**Database normalization** is a process used to organize a database into tables and columns to reduce data redundancy and improve data integrity. The process involves applying a series of rules known as normal forms. The first four normal forms (1NF, 2NF, 3NF, and BCNF) are commonly used. Here's a brief overview of each:

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

**Objective:** Ensure that the table is a valid relational table.

* **Rules:**
  + Each table cell should contain a single value (no repeating groups or arrays).
  + Each record needs to be unique.

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

**Objective:** Remove partial dependencies; that is, ensure that non-key attributes are fully dependent on the primary key.

* **Rules:**
  + The table must already be in 1NF.
  + All non-key attributes must be fully functionally dependent on the entire primary key (not just part of it). This means that there should be no partial dependency of any column on the primary key if the primary key is composite.

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

**Objective:** Eliminate transitive dependencies.

* **Rules:**
  + The table must already be in 2NF.
  + There should be no transitive dependency for non-prime attributes. This means that non-key attributes should not depend on other non-key attributes. Each non-key attribute must depend only on the primary key.

**4. Boyce-Codd Normal Form (BCNF)**

**Objective:** Address certain types of anomalies that 3NF does not cover.

* **Rules:**
  + The table must already be in 3NF.
  + For any dependency A → B, A should be a super key. This means that the left-hand side of every dependency must be a super key.

Each of these normal forms builds upon the previous one, progressively reducing redundancy and potential for anomalies in the database.

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

**Objective:** Ensure each table cell has a single value and each record is unique.

**Example:** Imagine a table of students and their favorite subjects:

| **StudentID** | **Name** | **FavoriteSubjects** |
| --- | --- | --- |
| 1 | Alice | Math, Science |
| 2 | Bob | English, History, Science |

**Problem:** The "FavoriteSubjects" column contains multiple values.

**Solution:** Split the subjects into separate rows:

| **StudentID** | **Name** | **FavoriteSubject** |
| --- | --- | --- |
| 1 | Alice | Math |
| 1 | Alice | Science |
| 2 | Bob | English |
| 2 | Bob | History |
| 2 | Bob | Science |

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

**Objective:** Ensure all non-key columns are fully dependent on the entire primary key.

**Example:** Consider a table of course enrollments:

| **StudentID** | **CourseID** | **StudentName** | **CourseName** |
| --- | --- | --- | --- |
| 1 | 101 | Alice | Math |
| 2 | 102 | Bob | English |

**Problem:** "StudentName" and "CourseName" depend only on "StudentID" and "CourseID" respectively, not on the combination of both.

**Solution:** Split the table into two tables:

**Students:**

| **StudentID** | **StudentName** |
| --- | --- |
| 1 | Alice |
| 2 | Bob |

**Courses:**

| **CourseID** | **CourseName** |
| --- | --- |
| 101 | Math |
| 102 | English |

**Enrollments:**

| **StudentID** | **CourseID** |
| --- | --- |
| 1 | 101 |
| 2 | 102 |

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

**Objective:** Eliminate transitive dependencies.

**Example:** Consider a table of employees:

| **EmployeeID** | **EmployeeName** | **Department** | **DepartmentHead** |
| --- | --- | --- | --- |
| 1 | John | IT | Alice |
| 2 | Sarah | HR | Bob |

**Problem:** "DepartmentHead" depends on "Department," which depends on "EmployeeID."

**Solution:** Split the table into two tables:

**Employees:**

| **EmployeeID** | **EmployeeName** | **Department** |
| --- | --- | --- |
| 1 | John | IT |
| 2 | Sarah | HR |

**Departments:**

| **Department** | **DepartmentHead** |
| --- | --- |
| IT | Alice |
| HR | Bob |

**4. Boyce-Codd Normal Form (BCNF)**

**Objective:** Handle cases where 3NF is not strict enough.

**Example:** Consider a table of students and their advisors:

| **StudentID** | **AdvisorID** | **AdvisorName** |
| --- | --- | --- |
| 1 | 101 | Dr. Smith |
| 2 | 102 | Dr. Jones |
| 3 | 101 | Dr. Smith |

**Problem:** If "AdvisorID" determines "AdvisorName," then "AdvisorID" should be a super key, but "StudentID" is not a super key.

**Solution:** Ensure that each non-trivial functional dependency has a super key:

**Students:**

| **StudentID** | **AdvisorID** |
| --- | --- |
| 1 | 101 |
| 2 | 102 |
| 3 | 101 |

**Advisors:**

| **AdvisorID** | **AdvisorName** |
| --- | --- |
| 101 | Dr. Smith |
| 102 | Dr. Jones |

In this new structure, each non-trivial functional dependency now has a super key, satisfying BCNF.

**In SQL (Structured Query Language),** commands are categorized into different types based on their functionality. Here are the definitions and examples of Data Control Language (DCL), Data Definition Language (DDL), Data Manipulation Language (DML), and Data Query Language (DQL) statements:

**1. Data Control Language (DCL)**

**Definition**: DCL statements are used to control access to data in a database. They grant or revoke permissions to users.

**Examples**:

* **GRANT**: Gives a user permission to perform certain tasks.

sql

Copy code

GRANT SELECT, INSERT ON Employees TO user1;

* **REVOKE**: Removes a user's permission to perform certain tasks.

sql

Copy code

REVOKE SELECT, INSERT ON Employees FROM user1;

**2. Data Definition Language (DDL)**

**Definition**: DDL statements are used to define and manage all database objects, such as tables, indexes, and schemas.

**Examples**:

* **CREATE**: Creates a new table, view, index, or other database object.

sql

Copy code

CREATE TABLE Employees (

ID INT PRIMARY KEY,

Name VARCHAR(100),

Position VARCHAR(50),

Salary DECIMAL(10, 2)

);

* **ALTER**: Modifies an existing database object.

sql

Copy code

ALTER TABLE Employees ADD COLUMN DateOfBirth DATE;

* **DROP**: Deletes a table, view, index, or other database object.

sql

Copy code

DROP TABLE Employees;

**3. Data Manipulation Language (DML)**

**Definition**: DML statements are used to manage data within schema objects. These statements are used for inserting, updating, deleting, and merging data.

**Examples**:

* **INSERT**: Adds new rows of data into a table.

sql

Copy code

INSERT INTO Employees (ID, Name, Position, Salary)

VALUES (1, 'John Doe', 'Manager', 75000.00);

* **UPDATE**: Modifies existing data within a table.

sql

Copy code

UPDATE Employees

SET Salary = 80000.00

WHERE ID = 1;

* **DELETE**: Removes rows of data from a table.

sql

Copy code

DELETE FROM Employees

WHERE ID = 1;

**4. Data Query Language (DQL)**

**Definition**: DQL statements are used to query the database to retrieve data. The primary DQL command is SELECT.

**Examples**:

* **SELECT**: Retrieves data from one or more tables.

sql

Copy code

SELECT Name, Position, Salary

FROM Employees

WHERE Salary > 50000.00;

**Summary**

* **DCL (Data Control Language)**: Controls access to data (e.g., GRANT, REVOKE).
* **DDL (Data Definition Language)**: Defines and manages database objects (e.g., CREATE, ALTER, DROP).
* **DML (Data Manipulation Language)**: Manages data within schema objects (e.g., INSERT, UPDATE, DELETE).
* **DQL (Data Query Language)**: Queries data from the database (e.g., SELECT).