Normalization is a process of organizing the data in database to avoid data redundancy, insertion anomaly, update anomaly & deletion anomaly.

OR

Normalization is the process of minimizing redundancy from a relation or set of relations. Redundancy in relation may cause insertion, deletion and update anomalies. So, it helps to minimize the redundancy in relations. Normal forms are used to eliminate or reduce redundancy in database tables.

Anomalies in DBMS

There are three types of anomalies that occur when the database is not normalized. These are – Insertion, update and deletion anomaly. Let's take an example to understand this.

Example: Suppose a manufacturing company stores the employee details in a table named employee that has four attributes: emp_id for storing employee's id, emp_name for storing employee's name, emp_address for storing employee's address and emp_dept for storing the department details in which the employee works. At some point of time the table looks like this:

emp_id	emp_name	emp_address	emp_dept
101	Rick	Delhi	D001
101	Rick	Delhi	D002
123	Maggie	Agra	D890
166	Glenn	Chennai	D900
166	Glenn	Chennai	D004

The above table is not normalized. We will see the problems that we face when a table is not normalized.

Update anomaly: In the above table we have two rows for employee Rick as he belongs to two departments of the company. If we want to update the address of Rick then we have to update the same in two rows or the data will become inconsistent. If somehow, the correct address gets updated in one department but not in other then as per the database, Rick would be having two different addresses, which is not correct and would lead to inconsistent data.

Insert anomaly: Suppose a new employee joins the company, who is under training and currently not assigned to any department then we would not be able to insert the data into the table if emp_dept field doesn’t allow nulls.

Delete anomaly: Suppose, if at a point of time the company closes the department D890 then deleting the rows that are having emp_dept as D890 would also delete the information of employee Maggie since she is assigned only to this department.

To overcome these anomalies we need to normalize the data. In the next section we will discuss about normalization**.**

The normal form of a relation refers to the highest normal form condition that it meets, and hence indicates the degree to which it has been normalized.

Type of Normal Form

First Normal Form (1NF):

If a relation contain composite or multi-valued attribute, it violates first normal form, or a relation is in first normal form if it does not contain any composite or multi-valued attribute. A relation is in first normal form if every attribute in that relation is singled valued attribute.

A table is in 1 NF iff:

There are only Single Valued Attributes.

Attribute Domain does not change.

There is a Unique name for every Attribute/Column.

The order in which data is stored, does not matter.

OR

In the relational model, we formalize this idea that attributes do not have any substructure. A domain is atomic if elements of the domain are considered to be indivisible units. We say that a relation schema R is in first normal form (1NF) if the domains of all attributes of R are atomic.

A set of names is an example of a nonatomic value. For example, if the schema of a relation employee included an attribute children whose domain elements are sets of names, the schema would not be in first normal form.

Second normal form:

A relation is in second normal form if,

1. it is in 1NF

2. Every non key attribute is fully functionally dependent on the primary key/candidate key of the relation R (For relations where primary key contains multiple attributes, no nonkey attribute should be functionally dependent on a part of the primary key)

Prime (key) and non-prime (non key) attributes

Attributes which are parts of any candidate key of relation are called as prime attribute or key attribute, others are non-prime or non key attributeattributes.

**In a relation R attribute B of R is “fully functional dependent” on an attribute or set of attribute A of R if B is functional dependent on A but not functional dependent on any proper subset of A.

OR

A functional dependency A→B is called a partial dependency if there is a proper subset Z of A such that Z → B. We say that B is partially dependent on A

For Example

Consider the following set F ={A → BC, CD → E, B → D, E → A of functional dependencies for relation schema R = (A, B, C, D, E). Is the relation in 2NF.

The candidate keys of relation R are A, BC, CD, and E

So, Prime attributes are A, B, C, D & E. Because their is no non prime attribute in relation R then Relation is in 2NF.

Example 2: Consider the following set F = { A → B, ACD → E, EF → GH } for the relation schema R = (A, B, C, D, E, F, G, H)

The candidate keys of relation R are ACDF

So, the prime attributes are A, C, D, & F & non prime attributes are B, E, G, H. Now as defined in the 2NF defination, A → B is not legal FD because A is a prime attribute and a non prime attribute B functionally dependent on a part of the candidate key ACDF i.e A

Third Normal Form

In general,a functional dependency of the form A → B is trivial if B ⊆ A

A relation schema R is in third normal form with respect to a set F of functional dependencies if, for all functional dependencies in F^+ of the form A → B , where

$A \subseteq R$ and $B \subseteq R$, **at least one of the following holds:**

- A → B is a trivial functional dependency.
- A is a superkey for R.
- Each attribute Z in B − A is contained in a candidate key for R.

let us consider the dept advisor relationship set, which has the following functional dependencies:

dept advisor (s_ID , i_ID , deptname)

i_ID → dept name

s_ID , dept name → i_ID

1. s_ID , dept name → i_ID is alegal FD because s_ID ,dept name is a candidate key of relation R
2. Note that here A= i_ID , B= dept name, and B − A = dept name. Since the functional dependency s_ID , dept name →i ID holds on dept advisor, the attribute dept name is contained in a candidate key(s_ID dept name is a candidate key of relation dept advisor) and, therefore, dept advisor is in 3NF.

Boyce–Codd normal form (BCNF)

A relation schema R is in third normal form with respect to a set F of functional dependencies if, for all functional dependencies in F^+ of the form A → B , where

$A \subseteq R$ and $B \subseteq R$, **at least one of the following holds:**

- A → B is a trivial functional dependency.
- A is a superkey for R.

For example in above example of 3NF, relation R is not in BCNF because FD i_ID → dept name is not legal under defination of BCNF i.e dept name is not a subset of i_ID & i_ID is not any key of relation R