



---

## Software Development

[Tutorials](#)   [Articles](#)   [Free Practice Tests](#)   [On-demand Webinars](#)

[Home](#)   [Resources](#)   [Software Development](#)   [SQL Tutorial for Beginners](#)   [What is Normalization in SQL?](#)  
1NF, 2NF, 3NF and BCNF in DBMS



# What is Normalization in SQL? 1NF, 2NF, 3NF and BCNF in DBMS

Lesson 6 of 6

By Ravikiran A S

Last updated on Nov 18, 2022

283131

[Previous](#)

---

## Tutorial Playlist

---

[1st Normal Form \(1NF\)](#)

[Second Normal Form \(2NF\)](#)

[Third Normal Form \(3NF\)](#)

[Boyce CoddNormal Form \(BCNF\)](#)

[View More](#)

As an SQL Developer, you often work with enormous amounts of data stored in different tables that are present inside multiple [databases](#). It often becomes strenuous to extract the data if it is not organized correctly. Using Normalization, you can solve the problem of data redundancy and organize the data using different forms. This tutorial will help you get to know the concept of Normalization in SQL.

In this tutorial, you will learn the following topics:



## What is Normalization in SQL?

© 2009 -2023- Simplilearn Solutions

- First Normal Form

Disclaimer

- Second Normal Form

PMP, PMI, PMBOK, CAPM, PgMP, PfMP, ACP, PBA, RMP, SP, and OPM3 are registered marks of the Project Management Institute, Inc.

- Third Normal Form

- Boyce Codd Normal Form

## Basics to Advanced - Learn It All!

Caltech PGP Full Stack Development

EXPLORE PROGRAM

## What Is Normalization in SQL?

Normalization is the process to eliminate data redundancy and enhance data integrity in the table. Normalization also helps to organize the data in the database. It is a multi-step process that sets the data into tabular form and removes the duplicated data from the relational tables.

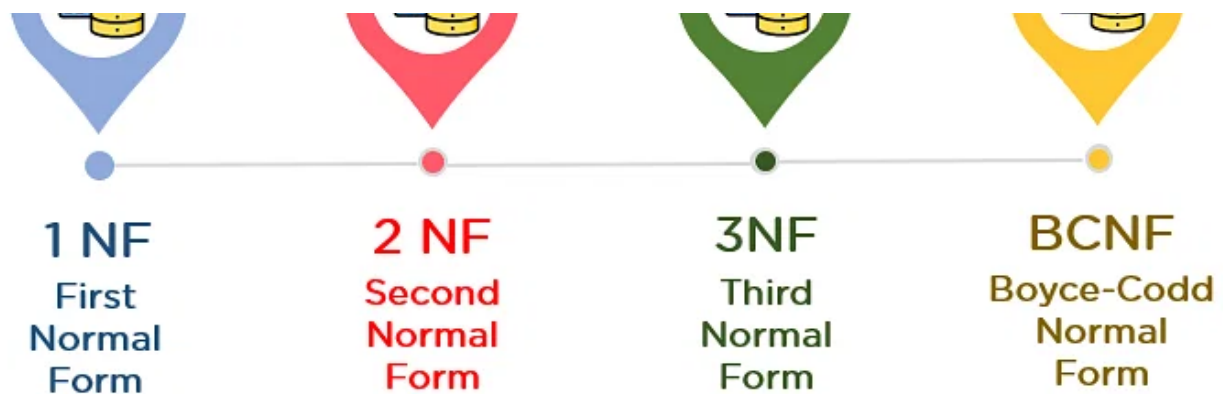
Normalization organizes the columns and tables of a database to ensure that database integrity constraints properly execute their dependencies. It is a systematic technique of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Update, and Deletion anomalies.

Also Read: [The Ultimate Guide on SQL Basics](#)

In 1970 Edgar F. Codd defined the First Normal Form.

Now let's understand the types of Normal forms with the help of examples.





## 1st Normal Form (1NF)

- A table is referred to as being in its First Normal Form if atomicity of the table is 1.
- Here, atomicity states that a single cell cannot hold multiple values. It must hold only a single-valued attribute.
- The First normal form disallows the multi-valued attribute, composite attribute, and their combinations.

Now you will understand the First Normal Form with the help of an example.

Below is a students' record table that has information about student roll number, student name, student course, and age of the student.

	rollno	name	course	age
▶	1	Rahul	c/c++	22
	2	Harsh	java	18
	3	Sahil	c/c++	23
	4	Adam	c/c++	22
	5	Lisa	java	24
	6	James	c/c++	19
*	NULL	NULL	NULL	NULL

In the studentsrecord table, you can see that the course column has two values. Thus it does not follow the First Normal Form. Now, if you use the First Normal Form to the above table, you get the below table as a result.

	rollno	name	course	age
▶	1	Rahul	c	22
	1	Rahul	c++	22
	2	Harsh	java	18
	3	Sahil	c	23
	3	Sahil	c++	23
	4	Adam	c	22

4	Adam	c++	22
5	Lisa	java	24
6	James	c	19
6	James	c++	19

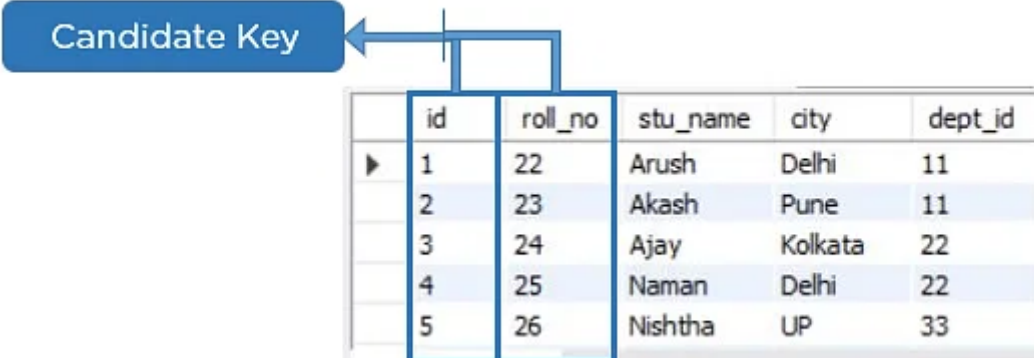
By applying the First Normal Form, you achieve atomicity, and also every column has unique values.

Before proceeding with the Second Normal Form, get familiar with Candidate Key and Super Key.

### Candidate Key

A candidate key is a set of one or more columns that can identify a record uniquely in a table, and YOU can use each candidate key as a [Primary Key](#).

Now, let's use an example to understand this better.



	id	roll_no	stu_name	city	dept_id
▶	1	22	Arush	Delhi	11
	2	23	Akash	Pune	11
	3	24	Ajay	Kolkata	22
	4	25	Naman	Delhi	22
	5	26	Nishtha	UP	33

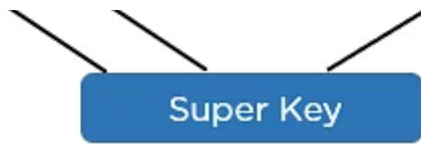
### Super Key

Super key is a set of over one key that can identify a record uniquely in a table, and the Primary Key is a subset of Super Key.

Let's understand this with the help of an example.



	id	roll_no	stu_name	enroll_no	city	dept_id
▶	1	22	Arush	223	Delhi	11
	2	23	Akash	224	Pune	11
	3	24	Ajay	225	Kolkata	22
	4	25	Naman	226	Delhi	22
	5	26	Nishtha	227	UP	33
	6	27	Lamba	228	Haryana	44



## Learn the Ins & Outs of Software Development

Caltech Coding Bootcamp

EXPLORE PROGRAM

### Second Normal Form (2NF)

The first condition for the table to be in Second Normal Form is that the table has to be in First Normal Form. **The table should not possess partial dependency.** The partial dependency here means the proper subset of the candidate key should give a non-prime attribute.

Now understand the Second Normal Form with the help of an example.

Consider the table Location:

	cust_id	storeid	store_location
▶	1	D1	Toronto
	2	D3	Miami
	3	T1	California
	4	F2	Florida
	5	H3	Texas

**The Location table possesses a composite primary key cust\_id, storeid. The non-key attribute is store\_location. In this case, store\_location only depends on storeid, which is a part of the primary key. Hence, this table does not fulfill the second normal form.**

To bring the table to Second Normal Form, you need to split the table into two parts. This will give you the below tables:

	cust_id	storeid
▶	1	D1
	2	D3
	3	T1

	4	F2
	5	H3

	storeid	store_location
►	D1	Toronto
	D3	Miami
	T1	California
	F2	Florida
	H3	Texas

As you have removed the partial functional dependency from the location table, the column store\_location entirely depends on the primary key of that table, storeid.

Now that you understood the 1st and 2nd Normal forms, you will look at the next part of this Normalization in SQL tutorial.

### Third Normal Form (3NF)

- The first condition for the table to be in Third Normal Form is that the table should be in the Second Normal Form.
- The second condition is that there should be no transitive dependency for non-prime attributes, which indicates that non-prime attributes (which are not a part of the candidate key) should not depend on other non-prime attributes in a table. Therefore, a transitive dependency is a functional dependency in which  $A \rightarrow C$  (A determines C) indirectly, because of  $A \rightarrow B$  and  $B \rightarrow C$  (where it is not the case that  $B \rightarrow A$ ).
- The third Normal Form ensures the reduction of data duplication. It is also used to achieve data integrity.

Below is a student table that has student id, student name, subject id, subject name, and address of the student as its columns.

	stu_id	name	subid	sub	address
►	1	Arun	11	SQL	Delhi
	2	Varun	12	Java	Bangalore
	3	Harsh	13	C++	Delhi
	4	Keshav	12	Java	Kochi

In the above student table, stu\_id determines subid, and subid determines sub. Therefore, stu\_id determines sub via subid. This implies that the table possesses a transitive functional dependency, and it does not fulfill the third normal form criteria.

Now to change the table to the third normal form, you need to divide the table as shown below:

	stu_id	name	subid	address
▶	1	Arun	11	Delhi
	2	Varun	12	Bangalore
	3	Harsh	13	Delhi
	4	Keshav	12	Kochi

	subid	subject
▶	11	SQL
	12	java
	13	C++
	12	Java

As you can see in both the tables, all the non-key attributes are now fully functional, dependent only on the primary key. In the first table, columns name, subid, and addresses only depend on stu\_id. In the second table, the sub only depends on subid.

## Learn How to Get Started in Blockchain

Free Webinar | 2 May, Tuesday | 9 PM IST

**REGISTER NOW**

## Boyce Codd Normal Form (BCNF)

Boyce Codd Normal Form is also known as 3.5 NF. It is the superior version of 3NF and was developed by Raymond F. Boyce and Edgar F. Codd to tackle certain types of anomalies which were not resolved with 3NF.

The first condition for the table to be in Boyce Codd Normal Form is that the table should be in the third normal form. Secondly, every Right-Hand Side (RHS) attribute of the functional dependencies should depend on the super key of that particular table.

For example :

You have a functional dependency  $X \rightarrow Y$ . In the particular functional dependency, X has to be the part of the super key of the provided table.



Consider the below subject table:

	stuid	subject	professor
▶	1	SQL	Prof. Mishra
	2	Java	Prof. Anand
	2	C++	Prof. Kanth
	3	Java	Prof. James
	4	DBMS	Prof. Lokesh

The subject table follows these conditions:

- Each student can enroll in multiple subjects.
- Multiple professors can teach a particular subject.
- For each subject, it assigns a professor to the student.

In the above table, student\_id and subject together form the primary key because using student\_id and subject; you can determine all the table columns.

Another important point to be noted here is that one professor teaches only one subject, but one subject may have two professors.

Which exhibit there is a dependency between subject and professor, i.e. subject depends on the professor's name.

to

The table is in 1st Normal form as all the column names are unique, all values are atomic, and all the values stored in a particular column are of the same domain.

The table also satisfies the 2nd Normal Form, as there is no Partial Dependency.

And, there is no Transitive Dependency; hence, the table also satisfies the 3rd Normal Form.

This table follows all the Normal forms except the Boyce Codd Normal Form.

As you can see stuid, and subject forms the primary key, which means the subject attribute is a prime attribute.

However, there exists yet another dependency - professor → subject.

BCNF does not follow in the table as a subject is a prime attribute, the professor is a non-prime

attribute.

To transform the table into the BCNF, you will divide the table into two parts. One table will hold stuid which already exists and the second table will hold a newly created column profid.

	stuid	profid
▶	1	101
	2	102
	2	103
	3	102
	4	104

And in the second table will have the columns profid, subject, and professor, which satisfies the BCNF.



With this, you have reached the conclusion of the 'Normalization in SQL' tutorial.

Learn best business analysis techniques by Purdue University, IB and EY experts. Sign-up for our [Professional Certificate Program in Business Analysis](#) TODAY!

## Here's How to Land a Top Software Developer Job

Full Stack Development-MEAN

[EXPLORE PROGRAM](#)

## Conclusion

In this tutorial, you have seen Normalization in SQL and understood the different Normal forms of Normalization. Now, you can organize the data in the database and remove the data redundancy and promote data integrity. This tutorial also helps beginners for their interview processes to understand the concept of Normalization in SQL.

If you have any questions or inputs for our editorial team regarding this "The Supreme Guide to

Normalization in SQL” tutorial, do share them in the comments section below. Our team will review them and help solve them for you very soon!

To get certified in SQL and start your career in it, check this course link: [SQL Training](#).

Happy learning!

### Find our Full Stack Java Developer Online Bootcamp in top cities:

Name	Date	Place
<a href="#">Full Stack Java Developer</a>	Cohort starts on 12th May 2023, Weekend batch	Your City
<a href="#">Full Stack Java Developer</a>	Cohort starts on 1st Jun 2023, Weekend batch	Your City

### About the Author

 [Ravikiran A S](#)

Ravikiran A S works with Simplilearn as a Research Analyst. He an enthusiastic geek always in the hunt to learn the latest technologies. He is proficient with Java Programming Language, Bi...

[View More](#)

### Recommended Resources

 <b>Implementing Stacks in Data Structures</b>	 <b>Blockchain Career Guide: A Comprehensiv...</b>	
---	---	---

