Lab 7: Database Connectivity with JDBC

Assignment Preparation

You may choose to work individually or in teams of 2 or 3 people. Your team may be composed of students from sections 01 or 03 of CSC 365.

Lab Assignment

Construct a Java JDBC proptotype to handle portions of an inn/hotel reservation system. Implement your application with the following broad goals in mind: (1) minimize the amount of data transferred between your Java application and the the database (2) minimize the number of individual SQL statements you send to the database. The volume of data in our sample dataset would allow an application to `SELECT *` from every table in the database upon startup, then work entirely from memory (neatly achieving both goals.) However, we will imagine that the Inn's full dataset does not fit into main memory. Furthermore, we will assume that there are many other users simultaneously reading and writing the same data.

Wherever possible and reasonable, calculations, filtering, summarization and other logic should be expressed in SQL for evaluation by the database.

Non-Functional Requirements

NFR1: Filenames. Use the Java programming language (JDK 8 or above). Name your main class `InnReservations`. The structure of your application and inclusion of any other files is left up to you. You are required to submit a `README` file which explains how to compile and run your program *from the command line.*

NFR2: Dataset. Create and populate (in your own embedded H2 database, based on the provided sample code) tables named `lab7_rooms` and `lab7_reservations`. Follow the schema from the standard INN database and add test data to your tables as needed. You are not required to rebuild the entire INN database; create a small sample of records for testing.

NFR3: DBMS Access. Your system shall access an embedded H2 database using JDBC.

Important Note: While H2 supports most of the ANSI-standard SQL syntax we have used this quarter, please make sure to consult H2 reference documentation for details about date/time functions, data types, and other implementation specifics:

- `https://h2database.com/html/main.html`
- `https://h2database.com/html/grammar.html`
- `https://h2database.com/html/functions.html`

NFR4: User Interface. Your system should use either a command line / text user interface. Exact output format is up to you. Please use consistent formatting.

NFR5: Protection against malicious input. Your application should assume that the end user is well-versed in SQL injection attacks. Construct and execute your SQL statements accordingly.

Functional Requirements

Your application must support the following features, accessed via a main menu. Again, you may choose to implement using either a GUI or command line interface.

FR1: Rooms and Rates. When this option is selected, the system shall output a list of rooms to the user sorted by name Include all information from the `lab7_rooms` table, as well as the following:

- Next available check-in date (or “Today” if the room is currently unoccupied)
- Date on which the next reservation in the room begins (or “None” if there are no future reservations)

FR2: Reservations. When this option is selected, your system shall accept from the user the following information:

- First name
- Last name
- A room code to indicate the specific room desired
- Begin date of stay
- End date of stay
- Number of children
- Number of adults

If the room is not available for the entire date range, or if the requested person count (children plus adults) exceeds the maximum capacity of the selected room, print a message indicating that the reservation cannot be completed. To reserve a block of rooms, it would be up to the user to submit multiple reservation requests.

If the reservation can be made without conflict, add a row to your `lab7_reservations` table and provide the user with a confirmation screen that displays the following:

- First name, last name
- Room code, room name, bed type
- Begin and end date of stay
- Number of adults
- Number of children

- Total cost of stay, based on a sum of the following:
  - Number of weekdays multipled by room base rate
  - Number of weekend days multiplied by 110% of the room base rate

FR3: Reservation Change. Allow the user to make changes to an existing reservation. Accept from the user a reservation code and new values for any of the following:

- First name
- Last name
- Begin date
- End date
- Number of children
- Number of adults

Allow the user to provide a new value or to indicate "no change" for a given field. Update the reservation based on any new information provided. If the user requests different begin and/or end dates, *make sure to check whether the new begin and end dates conflict with another reservation in the system.*

FR4: Reservation Cancellation. Allow the user to cancel an existing reservation. Accept from the user a reservation code, confirm the cancellation, then remove the reservation record from the database.

FR5: Revenue Summary. When this option is selected, your system shall provide a month-by-month overview of revenue for the current calendar year. For reservations that span multiple months, revenue should be computed based on the month in which the reservation ends.

Your system shall display a list of rooms, and, for each room, 13 columns: 12 columns showing dollar revenue for each month and a 13th column to display total yearly revenue for the room. There shall also be a "totals" row in the table, which contains column totals. All amounts should be rounded to the nearest whole dollar.

Submission

Submit one set of deliverables per team:

- The entire code for your application
- A `README` file which includes (a) the list of all members of the team, (b) any compilation/runtime instructions, and (c) information about any known bugs and/or deficiencies.

Submit the lab via Canvas as a single archive (either `lab7.zip` or `lab7.tar.gz`).