**Data Normalization (1NF, 2NF, 3NF)**

**Introduction to Data Normalization**

Data normalization is the process of organizing the fields and tables of a relational database to minimize redundancy and dependency. The goal is to ensure that the database structure is efficient, reduces anomalies, and improves data integrity. This is achieved by following specific rules and stages, which are called **Normal Forms**.

Normalization typically involves breaking down large tables into smaller ones, eliminating repetitive data, and ensuring that data dependencies are logical and systematic.

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

**Definition:**  
A table is in **1NF** if:

* It contains only atomic (indivisible) values.
* Each record (row) must have a unique identifier (primary key).
* Each column must contain unique values (no sets, lists, or arrays).

**Example of Redundant Data in 0NF (Before Normalization):**

|  |  |  |
| --- | --- | --- |
| **CustomerID** | **CustomerName** | **CustomerPhone** |
| 1 | John Doe | (555) 123-4567, (555) 765-4321 |
| 2 | Jane Smith | (555) 234-5678 |

In this case, John Doe has two phone numbers stored in the same column, which violates **1NF** because columns should contain atomic values.

**Normalized to 1NF:**

|  |  |  |
| --- | --- | --- |
| **CustomerID** | **CustomerName** | **CustomerPhone** |
| 1 | John Doe | (555) 123-4567 |
| 1 | John Doe | (555) 765-4321 |
| 2 | Jane Smith | (555) 234-5678 |

Now, each phone number is stored in a separate row, and the data is atomic.

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

**Definition:**  
A table is in **2NF** if:

* It is in **1NF**.
* There is no **partial dependency**: All non-key attributes must depend on the **entire primary key** (i.e., every non-key attribute must be fully dependent on the whole composite key).

**Example of Redundant Data in 1NF (Before Normalization):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **OrderID** | **ProductID** | **ProductName** | **Quantity** | **Price** |
| 1001 | 2001 | Laptop | 2 | 1500 |
| 1001 | 2002 | Mouse | 1 | 20 |
| 1002 | 2001 | Laptop | 1 | 1500 |

In this table, ProductName is dependent only on ProductID, not on the entire composite key (OrderID, ProductID). This violates **2NF** because ProductName is partially dependent on the composite primary key.

**Normalized to 2NF:**

**Orders Table:**

|  |  |  |
| --- | --- | --- |
| **OrderID** | **ProductID** | **Quantity** |
| 1001 | 2001 | 2 |
| 1001 | 2002 | 1 |
| 1002 | 2001 | 1 |

**Products Table:**

|  |  |  |
| --- | --- | --- |
| **ProductID** | **ProductName** | **Price** |
| 2001 | Laptop | 1500 |
| 2002 | Mouse | 20 |

Now, ProductName and Price are moved to a separate table where they depend only on ProductID, removing the partial dependency.

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

**Definition:**  
A table is in **3NF** if:

* It is in **2NF**.
* There is **no transitive dependency**: Non-key attributes should not depend on other non-key attributes.

**Example of Redundant Data in 2NF (Before Normalization):**

|  |  |  |  |
| --- | --- | --- | --- |
| **EmployeeID** | **EmployeeName** | **DepartmentID** | **DepartmentName** |
| 1001 | Alice | D01 | HR |
| 1002 | Bob | D02 | IT |

In this table, DepartmentName depends on DepartmentID, which is a non-key attribute. This creates a **transitive dependency** because DepartmentName depends indirectly on EmployeeID through DepartmentID.

**Normalized to 3NF:**

**Employees Table:**

|  |  |  |
| --- | --- | --- |
| **EmployeeID** | **EmployeeName** | **DepartmentID** |
| 1001 | Alice | D01 |
| 1002 | Bob | D02 |

**Departments Table:**

|  |  |
| --- | --- |
| **DepartmentID** | **DepartmentName** |
| D01 | HR |
| D02 | IT |

Now, DepartmentName is stored in a separate table, removing the transitive dependency.

**Summary of Redundancy Removal through Normalization:**

* **1NF** ensures that data is atomic, preventing repeated values in a single column.
* **2NF** removes partial dependencies, ensuring that all non-key attributes are fully dependent on the whole primary key.
* **3NF** eliminates transitive dependencies, ensuring that non-key attributes depend only on the primary key.

Each step of normalization progressively removes redundancy, improving the efficiency and integrity of the database by minimizing anomalies and improving update, insert, and delete operations.

**Conclusion**

Normalization is a fundamental concept in relational database design, ensuring that data is stored efficiently and consistently. By applying 1NF, 2NF, and 3NF, redundant data is eliminated, which results in easier maintenance and more accurate querying of the database.