Course: Data Acquisition — D205

TGM3 Task 1: Data Acquisition

Student name: Amy Pett

ID number: 012220223

Date: 8/7/2024

## A: Question

I will answer the question “What is the total of emergency admissions for patients who drink soda and have diabetes?” using the medical\_data database and the add-on file mservices.csv. This data could hypothetically/potentially be used in a public health campaign aimed at reducing emergency visits among individuals with diabetes who answered ‘Yes’ to drinking soda.

## A1: Question Justification

I will first import that mservices.csv file to the medical\_data database. I will then run an SQL query in the medical\_data database that joins the admissions, patient, and mservices tables. I will write code to filter a count of emergency admissions per patient that have diabetes and drink soft drinks. These three data points are in each of the three tables, respectively.

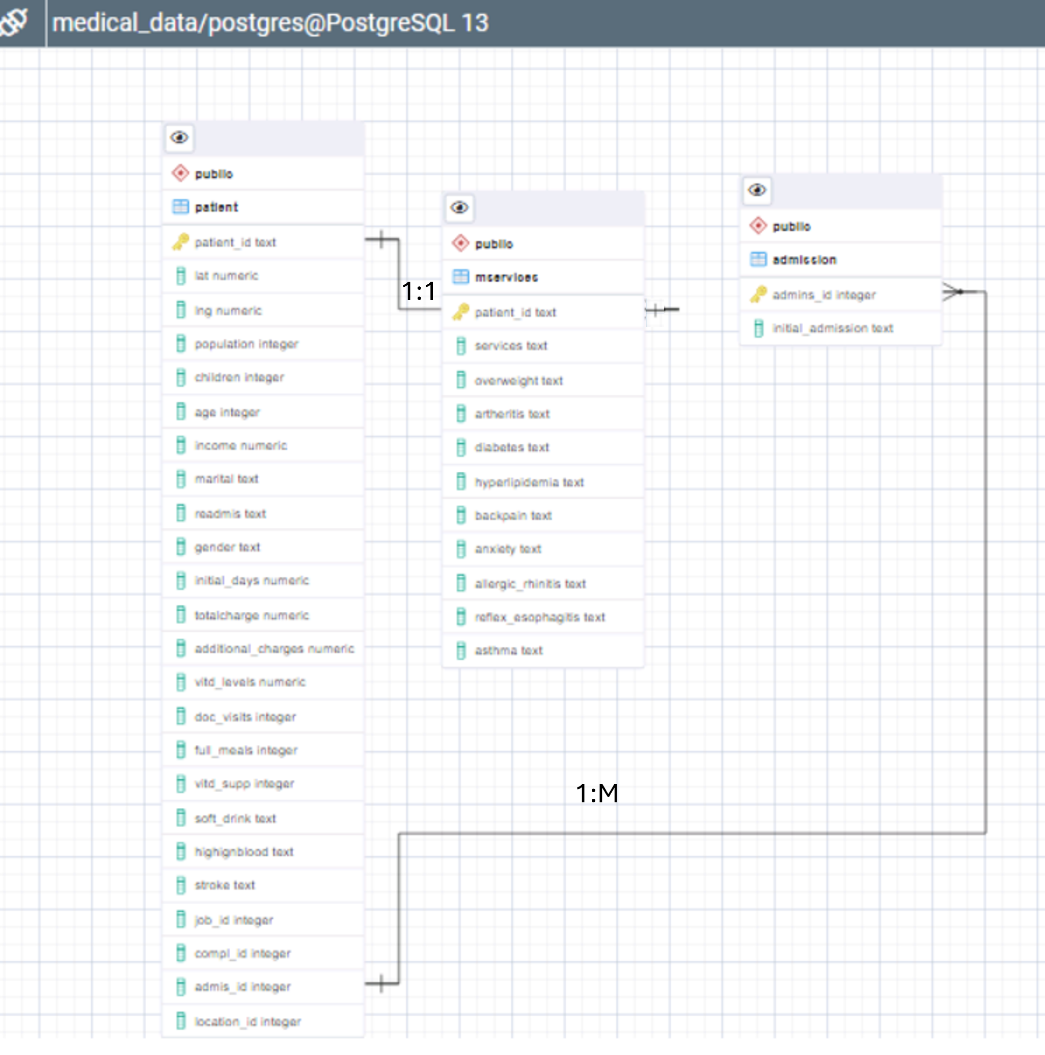
## A2: Identifying Data

I have chosen the medical\_data database. The tables needed to answer my question are the admissions table (original dataset), patient table (original dataset), and mservices table (which comes from the add-on dataset). I will create a new table and import the mservices table from the lab files to the original dataset.

The columns I will need to include in my code will be the primary and foreign keys for the joins and the columns I will use to filter the data. Those columns that will be the primary/foreign keys are patient\_id (from both patient and mservices tables) and admission\_id/admis\_id (from admission and patient tables). The columns I will be using in my “Where” statement to filter my data are soft\_drink (patient table), diabetes (mservices table), and initial\_admin (admission table).

|  |  |  |
| --- | --- | --- |
| Table | Column | Data Type |
| patient | patient\_id | Text |
| patient | soft\_drink | Text |
| patient | admis\_id | Integer |
| admission | admins\_id | Integer |
| admission | initial\_admin | Text |
| mservices | patient\_id | Text |
| mservices | diabetes | Text |

## B: Entity Relationship Diagram



B1: Relationship Discussion

The following table lists all the relevant keys from the ERD:

|  |  |
| --- | --- |
| Table | Keys |
| patient | patient\_id |
| patient | admins\_id |
| admission | admission\_id |
| mservices | patient\_id |

The table that shares a relationship with my add-on table is the patient table. Both the patient and mservices have the patient\_id. In this case, there is a 1:1 relationship between these tables. The patient table has a 1:M relationship to the admission table linked through the admission\_id column. The admission and mservices tables do not share any relationships.

I did need to manually update my diagram to show a 1:1 relationship between the patient and mservices tables. PGAdmin only had options to create a 1:M or M:M. Since those two tables are a 1:1 relationship, I corrected that after taking the screenshot. When I created the mservices table in PGAdmin I set the patient\_id as the primary key of it’s table and the foreign key to the Patient table using patient\_id on that table.

## B2: Statement for the ERD

This is the SQL code I used to create my add-on table:

CREATE TABLE mservices

(

     patient\_id text NOT NULL,

     Services text,

      Overweight text,

      Arthritis text,

      Diabetes text,

      Hyperlipidemia text,

      BackPain text,

      Anxiety text,

      Allergic\_rhinitis text,

      Reflex\_esophagitis text,

      Asthma text,

PRIMARY KEY (patient\_id),

FOREIGN KEY (patient\_id) REFERENCES patient(patient\_id)

);

## B3: Loading CSV Data

This is the SQL code I used to load the data from the mservices CSV add-on file to my newly created table in the medical\_data database:

COPY mservices

FROM 'c:\LabFiles\Medical\mservices.csv'

DELIMITER ','

CSV HEADER;

## C: SQL Query

The following is the SQL code I wrote to answer the question “What the total of emergency admissions for patients is who drink soda and have diabetes?”:

SELECT admission.initial\_admission, patient.soft\_drink, mservices.diabetes

FROM patient

JOIN admission

ON patient.admis\_id=admission.admins\_id

JOIN mservices

ON patient.patient\_id=mservices.patient\_id

WHERE mservices.diabetes='Yes'

      AND patient.soft\_drink='Yes'

      AND admission.initial\_admission='Emergency Admission'

## C1: CSV Files

See file ‘output.csv’ submitted with this document to verify the resulting data from my query. This is the code I ran to get that file:

COPY (SELECT admission.initial\_admission, patient.soft\_drink, mservices.diabetes

FROM patient

JOIN admission

ON patient.admis\_id=admission.admins\_id

JOIN mservices

ON patient.patient\_id=mservices.patient\_id

WHERE mservices.diabetes='Yes'

AND patient.soft\_drink='Yes'

AND admission.initial\_admission='Emergency Admission')

TO 'c:\labfiles\outpu.csv'

DELIMITER','

CSV HEADER;

## D: Add-on File Time Period

In a business case where this data may be tracked, it would be in tandem with a community health initiative to decrease soda consumption in the general population. Any campaign would take time to put together and then launch. Health changes will take weeks as well. A quarterly review of any trends in this data point seems reasonable.

## D1: Explanation of Time Period

A common belief is that making a new habit can take at least 21 days. Health improvements can also take a bit of time to be seen, so it seems unlikely that any changes in the trend will occur in a week or even a month. A quarterly update, however, allows for habit and health changes without going too long to monitor potential changes in the trend.

## E: Panopto Video

Please follow this link to view my submitted Panopto video showing my SQL code execution and results:

<https://d2y36twrtb17ty.cloudfront.net/sessions/e9478de7-6dde-4f04-b539-b2020130a362/0a395dfe-ba62-4bca-a742-b2020130a370-4de9e225-db90-4466-90b3-b2020132d013.mp4?invocationId=0a093cc6-db84-ef11-8293-12b1cb861383>

## F: Web Sources

Khan, Maham. (n.d.). *Joining data in SQL*. DataCamp. <https://app.datacamp.com/learn/courses/joining-data-in-sql>

Ludolf, Jasmin. (n.d.). *Intermediate SQL* DataCamp. <https://app.datacamp.com/learn/courses/intermediate-sql>

## W3Schools. (n.d.) *SQL FOREIGN KEY Constraint*.

## <https://www.w3schools.com/sql/sql_foreignkey.asp>

## G: Professional Communication