

1 Preliminaries

Before starting on this assignment, please be sure to read the General Instructions that are on Piazza (under Resources->General Resources). If you did Lab1, you should already know how to log in to the class PostgreSQL server. You'll get help on Lab2 in your Lab Section, not the Lectures, so *be sure to attend Lab Sections*.

2 Goal

The goal of the second assignment is to create a PostgreSQL data schema with 6 tables that are very similar to the tables that you created in Lab1. The tables have the same names, attributes and data types as the tables of Lab1, and the same primary keys but there are some UNIQUE constraints and some restrictions on NULLs.

After you create the data schema with the 6 tables, you will be required to write some SQL statements that use those tables. Under Resources→Lab2, we'll provide you with data that you can load into your tables so that you can test the results of your queries. Testing can prove that a query is wrong, but not that it is right, so be careful.

Lab2 is due in two weeks, so you will have an opportunity to discuss the assignment during the Discussion Section in the first week of the assignment, and to discuss issues you have had in writing a solution to the assignment during the Discussion Section of the second week. Instructions for submitting the assignment appear at the end of this document.

3 Lab2 Description

3.1 Create PostgreSQL Schema Lab2

You will create a Lab2 schema to set apart the database tables created in this lab from ones you will create in future, as well as from tables (and other objects) in the default (public) schema. Note that the meaning of schema here is specific to PostgreSQL and different from the general meaning. See [here](#) for more details on PostgreSQL schemas. You create the Lab2 schema like this:

```
CREATE SCHEMA Lab2;
```

Now that you have created the schema, you want to set Lab2 to be your default schema when you use psql. If you do not set Lab2 as the default schema, then you will have to qualify your table names with the schema name (e.g. Lab2.Customers). To set the default schema, you modify your search path. (For more details, see [here](#).)

```
ALTER ROLE username SET SEARCH_PATH to Lab2;
```

You will need to log out and log back in to the server for this default schema change to take effect. (Students often forget to do this.)

You do not have to include the CREATE SCHEMA or ALTER ROLE statements in your solution.

3.2 Create tables

You will create tables in schema Lab2 for the tables Products, Customers, Stores, Days, Sales and Payments. The attributes of these 6 tables are the same as for the tables of Lab1. Data types for the attribute names in these tables are also the same as the ones specified for the tables of Lab1, and the Primary Keys of these tables are also the same. However, the tables must have the additional constraints described in the next section.

3.2.1 Constraints

The following attributes cannot be NULL. All other attributes can be (but remember that attributes in Primary Keys also cannot be NULL).

- In Stores: storeName
- In Products: normalPrice
- In Customers: custName

Also, the following must be unique for the specified table. That is, there cannot be identical rows in that table that have exactly the same (non-NULL) values for all of those attributes (composite unique constraint).

- In Stores: the attribute storeName
- In Products: the 2 attributes productName and manuF
- In Customers: the 2 attributes custName and address

For example, the third constraint says that there can't be two rows in Customers that have the same values for both custName and address (if both custName and address are not NULL). Think of this as saying that there can't be two different customers who have both the same custName and the same address.

You will write a CREATE TABLE command for each of the 6 tables. Save the commands in the file create.sql

4 SQL Queries

Below are English descriptions of the five SQL queries that you need to write for this assignment, which you will include in files queryX.sql, where X is the number of the query, e.g., your SQL statement for Query 1 will be in the file query1.sql, and so forth. Follow the directions as given. You will lose points if you give extra tuples or attributes in your results, if you give attributes in with the wrong names or in the wrong order, or if you have missing or wrong results. You will also lose points if your queries are unnecessarily complex, even if they are correct.

Terminology: A product appears as a row in Products, a customer is someone who appears as a row in Customer, a store appears as a row in Stores, a day appears as a row in Days, a sale appears as a row in Sales, etc. The name of a product is given by productName, the name of a customer is given by custName, the name of a store is given by storeName, etc.

4.1 Query 1

Find the productID and name for each product whose name begins with 'Intelligent'. Your result should be ordered alphabetically by the product's name. No duplicates should appear in your result.

4.2 Query 2

Output the productid, customerID and storeID for each sale that was made on January 16, 2018 that had a paidPrice that was more than 20.00 and a quantity that was more than 5 . The attributes in your result should appear as theProduct, theCustomer and theStore . No duplicates should appear in your result.

4.3 Query 3

Find the storeName and dayDate of each store for which there was at least one sale on that dayDate. No duplicates should appear in your result.

4.4 Query 4

Find the customerID and name of each customer who bought at least two different products at the same North region store. No duplicates should appear in your result.

4.5 Query 5

If a customer's status is 'H', that's a high status customer. ('M' is for medium, 'L' is for low.) The revenue from a sale equals paidPrice * quantity for that sale.

For each sale in which all of the following are true:

- a) the revenue is greater than or equal to two hundred,
- b) the customer is a high status customer, and
- c) the customer's address isn't NULL,

output the customerID, customer address, revenue and dayDate for that sale. The attribute for the revenue of the sale should appear as theRevenue in your result. No duplicates should appear in your result.

5 Testing

While your solution is still a work in progress, it is a good idea to drop all objects from the database every time you run the script, so you can start fresh. Of course, dropping each object may be tedious, and sometimes there may be a particular order in which objects must be dropped. The following commands (which you can put at the top of `create.sql` if you want, but you don't have to), will drop your Lab2 schema (and all objects within it), and then create the (empty) schema again:

```
DROP SCHEMA Lab2 CASCADE;  
CREATE SCHEMA Lab2;
```

Before you submit, login to your database via `psql` and execute your script. As you've learned already, the command to execute a script is: `\i <filename>`.

Under Resources→Lab2 on Piazza, we will provide a load script named *lab2_data_loading.sql* that loads data into the 6 tables of the database. You can execute that script with the command:

```
\i lab2_data_loading.sql.
```

You can test your 5 queries using that data, but you will have to figure out on your own whether your query results are correct. We won't provide answers, and students should not share answers with other students. Also, your queries must be correct on any database instance, not just on the data that we provide. You may want to test your SQL statements on your own data as well.

6 Submitting

1. Save your scripts for table creations and query statements as create.sql and query1.sql through query5.sql. You may add informative comments inside your scripts if you want (the server interprets lines that start with two hyphens as comment lines).
2. Zip the file(s) to a single file with name Lab2_XXXXXXX.zip where XXXXXXX is your 7-digit student ID. For example, if a student's ID is 1234567, then the file that this student submits for Lab2 should be named Lab2_1234567.zip

To generate the zip file you can use the Unix command:

```
zip Lab2_1234567 create.sql query1.sql query2.sql query3.sql query4.sql query5.sql
```

(Of course, you use your own student ID, not 1234567.)

3. You should already know how to transfer the files from the UNIX timeshare to your local machine before submitting to Canvas. If you are still not familiar with the process, use the instructions we provided at the Lab1 assignment.
4. Lab2 is due by 11:59pm on Sunday, April 28. Late submissions will not be accepted, and there will be no make-up Lab assignments.
5. You may lose credit if your queries are unnecessarily complex, even if they are correct. For example, if you use DISTINCT but it's not needed, you might lose half a point. Of course, you will also lose credit if DISTINCT is needed and you omit it.
6. Be sure to follow directions about Academic Integrity that are in the Syllabus and Lecture1. If you have any questions about those directions, please speak to the instructor as soon as possible.