project. However, if you prefer to use ODBC or other programming languages, you are allowed to do so. But, please note that you will need to write the program from scratch without the help of skeleton code.

Although there are various convenient database application development frameworks available, such as View.js, Django, and Ruby-on-Rails, please refrain from using them for this project. This is an introductory course, and it's essential to have a solid understanding of the basics before diving into more advanced tools and frameworks.

2 Requirement

The following description is about an application for buying and selling items through auctions. The provided menu-driven code is for this auction program. However, note that the description and the skeleton code is only for reference. You are free to design and implement a program on a different topic you like, and in fact, it is recommended that you develop a program on a more interesting subject. Just keep in mind that you need to create a text-based program, not a web interface, so feel free to modify the menu as you wish, using the provided skeleton code as a reference. The only condition is that your application's database schema and queries should not be simpler than that of this auction program. If you are not confident in developing a more interesting application, you may start with the provided skeleton code for auction program, and add or modify the features and menus as you like.

2.1 Data and Queries

The auction system allows any user to list items to sell and also to browse and purchase items. To support this basic functionality, your database should be able to manage the following data, at a minimum (though you may need to manage additional data as well):

Please keep in mind that the following is just a rough sketch of the data model, and your application will need to refine and revise it to meet its specific needs.

- **User Data**

  user ID, password, etc.

- **Item Data**

  item category (used for category search), item description (used for keyword search), condition (new, like-new, good, acceptable), seller ID, buy-it-now price, date posted, status, etc.

- **Bid Data**

  item ID, bid price, bidder, date posted, bid closing date, etc.

- **Billing Data**

  sold item ID, purchase date, seller ID, buyer ID, amount due buyers need to pay, amount of money sellers need to get paid, etc.

Note: Your data model must be decomposed to satisfy one of the canonical forms (BCNF or 3NF):

Your auction company operates on the following business model: sellers are charged a commission fee of 10% of the item price, with any commissions less than 1 KRW rounded down. For example, if the item price is 85 KRW, the seller pays 8 KRW instead of 8.5 KRW to the auction company.

When the bidding period ends (i.e., when the bid closing time passes), you need to determine the winning bid and charge the winning bidder the price of the sold item. Since PostgreSQL does not support a time-driven event scheduler, your program will NOT continuously check if the bid closing time has expired or not for each auction item. Instead it should check the bid closing time in the following scenarios.

- First, when performing a select query to search for an item in 'Buy Item' menu, the bid closing date should be compared with the current time, and the items whose bids have ended should not be displayed.
- Second, when a user bids on an item, your update query should check the current timestamp against the bid closing date. If the bid closing date has already passed, your program should print out an error message - "Bid Ended."
- Third, when the 'Check your Account' or 'Check Status of your Bid' menu is selected, the corresponding functions must compare the bid closing date of the items currently being sold with the current time and perform appropriate transaction processing for the items whose bids have ended.

Please note that this project does not require you to implement a payment system. Instead, the administrator will check each user's account balance to determine how much money they owe to the auction company and how much money the company owes to the user (i.e., the seller).

- Your program should allow buyers to browse auction items by category, condition, keyword, seller, and date. The search results should be displayed in a well-formatted table that includes the following information: when the auction ends, the current highest bidding price, the current highest bidder, and the buy-it-now price.

  It's important to note that this requirement is in place to make it easy for TAs to check the output table. If the table is unreadable, TAs may not be able to thoroughly review it, which could result in point deductions.
- Sellers should have the ability to view a list of their auction items and their respective status.
- Buyers should also have the ability to view a list of the auction items that they have bidded on.
- For more details, please check the provided skeleton code.

3 Deliverables

You must submit a project report, source codes, and a demo video. The report should consist of the followings.

- Short introduction to your application
- Schema diagram using E-R model
- List of functionalities that your application supports, and SQL queries for each functionality.

The source codes must consist of the followings.

- ddl.sql: DDL statements that create the database schema of your application
- data.sql: INSERT statements that populate your database tables with some sample input records
- all other source codes that include DML (INSERT/UPDATE/DELETE/SELECT/etc) statements
- Makefile: If you prefer other compilation package, such as maven, you may do so. But, please do not expect TA to know how to compile your code. In the project report, please describe how to compile your code. Also, describe how to run your program. Note that TA will change the connection arguments such that he can run your program in his own database.

4 How to Submit

Please submit your project report and demo video in iCampus. Also, please submit your source code files in swji.skku.edu using db_submit command as follows.

```
$ db_submit  term  /your/code/directory/path
```

This command will compress and submit all files in your current directory. For example, if your codes are in the current directory, run the following command.

```
$ db_submit  term  ./
```

Note that you can submit multiple times. But only the last submission will be graded. Using the following command, you can check whether your file has been correctly submitted.

```
$ db_check_submission term
```

5 Late Submission Policy

20% will be deducted for every 24 hours.

6 Q&A Piazza

For any questions, please post them in Piazza so that we can share your questions and answers with other students and TAs. Please feel free to raise any issues and post any questions. Also, if you can answer other students' questions, you are welcome to do so.