26-05-2023

**NORMALISATION IN RELATIONAL DATABASE MANAGEMENT SYSTEMS**

**What Is normalization?**

* Normalization is the process of organizing the data in the database.
* Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate undesirable characteristics like Insertion, Update, and Deletion Anomalies.
* Normalization divides the larger table into smaller and links them using relationships.
* The normal form is used to reduce redundancy from the database table.

Normalization is an iterative process, and achieving higher normal forms ensures better database design, reduced data redundancy, improved data integrity, and increased flexibility for future modifications. However, it's important to strike a balance and not over-normalize, as excessively normalized databases can result in complex queries and performance issues. A good database design typically aims for a balance between normalization and practical considerations.

**Why do we need normalization?**

The main reason for normalizing the relations is removing these anomalies. Failure to eliminate anomalies leads to data redundancy and can cause data integrity and other problems as the database grows. Normalization consists of a series of guidelines that helps to guide you in creating a good database structure.

*Data modification anomalies can be categorized into three types:*

* Insertion Anomaly: Insertion Anomaly refers to when one cannot insert a new tuple into a relationship due to lack of data.
* Deletion Anomaly: The delete anomaly refers to the situation where the deletion of data results in the unintended loss of some other important data.
* Updation Anomaly: The update anomaly is when an update of a single data value requires multiple rows of data to be updated.

**Types of normalization in RDBMS:**

1. **First Normal Form (1NF):**

* First Normal Form ensures that each column in a table contains atomic values and eliminates repeating groups or arrays of values within a row.
* To achieve 1NF, the table must have a primary key that uniquely identifies each record, and each attribute (column) in the table should hold only atomic values.
* This means that values should not be grouped together or contain multiple values within a single attribute.

1. **Second Normal Form (2NF):**

* Second Normal Form focuses on eliminating partial dependencies. It requires that every non-key attribute in a table is functionally dependent on the entire primary key.
* In other words, if an attribute depends on only a part of the primary key, it should be moved to a separate table along with the part of the key it depends on. This ensures that each attribute is dependent on the whole primary key and not just a portion of it.

1. **Third Normal Form (3NF):**

* Third Normal Form aims to eliminate transitive dependencies.
* It states that no non-key attribute should depend on another non-key attribute within the same table.
* If such dependencies exist, they should be resolved by moving the dependent attribute(s) to a separate table.
* This ensures that each attribute directly depends only on the primary key and not on other non-key attributes.

1. **Boyce-Codd Normal Form (BCNF):**

* Boyce-Codd Normal Form addresses functional dependencies.
* It requires that for every non-trivial functional dependency (X -> Y) in a table, X (the determinant) must be a super key.
* In simpler terms, every determinant must be a candidate key, ensuring that there are no non-trivial dependencies among attributes.
* BCNF ensures that each attribute in a table is functionally dependent on the key, and there are no redundant dependencies.

1. **Fourth Normal Form (4NF):**

* Fourth Normal Form deals with multi-valued dependencies.
* It states that there should be no non-trivial multi-valued dependencies between non-key attributes.
* If such dependencies exist, they should be resolved by moving the dependent attributes to a separate table.
* 4NF ensures that each attribute is independent of other non-key attributes and eliminates redundancy related to multi-valued dependencies.

1. **Fifth Normal Form (5NF) or Project-Join Normal Form (PJNF):**

* Fifth Normal Form focuses on eliminating join dependencies.
* It ensures that all join dependencies are explicitly represented using foreign keys and primary keys, and no non-trivial join dependency can be inferred from the table's primary key.
* 5NF helps in optimizing join operations and maintaining data consistency.

1. **Domain-Key Normal Form (DK/NF):**

* Domain-Key Normal Form combines the aspects of data dependencies and the logical representation of a relation.
* It states that a table should be in 5NF, and every constraint should be expressed as a logical combination of attributes, rather than using external constraints or triggers.
* DK/NF helps in maintaining data integrity and providing a more comprehensive representation of constraints.

**Advantages of normalisation in RDBMS:**

* **Data Integrity:**

Normalization helps to maintain data integrity by reducing data redundancy and eliminating data anomalies. By organizing data into separate tables and reducing duplication, it ensures that each piece of data is stored in one place. This minimizes the chances of inconsistent or conflicting data, ensuring accuracy and reliability.

* **Efficient Data Storage:**

Normalization optimizes data storage by eliminating redundant data. It reduces the overall data footprint and improves disk space utilization. With normalized tables, data is stored only once, reducing storage requirements and improving performance for data retrieval and storage operations.

* **Data Consistency and Accuracy:**

By reducing data redundancy and eliminating anomalies, normalization enhances data consistency and accuracy. When data is stored in multiple places, inconsistencies can occur due to redundant updates or deletions. Normalization ensures that each piece of data is stored in one location, reducing the likelihood of inconsistencies and improving data quality.

* **Improved Data Retrieval:**

Normalization supports efficient data retrieval by enabling faster and more targeted queries. With well-structured tables and appropriate indexing, data retrieval becomes more efficient and less complex. Normalization helps to eliminate unnecessary joins and simplifies the query process, leading to improved query performance.

* **Flexibility and Adaptability:**

Normalized databases are more flexible and adaptable to changes in data requirements. As the database grows and evolves, normalization allows for easier modification and expansion without significant disruptions. New relationships and entities can be added, and modifications can be made with minimal impact on the overall database structure.

* **Reduced Update Anomalies:**

Normalization helps to eliminate update anomalies that can occur when data is stored redundantly. When updates are made to redundant data, inconsistencies may arise if the updates are not propagated correctly to all instances. By reducing redundancy, normalization minimizes the chances of update anomalies, ensuring that changes to data are correctly and consistently applied.

* **Simplified Database Maintenance:**

Normalized databases are generally easier to maintain. Since data is organized into separate tables with clear relationships, modifications and updates can be made more efficiently. Changes can be localized to specific tables, reducing the risk of unintended consequences and making maintenance tasks more manageable.

* **Scalability and Performance:**

Normalization supports scalability and performance optimization. As the database size and complexity increase, normalization allows for better management of data growth. It facilitates efficient indexing, partitioning, and distribution strategies, which can enhance performance and scalability as the database expands.