DBMS

UNIT -IV

Normalization in a Database

It is the processes of reducing the redundancy of data in the table and also improving the data integrity.

Without Normalization in SQL, we may face many issues such as

1. **Insertion anomaly** - It occurs when we cannot insert data to the table without the presence of another attribute
2. ***Update anomaly*** - It is a data inconsistency that results from data redundancy and a partial update of data.
3. ***Deletion Anomaly*** - It occurs when certain attributes are lost because of the deletion of other attributes.

Normalization is a way of organizing the data in the database. Normalization entails organizing the columns and tables of a database to ensure that their dependencies are properly enforced by database integrity constraints. It usually divides a large table into smaller ones, so it is more efficient.

1st Normal Form (1NF)

In this Normal Form, we tackle the problem of atomicity. Here atomicity means values in the table should not be further divided. In simple terms, a single cell cannot hold multiple values. If a table contains a composite or multi-valued attribute, it violates the First Normal Form.

Employee ID	Employee Name	Phone Number	Salary
1EDU001	Alex	+91 8553206126 +91 9449424949	60,131
1EDU002	Barry	+91 8762989672	48,302
1EDU003	Clair	+91 9916255225	22,900
1EDU004	David	+91 6363625811 +91 8762055007	81,538

In the above table, we can clearly see that the Phone Number column has two values. Thus it violated the 1st NF. Now if we apply the 1st NF to the above table we get the below table as the result.

Employee ID	Employee Name	Phone Number	Salary
1EDU001	Alex	+91 8553206126	60,131
1EDU001	Alex	+91 9449424949	60,131
1EDU002	Barry	+91 8762989672	48,302
1EDU003	Clair	+91 9916255225	22,900
1EDU004	David	+91 6363625811	81,538
1EDU004	David	+91 8762055007	81,538

By this, we have achieved atomicity and also each and every column have unique values.

2nd Normal Form (2NF)

The first condition in the 2nd NF is that the table has to be in 1st NF. The table also should not contain partial dependency. Here partial dependency means the proper subset of candidate key determines a non-prime attribute. For example

Consider the table

Employee Id	Department Id	Office Location
1EDU001	ED-T1	Pune
1EDU002	ED-S2	Bengaluru
1EDU003	ED-M1	Delhi
1EDU004	ED-T3	Mumbai

This table has a composite primary key Emplyoee ID, Department ID. The non-key attribute is Office Location. In this case, Office Location only depends on Department ID, which is only part of the primary key. Therefore, this table does not satisfy the second Normal Form.

To bring this table to Second Normal Form, we need to break the table into two parts. Which will give us the below tables:

Employee Id	Department Id
1EDU001	ED-T1
1EDU002	ED-S2
1EDU003	ED-M1
1EDU004	ED-T3

Department Id	Office Location
ED-T1	Pune
ED-S2	Bengaluru
ED-M1	Delhi
ED-T3	Mumbai

As you can see we have removed the partial functional dependency that we initially had. Now, in the table, the column Office Location is fully dependent on the primary key of that table, which is Department ID.

Now that we have learnt 1st and 2nd normal forms lets head to the next part of this Normalization in SQL article.

3rd Normal Form (3NF)

The same rule applies as before i.e., the table has to be in 2NF before proceeding to 3NF. The other condition is there should be no transitive dependency for non-prime attributes. That means non-prime attributes (which doesn't form a candidate key) should not be dependent on other non-prime attributes in a given table. So a transitive dependency is a functional dependency in which X → Z (X determines Z) indirectly, by virtue of X → Y and Y → Z (where it is not the case that Y → X)

Let's understand this more clearly with the help of an example:

Student Id	Student Name	Subject Id	Subject	Address
1DT15ENG01	Alex	15CS11	SQL	Goa
1DT15ENG02	Barry	15CS13	JAVA	Bengaluru
1DT15ENG03	Clair	15CS12	C++	Delhi
1DT15ENG04	David	15CS13	JAVA	Kochi

In the above table, Student ID determines Subject ID, and Subject ID determines Subject. Therefore, Student ID determines Subject via Subject ID. This implies that we have a transitive functional dependency, and this structure does not satisfy the third normal form.

Now in order to achieve third normal form, we need to divide the table as shown below:

Student Id	Student Name	Subject Id	Address
1DT15ENG01	Alex	15CS11	Goa
1DT15ENG02	Barry	15CS13	Bengaluru
1DT15ENG03	Clair	15CS12	Delhi
1DT15ENG04	David	15CS13	Kochi

Subject Id	Subject
15CS11	SQL
15CS13	JAVA
15CS12	C++
15CS13	JAVA

As you can see from the above tables all the non-key attributes are now fully functional dependent only on the primary key. In the first table, columns Student Name, Subject ID and Address are only dependent on Student ID. In the second table, Subject is only dependent on Subject ID.