**CPSC 304 Project Cover Page**

Milestone #: 4

Date: 2023-08-07

Group Number: 34

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Student Number** | **CS Alias (Userid)** | **Preferred E-mail Address** |
| Kyle Nguyen | 76276393 | k1s9g | kylnguyen@hotmail.com |
| Michael Paknys | 30618185 | k4r6m | mpaknys@icloud.com |
| Hoi Suen Wong | 18947440 | t8a8t | Johnnwong@hotmail.com |

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

**Repository Link**: <https://github.students.cs.ubc.ca/CPSC304-2023S-T2/project_k1s9g_k4r6m_t8a8t>

**Program Link:** <https://www.students.cs.ubc.ca/~kynguyen/Restaurants_Manager.php>

**Project Summary**

In this database project, we modelled a domain on restaurant logistics called Restaurants Manager. We have successfully created a relational database with Oracle SQL\*Plus, utilizing PHP to produce the backend to frontend of our project. Accessing the program page, we can access the database instances which includes Restaurants, Employees divided between Managers, Cooks and Servers, Menus, Menu Dishes, Ingredients, Suppliers, Customers, Orders, and more.

This program allows us to view any relation instance and view the tuples in each of them, such as orders for example, to see content such as the review given by a customer pertaining to dishes that they ordered, and the servers who served them. The program properly inserts new cooks into the corresponding program without violating constraints, and it can update or delete current employees in the database. We can select specific attributes of a table we want to view and add constraints to them. We can also simply project the attributes we want to see from the Orders table. We can obtain the info of the orders taken from a given server employee. There are other special queries which overall can give insight for the user to compare and critique restaurants, such as counting the number of employees at each restaurant or viewing the restaurants that offer lower average wages compared to the total average wage of all restaurant employees. We can also see which customers are ordering over a given amount of orders. Lastly we can find suppliers that supply all ingredients in the database, such that restaurants may want to convenient use them to supply their ingredients.

**Project Changes**

There has been several changes throughout the progress of the project, such as changing the participation constraints in the ER diagram, and realizing the changes on the DDL. This allowed us to not require assertions, and to have a more convenient method of adding tuples to tables such as employees or customers without having to make sure they have at least one relation they are in, such as at least one place of employment, or one order, respectively. The IsA constraints were changed to overlapping from disjoint to allow employees to have multiples, as seen in our sample data, where every employee should at least be a server. The ON UPDATE lines of code were removed as they were redundant, and the ON DELETE lines were updated to match the participation constraints in the one-to-many or one-to-one cardinalities in our relations. Lastly, a few more insertion statements were added for additional sample data that our queries can use. The latest and final ER diagram is attached below.

A diagram of a flowchart

Description automatically generated

**Queries Included**

Here is the list of queries the program provides, and sample visualization:

* **OPTIONAL: Reset**
  + Reset the program and run the DDL script to add all relations and their tuples
* **Insert**
  + HTML: Line 48, php: Line 972
  + Add a new cook to Cooks, while also adding a new tuple to Employees\_Main, and Employees\_FT if the given hours per week and FT status is new

A screenshot of a computer

Description automatically generated 🡪 A screenshot of a computer

Description automatically generated🡪 A screenshot of a computer

Description automatically generated

* **Delete**
  + HTML: Line 81, php: Line 1064
  + Remove an Employee by their ID, or by random

A screenshot of a computer screen

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer screen

Description automatically generated

* **Update**
  + HTML: Line 95, php: Line 1103
  + Update an Employee’s details given their ID

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a questionnaire

Description automatically generatedA screenshot of a computer screen

Description automatically generated



A table of numbers and names

Description automatically generated

* **Selection**
  + HTML: Line 108, php: Line 1181
  + From a selection of tables from a dropdown, type what attributes and conditions to select from

A screenshot of a computer

Description automatically generated🡪 A screenshot of a computer

Description automatically generated🡪 A group of address labels

Description automatically generated

* **Projection**
  + HTML: Line 144, php: Line 1206
  + Choose attributes to project from the Orders table

A screenshot of a computer

Description automatically generated**🡪** A screenshot of a computer

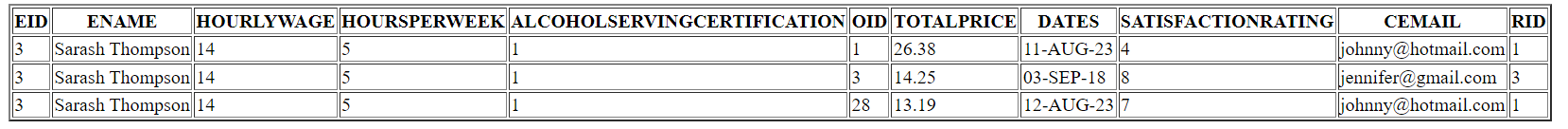
Description automatically generated

* **Join**
  + HTML: Line 191, php: Line 1243
  + Joins Employees, Servers and Orders given the EID

A close-up of a computer screen

Description automatically generated





* **Aggregation with Group By**
  + HTML: Line 203, php: Line 1258
  + Get Count of Employees at Each Restaurant

A close-up of a sign

Description automatically generated



A table with black text

Description automatically generated

* **Aggregation with Having**
  + HTML: Line 214, php: Line 1269
  + Find customers’ count of orders who order more than provided number of orders
  + A screenshot of a computer

    Description automatically generated**🡪**A white box with black text

    Description automatically generated
* **Nested Aggregation with Group By**
  + HTML: Line 227, php: Line 1281
  + List restaurants and their average hired employees’ wages if there are lower than the average wage of all employees in the database

A white background with black text

Description automatically generated



A close up of a sign

Description automatically generated

* **Division**
  + HTML: Line 238, php: Line 1311
  + List suppliers that supply every ingredient in the database

A close up of a sign

Description automatically generated🡪A close up of a sign

Description automatically generated

* **OPTIONAL: Count Tuples**
  + Lists the number tuples in each table
* **OPTIONAL: View Tuples**
  + Lists the tuples of a chosen table