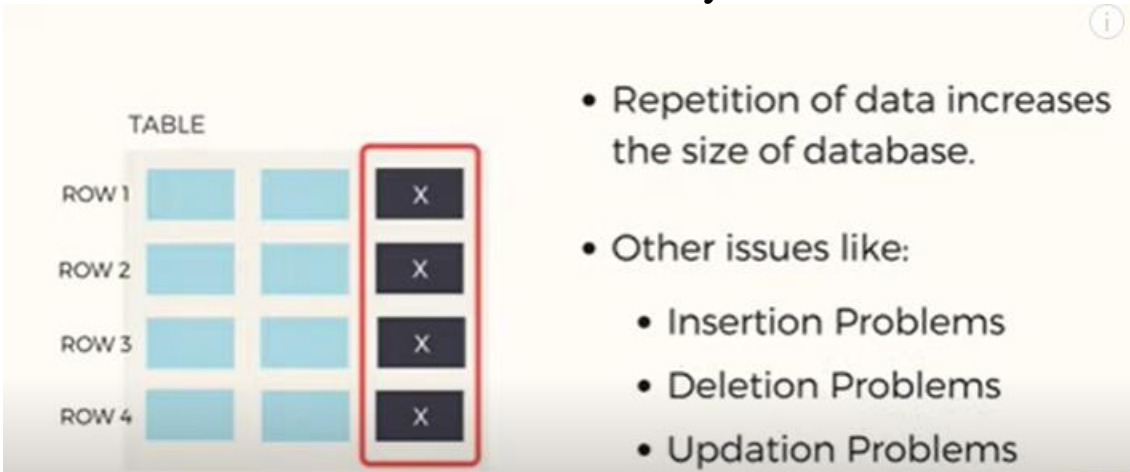


NORMALIZATION IN DATABASE

normalization is

a technique of organizing the data into multiple related tables, to minimize **DATA REDUNDANCY**.

Data redundancy



TABLE

ROW 1			X
ROW 2			X
ROW 3			X
ROW 4			X

- Repetition of data increases the size of database.
- Other issues like:
 - Insertion Problems
 - Deletion Problems
 - Updation Problems

EXAMPLE

STUDENTS TABLE				
rollno	name	branch	hod	office_tel

AFTER INSERTION OF DATA

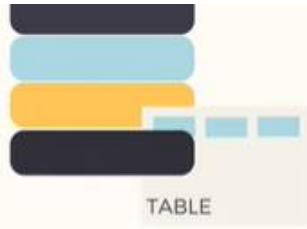
STUDENTS TABLE				
rollno	name	branch	hod	office_tel
1	Akon	CSE	Mr. X	53337
2	Bkon	CSE	Mr. X	53337
3	Ckon	CSE	Mr. X	53337
4	Dkon	CSE	Mr. X	53337

ISSUES

Unnecessary data
repetition increases the
size of the database.

And leads to more issues.

OTHER ISSUES



issues due to redundancy.

1. Insertion Anomaly
2. Deletion Anomaly
3. Updation Anomaly

INSERTION ANOMLY

Insertion Anomaly

To insert redundant data for every new row (of Student data in our case) is a data insertion problem or anomaly.

STUDENTS TABLE

rollno	name	branch	hod	office_tel
1	Akon	CSE	Mr. X	53337
2	Bkon	CSE	Mr. X	53337
3	Ckon	CSE	Mr. X	53337
4	Dkon	CSE	Mr. X	53337
5	Ekon	CSE	Mr. X	53337

STUDENTS TABLE

rollno	name	branch	hod	office_tel
--------	------	--------	-----	------------

Branch information deleted along
with Student data.

Deletion Anomaly

Loss of a related dataset when some other dataset is deleted.

STUDENTS TABLE				
rollno	name	branch	hod	office_tel
1	Akon	CSE	Mr. X	53337
2	Bkon	CSE	Mr. X	53337
3	Ckon	CSE	Mr. X	53337
4	Dkon	CSE	Mr. X	53337
<p>Mr. X leaves, and Mr. Y joins as the new HOD for CSE</p>				

UPDATION ANOMLY

**issue while
modifying the
data...**

STUDENTS TABLE				
rollno	name	branch	hod	office_tel
1	Akon	CSE	Mr. X Mr. Y	53337
2	Bkon	CSE	Mr. X Mr. Y	53337
3	Ckon	CSE	Mr. X Mr. Y	53337
4	Dkon	CSE	Mr. X Mr. Y	53337

INCONSISTANCY IN DATA

STUDENTS TABLE				
rollno	name	branch	hod	office_tel
1	Akon	CSE	Mr. X Mr. Y	53337
2	Bkon	CSE	Mr. X Mr. Y	53337
3	Ckon	CSE	Mr. X	53337
4	Dkon	CSE	Mr. X Mr. Y	53337

Data Redundancy

- Repetition of data hence needs extra space.
- Leads to insertion, deletion and updation issues.

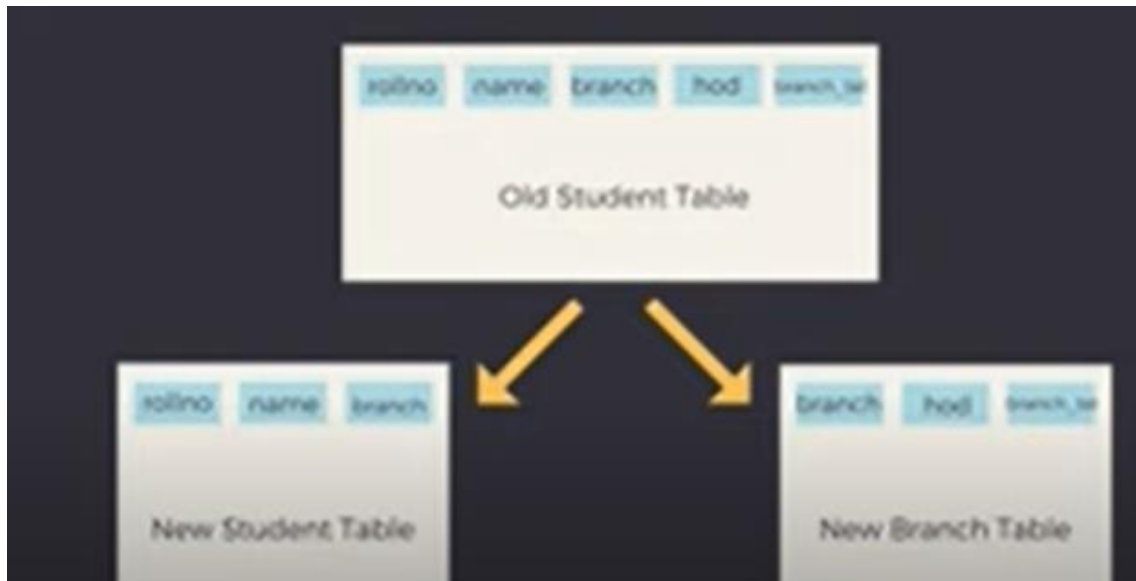
HOW NORMALIZATION SOLVE THESE PROBLEMS

How Normalization will solve all these problems?

Student Table



Student Table + Branch Table



BRANCH/DEPT. TABLE		
branch	hod	office_tel
CSE	Mr. Y	53337

STUDENTS TABLE		
rollno	name	branch
1	Akon	CSE

NORMALIZATION IS NOT ABOUT ELIMINATING REDUNDANCY

STUDENTS TABLE			BRANCH TABLE		
rollno	name	branch	branch	hod	office_tel
1	Akon	CSE	CSE	Mr. Y	53337
2	Bkon	CSE			
3	Ckon	CSE			

Not Eliminating Redundancy

BUT ABOUT MINIMIZING REDUNDANCY

STUDENTS TABLE			BRANCH TABLE		
rollno	name	branch	branch	hod	office_tel
1	Akon	CSE	CSE	Mr. Y	53337
2	Bkon	CSE			
3	Ckon	CSE			

But Minimizing Redundancy

STUDENTS TABLE			BRANCH TABLE		
rollno	name	branch	branch	hod	office_tel
1	Akon	CSE	CSE	Mr. Y	53337
2	Bkon	CSE			
3	Ckon	CSE			
4	Dkon	CSE			

Insertion problem solved.


DELETION PROBLEM SOLVED

STUDENTS TABLE			BRANCH TABLE		
rollno	name	branch	branch	hod	office_tel
			CSE	Mr. Y	53337


No Data

Deletion problem solved.

UPDATION PROBLEM SOLVED

STUDENTS TABLE			BRANCH TABLE		
rollno	name	branch	branch	hod	office_tel
1	Akon	CSE	CSE	Mr. Y	53337
2	Bkon	CSE	<div>  <p>Make changes here.</p> </div>		
3	Ckon	CSE			
4	Dkon	CSE			

Updation problem solved.

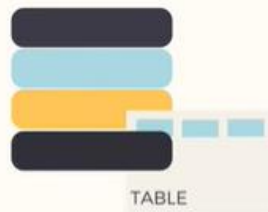


Normalisation is Good

**It follows,
Divide and Rule.**

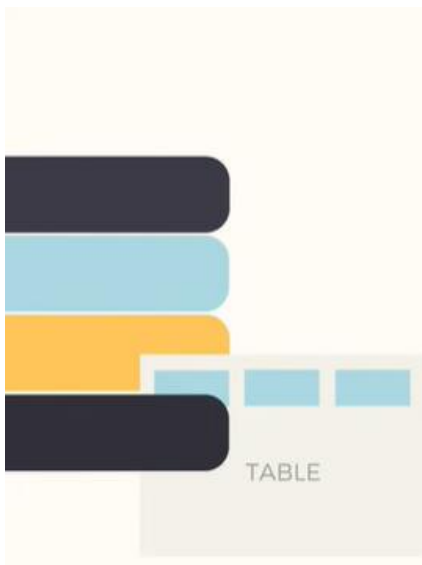
**Logical,
Independent, but
Related data.**

TYPES OF NORMALIZATION



Types of Normalization

1. 1st Normal Form
2. 2nd Normal Form
3. 3rd Normal Form



Basic concept of Normalization

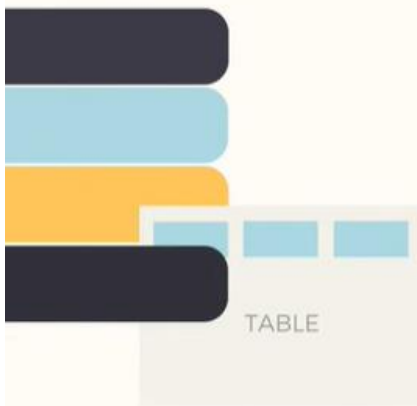
- What problems arise due to Data redundancy.
- How Normalization makes the Data more meaningful and useable.



1st Normal Form

DBMS NORMALIZATION

1st Normal Form



Step 1 of Normalisation process

- Scalable Table design which can be easily extended.
- If your table is not even in 1st Normal Form, it is considered poor DB design.

VERY BAD

1st Normal Form

Every Table in your Database should at least follow the 1st Normal Form, always.

or Stop using Database!

TABLE

Column 1	Column 2	
A	X, Y	
B	W, X	
C	Y	
D	Z	

RULE 1

- Each Column should contain atomic values.
- Entries like X, Y and W, X violate this rule.

TABLE

Column 1	Column 2	
A	X, Y	
B	W, X	
C	Y	
D	Z	

RULE 1

- Each Column should contain atomic values.
- Entries like X, Y and W, X violate this rule.

TABLE		
DOB	Name	
26-10-89	A	
13-2-92	SK	
16-11-65	SA	
R	8-9-86	

RULE 2

- A Column should contain values that are of the same type.
- Do not inter-mix different types of values in any column.

TABLE		
DOB	Name	
26-10-89	A	
13-2-92	SK	
16-11-65	SA	
R	8-9-86	



RULE 2

- A Column should contain values that are of the same type.
- Do not inter-mix different types of values in any column.

TABLE		
DOB	Name	Name
26-10-89	A	A
13-2-92	S	K
16-11-65	S	A
8-9-86	R	A

RULE 3

- Each column should have a unique name.
- Same names leads to confusion at the time of data retrieval



TABLE

DOB	Name	Name
26-10-89	A	A
13-2-92	S	K
16-11-65	S	A
8-9-86	R	A



RULE 3

- Each column should have a unique name.
- Same names leads to confusion at the time of data retrieval



TABLE

Roll_no	F_Name	L_Name
3	A	A
4	S	K
1	S	A
2	R	A

RULE 4

- Order in which data is saved doesn't matter.
- Using SQL query, you can easily fetch data in any order from a table.



Let's have an example.

Student Table

Roll no. Name Subject

STUDENTS TABLE		
rollno	name	subject



STUDENTS TABLE		
rollno	name	subject
101	Akon	OS, CN
103	Ckon	JAVA
102	Bkon	C, C++

VIOLATION OF ATOMICITY

STUDENTS TABLE		
rollno	name	subject
101	Akon	OS, CN
103	Ckon	JAVA
102	Bkon	C, C++

Violation of 1 NF

SOLOUTION

rollno	name	subject
101	Akon	OS  CN
103	Ckon	JAVA
102	Bkon	C  C++

NEW LOOK OF TABLE

rollno	name	subject
101	Akon	OS
101	Akon	CN
103	Ckon	JAVA
102	Bkon	C
102	Bkon	C++

Database Normalization With Examples

Database **Normalization Example** can be easily understood with the help of a case study. 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. Let's understand Normalization database with normalization example with solution:

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.

Here you see **Movies Rented** column has multiple values. Now let's move into 1st Normal Forms:

1NF (First Normal Form) Rules

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

The above table in 1NF-

1NF Example

1NF Example

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.

Example of 1NF in DBMS

Before we proceed let's understand a few things —

What is a KEY in SQL?

A **KEY in SQL** is a value used to identify records in a table uniquely. An SQL KEY is a single column or combination of multiple columns used to uniquely identify rows or tuples in the table. SQL Key is used to identify duplicate information, and it also helps establish a relationship between multiple tables in the database.

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

What is a Primary Key?

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

It has following attributes

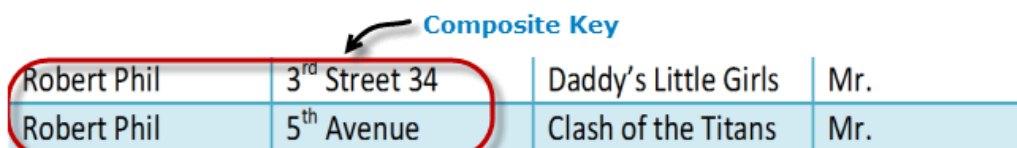
- A **primary key** cannot be NULL
- A primary key value must be unique
- The primary key values should rarely be changed
- The primary key must be given a value when a new record is inserted.

What is 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.

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.

Composite key in Database

Let's move into second normal form 2NF

2NF (Second Normal Form) Rules

- Rule 1- Be in 1NF
- Rule 2- Single Column Primary Key that does not functionally dependant on any subset of candidate key relation

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

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.

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

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

Database – Foreign Key

In Table 2, Membership_ID is the Foreign Key

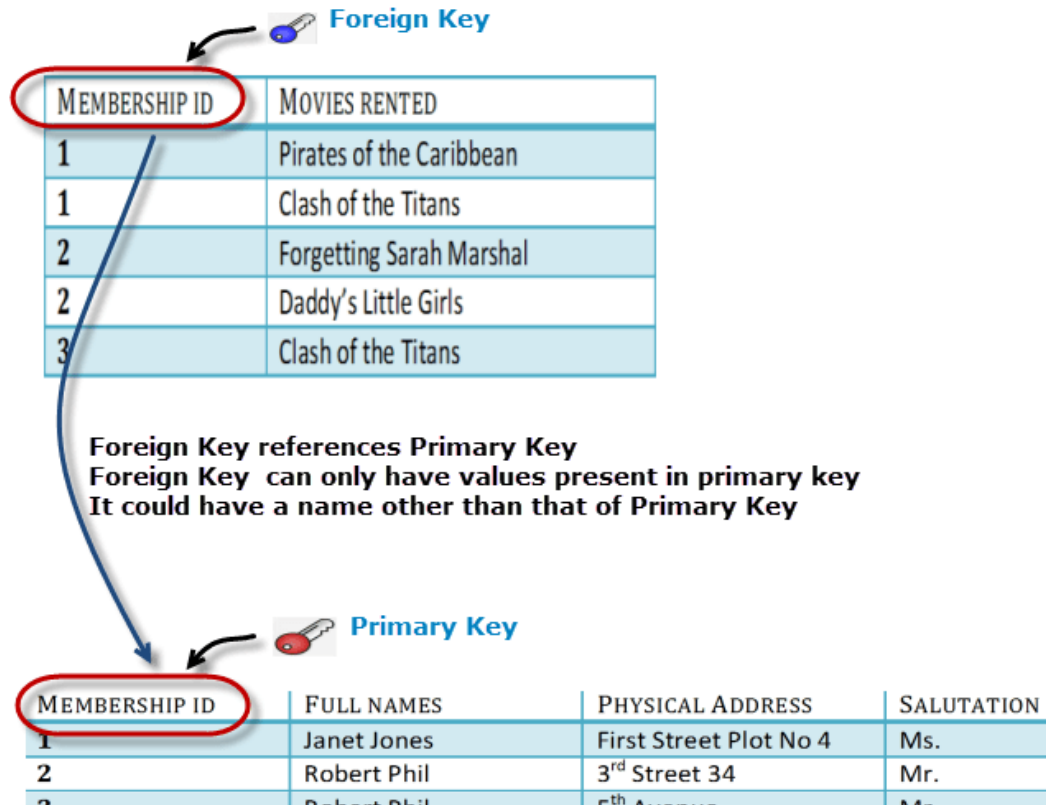
MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans



Foreign Key

Foreign Key references the 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 the 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?

Suppose, a novice inserts 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

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 are 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.

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.

Change in Name (under row 3, Full Names) → *May Change Salutation* (under row 3, Salutation)

Let's move into 3NF

3NF (Third Normal Form) Rules

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

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

3NF Example

Below is a 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

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

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

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

Now our little example is at a level that cannot further be decomposed to attain higher normal form types of normalization in DBMS. In fact, it is already in higher normalization forms. Separate efforts for moving into next levels of normalizing data are normally needed in complex databases. However, we will be discussing next levels of normalisation in DBMS in brief in the following.