Logo

Description automatically generated

**College of Professional Studies**

**Northeastern University San Jose**

**MPS Analytics**

**Course: ALY6030: Data Warehousing & SQL**

**Assignment:**

Module 1 Tech Crunch Assignment

**Submitted on:**

Mar 2, 2024

**Submitted to:**  **Submitted by:**

Prof: Kayal Chandrasekaran Nikshita Ranganathan

**Part 1. Normalization**

The dataset is structured as a collection of records, where each record represents a distinct funding event. The fields included within the dataset offer a detailed account of each event, encompassing a range of attributes that describe the funding transaction, the recipient company, and contextual details surrounding the investment. It contains 523 entries and 10 columns.

Normalization is a fundamental aspect of database theory, aimed at reducing redundancy and dependency by organizing data into a series of tables with well-defined relationships. Normalization involves various stages, known as normal forms, each targeting specific inefficiencies and anomalies. This critical process ensures the integrity of data, enhances the efficiency of queries, and simplifies the maintenance of databases.

* **Good choice - primary key :**

A primary key must uniquely identify each record in a table and must not be null. The “fund\_id” column serves as a robust choice for the primary key. It is an integer identifier that appears to be unique for each funding event (comprising distinct numbers ranging from 1 to 523), allowing for the identification of each row and easily retrievable.

**Not a valid primary key:**

An attribute that would not serve as a valid primary key is company since there are multiple funding events per company, which means this field will have duplicates. Similarly, fundedDate would not be a good primary key on its own because there could be multiple entries on the same date.

* **Does the table satisfy 1NF? Why or why not?**

The table appears to satisfy the First Normal Form (1NF) because:

* Each cell contains atomic values, meaning there are no repeating groups or arrays within cells.
* Each row appears to represent a single funding event.
* All entries in a column are of the same data type.

Every cell contains a single value, without multiple values being joined by a comma or other symbol. For example, in the company column, each row is dedicated to just one company and not like “MyCityFaces, Facebook”.

* **Does the table satisfy 2NF? Why or why not?**

First, a table must satisfy the First Normal Form (1NF) in order to satisfy Second Normal Form (2NF). Furthermore, all attributes that are not part of the primary key must be fully functionally dependent on the primary key. However, in this dataset, there are instances of partial dependency. For instance, the city attribute may be entirely dependent on the state attribute. This suggests the necessity for a new table where company could serve as a primary key to have information particular to the company, rather than the specific details of each funding occurrence.

* **Does the table satisfy 3NF? Why or why not?**

A table is in 3rd Normal Form (3NF) if it is in 2nd Normal Form and no non–key attribute is transitively dependent on the primary key. The table also does not satisfy the Third Normal Form (3NF). There are transitive dependencies present in the dataset. For example, city depends on the state and company columns. Similarly, numEmps is dependent on the company column. To eliminate these dependencies and meet the criteria for 3NF, it's necessary to reorganize the data into separate tables.

* **ERD Diagram**

A diagram of company data

Description automatically generated

***Figure 1 – ERD Diagram***

A single company\_id is associated with one or more fund\_ids, establishing a one-to-many relationship. Additionally, each company\_id is linked to a unique city\_id, forming a one-to-one relationship.

**Part 2. Case study**

* **Once you run the SQL code you’ll notice that each table has data populated for two recipes, Chicken Marsala and Absolute Brownies.**

**A screenshot of a computer

Description automatically generated**

***Figure 2 – Recipes already in the database***

**Two New recipes : Insertion**

The Recipe Database consists of the following tables:

* **categories**: Stores different recipe categories.
* **recipe\_main**: Contains main information about each recipe, such as title, description, category, preparation time, cooking time, servings, difficulty, and directions.
* **ingredients**: Lists all ingredients used in recipes.
* **rec\_ingredients**: Represents the many-to-many relationship between recipes and ingredients, along with the quantity of each ingredient used in a recipe.

Two new recipes are added to the database – Margherita Pizza and Choco Chip Cookie.

**Code –**

**# Recipe-1 Margherita Pizza**

A white background with text

Description automatically generated

**# Recipe-2 Chocochip cookie**

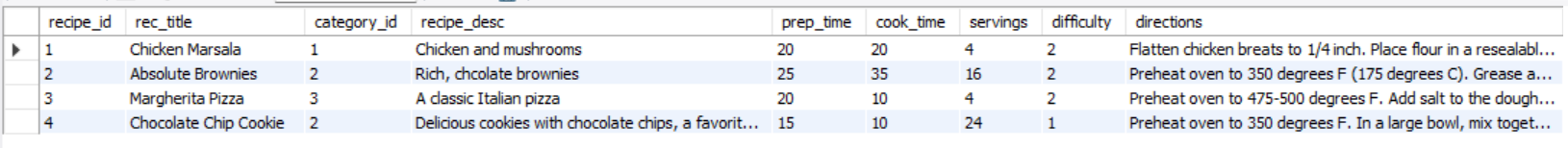
A close-up of a computer screen

Description automatically generated

**A screenshot of a computer

Description automatically generated**

***Figure 3 – categories table after insertion of two new recipes***

****

***Figure 4 – recipe\_main table after insertion of two new recipes***

**A list of food items

Description automatically generated**

***Figure 5 – ingredients table after insertion of two new recipes***

**A screenshot of a table

Description automatically generated**

***Figure 6 – rec\_ingredients table after insertion of two new recipes***

* **Write only one SQL query that returns all information on only the two new recipes you inserted from all the tables you created.**

**Code –**

**A computer screen shot of text

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

***Figure 7 – Code and Result of Part 2 Ques 2***

* **Write a SELECT query that identifies the recipe name, category name, and ingredient name, and ingredient amount. No other variables should be included.  
  Your output should be sorted first by descending category name, then by ascending recipe name, followed by descending ingredient name.**

**Code –**

A screenshot of a computer program

Description automatically generated

**A screenshot of a computer

Description automatically generated**

***Figure 8 – Code and Result of Part 2 Ques 3***

**CONCLUSION**

This project helped us understand the importance of data normalization for database efficiency and integrity, providing a hands-on experience with structuring data in 1NF, 2NF, and 3NF. Additionally, it offered practical SQL skills, particularly in data insertion and extraction using JOIN statements, emphasizing the significance of well-organized data for easy retrieval and analysis. Through these exercises, the project highlighted how normalization and SQL operations are fundamental to effective database management and data analysis.

**REFERENCES**

S, R. A. (2023b, October 11). *What is Normalization in SQL? 1NF, 2NF, 3NF and BCNF in DBMS*. Simplilearn.com. <https://www.simplilearn.com/tutorials/sql-tutorial/what-is-normalization-in-sql>

GfG. (2023, May 8). *Difference between Primary Key and Foreign Key*. GeeksforGeeks. <https://www.geeksforgeeks.org/difference-between-primary-key-and-foreign-key/?ref=header_search>