**What is Normalization? Why should we use it?**

Normalization is a database design technique which organizes tables in a manner that reduces redundancy and dependency of data. It divides larger tables to smaller tables and link them using relationships.

The inventor of the relational model Edgar Codd proposed the theory of normalization with the introduction of First Normal Form and he continued to extend theory with Second and Third Normal Form. Later he joined with Raymond F. Boyce to develop the theory of Boyce-Codd Normal Form.

**Example:**

Assume a video library maintains a database of movies rented out. Without any normalization all information is stored in one table as shown below.

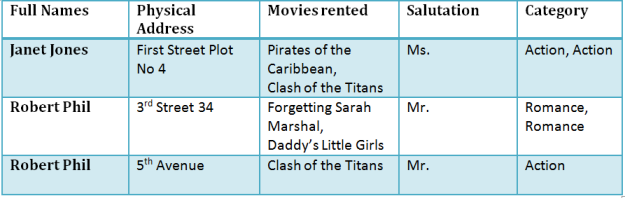


Table 1

Here you see **Movies Rented column has multiple values**.

**1NF Rules**

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

The above table in 1NF-

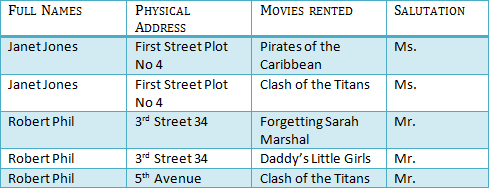


Table 1 : In 1NF Form

**2NF Rules**

* Rule 1- Be in 1NF
* Rule 2- Single Column Primary Key
* Every Non-Key Attribute must depend on the key attribute.

It is clear that we can’t move forward to make our simple database in 2nd Normalization form unless we partition the table above.

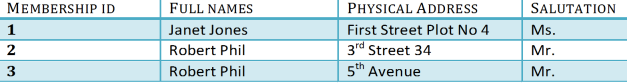


Table 1

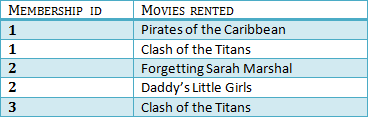


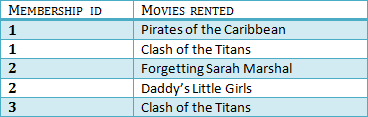
Table 2

We have divided our 1NF table into two tables viz. Table 1 and Table2. Table 1 contains member information. Table 2 contains information on movies rented.

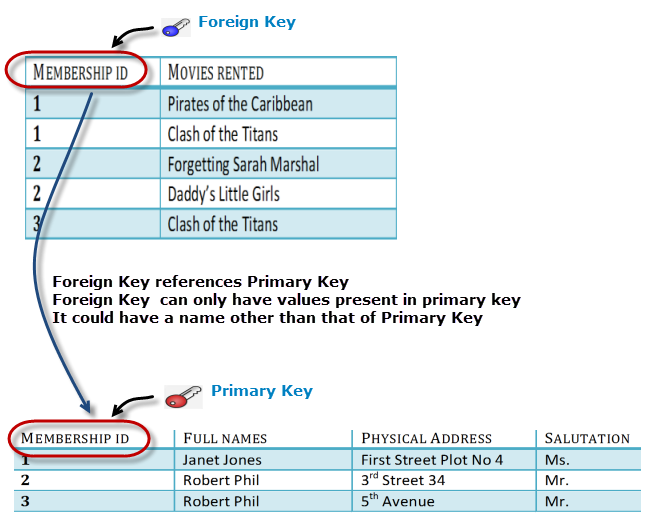
We have introduced a new column called Membership\_id which is the primary key for table 1. Records can be uniquely identified in Table 1 using membership id

Introducing Foreign Key!

In Table 2, Membership\_ID is the foreign Key

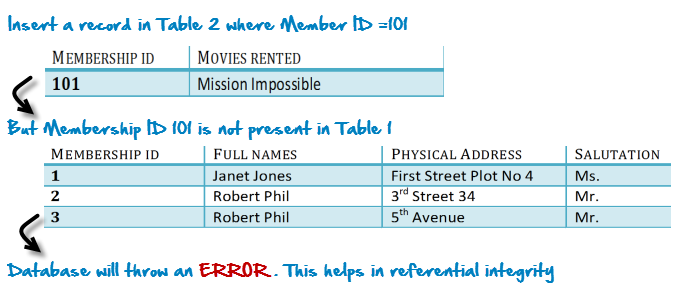


|  |  |
| --- | --- |
|  | Foreign Key references primary key of another Table!It helps connect your Tables     * A foreign key can have a different name from its primary key * It ensures rows in one table have corresponding rows in another * Unlike Primary key they do not have to be unique. Most often they aren’t * Foreign keys can be null even though primary keys can not |



Why do you need a foreign key ?

You will only be able to insert values into your foreign key that exist in the unique key in the parent table. This helps in referential integrity.



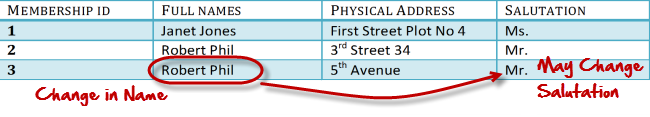
The above problem can be overcome by declaring membership id  from Table2  as foreign key of membership id  from Table1

Now , if somebody tries to insert a value in the membership id  field that does not exist in the parent table , an error will be shown!

What is a transitive functional dependencies?

A transitive functional dependency is when changing a non-key column , might cause any of the other non-key columns to change

Consider the table 1. Changing the non-key column Full Name , may change Salutation.



**3NF Rules**

* Rule 1- Be in 2NF
* Rule 2- Has no transitive functional dependencies

To move our 2NF table into 3NF we again need to need divide our table.

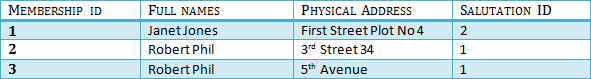


TABLE 1

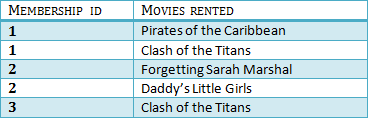


Table 2

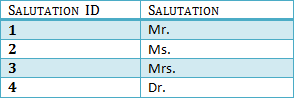


Table 3

We have again divided our tables and created a new table which stores Salutations.

There are no transitive functional dependencies and hence our table is in 3NF

In Table 3 Salutation ID is primary key and in Table 1 Salutation ID is foreign to primary key in Table 3