**STUDENT NAME: SAI KIRAN ETIKAPALLY**

**GIT HUB URL:** [**https://github.com/Saikiran91777/Databaseprojectpart3**](https://github.com/Saikiran91777/Databaseprojectpart3)

**TOPIC: ERD WITH NORMALIZATION**

**DESCRIPTION**

Normalization is a database design technique used in relational databases to eliminate duplication.In computerized data management, the relational database model is the most extensively used paradigm. Information in relational databases is kept as entries in tables linked by keys. A data record is made up of numerous value ranges that are allocated to certain characteristics through the use of table columns.

We'll step through the distinct phases of relational database normalization using the data in the table above as an example to demonstrate the conversion of a relational database into the first, second, and third normal forms.

The first normal form (1NF)

When a table in a relational database meets the following conditions, it conforms to the first normal form (1NF):

All information is atomic.

The values in all table columns are the same.

If each piece of information is allocated to a different data field, a data set is termed atomic.

The second standard form (2NF)

A table in accordance with the second normal form must meet all of the conditions of the first normal form as well as the following:

Each non-key attribute, which is dependent on the primary key, must be completely functioning.A relational database is defined in the introduction as a set of discrete tables that are linked together using keys.

In relational databases, keys are used to uniquely identify data records (tuples). A super key is a key that allows you to uniquely name the individual lines of a database table. A key of this kind can represent the values of a single column or the values of numerous columns together.

To convert a database table to the second normal form, you must first discover not only the main key and all non-key properties, ut also their relationships. Take the following steps:

Check to see if all non-key qualities are completely functionally reliant on the primary key. This type of dependence arises only if all primary key characteristics are required to uniquely identify the non-key attribute. This also means that tables with one-part primary keys automatically correspond to the second normal form if the first normal form's conditions are met.

Separate all non-key properties that are not totally functionally dependent on the whole main key.

Third normal form (3NF)

To convert a table to the third normal form, the conditions of the first and second normal forms must be met, as well as the following:

There can be no non-key attributes that are transitively reliant on a key candidate.

When a non-key attribute is reliant on another non-key attribute and hence indirectly on its key candidate, this is referred to as a transitive dependence.

To eliminate any dependencies between non-key attributes, the necessary attributes have been transferred to separate tables connected by foreign keys.

Even though database normalisation necessitates more programming work, 3NF - the third normal form - is widely accepted as the standard for relational database formulae and is only varied in unusual instances. Databases that conform to the third normal form, for example, are sometimes denormalized to the second normal form. This is due to the fact that joins across numerous tables are time-consuming in very big databases. Denormalization decreases the number of tables and, as a result, the query time.

The goal of normalization is to limit the number of instances of double values. By converting a database to one of the standard forms specified, the target schema gains less redundancy than the source schema. Normalization also simplifies database upkeep.

Database normalization, on the other hand, always entails storing attributes in different tables. This may necessitate the incorporation of foreign keys, which may result in key redundancy. The main drawback is that logically related data is no longer kept together in a normalized database. To integrate data that has been divided into many tables, a join is necessary.

Joins in database queries can be used to filter out complex information. Joins, on the other hand, are more difficult to construct than simple searches. This also takes a lot longer.