Normalization

Normalization is a database design technique used to eliminate data redundancy and improve data integrity by organizing data in a structured and efficient manner. It involves breaking down a relational database into multiple related tables while adhering to specific rules. These rules are defined by a set of normal forms, each addressing different types of anomalies that can arise from improperly structured data.

Benefits of Normalization:

1. Data Integrity: Normalization minimizes data duplication and inconsistencies, ensuring accurate and reliable information.

2. Efficiency: Smaller, more focused tables improve query performance and reduce storage requirements.

3. Flexibility: Normalized structures allow for easier data manipulation, updates, and maintenance.

Normal Forms:

1. First Normal Form (1NF):

- Each table cell should hold a single atomic value.

- Example: Consider a "Customers" table with a column "Phone Numbers" containing multiple phone numbers. To achieve 1NF, create a separate "Phone Numbers" table with a customer ID and phone number.

2. Second Normal Form (2NF):

- Meets 1NF requirements.

- No partial dependencies: Non-key attributes depend on the entire primary key, not just part of it.

- Example: In a "Sales" table with "Order ID," "Product ID," and "Quantity," where "Product ID" depends only on part of the primary key, create a separate "Products" table.

3. Third Normal Form (3NF):

- Meets 2NF requirements.

- No transitive dependencies: Non-key attributes depend only on the primary key, not on other non-key attributes.

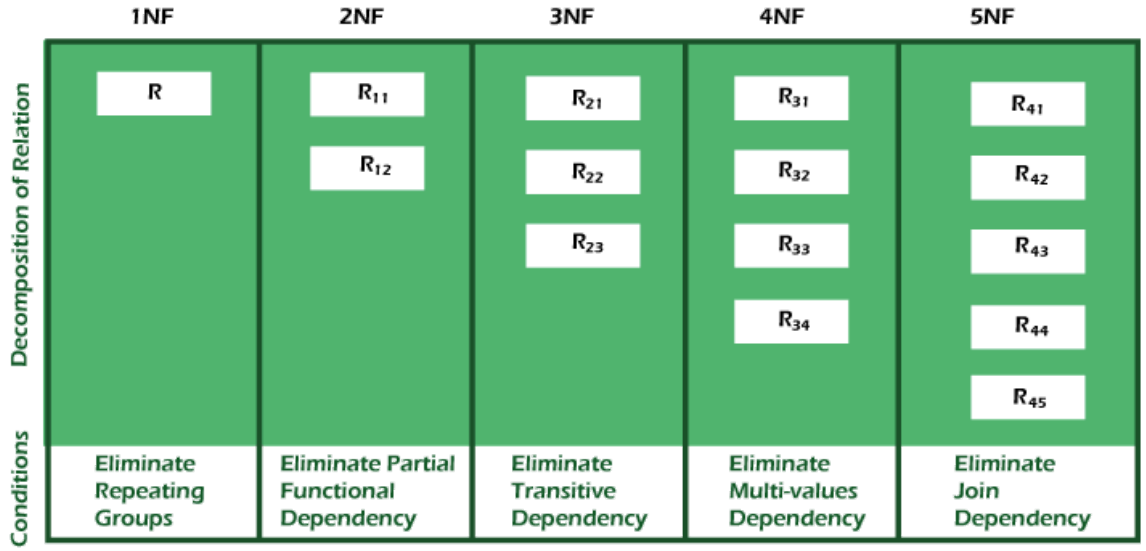
- Example: In a "Students" table with "Student ID," "Course ID," and "Instructor," where "Instructor" depends on "Course ID," move "Instructor" to a separate "Courses" table.

4. Boyce-Codd Normal Form (BCNF):

- Meets 3NF requirements.

- Every determinant (attributes that uniquely determine other attributes) must be a candidate key.

- Example: In an "Employees" table with "Employee ID," "Project ID," and "Project Manager," where "Project Manager" depends on "Project ID," separate "Project Managers" from "Projects."



Example 1:

Consider an example of a denormalized "Orders" table:

| Order ID | Customer Name | Customer Phone | Product Name | Product Category |

|----------|---------------|----------------|--------------|------------------|

| 101 | John Doe | 555-1234 | Laptop | Electronics |

| 102 | Jane Smith | 555-5678 | Smartphone | Electronics |

| 103 | John Doe | 555-1234 | Headphones | Electronics |

Applying normalization:

1. 1NF: Split customer and product information into separate tables:

- Customers: Customer ID, Customer Name, Customer Phone

- Products: Product ID, Product Name, Product Category

2. 2NF: Remove partial dependency:

- Orders: Order ID, Customer ID (foreign key), Product ID (foreign key)

3. 3NF: Remove transitive dependency:

- Orders: Order ID, Customer ID, Product ID

This normalized structure eliminates data redundancy and ensures that changes in customer or product information don't lead to inconsistencies in the Orders table.

Example 2: Library Management System

Consider a denormalized "Books" table:

Book ID	Title	Author	Genre	Library Branch
101	"The Catcher in the Rye"	J.D. Salinger	Fiction	Main Library
102	"1984"	George Orwell	Fiction	Branch A
103	"The Hobbit"	J.R.R. Tolkien	Fantasy	Main Library

Applying normalization:

1. 1NF: Separate author information and library branch information:

- Authors: Author ID, Author Name
- Library Branches: Branch ID, Branch Name

2. 2NF: Identify partial dependencies and remove them:

- Books: Book ID, Title, Author ID (foreign key), Genre

- Books_Library: Book ID (foreign key), Branch ID (foreign key)

3. 3NF: Remove transitive dependency:
   - Books: Book ID, Title, Author ID, Genre
   - Authors: Author ID, Author Name
   - Library Branches: Branch ID, Branch Name
   - Books_Library: Book ID, Branch ID

Example 3: Employee Management System

Consider a denormalized "Employees" table:

Employee ID	Full Name	Department	Manager	Salary
101	John Doe	HR	Jane Smith	$60,000
102	Jane Smith	HR	NULL	$75,000
103	Mary Johnson	IT	John Doe	$80,000

Applying normalization:

1. 1NF: Separate department and manager information:
   - Departments: Department ID, Department Name
   - Managers: Manager ID, Manager Name

2. 2NF: Eliminate partial dependencies:
   - Employees: Employee ID, Full Name, Department ID (foreign key), Salary
   - Employees_Managers: Employee ID (foreign key), Manager ID (foreign key)

3. 3NF: Remove transitive dependency:
   - Employees: Employee ID, Full Name, Department ID, Salary
   - Departments: Department ID, Department Name

- Managers: Manager ID, Manager Name
- Employees_Managers: Employee ID, Manager ID

These examples highlight the step-by-step process of normalization, starting from denormalized tables and progressively organizing the data into separate, related tables to eliminate anomalies and improve data integrity.

For more detailes refer:

DBMS Normalization: 1NF, 2NF, 3NF and BCNF with Examples - javatpoint

Here are 10 multiple-choice questions along with their answers based on normalization in a database management system (DBMS):

Question 1: What is the primary goal of database normalization?

A) To eliminate all redundancy in the database.

B) To increase data storage capacity.

C) To simplify complex queries.

D) To improve data integrity and reduce anomalies.

Answer: D) To improve data integrity and reduce anomalies.

Question 2: In which normal form is a relation if all non-prime attributes are fully functionally dependent on the primary key?

A) First Normal Form (1NF)

B) Second Normal Form (2NF)

C) Third Normal Form (3NF)

D) Boyce-Codd Normal Form (BCNF)

Answer: B) Second Normal Form (2NF)

Question 3: Which normal form ensures that every non-prime attribute is non-transitively dependent on the primary key?

A) First Normal Form (1NF)

B) Second Normal Form (2NF)

C) Third Normal Form (3NF)

D) Boyce-Codd Normal Form (BCNF)

Answer: C) Third Normal Form (3NF)

Question 4: Which of the following is a potential drawback of excessive normalization?

A) Improved query performance.

B) Increased storage requirements.

C) Simplified data maintenance.

D) Enhanced data integrity.

Answer: B) Increased storage requirements.

Question 5: Which normal form is based on functional dependency and multivalued dependency concepts?

A) First Normal Form (1NF)

B) Second Normal Form (2NF)

C) Third Normal Form (3NF)

D) Fourth Normal Form (4NF)

Answer: D) Fourth Normal Form (4NF)

Question 6: Which normal form addresses partial dependency?

A) First Normal Form (1NF)

B) Second Normal Form (2NF)

C) Third Normal Form (3NF)

D) Boyce-Codd Normal Form (BCNF)

Answer: B) Second Normal Form (2NF)

Question 7: In which normal form is a relation if it is in Second Normal Form (2NF) and has no partial dependencies?

A) First Normal Form (1NF)

B) Second Normal Form (2NF)

C) Third Normal Form (3NF)

D) Boyce-Codd Normal Form (BCNF)

Answer: D) Boyce-Codd Normal Form (BCNF)

Question 8: Which of the following normal forms allows for the presence of transitive dependencies?

A) First Normal Form (1NF)

B) Second Normal Form (2NF)

C) Third Normal Form (3NF)

D) Boyce-Codd Normal Form (BCNF)

Answer: C) Third Normal Form (3NF)

Question 9: Which normal form eliminates transitive dependencies as well as partial dependencies?

A) Third Normal Form (3NF)

B) Boyce-Codd Normal Form (BCNF)

C) Fourth Normal Form (4NF)

D) Fifth Normal Form (5NF)

Answer: B) Boyce-Codd Normal Form (BCNF)

Question 10: What is the key principle of normalization in database design?

A) Minimizing the number of tables.

B) Maximizing data redundancy.

C) Balancing data and index storage.

D) Reducing data duplication and anomalies.

Answer: D) Reducing data duplication and anomalies.