Data Modeling

Shahwar Saleem, July 14, 2021

1 Data Modeling with Relational Databases

Learn about fundamentals of how to do relational data modeling by studying:

- Normalization
- DeNormalization
- Fact/Dimension tables
- Different schema models

Some definitions:

- **Database** A set of related data and the way it is organized.
- **Database Manegment System** Computer software that allows users to interact with databases. Provides access to all of the data.

RULE 1: The Information Rule

All information in a relational database is represented explicitly at the logical level and in exactly one way - **by values in tables.**

1.1 Importance of Relational Databses

- **Standardization of data model**: Once your data is transformed into the rows and columns format, your data is standardized and you can query it with SQL
- **Flexibility in adding and altering tables**: Relational databases gives you flexibility to add tables, alter tables, add and remove data.
- **Data Integrity**: Data Integrity is the backbone of using a relational database. ACID
- **Structured Query Language (SQL)**: A standard language can be used to access the data with a predefined language.
- **Simplicity**: Data is systematically stored and modeled in tabular format.
- **Intuitive Organization**: The spreadsheet format is intuitive but intuitive to data modeling in relational databases.

1.2 OLAP Vs OLTP

Databases

Online Analytical Processing OLAP

Databases optimized for these workloads allow for **complex analytical and ad-hoc queries**. **These type of databases are optimized for reads.**

Online Transactional Processing OLTP

Databases optimized for these workloads allow for **less queries in large volume**. Types of queries: **read, insert, update, delete**

KEY DIFFERENCE

The key to remember the difference between OLAP and OLTP is analytics (A) vs transactions (T). If you want to get the price of a shoe then you are using OLTP (this has very little or no aggregations). If you want to know the total stock of shoes a particular store sold, then this requires using OLAP (since this will require aggregations).

1.3 Normalization

To reduce data redundancy and increase data integrity

Definition : The process of structuring a relational database in accordance with a series of **normal forms** in order to reduce data redundancy and increase data integrity.

Normalization organizes the columns and tables in a database to ensure that their **dependencies are properly enforced by the database integrity constraints.**

IMPORTANT

No need of extra copies of data. (Redundancy) We want data to be in one place which is source of truth (Integrity) More copies of data means more likely it is not properly updated.

1.3.1 Objectives of Normalization

- To Free database from unwanted insertions, updates and deletion dependencies. (If I wanna update my data, I would want to update it in ONE place).
- To reduce need for re-factoring the database as new types of data are introduced.
- To make relational model more informative to users. (Model real life concepts)
- To make the database neutral to query statistics. (Not make table for specific queries, all queries should be gathered over time).

1.3.2 Normal Forms

First Normal Form 1NF

- Atomic values: Each cell contains unique single values
- Be able to add data without altering tables.
- Separate different relations into different tables. Customer and Sales separate table.
- Keep relationships between tables together with foreign keys. (Ability to relate separate relations with each other)

Second Normal Form 2NF

- Have reached 1NF
- All columns in a table must rely only on Primary Key. Should not need 2 columns to reach the third. IF this is the case, then might need to split tables. If the column present in the table does not describe the entity, it may not fit the table.

Third Normal Form 3NF

- Must be in 2NF
- No transitive dependencies => Remember, transitive dependencies you are trying to maintain is that to get from A-> C, you want to avoid going through B.

When to use 3NF

When you want to update data, we want to be able to do in just 1 place. We want to avoid updating the table in the Customers Detail table

1.4 Denormalization

Must be done in read heavy workloads to increase performance.

Definition The process of trying to **improve the read performance of a database at the expense of losing some write performance** by adding redundant copies of data.

- Process of Denormalization is all about performance (read)
- **JOINs allow for flexibility but JOINS are really slow.** Therefore, sometimes, heavy read tables might need some duplicate data to avoid unnecessary JOINS.
- While dealing with Heavy Reads, you might want to think about Denormalization.
- Denormalization always comes after Normalization is done.
- Denormalization always needs more space.
- SELECTs will be faster => INSERT, UPDATE and DELETE are slower after Denormalization.

1.4.1 Fact and Dimension Tables

A general rule of thumb to help divide information under different categories of tables:

- **Fact table** consists of the measurements, metrics or facts of a business process. They are normally int or numbers. [How many of the products were bought?]
- **Dimension Table** A structure that categorizes facts and measures in order to enable users to answer business questions. Dimensions are people, products, place and time. [Where/When/what product was bought?]

IMPLEMENTING DIFFERENT SCHEMAS

Based on facts and tables, we can design **data mart schemas** in two different styles:

- Star Schema
- Snowflake Schema

1.5 Star Schema

- Simplest style of data mart schemas.
- Star Schema consists of 1 or more fact tables.
- Fact tables reference any number of dimension tables.
- Why Star? Fact Table is in the center.
- Dimension table surround a Fact table they want to reference.

BENEFITS of STAR SCHEMA

- Can denormalize tables
- Simplifies queries
- Fast aggregations [The work of creating tables to make easily accessible aggregations is push out of application development]

DRAWBACKS of STAR SCHEMA

- Denormalization issues
- Data integrity - Duplicated data
- Decrease query flexibility (Now modeling to query)
- Many to many relationships are hard to support.

1.6 Snowflake Schema

A snowflake schema has

- Multiple levels of relationships
- child tables having multiple parents
- A complex shape emerges when dimensions of snowflake schema are elaborated.

STAR Vs SNOWFLAKE

- Star schema is a very simplified and special case of **Snowflake** schema.
- One to Many relationships are NOT allowed in Star Schema while Snowflake schema allows it.
- Snowflake Schema is **MORE NORMALIZED** than Star schema but only in 1NF or 2NF.

1.7 Data Definition And Constraints

- NOT NULL: Column cannot contain a NULL value. (Can add NOT NULL to multiple columns in case of composite keys)
- UNIQUE: Specifies that the data accross all the rows in one column are unique.(Also used for multiple columns)
- PRIMARY KEY: defined on a single column, and every table should contain a primary key. The values in this column uniquely identify the rows in the table. If a group of columns are defined as a primary key, they are called a composite key. That means the combination of values in these columns will uniquely identify the rows in the table. By default, the PRIMARY KEY constraint has the unique and not null constraint built into it.