Blog: What is Data Modelling and Normalization?

Author Name: Vaibhav Agrawal

Due Date: 7, July,2022

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Introduction**

Data modelling is the process of creating a simplified diagram of a software system and the data elements it contains, using text and symbols to represent the data and how it flows. Data models provide a blueprint for designing a new database or reengineering a legacy application. Overall, data modelling helps an organization use its data effectively to meet business needs for information.

**Need of Database Modelling**

Data modelling is a core data management discipline. By providing a visual representation of data sets and their business context, it helps pinpoint information needs for different business processes. It then specifies the characteristics of the data elements that will be included in applications and in the database or file system structures used to process, store and manage the data.

It plays a big role in data architecture processes that document data assets, map how data moves through IT systems and create a conceptual data management framework.

**Types of Modelling**

### **1. ER (Entity-Relationship) Model**

This model is based on the notion of real-world entities and relationships among them. It creates an entity set, relationship set, general attributes, and constraints.

Here, an entity is a real-world object; for instance, an employee is an entity in an employee database. An attribute is a property with value, and entity sets share attributes of identical value. Finally, there is the relationship between entities.

### **2. Hierarchical Model**

This data model arranges the data in the form of a tree with one root, to which other data is connected. The hierarchy begins with the root and extends like a tree. This model effectively explains several real-time relationships with a single one-to-many relationship between two different kinds of data.

### **3. Network Model**

This database model enables many-to-many relationships among the connected nodes. The data is arranged in a graph-like structure, and here ‘child’ nodes can have multiple ‘parent’ nodes. The parent nodes are known as owners, and the child nodes are called members.

### **4. Relational Model**

This popular data model example arranges the data into tables. The tables have columns and rows, each cataloguing an attribute present in the entity. It makes relationships between data points easy to identify.

For example, e-commerce websites can process purchases and track inventory using the relational model.

### **5. Object-Oriented Database Model**

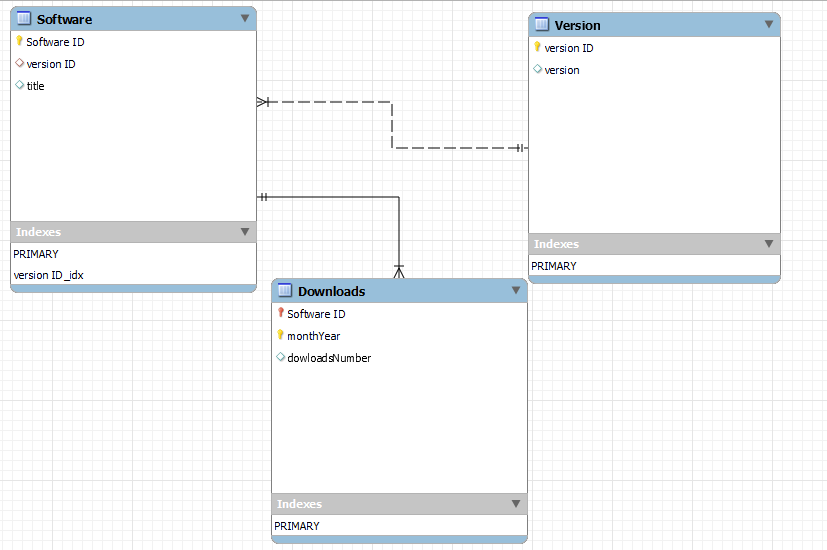
This data model defines a database as an object collection, or recyclable software components, with related methods and features.

For instance, architectural and engineering real-time systems used in 3D modelling use this data modelling process.

### **6. Object-Relational Model**

This model is a combination of an object-oriented database model and a relational database model. Therefore, it blends the advanced functionalities of the object-oriented model with the ease of the relational data model.

The data modelling process helps organizations to become more data-driven. This starts with cleaning and modelling data. Let us look at how data modelling occurs at different levels.



**Fig:** Sample Database Model given for evaluation

**Normalization**

**Normalization** is the process of minimizing **redundancy** and increase **automicity** from a relation or set of relations. Redundancy in relation may cause insertion, deletion, and update anomalies. So, it helps to minimize the redundancy in relations.

**Normal forms** are used to eliminate or reduce redundancy in database tables.

1. **First normal form - (1NF):** A relation is in 1NF if it contains an atomic value.

Example: Consider following table which is not normalized:

Employee Table:

|  |  |  |
| --- | --- | --- |
| Employee No | Employee Name | Department |
| 1 | Amit | OBIEE, ETL |
| 2 | Divya | COGNOS |
| 3 | Rama | Administrator |

**After 1NF form-**

First table: Employee Table

|  |  |
| --- | --- |
| Employee No | Employee Name |
| 1 | Amit |
| 2 | Divya |
| 3 | Rama |

Second Table: Department table

|  |  |
| --- | --- |
| Employee No | Department |
| 1 | OBIEE |
| 1 | ETL |
| 2 | COGNOS |
| 3 | Administrator |

1. **Second normal form (2NF):** A relation will be in 2NF if it is in 1NF and all non-key attributes are fully functional dependent on the primary key.

Example:

|  |  |  |  |
| --- | --- | --- | --- |
| Employee No | Department No | Employee Name | Department |
| 1 | 101 | Amit | OBIEE |
| 2 | 102 | Divya | COGNOS |
| 3 | 101 | Rama | OBIEE |

**After 2NF form-**

Table 1: Employee NO table

|  |  |  |
| --- | --- | --- |
| Employee No | Department No | Employee Name |
| 1 | 101 | Amit |
| 2 | 102 | Divya |
| 3 | 101 | Rama |

Table 2: Department table

|  |  |
| --- | --- |
| Department No | Department |
| 101 | OBIEE |
| 102 | COGNOS |

1. **Third normal form (3NF):** A relation will be in 3NF if it is in 2NF and no transition dependency exists.

**Example:**

Consider following table:

|  |  |  |  |
| --- | --- | --- | --- |
| Employee No | Salary Slip No | Employee Name | Salary |
| 1 | 0001 | Amit | 50000 |
| 2 | 0002 | Divya | 40000 |
| 3 | 0003 | Rama | 57000 |

**After 3NF form-**

Employee table:

|  |  |  |
| --- | --- | --- |
| Employee No | Salary Slip No | Employee Name |
| 1 | 0001 | Amit |
| 2 | 0002 | Divya |
| 3 | 0003 | Rama |

Salary Table:

|  |  |
| --- | --- |
| Salary Slip No | Salary |
| 0001 | 50000 |
| 0002 | 40000 |
| 0003 | 57000 |