CPSC 304 Project Part 3: Partial Implementation

Export your project to a zip file and submit it on Canvas. If your project file is too large to submit to Canvas, submit the URL to your group's repository and the TAs will grab the files from there. Refer to the Syllabus for information on how late submissions will be treated.

Your submitted code must compile without problems. When grading, the TAs will not be debugging or fixing code to see what works.

Introduction

In this part of the project, you will implement part of the database design for the SuperRent enterprise and create a user friendly graphical interface for the system. This will require you to: 1) define the tables, 2) write code for the transactions, and 3) design a reasonable user interface. We highly recommend reading the Java/JDBC Tutorials (or the PHP tutorial) found on the Resources page before you get started.

As we would like all the groups to implement the same database design, please use the solution to part 2 when creating tables in the database.

In order to simplify the project, we have removed the requirements associated with equipment, club customers, and vehicle sales. All vehicles are for rent. A vehicle is identified by its plate number (vlicense) and the status attribute in the Vehicle table indicates whether a vehicle is rented, in the shop for maintenance, or is available. Customers are now identified by their driver's license (dlicense) instead of their phone number. The rest of the tables have been adjusted accordingly. Everything else will follow what is listed in the solution for part II.

Do note that you may not need to use every single table listed in the solution for part 2. In fact, you will more than likely just need to create tables for:

- Reservations
- Rentals
- Vehicles
- VehicleTypes
- Customers
- Returns

If you feel you need more tables than what is listed above, that's fine too. The list above is only meant as a suggestion for where you can start.

CPSC 304 Project Part 3: Partial Implementation

Using a Different Programming Language/Database From Java/Oracle or PHP/Oracle

You are welcome to use another programming language/database combination subject to these conditions:

1. The query language used for the database must be SQL-like enough for the TAs to be able to judge if the query you have used is correct.
2. The TAs may decide to run your code on their machine when checking the details of your project. You will need to include a README with instructions on how to run your project. Note that this README should not contain instructions that require the TA to go through many configuration steps or have to download external software onto their machines.
3. If your project uses a database that is local to your machine, the TA will not able to run and verify your code. You will need to have a backup plan for this situation.
4. If you run into any technical issues, the course staff will be unable to help.

Functionality

Your project should support the following actions. When results are required to be returned, there should be about 10-15 tuples in your result set.

<u>Transactions performed by a customer</u>

- View the number of available vehicles for a specific car type, location, and time interval. The user should be able to provide any subset of {car type, location, time interval} to view the available vehicles. If the user provides no information, your application should automatically return a list of all vehicles (at that branch) sorted in some reasonable way for the user to peruse.

  The actual number of available vehicles should be displayed. After seeing the number of vehicles, there should be a way for the user to see the details of the available vehicles if the user desires to do so (e.g., if the user clicks on the number of available vehicles, a list with the vehicles' details should be displayed).

- Make a reservation. If a customer is new, add the customer's details to the database. (You may choose to have a different interface for this).

  Upon successful completion, a confirmation number for the reservation should be shown along with the details entered during the reservation. Refer to the project description for details on what kind of information the user needs to provide when making a reservation.

  If the customer's desired vehicle is not available, an appropriate error message should be shown.

CPSC 304 Project Part 3: Partial Implementation

Transactions performed by a clerk

- Renting a Vehicle: The system will display a receipt with the necessary details (e.g., confirmation number, date of reservation, type of car, location, how long the rental period lasts for, etc.) for the customer.

  Note: It is not necessary for a user to have made a reservation prior to renting a vehicle. If this is the case, then you will need to determine how to appropriately handle this situation.

- Returning a Vehicle: Only a rented vehicle can be returned. Trying to return a vehicle that has not been rented should generate some type of error message for the clerk.

  When returning a vehicle, the system will display a receipt with the necessary details (e.g., reservation confirmation number, date of return, how the total was calculated etc.) for the customer.

- Generate a report for:
  - Daily Rentals: This report contains information on all the vehicles rented out during the day. The entries are grouped by branch, and within each branch, the entries are grouped by vehicle category. The report also displays the number of vehicles rented per category (e.g., 5 sedan rentals, 2 SUV rentals, etc.), the number of rentals at each branch, and the total number of new rentals across the whole company

  - Daily Rentals for Branch: This is the same as the Daily Rental report but it is for one specified branch

  - Daily Returns: The report contains information on all the vehicles returned during the day. The entries are grouped by branch, and within each branch, the entries are grouped by vehicle category. The report also shows the number of vehicles returned per category, the revenue per category, subtotals for the number of vehicles and revenue per branch, and the grand totals for the day.

  - Daily Returns for Branch: This is the same as the Daily Returns report, but it is for one specified branch.

For each transaction, make sure to check for basic errors and produce appropriate error messages for the user. For example, if the “make a reservation” feature failed for a customer, the GUI should clearly indicate that this is the case.

CPSC 304 Project Part 3: Partial Implementation

User Interface

Create an easy to use GUI that allows users to execute all the operations and transactions provided by the system. In other words, don’t worry about having the GUI change based on which user is accessing the system. In this case, any user accessing your system can use either the customer or clerk menu.

Make sure your interface is designed in a way where all error messages appear in separate pop-up boxes, or in a designated area of the main window, so that they don't interfere with other information.

Your GUI should also enable the user to display all the rows, insert a tuple, and delete a tuple on any user defined table in the database.

We are not expecting a complicated GUI; the bulk of your effort should be concentrated on implementing functionality, not the user interface. The goal of the project is to give you some experience with SQL, programming something that interacts with a database, and understanding how to structure your classes to work with GUI components.