Metropolitan State University

Computer Sciences Department

CS 3810 - Principles of Database Systems – Spring 2022

Database Project 02

Deadline: March 27, 11:59pm

1. Introduction

The Centers for Medicare and Medicaid Services (CMS) is a U.S. agency that administers the Medicare program and works in partnership with state governments to administer Medicaid. CMS makes health related data available to the public through its website data.cms.gov.

In this assignment you will use one of CMS' datasets that contains hospital discharges by Inpatient Prospective Payment System (IPPS) hospitals, referred to as "Medicare Inpatient Hospitals - by Provider and Service" and available here (2019). This CMS dataset contains information about charges of top groups of similar clinical conditions (diagnosis) by different health providers in the U.S. and the correspondent amounts covered by health insurance. It shows how treatments for similar clinical conditions can be charged very differently depending on the health care provider.

2. Normalization

The CMS dataset has 188,806 rows and 15 columns. The goal of this assignment is to have you download this dataset in a CSV format so you can later load all of its data content into a Postgres database named `cms` that you will be creating. The `cms` database has to be designed so that all of its tables are normalized up to 3NF (third normal form). You should refer to the data dictionary, available here, to understand the meaning of each of the fields in the CSV file and the information below:

- `Avg_Submtd_Cvrd_Chrg`: that's what the provider billed Medicare (on average).
- `Avg_Tot_Pymt_Amt`: that's what Medicare actually paid, on average (including amounts paid by the patients: deductibles, co-payments, etc.)
- `Avg_Mdcr_Pymt_Amt`: that's what Medicare actually paid, on average (NOT including amounts paid by the patients: deductibles, co-payments, etc.)

We strongly recommend using the same names of the columns in the data dictionary when you name attributes on your tables.

All Data Definition Language (DDL) SQL statements (`CREATE DATABASE` and `CREATE TABLE` statements), DCL (Data Control Language) statements (`CREATE USER`, `GRANT` statements), and DQL (Data Query Language) statements should be submitted in a file named `cms.sql`. Get the template from here.

Other than being normalized up to 3NF, all `cms`'s tables of your database should have primary keys and, when applicable, appropriate foreign keys with referential integrity constraints in place. You should also create two users on this database: one named `cms_admin` with *full control* of all tables, and another one named `cms` with only *select* capabilities.

3. Data Load Script

In order to load the CMS CSV file into your (normalized) `cms` database you will have to write a data load program (accepted languages are Python or Java). This program should be named `cms.[py|java]` (extension depends on the programming language of your choice) and it is the second deliverable of this assignment. Data access secrets (user and password) should be protected in the code using best practices discussed in class. SQL statements should use prepared statements to follow good coding practices. I will also look for a comment header section that describes what the script is about, authors, and date when it was written.

Please note that **you are not allowed to pre-process or modify the CSV file** using an external program, like a spreadsheet application, for example. I will test your data load program using the CSV file obtained directly from data.cms.gov's download - CSV option. The CSV file should be the `MUP_IHP_RY21_P02_V10_DY19_PrvSvc_0.csv`. Because this file is large (47.7MB) and it can be downloaded from the internet, there is no need to submit it through Canvas.

4. Queries

Your final task in this project is to answer the following queries using SQL. Write your answers to all queries updating the `cms.sql` file. That's one of the files you are expected to submit through Canvas.

a) List all diagnostic names in alphabetical order.
b) List the names and correspondent states (including Washington D.C.) of all of the providers in alphabetical order (state first, provider name next, no repetition).
c) List the total number of providers.
d) List the total number of providers per state (including Washington D.C.) in alphabetical order (also printing out the state).
e) List the providers names in Denver (CO) or in Lakewood (CO) in alphabetical order
f) List the number of providers per RUCA code (showing the code and description)
g) Show the DRG description for code 308
h) List the top 10 providers (with their correspondent state) that charged (as described in Avg_Submtd_Cvrd_Chrg) the most for the DRG code 308. Output should display the provider name, their city, state, and the average charged amount in descending order.
i) List the average charges (as described in Avg_Submtd_Cvrd_Chrg) of all providers per state for the DRG code 308. Output should display the state and the average charged amount per state in descending order (of the charged amount) using only two decimals.
j) Which provider and clinical condition pair had the highest difference between the amount charged (as described in Avg_Submtd_Cvrd_Chrg) and the amount covered by Medicare only (as described in Avg_Mdcr_Pymt_Amt)?

5. Final Instructions

Get the SQL file template from here, where you can find a list of TO-DO's for this assignment, including the queries that you are required to answer using your postgres database. Remember, in this assignment you are required to submit 2 files:

- `cms.sql` AND
- `cms.py` (or `cms.java`)

There is no need to submit the CSV file!

6. Rubric

This programming assignment is worth 100 points, distributed in the following way:

+2 the two deliverables (source code and SQL file) use the naming format requested, with comment headers as suggested;
+15 Postgres database tables are normalized to 3NF;
+15 All tables are created correctly, having appropriate use of data types, names matching the data dictionary, NOT NULL used when appropriate, primary key defined correctly, and foreign key defined whenever is needed
+3 the requested users were created with privileges;
+15 data load program works as expected
+50 queries (+5 for each query)

I will not accept late submissions, unless for college-accepted excused reasons.

7. Bonus Points

Prove that you didn't do the normalization only using your "guts" but also what you learned in class. Show all 2NF or 3NF violations and normalization steps in detail and you will get up to +10 points.