

DATA NORMALIZATION

Normalization 1NF, 2NF, 3NF

- **Normalization** is a database design technique that reduces data redundancy and eliminates undesirable characteristics like Insertion, Update and Deletion Anomalies.
- **Normalization rules** divides larger tables into smaller tables and links them using relationships. The purpose of Normalization in SQL is to eliminate redundant (repetitive) data and ensure data is stored logically.

Normalization Example

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

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean, Clash of the Titans	Ms.
Robert Phil	3 rd Street 34	Forgetting Sarah Marshal, Daddy's Little Girls	Mr.
Robert Phil	5 th Avenue	Clash of the Titans	Mr.

Table 1

1NF (First Normal Form) Rules

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

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3 rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3 rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5 th Avenue	Clash of the Titans	Mr.

Table 1: In 1NF Form

KEY

A **KEY** is a value used to identify a record in a table uniquely.

A KEY could be a single column or combination of multiple columns

Note: Columns in a table that are NOT used to identify a record uniquely are called non-key columns.

Primary Key

A primary is a single column value used to identify a database record uniquely.

It has following attributes:

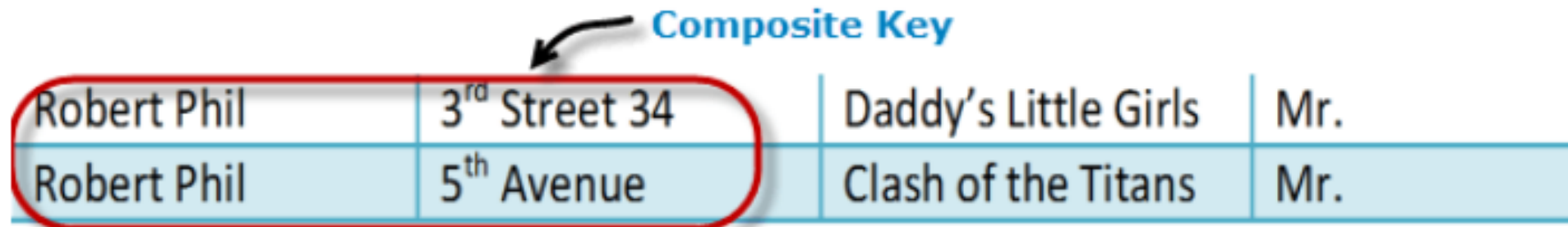
- A primary key cannot be NULL
- A primary key value must be unique
- The primary key values should rarely be changed
- The primary key must be given a value when a new record is inserted

Composite Key

- A composite key is a primary key composed of multiple columns used to identify a record uniquely

In our database, we have two people with the same name Robert Phil, but they live in different places. Hence, we require both Full Name and Address to identify a record uniquely. That is a composite key. Now we can move to 2NF...

Composite Key



Robert Phil	3 rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5 th Avenue	Clash of the Titans	Mr.

Names are common. Hence you need name as well Address to uniquely identify a record.

2NF (Second Normal Form) Rules

- Rule 1- Be in 1NF
- Rule 2- Single Column Primary Key

Considering both rules, it is very clear that we can't move forward to make our simple database in 2NF unless we partition the table

2NF (Second Normal Form) Rules – cont'd

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

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3 rd Street 34	Mr.
3	Robert Phil	5 th Avenue	Mr.

Table 1

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Table 2

Foreign Key

In Table 2, Membership_ID is the Foreign Key

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Foreign Key – cont'd

Foreign Key references the **primary key** of another Table, and thus it helps connect your Tables.

Characteristics:

- Foreign keys can have a different name from its primary keys
- Foreign keys can only have values present in primary keys
- It ensures rows in one table have corresponding rows in another
- Unlike the Primary key, they don't have to be unique. Most often they aren't

Foreign Key



MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Primary Key



MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3 rd Street 34	Mr.
3	Robert Phil	5 th Avenue	Mr.

Referential Integrity

Suppose, you insert a record in Table B such as

Insert a record in Table 2 where Member ID = 101

MEMBERSHIP ID	MOVIES RENTED
101	Mission Impossible

But Membership ID 101 is not present in Table 1

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3 rd Street 34	Mr.
3	Robert Phil	5 th Avenue	Mr.

Database will throw an **ERROR**. This helps in referential integrity

Referential Integrity – cont'd

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 problem can be solved by declaring `membership_id` from Table2 as foreign key of `membership_id` from Table1

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

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 (i.e.: from Robert Phil to Mary Smith)

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3 rd Street 34	Mr.
3	Robert Phil	5 th Avenue	Mr. <i>May Change</i>

Change in Name (circled around 'Robert Phil' in row 3)

Salutation (with arrow pointing from the circled name to the salutation in row 3)

We will use all the concepts explained before to move into 3NF now

3NF (Third Normal Form) Rules

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

To move our 2NF table into 3NF, first we need to **eliminate the transitive functional dependencies**. To do that, we again divide our table to create a new Table 3 which stores Salutations. In Table 3 Salutation ID is primary key, and in Table 1 Salutation ID is foreign to primary key in Table 3

Now, the table is in 3NF

3NF Example in SQL database:

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION ID
1	Janet Jones	First Street Plot No 4	2
2	Robert Phil	3 rd Street 34	1
3	Robert Phil	5 th Avenue	1

TABLE 1

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Table 2

SALUTATION ID	SALUTATION
1	Mr.
2	Ms.
3	Mrs.
4	Dr.

Table 3

- Now our example is at a level that cannot further be decomposed to attain higher normal forms of normalization.
- Additional efforts for moving into next levels of normalizing data are normally needed in complex databases. Refer to the complete course content for further details on how to go from 3NF, Boyce-Codd NF, 4NF, 5NF, and 6NF (not standardized)

Summary

- Database designing is critical to the successful implementation of a database management system (DBMS) that meets data requirements
- Normalized DBMS are cost-effective and have better security models
- Functional dependencies are important component of the normalization data process
- Most database systems are normalized database up to the 3NF
- A primary key uniquely identifies a record in a Table and cannot be null
- A foreign key helps connect table and references a primary key

