# Normalization of Database

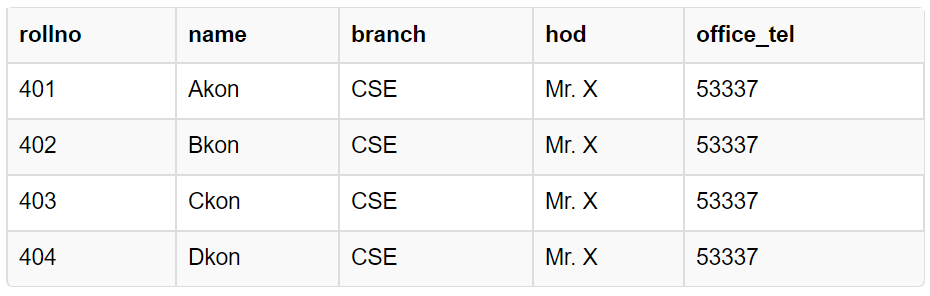
Database Normalization is a technique of organizing the data in the database.  to eliminate data redundancy(repetition) and undesirable characteristics like Insertion, Update and Deletion Anomalies.

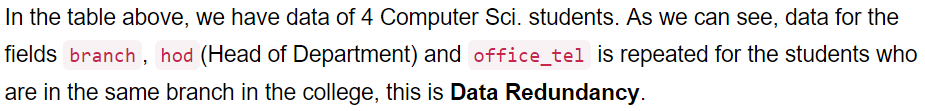
Normalization is used for mainly two purposes,

* Eliminating redundant (useless) data.
* Ensuring data dependencies make sense i.e data is logically stored.

## Problems without Normalization

If a table is not properly normalized and have data redundancy then it will not only eat up extra memory space but will also make it difficult to handle and update the database, without facing data loss. Insertion, Updation and Deletion Anamolies are very frequent if database is not normalized. To understand these anomalies let us take an example of a **Student** table.





#### Insertion Anomaly

Also, if we have to insert data of 100 students of same branch, then the branch information will be repeated for all those 100 students.

These scenarios are nothing but **Insertion anomalies**.

#### Updation Anomaly

What if Mr. X leaves the college? or is no longer the HOD of computer science department? In that case all the student records will have to be updated, and if by mistake we miss any record, it will lead to data inconsistency. This is Updation anomaly.

#### Deletion Anomaly

In our **Student** table, two different informations are kept together, Student information and Branch information. Hence, at the end of the academic year, if student records are deleted, we will also lose the branch information. This is Deletion anomaly.

## Normalization Rule

Normalization rules are divided into the following normal forms:

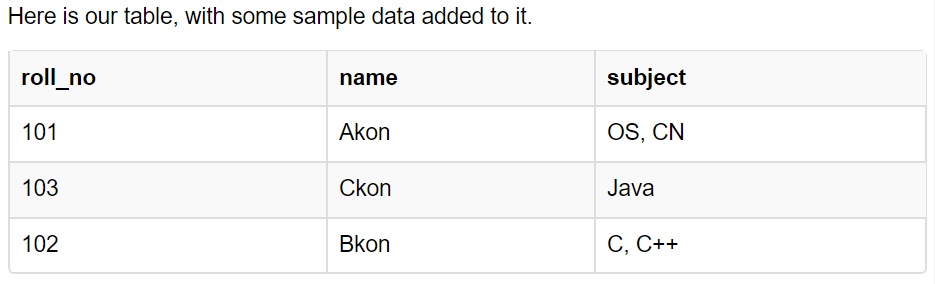
1. First Normal Form
2. Second Normal Form
3. Third Normal Form
4. BCNF
5. Fourth Normal Form

### First Normal Form (1NF)

For a table to be in the First Normal Form, it should follow the following 4 rules:

1. It should only have single(atomic) valued attributes/columns.
2. Values stored in a column should be of the same domain
3. All the columns in a table should have unique names.
4. And the order in which data is stored, does not matter.

## Example

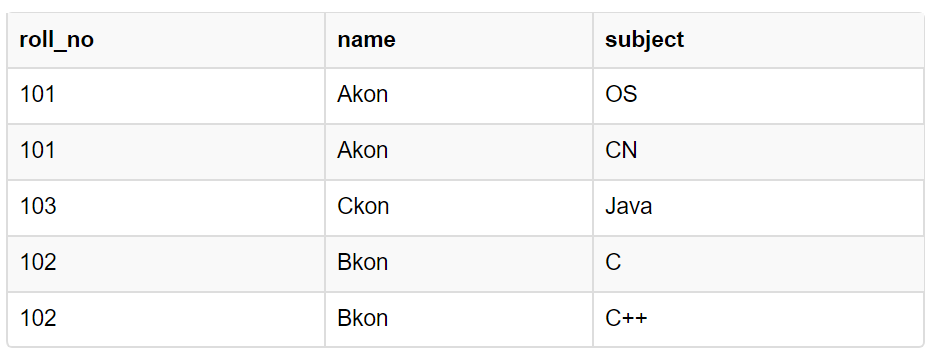


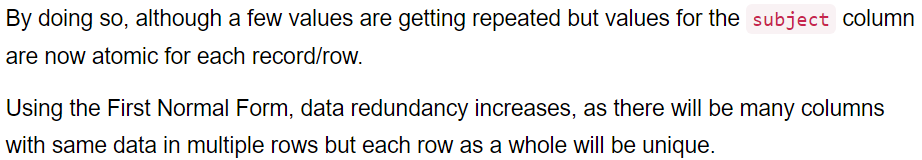
All properties are satisfied except atomic value property.

But out of the 3 different students in our table, 2 have opted for more than 1 subject. And we have stored the subject names in a single column. But as per the 1st Normal form each column must contain atomic value.

### How to solve this Problem?

It's very simple, because all we have to do is break the values into atomic values.





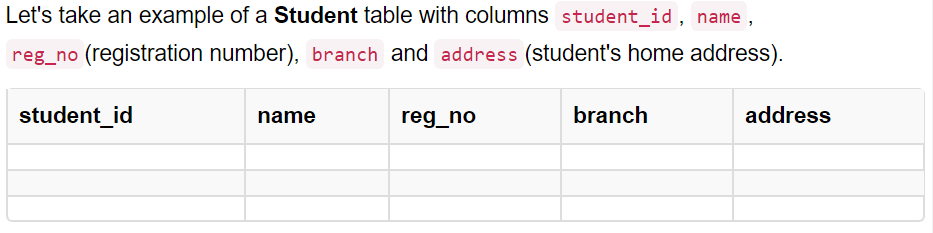
### Second Normal Form (2NF)

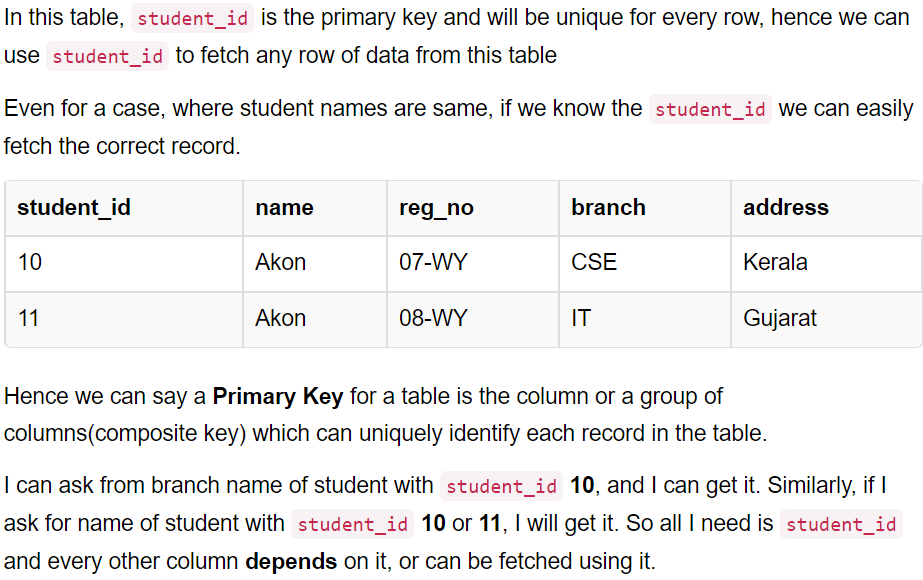
For a table to be in the Second Normal Form,

1. It should be in the First Normal form.
2. And, it should not have Partial Dependency.

What is **Partial Dependency**? Do not worry about it. First let's understand what is **Dependency** in a table?

## What is Dependency?



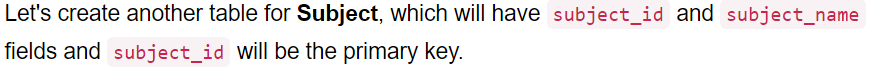


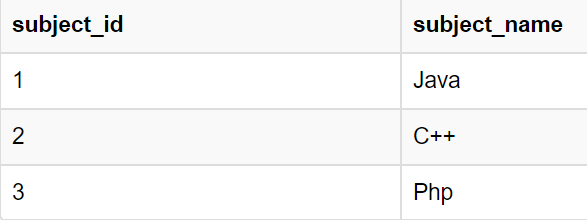
This is **Dependency** and we also call it **Functional Dependency**.

## What is Partial Dependency?

For a simple table like Student, a single column like student\_id can uniquely identfy all the records in a table.

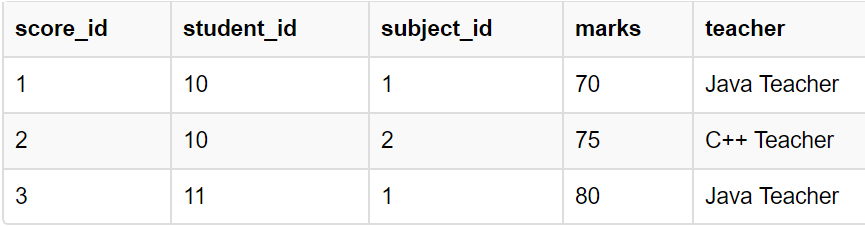
But this is not true all the time. So now let's extend our example to see if more than 1 column together can act as a primary key.



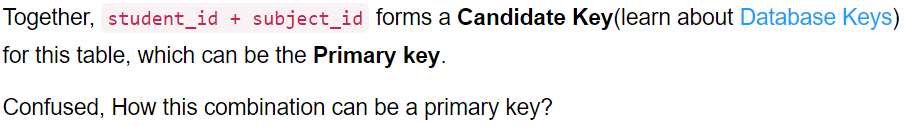


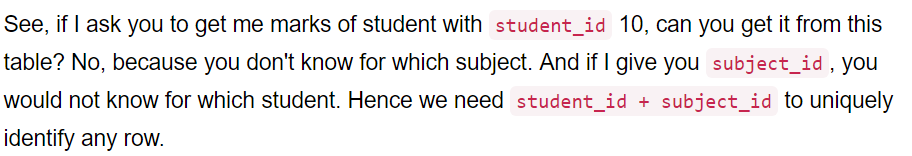
Now we have a **Student** table with student information and another table **Subject** for storing subject information.

Let's create another table **Score**, to store the **marks** obtained by students in the respective subjects. We will also be saving **name of the teacher** who teaches that subject along with marks.

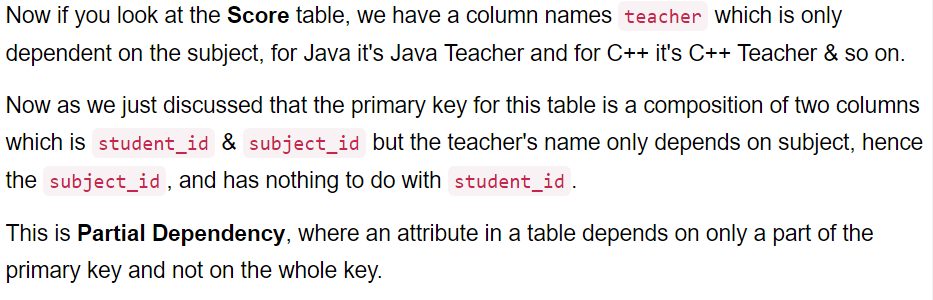


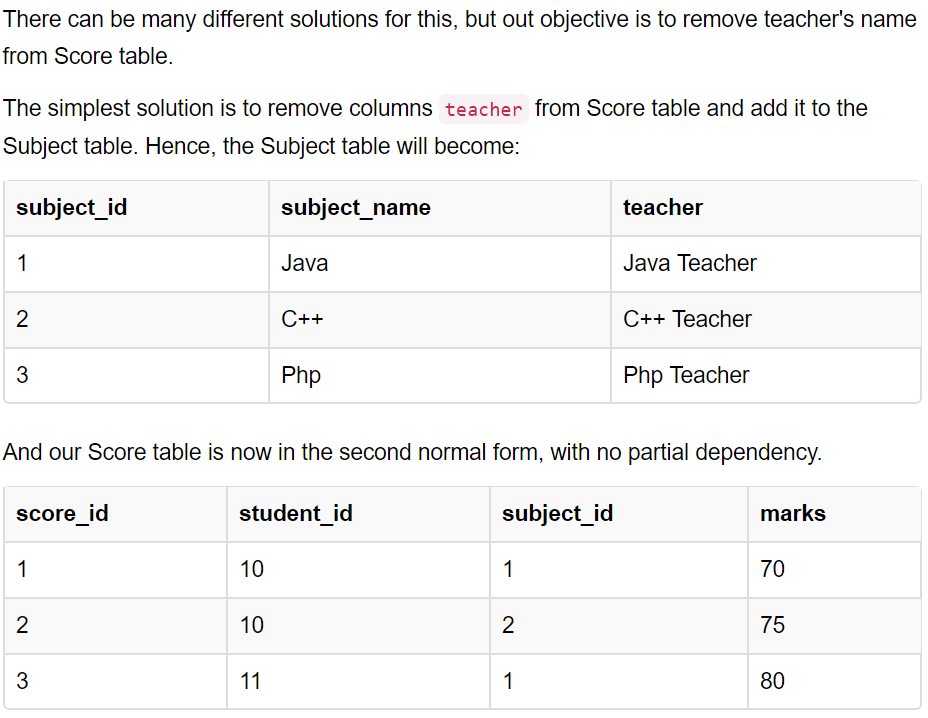
In the score table we are saving the **student\_id** to know which student's marks are these and **subject\_id** to know for which subject the marks are for.



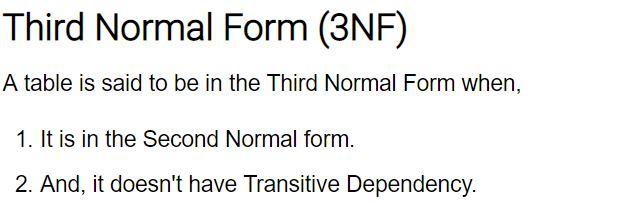


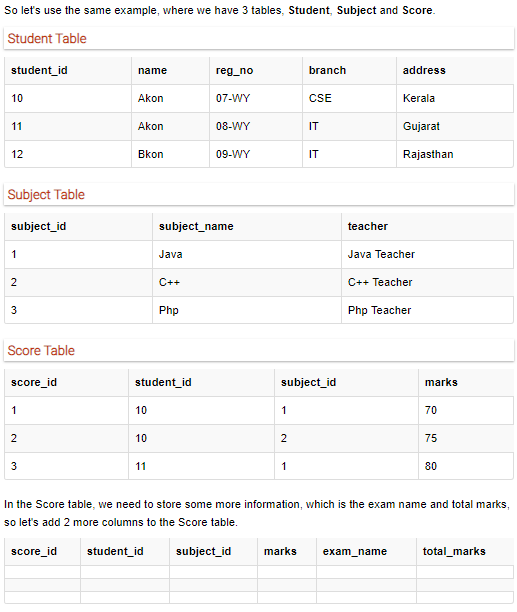
### But where is Partial Dependency?





To remove Partial dependency, we can divide the table, remove the attribute which is causing partial dependency, and move it to some other table where it fits in well.

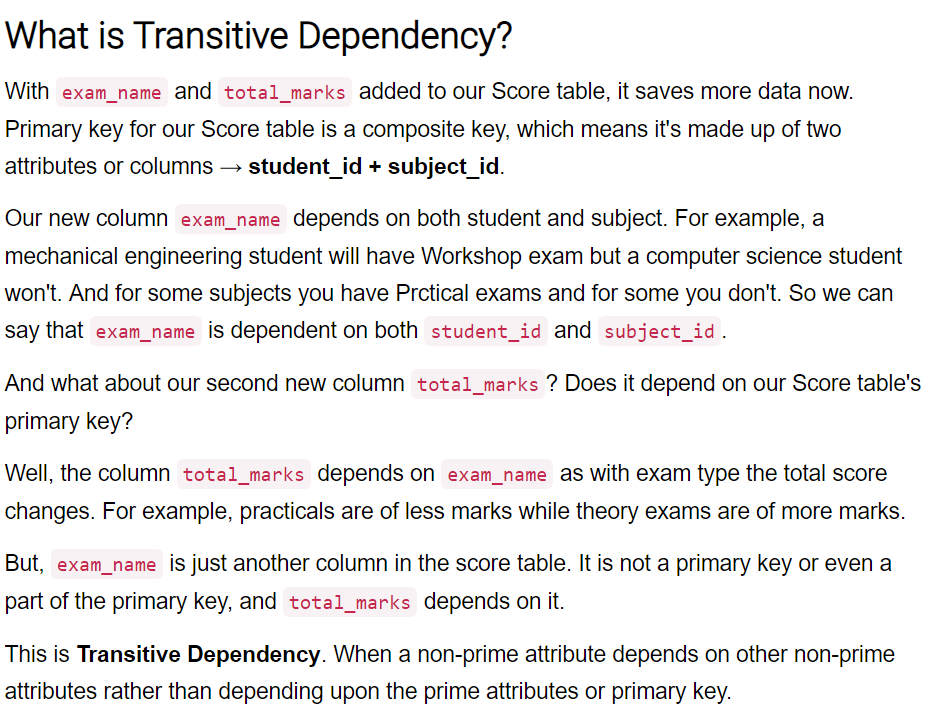


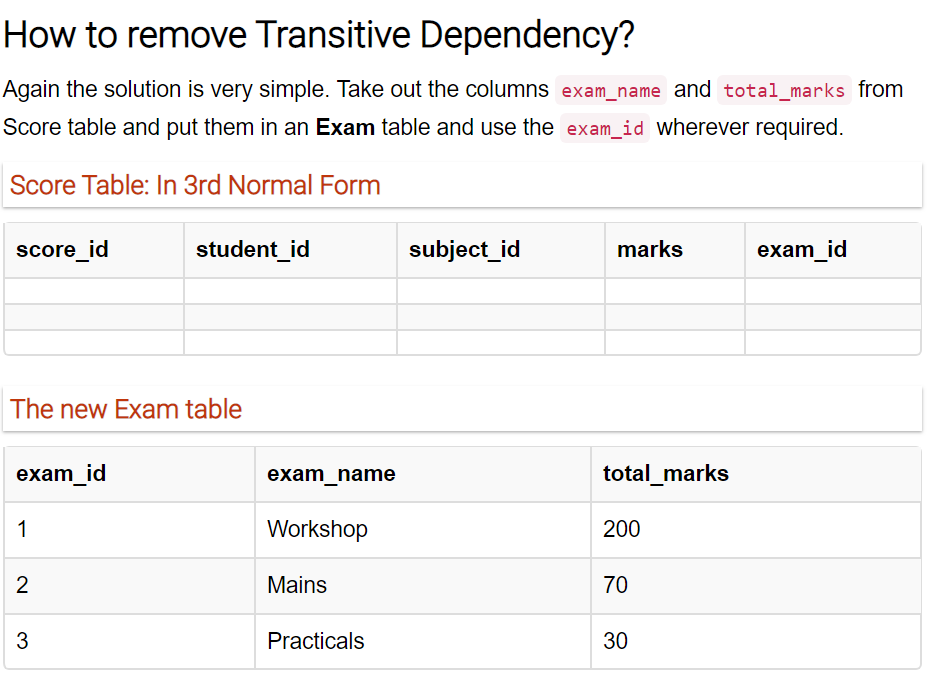


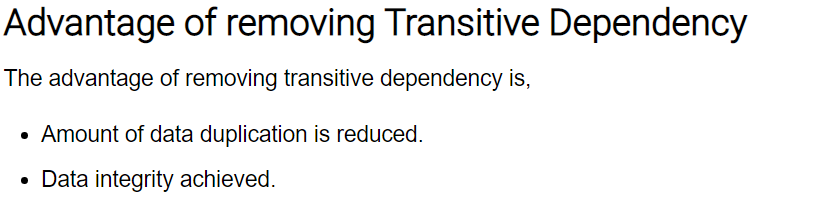
## Requirements for Third Normal Form

For a table to be in the third normal form,

1. It should be in the Second Normal form.
2. And it should not have Transitive Dependency.



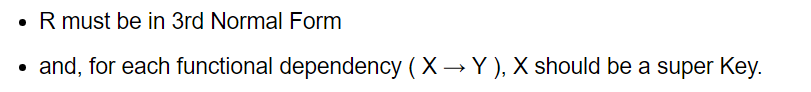


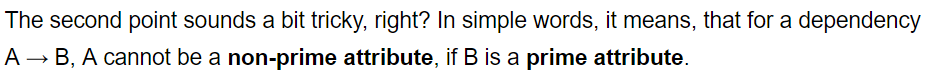


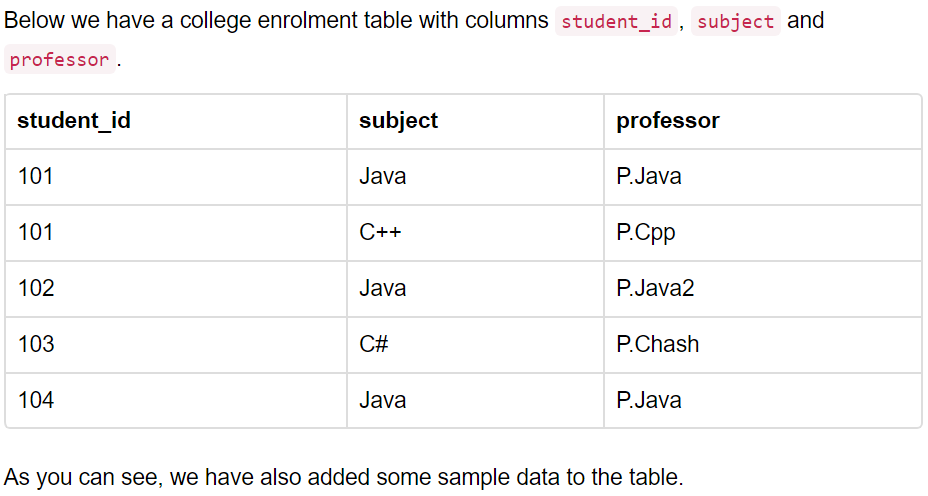


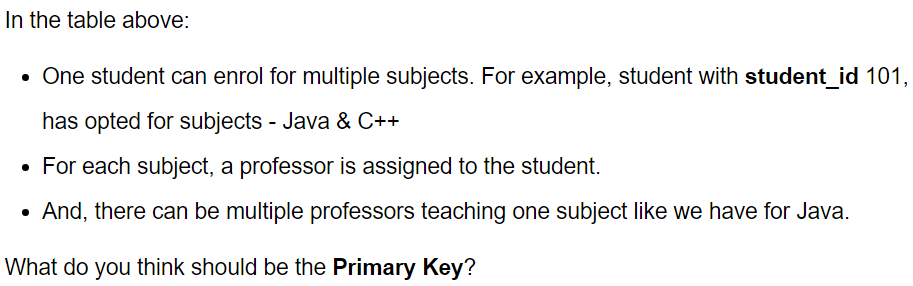
**Boyce and Codd Normal Form** is a higher version of the Third Normal form.

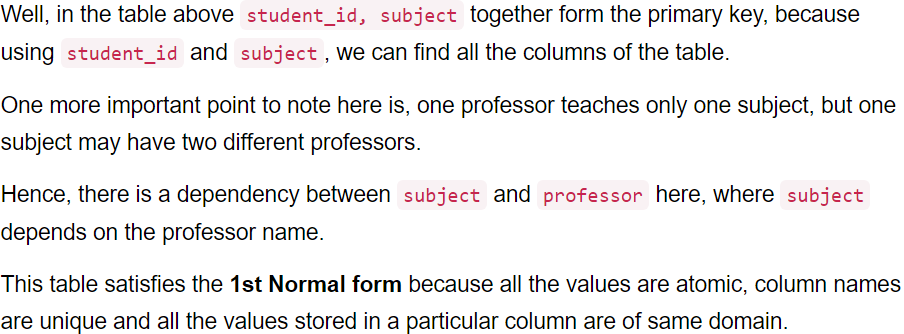
A 3NF table which does not have multiple overlapping candidate keys is said to be in BCNF. For a table to be in BCNF, following conditions must be satisfied:

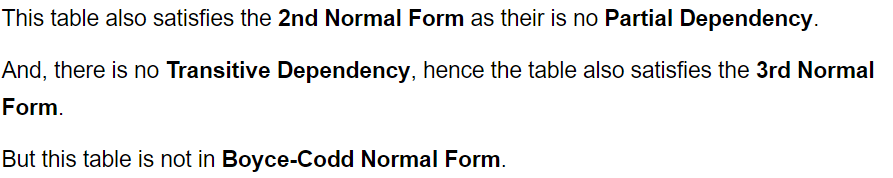


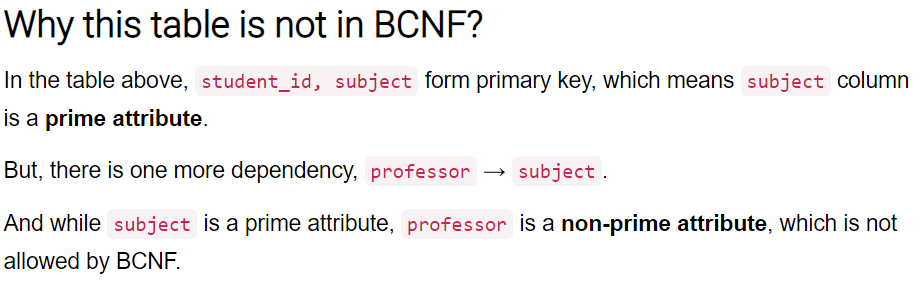












­ 