Normalization in Database Design

Introduction

Normalization is a **systematic process** in database design used to minimize **data redundancy** and **eliminate anomalies** by organizing data efficiently across multiple related tables. It follows a set of rules known as **Normal Forms (1NF, 2NF, 3NF, BCNF, etc.)** to ensure a well-structured database.

Purpose of Normalization

1. Eliminating Data Redundancy

- Avoids storing duplicate data across multiple tables.
- Saves storage space and ensures consistency.

2. Improving Data Integrity and Consistency

- Ensures that changes (insert, update, delete) do not cause inconsistencies.
- Helps in maintaining data accuracy and reliability.

3. Enhancing Data Organization and Efficiency

- Large tables are divided into smaller, logically related tables.
- Simplifies database management and improves structure.

4. Preventing Anomalies

Normalization eliminates the following anomalies:

- **Insertion Anomaly** – Prevents unnecessary restrictions while adding new data.
- **Update Anomaly** – Ensures consistency by avoiding partial updates.
- **Deletion Anomaly** – Prevents loss of important related information when deleting records.

5. Improving Query Performance

- Reduces redundant data retrieval, making queries more efficient.
- Relationships between tables help maintain data integrity and optimize searches.

Normalization Process (Normal Forms)

Normalization is carried out in multiple stages, called **Normal Forms (NF):**

1NF (First Normal Form)

- Ensures **atomicity** (each column contains indivisible values).
- Eliminates **repeating groups** within a table.

2NF (Second Normal Form)

- Table must be in **1NF**.
- Ensures **no partial dependency** (all non-key attributes must depend on the entire primary key).

3NF (Third Normal Form)

- Table must be in **2NF**.
- Eliminates **transitive dependencies** (non-key attributes should depend only on the primary key).

BCNF (Boyce-Codd Normal Form)

- Stronger version of **3NF** where every determinant is a **candidate key**.
- Removes dependencies where a **non-prime attribute determines another non-prime attribute**.

Example of Normalization

Unnormalized Table (UNF)

OrderID	CustomerName	Product	Quantity	Supplier
101	John Doe	Laptop	1	Dell
101	John Doe	Mouse	2	Logitech
102	Alice Smith	Keyboard	1	HP

Issues:

- Redundant CustomerName for the same OrderID.
- Multiple products in a single order lead to redundancy.

1NF – Remove Repeating Groups

OrderID	CustomerName	Product	Quantity	Supplier
101	John Doe	Laptop	1	Dell
101	John Doe	Mouse	2	Logitech
102	Alice Smith	Keyboard	1	HP

2NF – Remove Partial Dependencies

Orders Table:

OrderID	CustomerName
101	John Doe
102	Alice Smith

OrderDetails Table:

OrderID	Product	Quantity	Supplier
101	Laptop	1	Dell
101	Mouse	2	Logitech
102	Keyboard	1	HP

3NF – Remove Transitive Dependencies

Customers Table:

CustomerID	CustomerName
1	John Doe
2	Alice Smith

Orders Table:

OrderID	CustomerID
101	1
102	2

OrderDetails Table:

OrderID	Product	Quantity	Supplier
101	Laptop	1	Dell
101	Mouse	2	Logitech
102	Keyboard	1	HP

Advantages of Normalization

✓ Reduces **data redundancy**
✓ Improves **data consistency and integrity**
✓ Prevents **insertion, update, and deletion anomalies**
✓ Optimizes **storage and query performance**