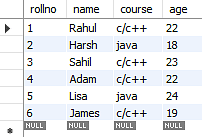
**NORMALIZATION**

**First Normal Form (1NF)**

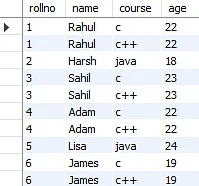
* Only atomic attributes (atomicity states that a single cell cannot hold multiple values. It must hold only a single-valued attribute.)

Example :

Table: Students



After 1NF table become :



In another method

| Roll.no | Name | Age |
| --- | --- | --- |
| 1 | Rahul | 22 |
| 2 | Harsh | 18 |
| 3 | Sahil | 23 |
| 4 | Adam | 22 |
| 5 | Lisa | 24 |
| 6 | James | 19 |

| Roll.No | Course |
| --- | --- |
| 1 | C |
| 1 | C++ |
| 2 | Java |
| 3 | C |
| 3 | C++ |
| 4 | C |
| 4 | C++ |
| 5 | Java |
| 6 | C |
| 6 | C++ |

**Second Normal Form (2NF)**

* The table has to be in 1NF.
* Every non-key attribute is fully functionally dependent on the ENTIRE primary key.
* The table should not possess partial dependency.

Partial dependency occurs when one primary key determines some other attribute/attributes.

Example :

Consider a table called "Orders" with columns: Order ID, Product ID, Product Name, and Order Quantity. Here, the primary key is (Order ID, Product ID), and Product Name depends only on Product ID, not the entire primary key. To normalize to 2NF, we would create two tables: "Orders" with columns (Order ID, Product ID, Order Quantity) and "Products" with columns (Product ID, Product Name).

| Order ID | Product ID | Product Name | Order Quantity |
| --- | --- | --- | --- |
| 1 | 100 | Laptop | 2 |
| 2 | 101 | Printer | 1 |
| 3 | 100 | Laptop | 1 |

After normalaization

Orders table

| Order ID | Product ID | Order Quantity |
| --- | --- | --- |
| 1 | 100 | 2 |
| 2 | 101 | 1 |
| 3 | 100 | 1 |

Products table

| Product ID | Product Name |
| --- | --- |
| 100 | Laptop |
| 101 | Printer |

In the original "Orders" table, the "Product Name" column depends only on the "Product ID" column and not on the entire primary key (Order ID, Product ID). To normalize this table to 2NF, we separate the dependent column "Product Name" into a separate table called "Products."

The "Orders" table now contains only the columns directly related to the orders, namely "Order ID," "Product ID," and "Order Quantity." The "Products" table stores the product information, with each row representing a unique product identified by its "Product ID."

By separating the dependent column into a separate table, we eliminate the partial dependency and adhere to the requirements of 2NF. Each table now represents a distinct entity and satisfies the normalization principles.

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

* + The table must satisfy 2NF
  + Non-key columns should depend only on the primary key, not on other non-key columns.

That is there should be no transitive dependency in table

Example:

Suppose we have a table called "student" with columns Roll.no,states and city. Here the primary key is Roll no. Roll.no determines state and state determines city. By the transitivity property roll.no determines the city also. To normalize this table to 3NF,

| Rool.no | State | City |
| --- | --- | --- |
| 1 | Punjab | Mohali |
| 2 | Haryana | Ambala |
| 3 | Punjab | Mohali |
| 4 | Haryana | Ambala |
| 5 | Bihar | Patna |

After normalization

| Rool.no | State |
| --- | --- |
| 1 | Punjab |
| 2 | Haryana |
| 3 | Punjab |
| 4 | Haryana |
| 5 | Bihar |

| State | City |
| --- | --- |
| Punjab | Mohali |
| Haryana | Ambala |
| Bihar | Patna |

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

* + The table must satisfy 3NF
  + Every determinant (a column or a set of columns determining other columns) must be a candidate key.

Example:

Consider a table called "Teachers" with columns: Teacher ID, Course ID, and Course Name. Here, both Teacher ID and Course ID determine Course Name, but Course ID is not a candidate key. Normalize to BCNF, we would create two tables: "Teachers" with columns (Teacher ID, Course ID) and "Courses" with columns (Course ID, Course Name).

| Teacher ID | Course ID | Course Name |
| --- | --- | --- |
| 1 | 101 | Math |
| 2 | 101 | Math |
| 3 | 102 | Science |

After normalization

| Teacher ID | Course ID |
| --- | --- |
| 1 | 101 |
| 2 | 101 |
| 3 | 102 |

| Course ID | Course Name |
| --- | --- |
| 101 | Math |
| 102 | Science |

In the original "Teachers" table, both the "Teacher ID" and "Course ID" determine the "Course Name," but "Course ID" is not a candidate key on its own. To normalize this table to BCNF, we separate the dependent column "Course Name" into a separate table called "Courses."

The "Teachers" table now contains only the columns directly related to the teachers, namely "Teacher ID" and "Course ID." The "Courses" table stores the course information, with each row representing a unique course identified by its "Course ID" and including the corresponding "Course Name."

By separating the dependent column into a separate table, we eliminate the partial dependency and adhere to the requirements of BCNF. Each table represents a distinct entity, and the relationships between the tables are established through the common "Course ID" column.

**The Fourth Normal Form**

4NF is an extension of the Third Normal Form (3NF) and addresses a specific type of data redundancy called multi-valued dependency. 4NF is designed to eliminate redundancy and dependency among multi-valued facts in a relational database.

In 4NF, a table must first satisfy the requirements of 3NF. Additionally, it should not have any non-trivial multivalued dependencies. A multi-valued dependency occurs when a table contains attributes that depend on a subset of the primary key but are independent of the remaining attributes

consider an example:

Suppose we have a table called "Employees" with the following columns: Employee ID, Project ID, Employee Name, and Project Responsibilities. The primary key is (Employee ID, Project ID), and the "Project Responsibilities" column stores multiple responsibilities for each employee-project combination.

| Employee ID | Project ID | Employee Name | Project Responsibilities |
| --- | --- | --- | --- |
| 1 | 100 | John | Analysis, Coding |
| 1 | 101 | John | Testing, Documentation |
| 2 | 100 | Alice | Analysis, Testing |

In this table, we have a multivalued dependency. The "Employee Name" depends on the Employee ID, but it is independent of the "Project Responsibilities'' column.

To normalize to 4NF, we would split the table into two separate tables:

Employees Table:

| Employee ID | Project ID |
| --- | --- |
| 1 | John |
| 2 | Alice |

Responsibilities Table:

| Employee ID | Project ID | Project Responsibilities |
| --- | --- | --- |
| 1 | 100 | Analysis |
| 1 | 100 | Coding |
| 1 | 101 | Testing |
| 1 | 101 | Documentation |
| 2 | 100 | Analysis |
| 2 | 100 | Testing |

In this normalized state, we have removed the redundancy in the "Employees" table and eliminated the multi-valued dependency. Each table represents a separate entity, and the relationships between the tables are established through the common "Employee ID" and "Project ID" columns.