CS166: Project Phase 3

October 27, 2022

Introduction

In this phase you are provided a package including SQL schema, dataset, and a template user interface in Java. The dataset consists of data records that should be loaded into your database system. The template user interface in Java shows you how to connect to the PostgreSQL database and how to execute SQL queries.

Additional extra points will be awarded to projects that use GUI interfaces and properly handle exceptional situation by providing user-friendly error messages.

Please follow the below steps to get started:

1. Your database should of been created in Lab 6(Java and SQL). The first line is to stop the current PostgreSQL server if you forgot to stop the server from the last session. The second line is to start the server. The third line is to create the database.

   ```
   source ./lab5/stopPostgreDB.sh
   source ./lab5/startPostgreSQL.sh
   source ./lab5/createPostgreDB.sh
   ```

2. Download project.zip from elearn (Canvas).

3. In the download directory execute the following command:

   ```
   unzip project.zip
   ```

   You will see that a directory named "project" will be created. This directory contains all necessary files to get started. More specifically it contains the following things:

   - **project/data** - contains the files, which will be used to populate your database

- **project/sql/src/create_tables.sql** - SQL script creating the database relational schema. It also includes the commands to drop these tables. Some tables have been created in slightly different way from Phase 2 ER Diagram for simplicity. Check this .sql file before starting working on the project.
- **project/sql/src/create_indexes.sql** - SQL script which creates database indexes. Initially is empty, you should add all your indexes to this file.
- **project/sql/src/load_data.sql** - SQL script for loading the data in your tables. The script loads each text file into the appropriate table. **Note that the file paths have to be changed to absolute paths in order to make it work**.
- **project/sql/scripts/create_db.sh** - shell script, which you should to setup your database.
- **project/java/src/Retail.java** - A basic java User Interface to your Postgres database. You should modify this program and write all your SQL-specific code there.
- **project/java/scripts/compile.sh** - compiles&runs your java code.

4. Change path to data files in **project/sql/src/load_data.sql**. Use absolute paths to avoid ambiguity. After that your load statements should look like this:

```
COPY USER_LIST
FROM '/home/user/project/data/usr_list.csv'
WITH DELIMITER ';';
```

5. Execute **project/sql/scripts/create_db.sh** to load your database
6. Execute **project/java/scripts/compile.sh** to compile and run your Java client.

Java console application

If this is the first time you work with Java there is no need to be worried. You are provided with a template client written in Java. You are expected to extend this basic client and add the functionality, required for a complete system. An Introduction to Java/JDBC can also be found in your Textbook (Sections 6.2 – 6.3 of Database Management Systems (Third edition)).

In this phase we basically want to create an interactive console for non-sql users (i.e. Customers and Managers of the Retail Store). You should

therefore pay special attention to making the interface as intuitive as possible for regular user.

In order to run the template Java program you should first start the Postgres database as it was described in Lab 6. Then you should execute shell script *project/java/scripts/compile.sh*

The script will compile and run Java source for the file *project/java/src/Retail.java*. If you will add other Java source files to your project, please modify the script accordingly.

Functionality of the Online Retail Store

Below you will find the basic requirements for the functionality of the system. On top of them you may implement as many additional features as you want. Throughout the project you should keep in mind that the users are not SQL-aware. You should therefore make the user interface as intuitive as possible. We initially planned to provide functionality in the following three groups:

Below we provide the basic operations you must implement in your system.

1. Users

    - New User Registration: when a new user comes to the system, he/she can setup a new account through your interface, by providing necessary information.
    - User Login/Logout: user can use his/her name and password to login into the system. Once he/she logs in, a session will be maintained for him/her until logout (an example has been provided in the Java source code).
    - Browse Stores. Allows the user to see the list of stores within 30 miles of his/her location. (**Note**: We have provided a calculateDistance() method in *Retail.java* file which takes two latitude, longitude pairs and calculate euclidean distance. We avoided original latitude, longitude distance calculation (haversine distance) for simplicity.)
    - Browse Products. Allows the user to input the storeID of a store and then returns the list of products of that store with all the information like name, number of units available, and price per unit.
    - Order Products. User can order any product from the store within 30 miles radius of his/her location. User will be asked to input

storeID, productName, and numberofUnits. After placing the order, the order information needs to be inserted in the Orders table. Product tables will need to be updated accordingly.

- Update Product Information: For Managers, they can update the information of any product given the storeID. Manager can only update the product information (number of units, price per unit) of the store he/she manages. Product and ProductUpdates tables will need to be updated accordingly if any updates take place. Manager can also view the information of last 5 recent updates of his/her store(s).
- Admin: Admins will be able view and update the information of all users and products information of the database.

2. Orders

   - Browse Orders List: Customers will be able to see the last 5 of his/her recent orders from the Orders table. They will be able to see storeID, storeName, productName, number of units ordered and date ordered. A customer is not allowed to see the order list of other customers.
   - Manager can see all the orders information of the store(s) he/she manages. They will be able to see orderID, customer name, storeID, productName, and date of order for each order.
   - Popular product and customer: Manager will be able to see top 5 most popular products (product name) in his/her store(s) (Based on the order count of Product). Manager can also view the top 5 customer's information who placed the most orders in his/her store(s).

3. ProductRequests

   - Put Supply Request: Manager can put product supply request for any product of his/her store(s).For that, they will need to input storeID, productName, number of units needed, and warehouseID of the warehouse which will supply the supply request. After placing the request we can assume that warehouse has enough products to process the request. So Product table and ProductRequests table should be updated accordingly after placing the supply request.

Extra Credit

1. Good User Interface.

   A good user interface will bring more users to your application, and also more points in this phase. A user interface is good if it is:

   - Easy for users to explore features;
   - Robust in exceptional situations, like unexpected inputs;
   - Graphic interface supporting all required features.

2. Triggers and Stored Procedures.

   Instead of processing the workflow step by step, triggers and stored procedures can be used to handle a sequence of SQL command and to execute these sequences on some table events (inserts, updates, etc). For example, you can write a trigger which will be adding the order information in the orders table after a customer places an order. Or you can write a trigger to add product update information on ProductUpdates table when a manager updates a product. You can also find other opportunities for triggers and stored procedures.

   To submit your triggers and stored procedures with your project, please include them in the following location *project/sql/src/triggers.sql*

3. Performance Tuning

   The performance can be improved if index are used properly. **You will receive points only if the index will actually be used in your queries**. You should defend why you have chosen to use an index at some particular table. For your submission, you should put index declarations in *project/sql/src/create_indexes.sql*

4. Any Other Fancy Stuff...

   Please feel free to show any of your ideas to improve your project! The only thing required will be clear documentation!

Submission Guideline

1. Project Report

   You should provide a high level description (1-2 pages) of your implementation. You should describe which part each of you was working

on and any problems/findings, you found along your way. In this document you should also include whatever is asked in the below individual descriptions.

2. Files

   The following files should be submitted:

   (a) Create Tables, Bulkload Data Scripts.

   If you have any schema or tables modifications you should include your changes into the files and leave necessary comments for them. Modify the following files: *project/sql/src/create_tables.sql*, *project/sql/src/load_data.sql*

   (b) System Implementation.

   Submit your source file(s). You should make sure that your code can be compiled and run successfully. **Any special requirement for compiling and running should be stated clearly in your project report, or a README file which comes with your source code**. Please put all your source code within the *project/java/src* directory.

   (c) Other scripts, like triggers, stored procedures, and indexes.

   You should provide descriptions for these features and include all your scripts within the *project/sql/src* directory.

   You should submit a single zip archive (**1 submission per group!**) via elearn (Canvas) within the due date. The due date for this phase is **Friday, December 2nd, 11:59PM.**.

   NB: There will be demonstration sessions for each project group on **December 5-6, 2022**. We will post the schedule for the demo sessions later, and you can choose your slot from them. **Both of the group members must be present during their scheduled demo session.**