Database Design and Normalization Notes

Hia Al Saleh

November 25th, 2024

Contents

1 Introduction

1.1 Overview of the Relational Model

The relational model forms the foundation of modern database systems. It organizes data into relations, commonly referred to as tables, each representing a specific entity type.

Key Characteristics of a Table:

- A table contains rows and columns, where:
  - **Rows:** Represent individual records or facts about an entity.
  - **Columns:** Represent attributes or properties of the entity.
- Tables must have a unique name within the database.
- Data items are stored at the intersection of rows and columns, ensuring a structured organization.

Example: Consider a table called `Student`, which stores information about students:

- Columns: `Student_ID`, `Name`, `Date_of_Birth`.
- Rows: Each row corresponds to a specific student and includes values for the attributes.

1.2 Entities and Attributes

An **entity type** refers to a class of real-world objects, such as patients, movies, or invoices. Each entity type has attributes that define its properties. For example:

- **Entity Type:** `Movie`
- **Attributes:** `Title`, `Release Year`, `Director`.

1.3 Rows and Columns in Detail

Columns (Attributes):

- Each column has a name and a **domain**, which defines the allowable data types and values.
- Values in columns must be single-valued (atomic).
- The order of columns in a table is unimportant.

Rows (Records):

- Each row describes a specific instance of the entity.
- Rows must be unique and are identified using a **primary key**.
- The order of rows is unimportant.

2 Relationships

2.1 Definition

A **relationship** is an association between tables established through common columns. Relationships ensure that data in different tables is interconnected, forming the backbone of a relational database.

2.2 Types of Relationships

One-to-One:

- Each row in Table A is associated with at most one row in Table B, and vice versa.
- Example: A table of `Employees` linked to a table of `Company Cars`.
- Implementation: The primary key of one table also acts as the foreign key in the related table.

One-to-Many:

- A single row in Table A can be linked to multiple rows in Table B.
- Example: A table of `Authors` linked to a table of `Books`.
- Implementation: The primary key of Table A is included as a foreign key in Table B.

Many-to-Many:

- Rows in Table A can relate to multiple rows in Table B, and vice versa.
- Example: A table of `Students` linked to a table of `Courses`.
- Implementation: A junction table is created to store the primary keys of both tables.

3 Normalization

Normalization is the process of organizing data in a database to reduce redundancy, avoid update anomalies, and maintain consistency.

3.1 First Normal Form (1NF)

Definition: A table is in 1NF if it has no repeating groups and all its columns contain atomic (indivisible) values.

Steps to Achieve 1NF:

- Eliminate repeating groups by storing each value in its own row.
- Ensure each column contains a single value.

Example: Unnormalized Table:

Student ID	Student Name	Courses Enrolled
1	John	Math, Science

1NF Table:

Student ID	Student Name	Course
1	John	Math
1	John	Science

3.2 Second Normal Form (2NF)

Definition: A table is in 2NF if:

- It is in 1NF.
- Every non-key attribute depends on the entire primary key.

Steps to Achieve 2NF:

- Identify attributes that depend only on a part of a composite key.
- Create separate tables for these attributes.

Example: Splitting a table of `Students` and `Courses` into separate `Student` and `Course` tables.

3.3 Third Normal Form (3NF)

Definition: A table is in 3NF if:

- It is in 2NF.
- No attribute depends on a non-key attribute (transitive dependency).

Steps to Achieve 3NF:

- Identify attributes dependent on non-key attributes.
- Move these attributes into a new table.

Example: Splitting a table of `Students`, `Courses`, and `Coordinators` into separate tables.

4 Conclusion

The relational model and normalization techniques are essential for designing efficient databases. By ensuring data integrity and minimizing redundancy, these principles lay the groundwork for scalable and maintainable database systems.