Johnny Basurto Rosales

CSc 22100

Hesham Auda

***Assignment 4***

**Problem**

1. Implement the database schema:
   1. Students(empID, firstName, lastName, email, gender)
   2. Courses(courseID, courseTitle, department)
   3. Classes(courseID, studentID, sectionNo, year, semester, grade)
   4. Use “F”, “M”, or “U” for gender.
   5. Use letter grades A, B, C, D , F, or W only.
2. Using Java
   1. Connect to PostgreSQL
   2. Create tables on PostgreSQL with the database schema.
   3. Update Student, Courses, Classes tables.
   4. Use PreparedStatement objects
3. Use the data from scheduleSpring2021 to populate Courses and Classes table.
   1. Set grades initially to NULL.
4. Use the SQL UPDATE functionality to update the student grades.
5. Using GROUP BY:
   1. Calculate the number of students for each letter grade in Csc 22100.
   2. Display the number of students for each letter grade in Csc 22100.
6. Use MyPieChart showing:
   1. Different color
   2. Legend
   3. Grades in alphabetical order **Clockwise**.
7. Show the sample input tables
8. Show the output table for the stated query.
9. Code is applicable to canvases of variable height and width.
10. The size of the pie chart is proportional to the smallest dimension of the canvas.
11. Use any RDBMS of my choice. I use **PostgreSQL**.

**Solution & Methods**

**Student**

Includes variables specified by database schema for Students and methods to get and set the variables.Graphical user interface, text, application, email

Description automatically generated

Table

Description automatically generated with medium confidence

**Stu\_Methods**

Includes code to create a table in PostgreSQL, insert data into a table, and delete a student.

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

**Courses**

Class that includes variables specified for the Class schema and methods to set and get variables.

Table

Description automatically generated

**Cours\_Methods**

Includes methods to create a table, and insert data.Text

Description automatically generated

**Classes** -Class that initializes variables specified by the database schema for Class. Graphical user interface, text, application, email

Description automatically generatedA picture containing table

Description automatically generated

**Clas\_Methods**

Includes methods that create a table in postgreSQL as specified by the scheme for Classes, insert data into the Classes table and update grades in the table.

**Text

Description automatically generated**

**Connect**

Class that uses the jdvc driver to connect to my database in postgresql. It also includes a close method that closes the connection. I wrote a separate class for these steps because they are repeatedly used in my other classes!

Graphical user interface, text, application, email

Description automatically generated

**HistogramAlphaBet**

The class extends application. It is the same class as in HistogramAlphaBet in assignment 3. The difference were placed in brackets and mainly serve to process the data from my database.

The following code prints out the grades and their frequency as required for this assignment.

**Graphical user interface, application, Teams

Description automatically generated**

Insert all grades into a MyPieChart object for the MyPieChart object to be drawn.

**Text

Description automatically generated**

The following is the main method. It is my **sample table input**. Class objects were initialized with grades set to null.

**Text

Description automatically generated**

**Table

Description automatically generated**

**SQL Statements**

**SELECT grade, COUNT(grade) AS counts FROM classes WHERE courseID = ? GROUP BY grade**

Selects the columns called grade and creates a column called counts that groups the grades if that are equal to eachother and counts their frequency where the courseID is specified – in our case “22100 F”.

**SELECT \* from classes**

Selects every column in the table classes. This was used to insert every grade into a MyPieChart object.

**CREATE TABLE IF NOT EXISTS Classes (courseID varchar(30) NOT NULL, empID int NOT NULL, sectionNO int NOT NULL, year varchar(4), semester varchar(10), grade varchar(1)," PRIMARY KEY (courseID, empID, sectionNO)**

If a table does not already exists, it create ones. Then it creates columns in the following order: courseID, empID, sectionNO, year, semester, grade. The primary keys are set to courseID, empID, and sectionNO.

**"INSERT INTO Classes (courseID, empID, sectionNO, year, semester, grade) VALUES (?, ?, ?, ?, ?, ?)"**

Used to insert new data into the classes table. Every column data must be provided.

**UPDATE Classes SET grade = ? WHEREempID = ? AND courseID = ?**

Updates the classes table with a grade for the students with a specified empID and courseID.

**CREATE TABLE IF NOT EXISTS Courses (courseID varchar(30) NOT NULL, courseTitle varchar(100), department varchar(100), PRIMARY KEY (courseID))**

If a table does not already exists, it create ones. Then it creates columns in the order it is typed. The varchar keyword specifies how long the string will be. I gave it a larger integer to make sure that my data does not go over the limit.

**INSERT INTO Courses (courseID, courseTitle, department) VALUES (?, ?, ?)")**

Inserts new data into the courses table with a specified value using setString(). Every column data must be provided.

**CREATE TABLE IF NOT EXISTS Students(empID bigint NOT NULL, firstName varchar(52), lastName varchar(52), email varchar(100) gender varchar(100), PRIMARY KEY (empID)**

If a table does not already exists, it create ones. Then it creates columns in the order it is typed. All the columns were chosen to match the database schema given.

**INSERT INTO Students (empID, firstName, lastName, email, gender) VALUES(?,?,?,?,?)**

Inserts new data into the students table with a specified value using setString(). Every column data must be provided.

**DELETE FROM Students WHERE empID = ?**

Deletes every row that matches the where condition that empID equals a specified integer.

**Proportionality**

****

**A picture containing timeline

Description automatically generated**

**Results**

Output tables using data provided.

**Graphical user interface, application

Description automatically generatedGraphical user interface

Description automatically generatedTable

Description automatically generated**

After using the method update, the class table was updated with grades.**Text

Description automatically generated**

**Graphical user interface, application

Description automatically generated**

New output table showing the all students that take 22100 F. This was done by initializing the class object’s courseID = “22100 F”.

**Text

Description automatically generated**

**Graphical user interface, application

Description automatically generated**

**Graphical user interface, text

Description automatically generated**

**Chart, pie chart

Description automatically generatedA picture containing text

Description automatically generated**

Prints out the grade A, B, C, or W followed right after by the frequency of that grade. For example, for A it is 2.