upGrad

Data Management and Relational Modelling

Course: Data Engineering - I

Lecture On: Data Management and Relational Modelling

Instructor: Vishwa Mohan

Session 4 | Data Normalization

Session Overview

Segment 02 Understanding the Database anomalies.

Segment 03 Discussion on 1NF

Segment 04 Discussion on 2NF

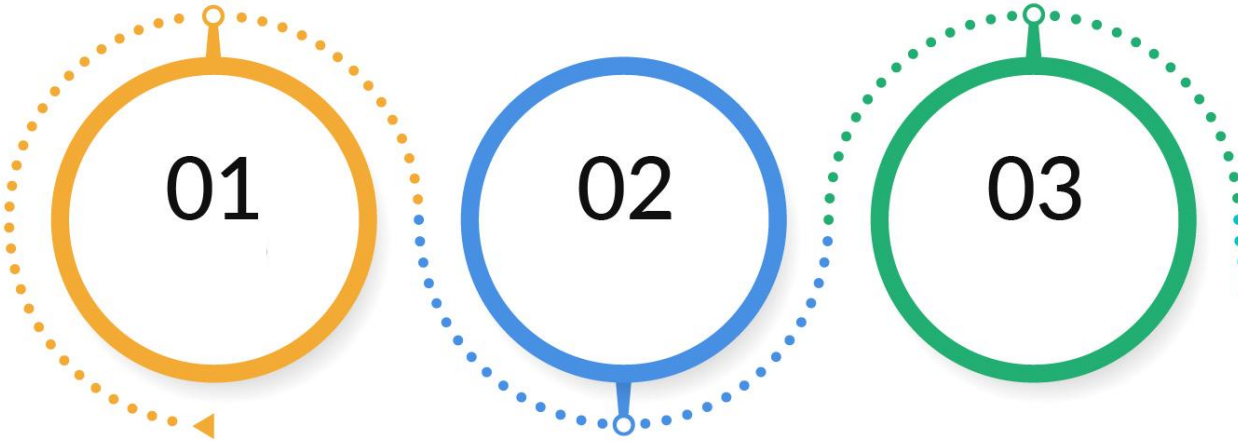
Segment 05 Discussion on 3NF

Segment 2 | Anomalies in a Database

In This Segment

Why do data anomalies occur in a database?

How do we handle such anomalies?



Updation, deletion and insertion anomalies

Why Different Anomalies Occur?

Information about three different business concepts is stored in the table given below.

For every student, the course information and the instructor information have to be consistent.

Student ID	Student Name	Student Cohort	Course ID	Course Name	Course Instructor
ID101	Sunil	C1	P101	Data Science	Mr. Ganguly
ID102	Sourav	C1	P102	Big Data Engineering	Mr. Dev
ID103	Sachin	C1	P103	Product Management	Mr. Ishant
ID104	Kapil	C1	P104	Marketing	Mr. John
ID105	Rahul	C1	P105	Digital Marketing	Mr. Irfan
ID106	Rohit	C1	P106	Data Analysis	Mr. Sachin

Updation Anomalies

Updation anomalies occur when one of the data fields is updated, which causes the previous value to be deleted.

In the first row, the course instructor is changed from Mr. Ganguly to Mr. Virat. However, the information about Mr. Virat does not exist.

Student ID	Student Name	Student Cohort	Course ID	Course Name	Course Instructor
ID101	Sunil	C1	P101	Data Science	Mr. Virat
ID102	Sourav	C1	P102	Big Data Engineering	Mr. Dev
ID103	Sachin	C1	P103	Product Management	Mr. Ishant
ID104	Kapil	C1	P104	Marketing	Mr. John
ID105	Rahul	C1	P105	Digital Marketing	Mr. Irfan
ID106	Rohit	C1	P106	Data Analysis	Mr. Sachin

Deletion Anomalies

Deletion anomalies occur when the deletion of data in one field causes the data in other fields to be deleted.

In this table, the course Product Management was deleted, which caused the other fields in the same row to be deleted. Student Information is also deleted here.

Student ID	Student Name	Student Cohort	Course ID	Course Name	Course Instructor
ID101	Sunil	C1	P101	Data Science	Mr. Ganguly
ID102	Sourav	C1	P102	Big Data Engineering	Mr. Dev
ID104	Kapil	C1	P104	Marketing	Mr. John
ID105	Rahul	C1	P105	Digital Marketing	Mr. Irfan
ID106	Rohit	C1	P106	Data Analysis	Mr. Sachin

Insertion Anomalies

Insertion anomalies occur when data about one field is added, but the details about other fields are not available.

In this table, the course Cloud Computing is added. However, the details about the student or instructor are not available.

Student ID	Student Name	Student Cohort	Course ID	Course Name	Course Instructor
ID101	Sunil	C1	P101	Data Science	Mr. Ganguly
ID102	Sourav	C1	P102	Big Data Engineering	Mr. Dev
ID104	Kapil	C1	P104	Marketing	Mr. John
ID105	Rahul	C1	P105	Digital Marketing	Mr. Irfan
ID106	Rohit	C1	P106	Data Analysis	Mr. Sachin
-	-	-	P107	Cloud Computing	-

How to Handle Such Anomalies?

01 Different attributes are separated into different tables and those tables are related using foreign keys.

02 The attributes in a table can be separated into different tables by finding the functional and transitive dependencies among the attributes.

03 The process of Data Normalisation is used to remove these anomalies in the database.

Summary | Anomalies in a Database

01 Insertion anomalies occur when data about one field is added, but the details about other fields are not available.

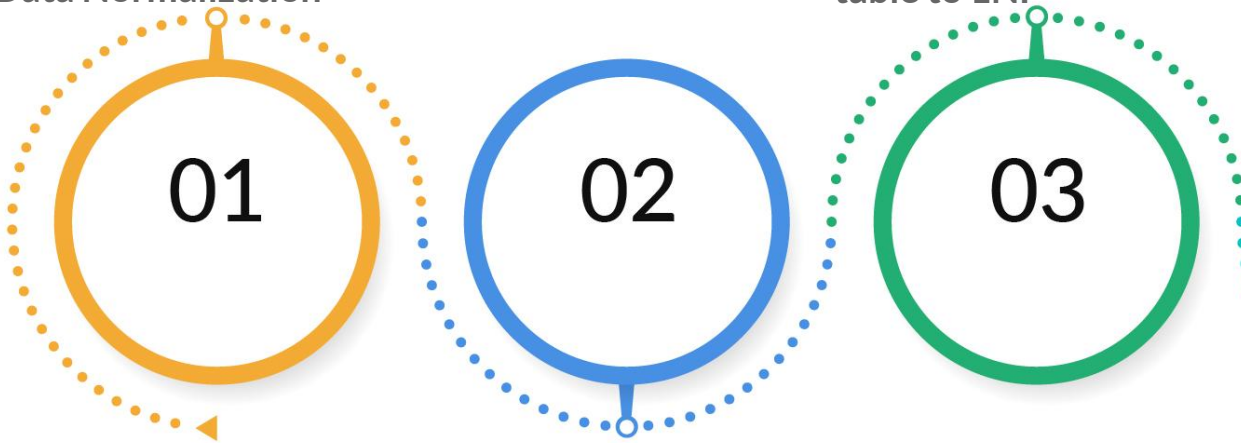
02 Deletion anomalies occur when the deletion of data in one field causes the data in the other fields of the same row to be deleted.

03 Updation anomalies occur when one of the data fields is updated, which causes the previous value to be deleted.

Segment 3 | 1st Normal Form

In This Segment

Understanding the process of Data Normalization



Converting an example table to 1NF

Understanding the criteria for 1NF

Data Normalization

DN is the process of organising data in a database such that a piece of data is not repeated at various places.

The data becomes consistent, as it is not repeated in different rows of the same table. The information about one business entity is stored in one table.

Different business entities are stored in different tables.

- The 1NF step removes multiple values from one field.
- The 2NF step removes partial functional dependencies from the table.
- The 3NF step removes transitive dependencies from the table.

Criteria for 1NF

01

For a table to be in 1NF, every field in the table must contain not more than one value. If you store multiple values in one field, separating them by commas, then the relational database (RDBMS) will consider the entire line to be one string, not a list. Hence, the application written to retrieve the data from the table will have to convert this string to a list in order to access each data element separately.

Product Type	Product Name
Clothes	Jeans, Shirts, T-Shirts, Shoes

If you want to add additional information about each product, then it will be difficult to manipulate the data.

02

For a table to be in 1NF, every table must have a primary key or a composite key to uniquely identify the table.
Unique identification of each data record in a table is important for querying the database for the right information.

Product Type	Product Name	Product Manager
Clothes	Jeans, Shirts, T-Shirts, Shoes	Virat, Rohit, Shikhar, Rahul

Consider This Table

Customer ID	Customer Name	Car Number Plate	Car Name	Date of Transaction	Owner ID	Owner Name
C12	Sachin	CarQ1234	Swift	12/01/2020	O76	Dev
		CarQ5436	Thar	18/01/2020	O78	Irfan
C46	Rahul	CarQ3421	Baleno	12/01/2020	O54	Rohit
		CarQ6534	Honda City	14/01/2020	O65	Shikhar
		CarQ3789	Swift	15/01/2020	O86	Irfan

A company borrows cars from car owners and lends them to customers for a certain duration.

As a customer can rent many cars from different owners, there can be multiple values in many fields.

Converting the Table to 1NF

Customer ID	Customer Name	Car Number Plate	Car Name	Date of Transaction	Owner ID	Owner Name
C12	Sachin	CarQ1234	Swift	12/01/2020	O76	Dev
C12	Sachin	CarQ5436	Thar	18/01/2020	O78	Irfan
C46	Rahul	CarQ3421	Baleno	12/01/2020	O54	Rohit
C46	Rahul	CarQ6534	Honda City	14/01/2020	O65	Shikhar
C46	Rahul	CarQ3789	Swift	15/01/2020	O86	Irfan

There is one row for each data record.

Every field must contain not more than one value.
The two customers' data are separated into five rows.
Every row contains a single transaction record.

Converting the Table to 1NF

Customer ID	Customer Name	Car Number Plate	Car Name	Date of Transaction	Owner ID	Owner Name
C12	Sachin	CarQ1234	Swift	12/01/2020	O76	Dev
C12	Sachin	CarQ5436	Thar	18/01/2020	O78	Irfan
C46	Rahul	CarQ3421	Baleno	12/01/2020	O54	Rohit
C46	Rahul	CarQ6534	Honda City	14/01/2020	O65	Shikhar
C46	Rahul	CarQ3789	Swift	15/01/2020	O86	Irfan

Single attribute keys:

- Customer ID: One customer can rent many cars.
- Car Number Plate: One car can be rented multiple times.
- Date of Transaction: There can be many transactions in one day.

Multiple attribute keys:

- Customer ID, Car Number Plate: One customer can rent many cars.
- Customer ID, Date of Transaction: One customer can rent two cars on the same day.
- **Car Number Plate, Date of Transaction: One car can be rented only once in a day.**

Car Number Plate and Date of Transaction together form the composite key.

Converting the Table to 1NF

Customer ID	Customer Name	Car Number Plate	Car Name	Date of Transaction	Owner ID	Owner Name
C12	Sachin	CarQ1234	Swift	12/01/2020	O76	Dev
C12	Sachin	CarQ5436	Thar	18/01/2020	O78	Irfan
C46	Rahul	CarQ3421	Baleno	12/01/2020	O54	Rohit
C46	Rahul	CarQ6534	Honda City	14/01/2020	O65	Shikhar
C46	Rahul	CarQ3789	Swift	15/01/2020	O86	Irfan

Are all anomalies handled in this 1NF table?

Is the table structure given above effective in storing data without multiple occurences of the same data?

- Customer information: All the details of a customer are repeated every time the customer makes a transaction.
- Car information: All the details of a car are repeated every time it is rented by customers.
- Owner information: All the details of an owner are repeated every time the owner's car is rented by customers.

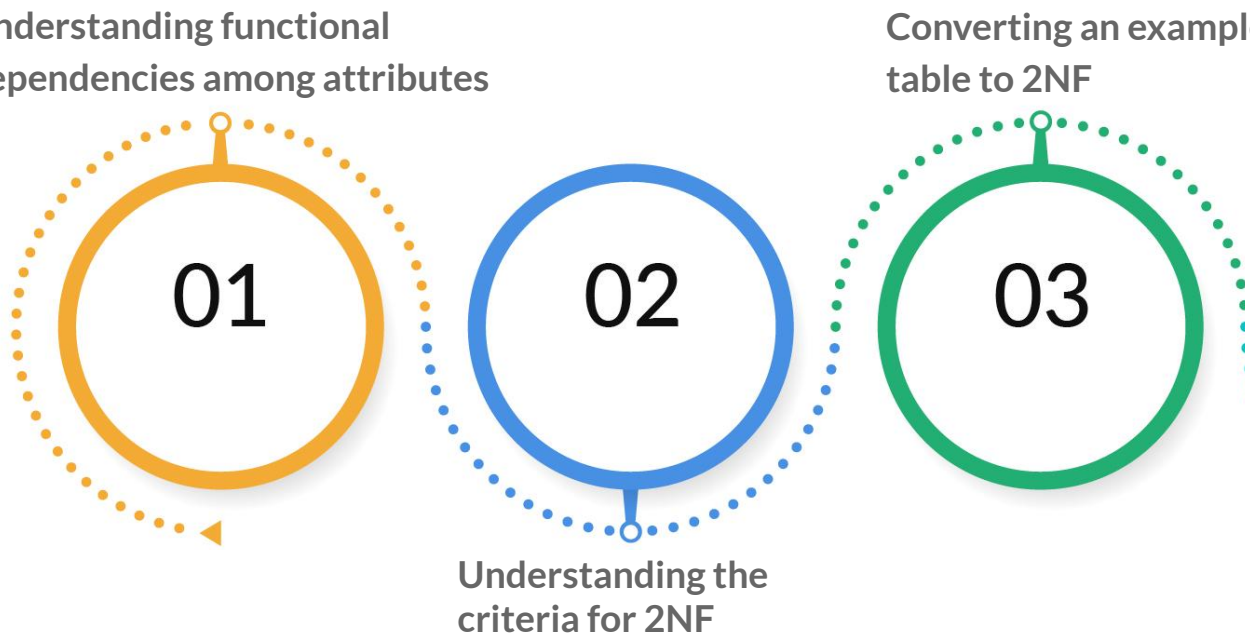
Summary | 1NF

For a table to be in 1NF, every field must contain not more than one value.

For a table to be in 1NF, there must be a primary key or a composite key.

Segment 4 | 2nd Normal Form

In This Segment



Functional Dependencies

Customer ID	Customer Name
C12	Virat
C46	Rohit
C14	Virat

Consider the two attributes: attribute A and attribute B

- If for every value of attribute A, there is only one value of attribute B, then attribute A can determine the value of attribute B.
- If attribute A can determine attribute B, then attribute B is functionally dependent on attribute A.
- A -> B; B is functionally dependent on A.

Consider two attributes: Customer ID and Customer Name

- For C12, there is only one unique value of Customer Name Virat.
- Customer ID -> Customer Name
- For Virat, there are two values of Customer ID: C12 and C14. Thus, Customer Name cannot determine Customer ID.
- Customer Name is functionally dependent on Customer ID, but Customer ID is not functionally dependent on Customer Name.

- **A -> B**
- **A can determine the value of B.**
- **B is functionally dependent on A.**

Full and Partial Functional Dependencies

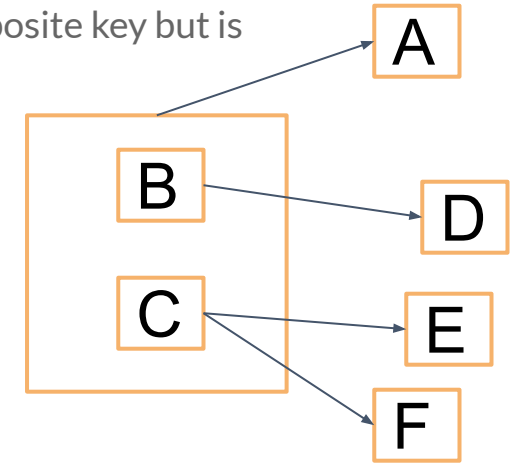
If the non-prime attribute A is functionally dependent on the complete composite key formed by B and C, then attribute A is fully functionally dependent on the composite key B and C.

If attribute B and attribute C form the composite key for a table and attribute D can be determined by the value of attribute B, then attribute B is not fully dependent on the composite key but is partially dependent on a part of the composite key B.

If only the value of B is known:
- **A cannot be determined,**
- **D can be determined.**

If only the value of C is known:
- **A cannot be determined,**
- **E and F can be determined.**

If the values of both B and C are known:
- **A can be determined,**
- **D, E and F can be determined.**



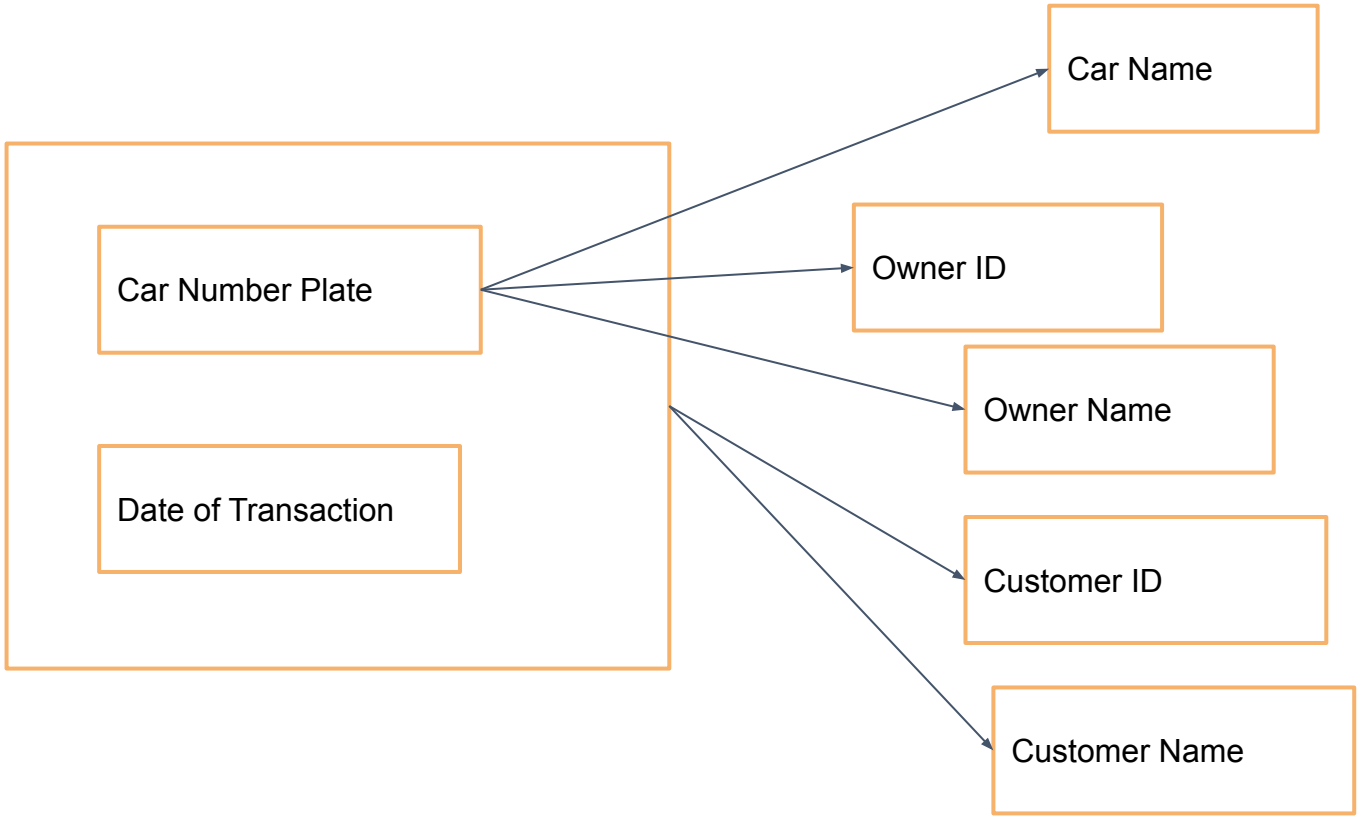
- **(B, C) -> A**
- **B -> D**
- **C -> E, F**

Functional Dependencies in the Table

Customer ID	Customer Name	Car Number Plate	Car Name	Date of Transaction	Owner ID	Owner Name
C12	Sachin	CarQ1234	Swift	12/01/2020	O76	Dev
C12	Sachin	CarQ5436	Thar	18/01/2020	O78	Irfan
C46	Rahul	CarQ3421	Baleno	12/01/2020	O54	Rohit
C46	Rahul	CarQ6534	Honda City	14/01/2020	O65	Shikhar
C46	Rahul	CarQ3789	Swift	15/01/2020	O86	Irfan

- (Car Number Plate, Date of Transaction) -> Customer ID, Customer Name, Car Name, Owner ID, Owner Name
- Car Number Plate -> Car Name, Owner ID, Owner Name
- Customer ID -> Customer Name (not partial dependencies)
- Owner ID -> Owner Name (not partial dependencies)

Functional Dependencies in the Table



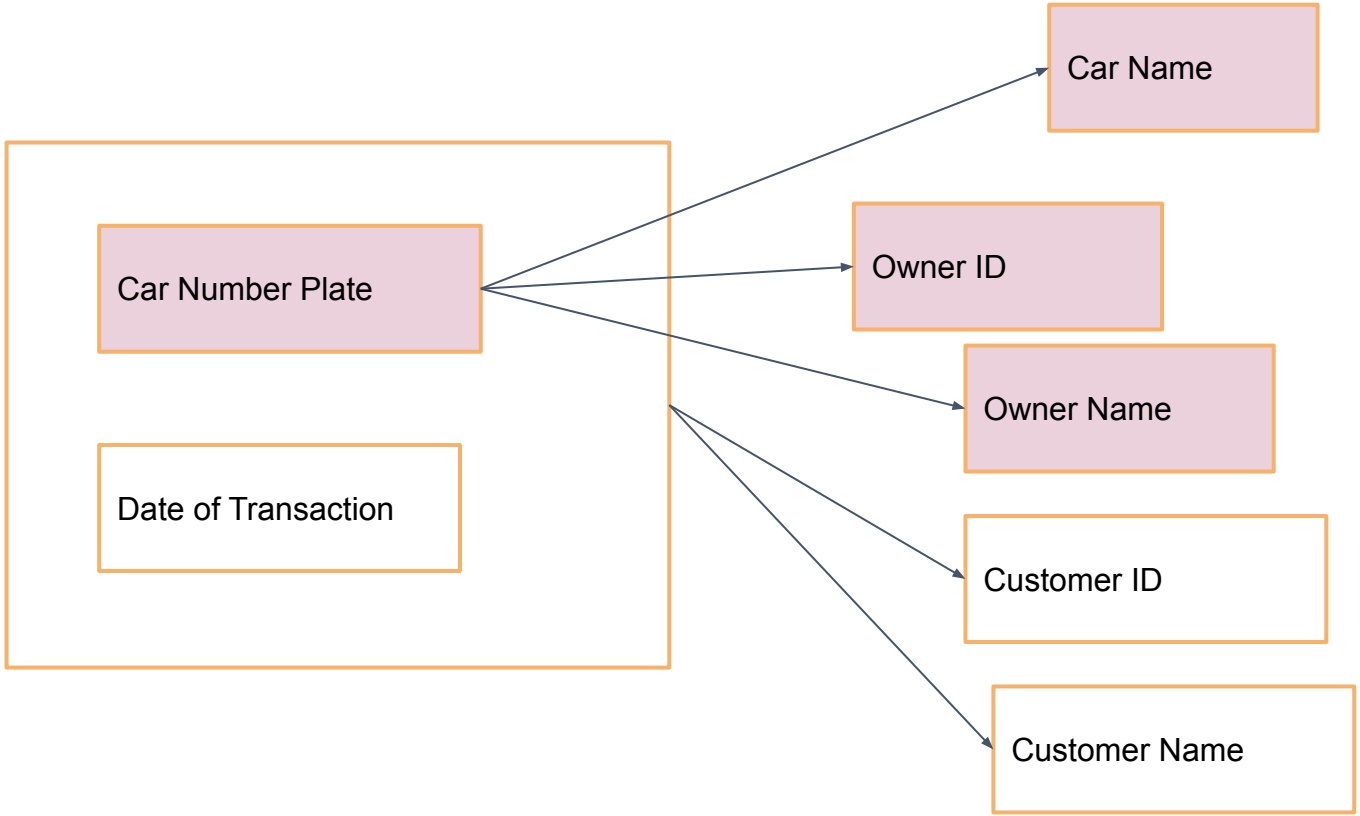
Criteria for 2NF

For a table to be in 2NF, a table must be first in 1NF.

For a table to be in 2NF, all the non-prime attributes must be fully functionally dependent on the composite key or the primary key of the table.

For a table to be in 2NF, there must be no partial dependencies on the composite key of the table.

Converting the Table to 2NF



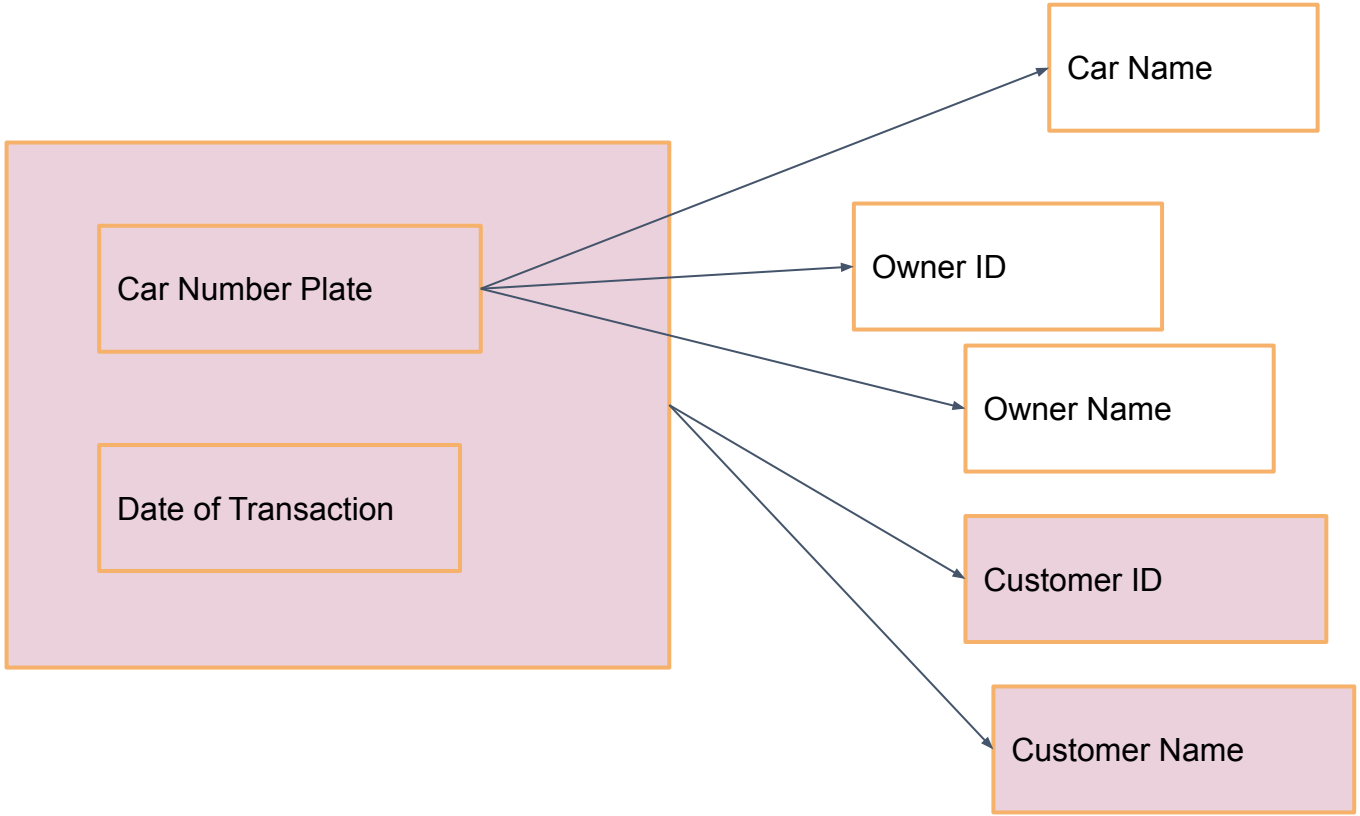
Converting the Table to 2NF

Car Number Plate	Car Name	Owner ID	Owner Name
CarQ1234	Swift	O76	Dev
CarQ5436	Thar	O78	Irfan
CarQ3421	Baleno	O54	Rohit
CarQ6534	Honda City	O65	Shikhar
CarQ3789	Swift	O86	Irfan

Car Name, Owner Name and Owner ID can be determined by Car Number Plate only. These attributes can be separated into another table.

Car Number Plate becomes the primary key of this table.

Converting the Table to 2NF



Converting the Table to 2NF

Car Number Plate	Date of Transaction	Customer ID	Customer Name
CarQ1234	12/01/2020	C12	Sachin
CarQ5436	18/01/2020	C12	Sachin
CarQ3421	12/01/2020	C46	Rahul
CarQ6534	14/01/2020	C46	Rahul
CarQ3789	15/01/2020	C46	Rahul

Both Car Number Plate and Date of Transaction can determine Customer ID and Customer Name.

Car Number Plate and Date of Transaction together become the composite key of this table.

Tables in 2NF

Car Number Plate	Date of Transaction	Customer ID	Customer Name
CarQ1234	12/01/2020	C12	Sachin
CarQ5436	18/01/2020	C12	Sachin
CarQ3421	12/01/2020	C46	Rahul
CarQ6534	14/01/2020	C46	Rahul
CarQ3789	15/01/2020	C46	Rahul

Car Number Plate in this table acts as a foreign key in the other table.

Car Number Plate	Car Name	Owner ID	Owner Name
CarQ1234	Swift	O76	Dev
CarQ5436	Thar	O78	Irfan
CarQ3421	Baleno	O54	Rohit
CarQ6534	Honda City	O65	Shikhar
CarQ3789	Swift	O86	Irfan

Are all the anomalies handled in these tables?
Car information is not repeated. Customer information and owner information are repeated.

Summary | 2NF

01 For a table to be in 2NF, a table must be first in 1NF.

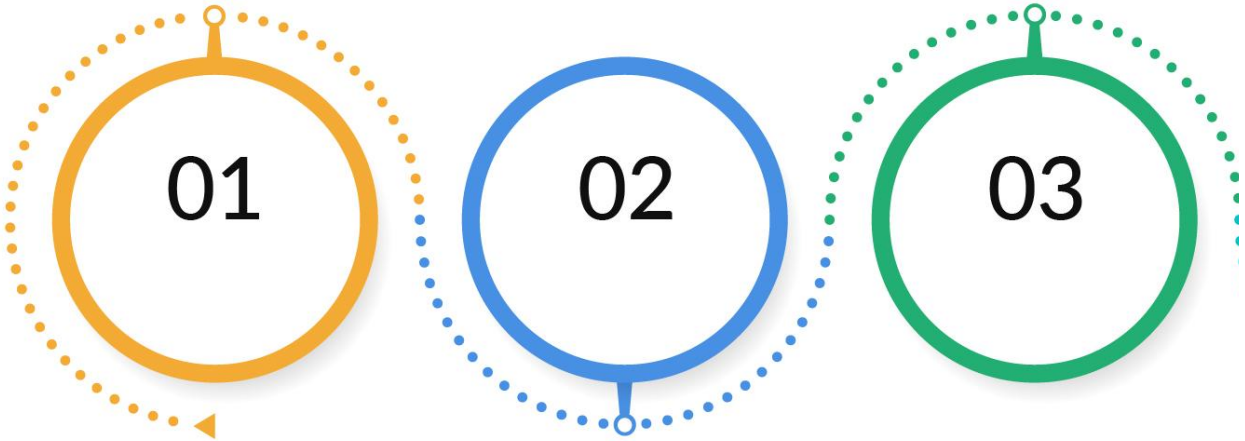
02 For a table to be in 2NF, all the non-prime attributes must be fully functionally dependent on the composite key or the primary key of the table.

03 For a table to be in 2NF, there must be no partial dependencies on the composite key of the table.

Segment 5 | 3rd Normal Form

In This Segment

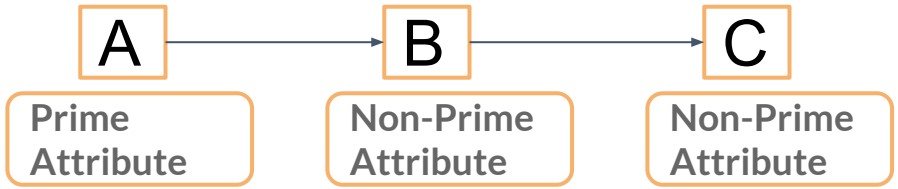
Understanding transitive dependencies



Converting an example table to 3NF

Understanding the criteria for 3NF

Transitive Dependencies



- Attribute C is functionally dependent on attribute B, and attribute B is functionally dependent on attribute A. Both attribute B and attribute C are non-prime attributes.
- A -> (B, C)
- B -> C
- This is a case of transitive dependency.

Checking for Transitive Dependencies

Car Number Plate	Date of Transaction	Customer ID	Customer Name
CarQ1234	12/01/2020	C12	Sachin
CarQ5436	18/01/2020	C12	Sachin
CarQ3421	12/01/2020	C46	Rahul
CarQ6534	14/01/2020	C46	Rahul
CarQ3789	15/01/2020	C46	Rahul

(Car Number Plate, Date of Transaction) -> Customer ID -> Customer Name

Car Number Plate	Car Name	Owner ID	Owner Name
CarQ1234	Swift	O76	Dev
CarQ5436	Thar	O78	Irfan
CarQ3421	Baleno	O54	Rohit
CarQ6534	Honda City	O65	Shikhar
CarQ3789	Swift	O86	Irfan

Car Number Plate -> Owner ID -> Owner Name

Criteria for 3NF

For a table to be in 3NF, the table must first be in 2NF.

For a table to be in 3NF, there must be no transitive dependencies in the table.

Tables in 3NF

Car Number Plate	Date of Transaction	Customer ID
CarQ1234	12/01/2020	C12
CarQ5436	18/01/2020	C12
CarQ3421	12/01/2020	C46
CarQ6534	14/01/2020	C46
CarQ3789	15/01/2020	C46

Customer ID	Customer Name
C12	Virat
C46	Rohit

Car Number Plate	Car Name	Owner ID
CarQ1234	Swift	O76
CarQ5436	Thar	O78
CarQ3421	Baleno	O54
CarQ6534	Honda City	O65
CarQ3789	Swift	O86

Owner ID	Owner Name
O76	Sachin
O78	Gavaskar
O54	Sehwag
O65	Shikhar
O86	Ganguly

Summary | 3NF

01 Every table in the database must be in 3NF to avoid anomalies.

02 There must be no field with more than one value. All the non-prime attributes must be fully functionally dependent on the composite key or the primary key of the table.

03 There must be no transitive dependencies in the table.

Session Summary

01 Data Normalization is the process of organising data in a database such that a piece of data is not repeated at various places.

02 For a table to be in 1NF, there must not be more than one value in any field. There must be a primary key to uniquely identify each row.

03 Functional dependency defines if one attribute can be determined by another attribute. If A is functionally dependent on B, B can determine the value of A.

04 Full Functional Dependency occurs when a non-prime attribute can be determined only by the entire composite key

05 Partial Dependency occurs when a non-prime attribute can be determined only by a part of composite key.

06 For a table to be in 2NF, there must not be any partial dependency. All non-prime attributes must fully functionally depend on the composite key

07 A transitive dependency occurs when a non-prime attribute A can be determined by another non-prime attribute B and B is fully functionally dependent on primary key.

08 For a table to be in 3NF, there must not be any transitive dependencies

09 For a table to be in 2 NF, a table must be in 1NF.

10 For a table to be in 3NF, a table must be in 2NF.

Thank you