# Normalization in DBMS

A lot of anomalies may be caused in a database because of various cause **insertion, deletion and updation operation or concurrency errors**.

There are 6 different normalisation benchmarks used which are –

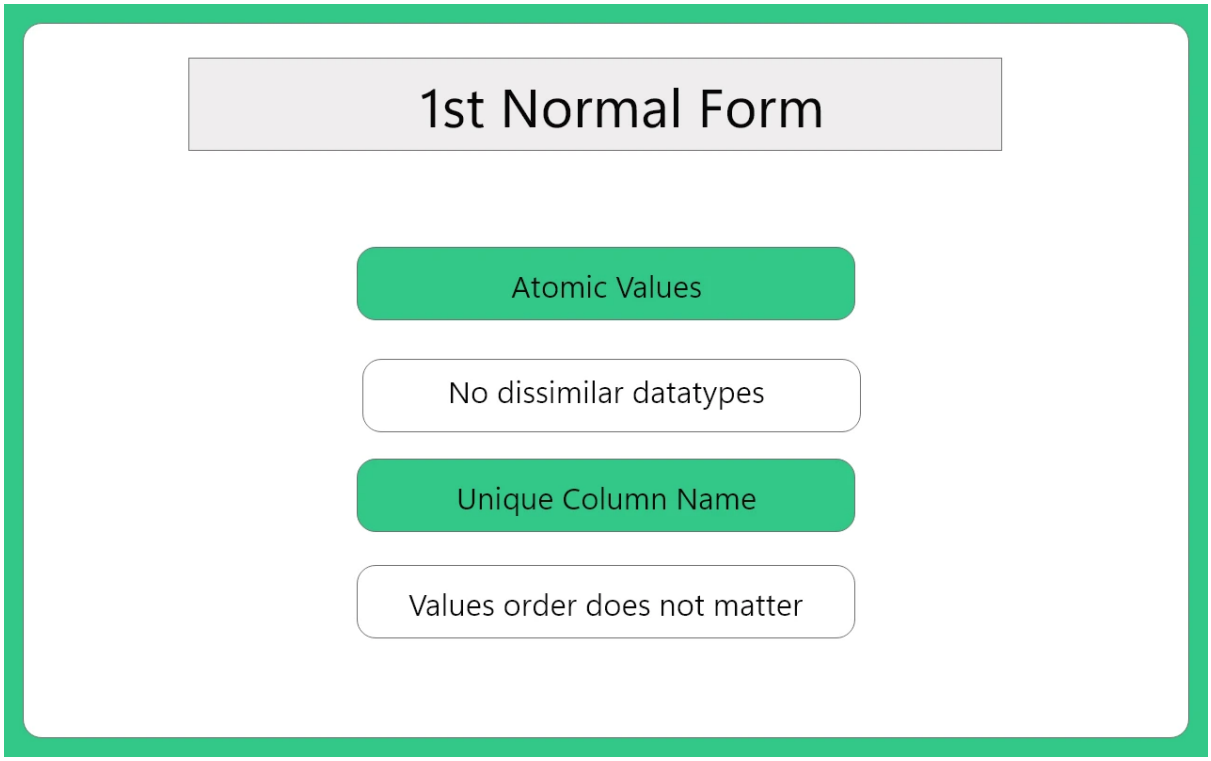
- 1 NF (normal form)
- 2 NF
- 3 NF
- BCNF (Boyce-Codd Normal Form
- 4NF
- 5NF (not used anymore)
- 6NF (not used anymore)

## 1NF

**Name –** 1 Normal Form

**Prerequisite –**

The relation **should not have any multivalued attribute** at all in it.




**How to solve –**

Decompose the table into single values attributes. Either the second phone number must be deleted or it can be decomposed into primary and secondary number.

**Not 1 NF**

| ID | FName | Phone Number |
|---|---|---|
| 1 | Bran | 915988589, 989458602 |
| 2 | Sansa | 790614709 |
| 3 | Jon | 700293958 |

**Is 1 NF**

| ID | FName | Primary No | Secondary No |
|---|---|---|---|
| 1 | Bran | 915988589 | 989458602 |
| 2 | Sansa | 790614709 | – |
| 3 | Jon | 700293958 | – |

**Is 1 NF**

| ID | FName | Phone Number |
|---|---|---|
| | | |

| | | |
|---|---|---|
| 1 | Bran | 915988589 |
| 2 | Sansa | 790614709 |
| 3 | Jon | 700293958 |

# 2NF

To understand the conditions for 2NF, one must know the following –

- What is non prime attribute
- What is an candidate Key

**Candidate Key –** A candidate key is a column, or set of columns, in a table that can uniquely identify any database record without referring to any other data. The best set of columns which identity it uniquely is chosen as candidate key.

**Non Prime Attribute –** Is the row that doesn't belong to the candidate key for the table.

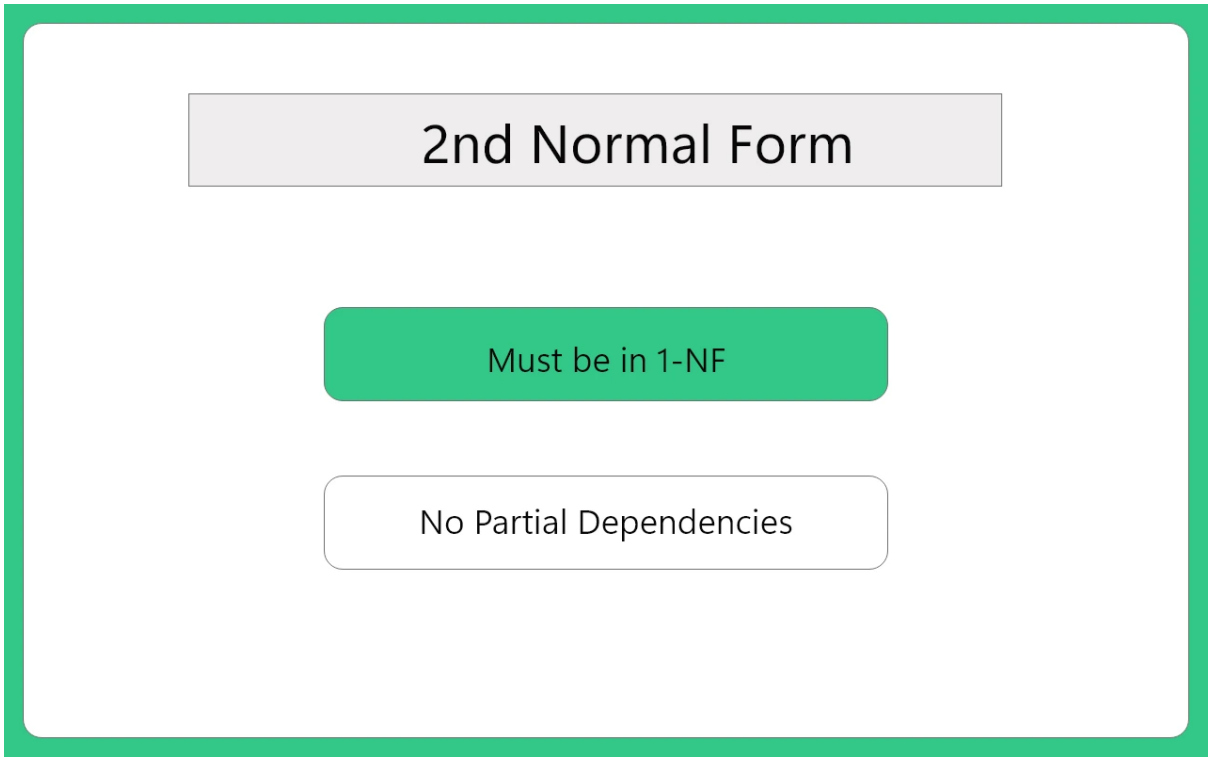
**Example –**

In the table below, if you want to identify any row of the table uniquely, we need both `{StudentID, CourseID}` thus, it becomes candidate key.

And since Course Name is not part of candidate key. It becomes non-prime attribute.

**Enrolled Courses Table**

| Student_ID | Course_ID | Course_Name |
|---|---|---|
| 1 | CSE101 | C |
| 1 | CSE102 | C++ |
| 2 | CSE101 | C |
| 2 | CSE102 | C++ |
| 3 | CSE103 | DBMS |




## What is 2NF Finally?

To be 2NF in DBMS. The relation should be –

- Already 1NF
- No Non Prime attribute should be dependent on any candidate key element, which is called not having partial dependency. This maybe is difficult to understand from definitions but easier from example.

In table, the candidate key is `{StudentID, CourseID}`,

the name of the course is obviously dependent on the CourseID right? CourseID, is non prime also. Thus there is partial dependency in it. Thus, the relation is not 2NF.

To make it 2NF we can break the table as follows –

**Enrolled Courses Table**

| ID | Student ID | Course ID |
|---|---|---|
| 1 | 1 | CSE101 |
| 2 | 1 | CSE102 |
| 3 | 2 | CSE101 |
| 4 | 2 | CSE102 |

| | | |
|---|---|---|
| 5 | 3 | CSE103 |

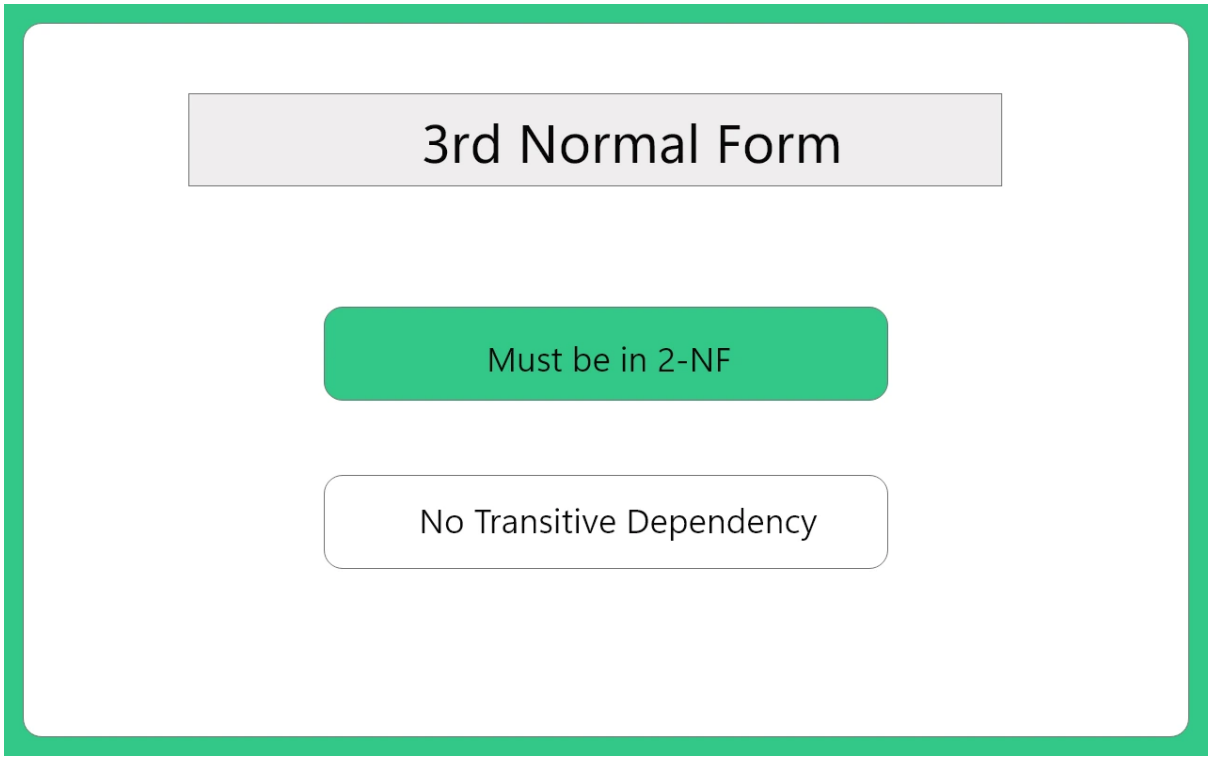
**Courses Table**

| Course ID | Name |
|---|---|
| CSE101 | C |
| CSE102 | C++ |
| CSE103 | DBMS |

# 3NF

For a relation to be in 3NF form the following must hold true –

- Should be 2NF already
- There should not be any transitive dependency for non prime attributes




## What is a transitive dependency?

When non-prime attribute depends on another non-prime attribute rather than depending upon the prime attributes or primary key

**Employee table**

| Emp_id | Name | Project-id | Date-completion |
|---|---|---|---|
| 64 | Arun | 101 | 20-4-21 |
| 65 | Sindhu | 102 | 30-4-21 |
| 66 | Trishaank | 101 | 20-4-21 |
| 67 | Vamshi | 103 | 10-2-21 |

- In the employee table, we have dependency as shownProject id -> date completion
- Project id rather than depending on the primary key of the table *employee ID* it is depending on a non-prime attribute *date completion*

The table is in First Normal Form because it does not have any multivalued attributes The table is in the second form because there is no composite and no chances of any partial dependency

**But still, the stables office from the insert, update and delete anomalies**

- Insert anomaly: Project ID cannot be assigned until an employee is assigned to it
- Update anomaly: Updating the values of date completion requires multiple updates because it is stored redundantly table
- Delete anomaly: The deletion of employee results in the deletion of entire project data even if there is even a single employee in that project the tables suffers from
- Even if the table is in second Normal Form
  it suffers from all the anomalies because of the presence of a
  transitive dependency as shown

**Note**: Table with any kind anomaly will have a transitive dependency.

## Solution steps

- Transitive dependency is converted into a separate table
- The determinant of each transitive dependency becomes a primary key attribute of the corresponding table
- Add primary key of each table resulted from each transitive dependency as the foreign key in an original table along with all remaining attributes

we have a transitive dependency *[ project id -> date completion ]*

- Break the original employee table in two tables where one table project consists of transitive dependency second table employees consistently meaning attributes
- In a newly formed projects table would
  contain project ID, date completion and determinant project ID would serve as a primary key attribute of this newly formed table

**Project table**

| Project_id | Date-Completion |
|---|---|
| 101 | 20-4-21 |
| 102 | 30-4-21 |
| 103 | 10-2-21 |

**Employee table**

| Emp_id | Name | Project-id |
|---|---|---|
| 64 | Arun | 101 |
| 65 | Sindhu | 102 |
| 66 | Trishaank | 101 |
| 67 | Vamshi | 103 |

- In the newly formed employees, table project id is derived as a foreign key attribute from the original employee table along with the remaining attributes.

**Now compare this newly formed  tables the previous problem table**

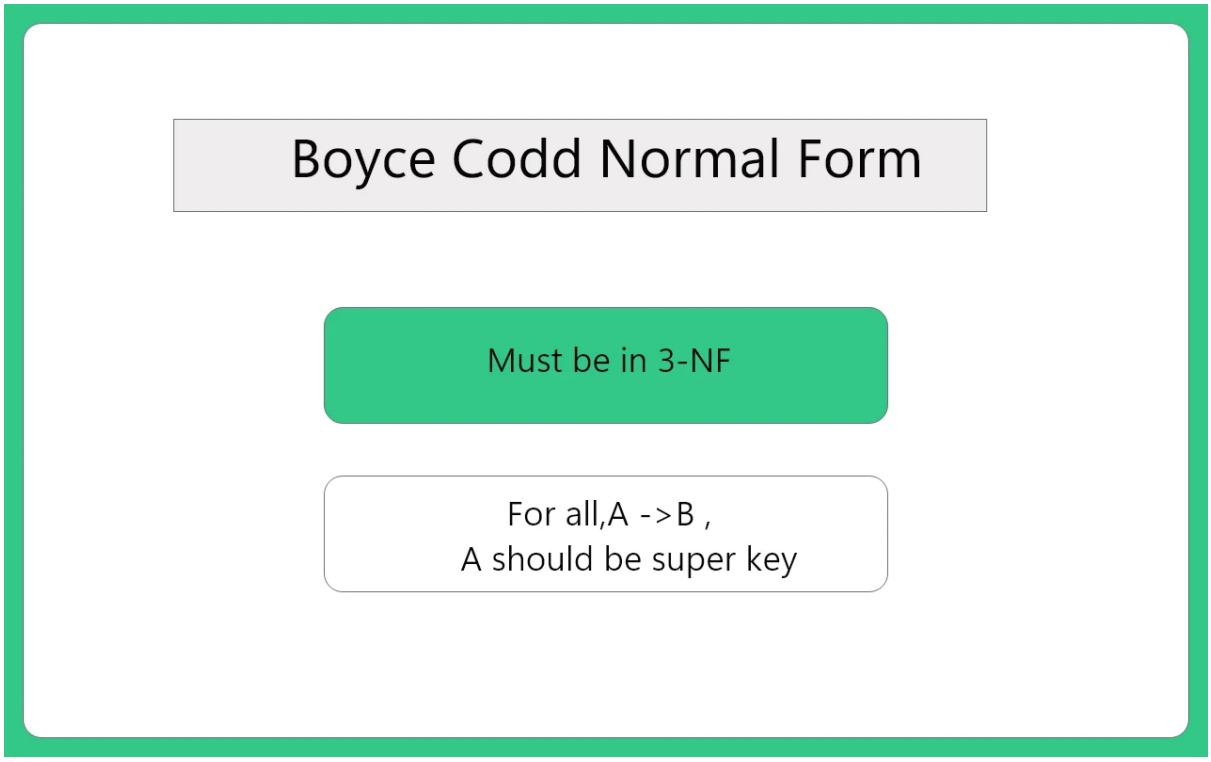
- Project ID could be added without the employee information
- Deletion of employee data would not result in the deletion of project data in the project table, updating the values of project-completion would not result in multiple updates because it is free from redundancy

# BCNF

The BCNF is the short for Boyce-Codd Normal Form the following are the conditions for a relation to be in BCNF form –

- Should be 3NF

- If every non-trivial functional dependency A –> B, A is a super key, i.e. LHS should always be a super key.




**Course Registered Table**

| Student ID | Name | Course Name | Branch ID | Course ID |
|---|---|---|---|---|
| 1 | Bran | C | CSE | 101 |
| 1 | Bran | Fluid Mechanics | MECH | 104 |
| 1 | Bran | C++ | CSE | 102 |
| 2 | Jamie | C | CSE | 101 |
| 2 | Jamie | Basic Electronics | EEE | 103 |

**Candidate Key –**

{StudentID, CourseName}

- To identify each row uniquely we need {StudentID, CourseName}
- Thus, its the candidate key

**Functional dependencies are –**

1. StudentID -> Name

2. CourseName -> {BranchID, CourseID}

Now, neither the 2nd is a dependency of type X -> Y, nor CourseName is a super key.

**Note –**

Set of attributes that can help us uniquely identify a record or tuple in the database is called super key.

**Student Table**

| Student ID | Name |
|---|---|
| 1 | Bran |
| 2 | Jamie |

**Course Table**

| Course ID | Course Name | Branch ID |
|---|---|---|
| 101 | C | CSE |
| 102 | C++ | CSE |
| 103 | Basic Electronics | EEE |
| 104 | Fluid Mechanics | MECH |

**Course Registered Table**

| ID | Student ID | Course ID |
|---|---|---|
| 1 | 1 | 101 |
| 2 | 1 | 101 |
| 3 | 1 | 102 |
| 4 | 2 | 101 |
| 5 | 2 | 101 |

# 4NF

***Normalization*** is a process of breaking a table into two smaller ones that are free from insertion and deletion and update  problems

4-NF  is the highest level of normalization.

## Definition of 4th Normal Form

The table is said to be  in 4th Normal Form it is it satisfies the following properties

- It should be in Boyce Codd normal form(BCNF)It is free from multivalued dependency.

## What is a multivalued dependency

A Dependency A->B for a single value of  A  ***there exist multiple values of B,*** then search dependencies are called as a multivalued dependency

**Problem table:**

**STUDENT**

| STU_ID | Course | Interest |
|---|---|---|
| 66_3sk | Programming | Drums |
| 66_3sk | Math | Ayurveda |
| 73_prash | Data Mining | Drums |
| 79_sanju | WPS | Sports |
| 82_srinu | Math | Aerobics |

- Given table is in First Normal Form because there is no multivalued attribute, it is in second normal form and third Normal Form because it is free from all kinds of partial dependency, full dependency and transitive dependency and it is in BCNF because in all the dependencies left side attribute is a super key
- Here student id 66 appears two times in courses and interests column and it leads to unnecessary repetition of data.

## Procedure to Convert a table to 4-NF

Break the given tables into two different tables ensure that on multiple values are not mapped for a single attribute

**STUDENT_COURSE**

| STU_ID | COURSE |
|---|---|
| 66_3sk | Programming |
| 66_3sk | Math |
| 73_prash | Data Mining |
| 79_sanju | WPS |
| | |

| | |
|---|---|
| 82_srinu | Math |

**STUDENT_INTEREST**

| STU_ID | Interest |
|---|---|
| 66_3sk | Drums |
| 66_3sk | Ayurveda |
| 73_prash | Drums |
| 79_sanju | Sports |
| 82_srinu | Aerobics |