**DATABASE MODELING AND NORMALIZATION**

1. **What is database modeling?.**

A database model shows the logical structure of a database, including the relationships and constraints that determine how data can be stored and accessed. Individual database models are designed based on the rules and concepts of whichever broader data model the designers adopt. **In other way "A** Database model defines the logical design and structure of a database and defines how data will be stored, accessed and updated in a database management system.”

## **Types of database models**

There are many kinds of data models. Some of the most common ones include:

1. Hierarchical database model
2. Relational model
3. Network model
4. Object-oriented database model
5. Entity-relationship model
6. Document model
7. Entity-attribute-value model
8. Star schema

**2.What is a normalization?.**

Normalization is a process and it should be carried out for every database you design. The process of taking a database design, and apply a set of formal criteria and rules, is called Normal Forms.Database normalization is a database schema design technique, by which an existing schema is modified to minimize redundancy and dependency of data.Normalization split a large table into smaller tables and define relationships between them to increases the clarity in organizing data.

## **Types of Normalization**

1. First Normal Form(1NF)

1. Second Normal Form(2NF)

1. Third Normal Form(3NF)

1. Boyce-Codd Normal Form (BCNF)

**1NF** -

* Each table cell should contain a single value.
* Each record needs to be unique.

|  |  |  |
| --- | --- | --- |
| **Roll\_Number** | **Full\_Name** | **Subject** |
| 101 | John Doe | DBMS,OS |
| 102 | Robert Pill | Java |

**1NF EXAMPLE**

|  |  |  |
| --- | --- | --- |
| **Roll\_Number** | **Full\_Name** | **Subject** |
| 101 | John Doe | DBMS |
| 101 | John Doe | OS |
| 102 | Robert Pill | java |

**2NF**

The entity should be considered already in 1NF, and all attributes within the entity should depend solely on the unique identifier of the entity.

**2NF EXAMPLE -**

Sample Table

|  |  |  |
| --- | --- | --- |
| **Product\_ID** | **Product** | **BRAND** |
| 1 | Monitor | Apple |
| 2 | Monitor | Samsung |
| 3 | Scanner | HP |

Product table following 2NF table:

|  |  |
| --- | --- |
| **Product\_ID** | **Product** |
| 1 | Monitor |
| 2 | Monitor |
| 3 | Scanner |

Brand table:

|  |  |
| --- | --- |
| **Brand\_ID** | **BRAND** |
| 1 | Apple |
| 2 | Samsung |
| 3 | HP |

Product Brand table :

|  |  |  |
| --- | --- | --- |
| **PBID** | **Product\_ID** | **Brand\_ID** |
| 1 | 1 | 1 |
| 2 | 2 | 2 |
| 3 | 3 | 3 |

**3NF**

1. The entity should be considered already in 2NF, and no column entry should be dependent on any other entry (value) other than the key for the table.
2. If such an entity exists, move it outside into a new table.
3. 3NF is achieved, considered as the database is normalized.

**3NF EXAMPLE-**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EMP\_ID** | **EMP\_NAME** | **EMP\_ZIP** | **EMP\_STATE** | **EMP\_CITY** |
| 1 | John | 223 | UK | Norwich |
| 2 | Robert | 334 | US | Chicago |
| 3 | Sam | 232 | UP | Noida |

Employee table

Non-prime attributes: Here all attributes except {EMP\_ID} are non-prime.

Here, EMP\_STATE & EMP\_CITY dependent on EMP\_ZIP and EMP\_ZIP dependent on EMP\_ID. The non-prime attributes (EMP\_STATE, EMP\_CITY) transitively dependent on super key(EMP\_ID). This violates the rules of the third normal form. We need to move the EMP\_CITY and EMP\_STATE to the new <EMPLOYEE\_ZIP> table, with EMP\_ZIP as a Primary key.

|  |  |  |
| --- | --- | --- |
| **EMP\_ID** | **EMP\_NAME** | **EMP\_ZIP** |
| 1 | John | 223 |
| 2 | Robert | 334 |
| 3 | Sam | 232 |

|  |  |  |
| --- | --- | --- |
| **EMP\_ZIP** | **EMP\_STATE** | **EMP\_CITY** |
| 223 | UK | Norwich |
| 334 | US | Chicago |
| 232 | UP | Noida |