**Pharmacy Claims**

Sakshi Mude

MPS Analytics, Northeastern University

ALY 6030 – Data Warehousing & SQL

Prof. Kayal Chandrasekharan

December 16, 2023

**Introduction**

This project delves into the realm of data normalization and database design, using a dataset of pharmacy claims from an insurance company managed by a Pharmacy Benefit Manager (PBM). The primary objective is to transform this raw data into a structured, relational database format that meets the Third Normal Form (3NF) standards. Through this process, we aim to create a test database that will facilitate the development of common SQL query reports, setting a foundation for efficient data analysis and management. This initial stage is crucial for ensuring a smooth transition to a fully functional data warehouse in a real-world business environment.

**Analysis**

**Part 1 : Normalization**

To normalize this data for a database, I would want to separate it into different tables to reduce redundancy and ensure each table represents one entity. Common entities I see in this dataset are 'Members', 'Drugs', and 'Claims'.

In the normalized structure I would create:

* I can minimize redundancies by ensuring member and drug information is not repeated with each claim, but rather connected to claims through foreign keys.

* Each table would represent a distinct entity in my design - Members, Drugs, and Claims.

* The relationships among these core tables can be established by adding foreign key columns that link records across the tables. For example, the Claims table would contain foreign keys referring to Members and Drugs. This defines the relationships without duplicating data.

**Q1:Fact Variable Types**

* Additive Facts: copay1, copay2, copay3, insurancepaid1, insurancepaid2, insurancepaid3

Can sum across dimensions like member\_id and drug\_ndc.

* Semi-Additive Facts: None in this dataset
* Non-Additive Facts: fill\_date1, fill\_date2, fill\_date3

Represent points in time, cannot be aggregated.

**Q2:Grain of the Fact Table:**

The grain of the fact table **fact\_claims.csv** is each individual pharmacy claim record. Each row represents a unique combination of a member's drug claim, including the dates of drug fills, copays, and insurance payments for each of these fills.

**Part 2 : Primary and Foreign Key Setup in MySQL**

**Q1: Primary Keys Designation**

* **dim\_member Table**:

Primary Key (PK): **member\_id**

Key Type: Natural Key. This is a unique identifier inherent to the data, representing each member.

* **dim\_drug Table**:

Primary Key (PK): **drug\_ndc**

Key Type: Natural Key. The National Drug Code (NDC) uniquely identifies each drug, making it a natural choice for a primary key.

* **fact\_claims Table**:

Primary Key (PK): If a surrogate key like **claim\_id** is used, then it's a Surrogate Key. Alternatively, a composite key comprising **member\_id**, **drug\_ndc**, and one of the **fill\_date** columns can be used.

Key Type: Surrogate Key (if **claim\_id** is used). This is an artificial key created specifically for the table to uniquely identify each claim.

**Q2: Foreign Keys Designation**

**fact\_claims** Table:

Foreign Key (FK): member\_id

References: member\_id in the dim\_member table.

Foreign Key (FK): drug\_ndc

References: drug\_ndc in the dim\_drug table.

**Q3:** **Foreign Keys Actions**

* For **member\_id** in **fact\_claims:**

On DELETE: SET NULL. This action is chosen because if a member record is deleted, we don't want to lose the claim records. Setting the member\_id to NULL in fact\_claims will maintain the integrity of the claim data while indicating that the member information is no longer available.

On UPDATE: CASCADE. If a member's details change (like an update in member\_id), this change should be reflected in the claims table to maintain data consistency.

* For **drug\_ndc** in **fact\_claims:**

On DELETE: RESTRICT. This prevents deletion of a drug record if there are existing claims associated with it, thus maintaining referential integrity.

On UPDATE: CASCADE. If the drug details are updated (like changes in drug\_ndc), these updates should be propagated to the claims to ensure accurate and up-to-date information.

**Part 3 : Entity Relationship Diagram**

A screenshot of a computer

Description automatically generated

**Part 4 : Analytics and Reporting**

**Q1: Count of Prescriptions by Drug Name**

**A screenshot of a medical form

Description automatically generated**

* For the drug “Ambien” there are 2 prescriptions filled.

**Q2: Prescriptions and Members by Age Group**

A screenshot of a phone

Description automatically generated

* There is 1 unique member over the age of 65.
* The unique member over the age of 65 filled 2 prescriptions.

**Q3: Insurance Payment for Most Recent Prescription**

**A screenshot of a computer

Description automatically generated**

* For member ID 10003, the drug name listed on their most recent fill date is Ambien.
* The insurance paid 250 for that medication on the most recent fill date.

**Conclusion**

In conclusion, this report has meticulously navigated the complexities of database normalization, achieving a structurally sound and query-efficient design through the application of Third Normal Form principles. We successfully imported our data into MySQL, creating a star schema that delineates clear relationships between our fact and dimension tables—a design that not only aligns with theoretical best practices but also promises practical ease-of-use for future business analytics. Through the establishment of primary and foreign keys, we have ensured the integrity and navigability of our dataset, laying a robust groundwork for the seamless integration of production data. The accompanying Entity Relationship Diagram (ERD) serves as a clear visual representation of our database structure, offering an intuitive guide for both technical and non-technical stakeholders.

**References**

1. *Third Normal Form  3NF*. (2023, December 4). GeeksforGeeks. https://www.geeksforgeeks.org/third-normal-form-3nf/
2. *SQL Window Functions | Advanced SQL - Mode*. (2016, May 23). Mode Resources. https://mode.com/sql-tutorial/sql-window-functions