Introduction 1 H Total 4H

- What is Data
- What is Database
- What is Database Management System?
- Relational Database Management System (RDBMS)
- DBMS Database Models
- File Systems vs DBMS (Why we shifted to DBMS)
- Introduction to File Management System
- Drawbacks for File Management System
- Need for DBMS
- RDBMS and DBMS

DBMS Architecture 20M

- 1-Tier Architecture
- 2-Tier Architecture
- 3-Tier Architecture

Building blocks of database 10M

- Columns (Fields, Attributes)
- Rows (Tuples, Record)
- Tables (Relation) :

DBMS Database Models 30M

- Centralized Database
- Hierarchical databases
- Network databases
- Relational databases
- Object-oriented databases
- NoSQL databases
- Personal Database
- Cloud Database
- Distributed Database
- Entity-relationship Model

ER Model ***30M***

- *Entities*
- *Attributes*
- Relationships

Key in database **30M**

why we need keys in dbms

- Primary Key
- Super Key
- Candidate Key
- Alternate Key
- Foreign Key
- Composite Key
- Unique Key

Normalization in DBMS **30M**

- First normal form (1NF)
- Second normal form (2NF)
- Third normal form (3NF)
- Boyce-Codd Normal Form(BCNF)
- Fourth normal form (4NF)
- Fifth normal form (5NF

ACID Properties **20M**

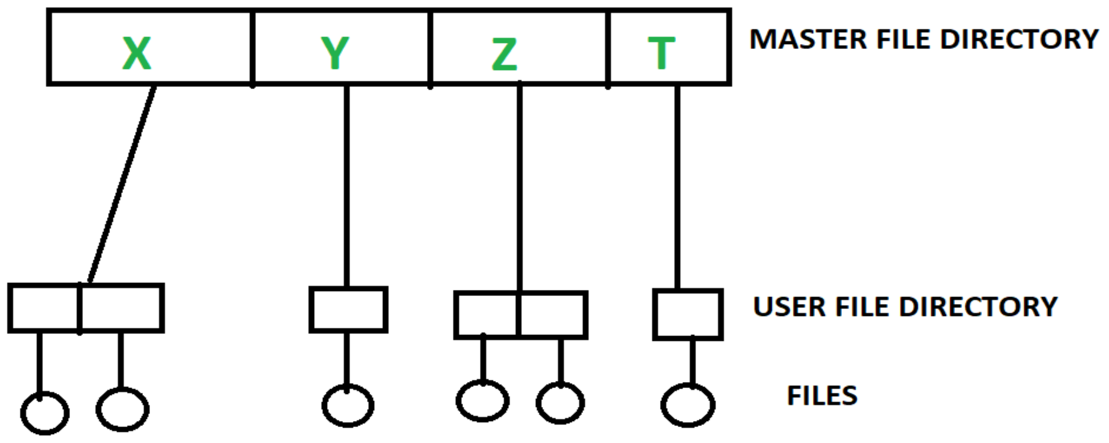
- Atomicity,
- Consistency,
- Isolation
- Durability.

Interview Questions

1-Paper based   before 1950

There are the following differences between DBMS and File systems:

File System: 1990
A file system is a technique arranging the files in a storage medium like a hard disk. pen drive, DVD CD, The file system organizes the files and helps in the retrieval of files when they are required. File systems consist of different files which are grouped into directories. The directories further contain other folders and files. The file system performs basic operations like management, file naming, giving access rules, etc. different file types, such as mp3, doc, txt, mp4, etc
Example: NTFS (New Technology File System), EXT (Extended File System).



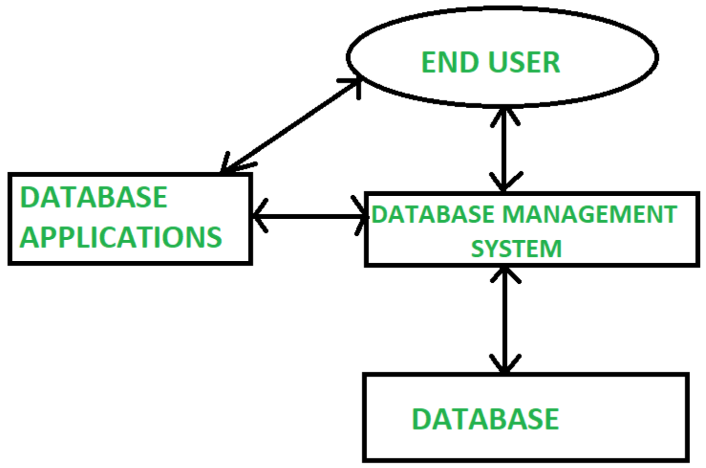
Problems

- duplicated in more than one file
- Data redundancy.
- It isn't easy to protect a file under the file system.
- Size of files KB MB GB TB..

DBMS (Database Management System) :

Database Management System is basically **software** that manages the collection of related data. It is used for **storing** data and **retrieving** the data effectively when it is needed. It also provides proper **security** measures for **protecting** the data from **unauthorized access**. In Database Management System the data can be fetched by SQL queries and relational algebra. It also provides mechanisms for data **recovery** and **data backup.**

Example:
Oracle, MySQL, MS SQL server.



DATA

- **Data** is a collection raw material and unstructured fact figure and unorganized it can be anything
- Data is raw material unprocessed unorganized unstructured fact it can be anything
- Data is a collection of information

Example–market data research or analysis., new papers employee name, Product name, Name of the student, Marks of the student, Mobile number, Image Sound, Video, Single character, Number (integer or floating-point), Picture, Boolean (true or false), Text (string) etc.

Types of data

Structured Data Structured data is typically stored in tabular form and managed in a relational database (RDBMS).table

Unstructured Data- Unstructured data includes various content such as documents, videos, audio files, posts on social media, and emails. research or analysis., new papers employee Product Word, PDF, Text, Media logs.

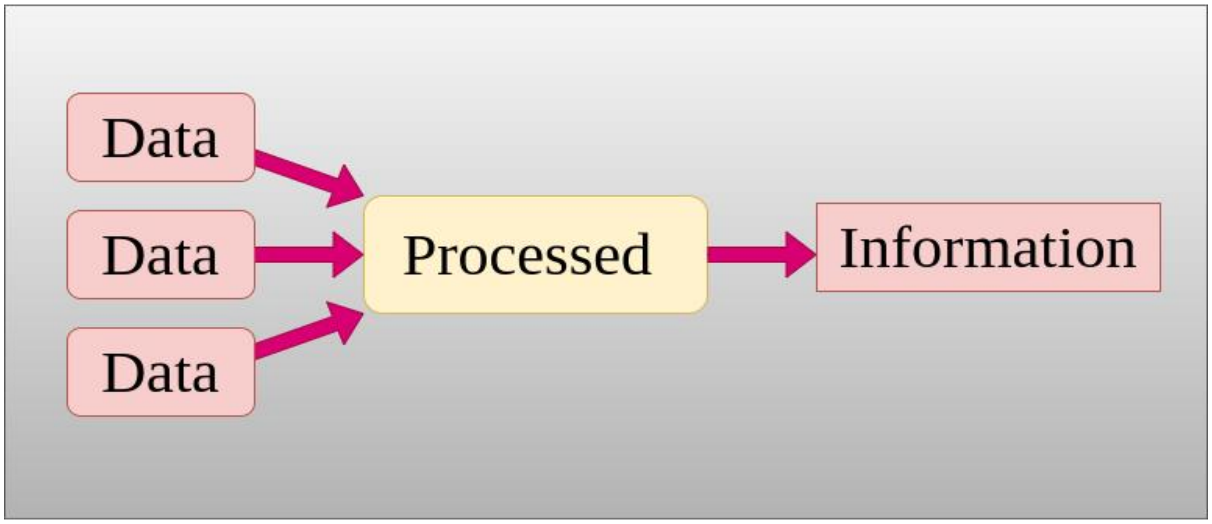
Information

- Information is collection of data which is processed in a meaningful Representation called information
- Information is the data that has been converted into more useful or intelligent form.
- Information is a processed form of data

Example: Report card sheet.

The information is needed for the following reasons –

- To gain knowledge about the surroundings.
- To keep the system up to date.
- To know about the rules and regulations of the society.

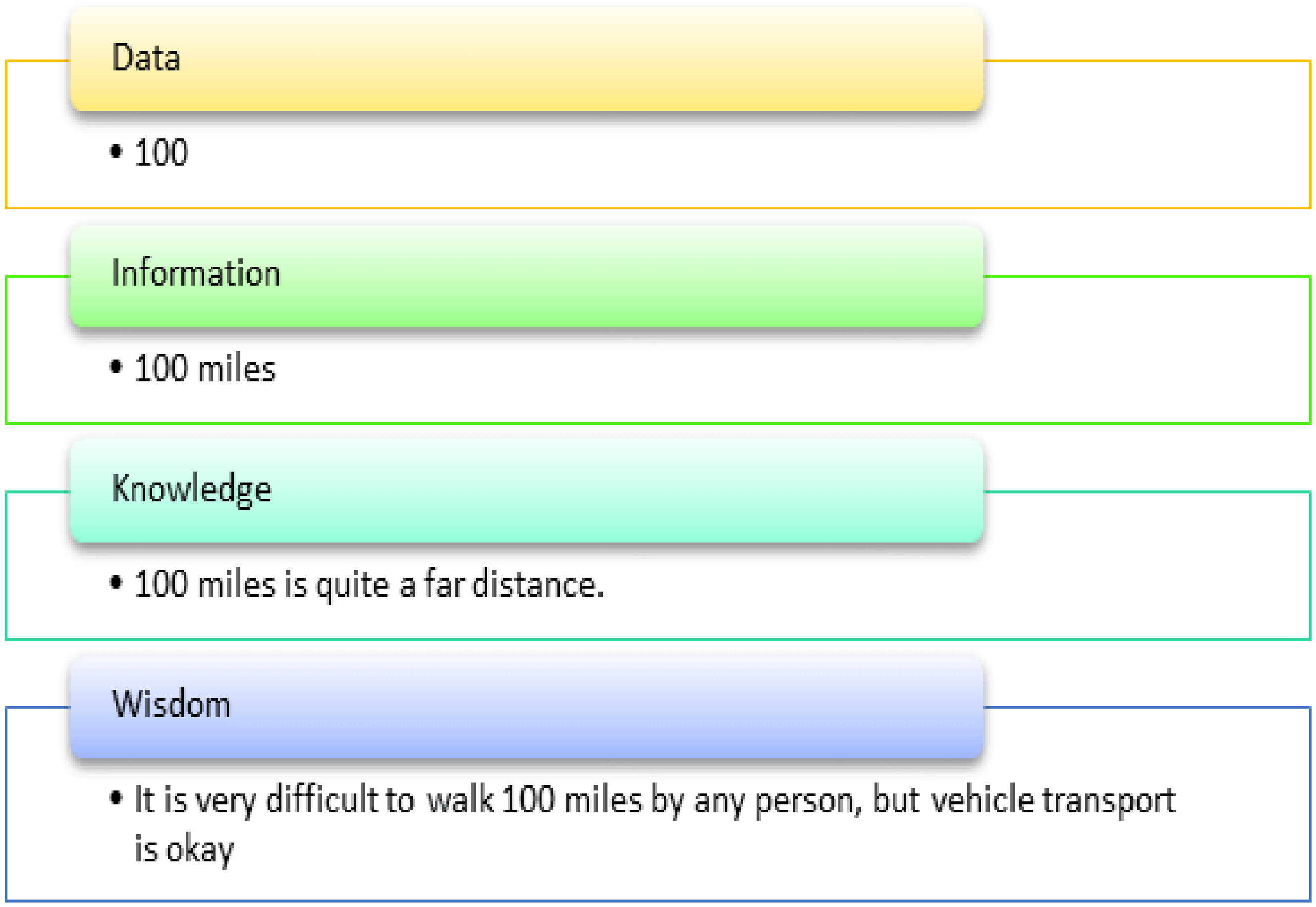


Knowledge

The human mind organizes meaningful information and evaluates it to produce knowledge.

Example of data, information and knowledge

- A student secures **450** marks. Here **450 is data,** marks of the student **is the information** and **hard work** required to get the marks is **knowledge**.


Data
• 100
Information
• 100 miles
Knowledge
• 100 miles is quite a far distance.
Wisdom
• It is very difficult to walk 100 miles by any person, but vehicle transport is okay

Database

- A **database** is an organized collection of structured data stored electronically in a computer system can be easily accessed and managed data into the form of tables, rows, columns, and index it to make it easier to find relevant information.

- **database** is collection of tables in the form of, rows, and columns, with index

- Databases are used to store a large number of dynamic websites on the Internet World Wide Webtoday.

telephone directory	customer's data
Visitor Register Book	university records
shopping data	IRCTC ticket booking

Note:

The **main purpose** of the database is to operate a large amount of information by storing, retrieving, and managing data.

Databases software like MySQL, Sybase, Oracle, MongoDB, Informix, PostgreSQL, SQL Server, etc.

Building blocks of database

1- Tables (Relation) : Table is contains columns & rows and each row has value.

2- Columns (Fields, Attributes) : Column is specific type of attributes contains one or more rows.

3- Rows (Tuples, Record) : Each columns has one or more rows and rows contains data (values).

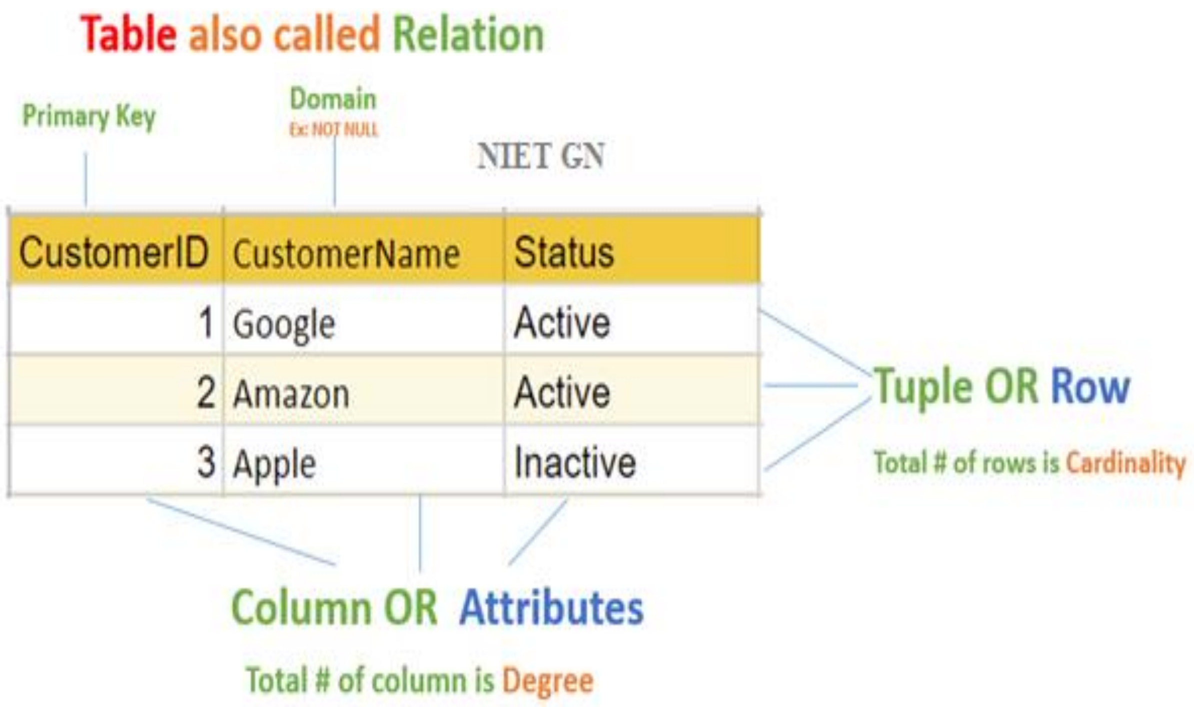
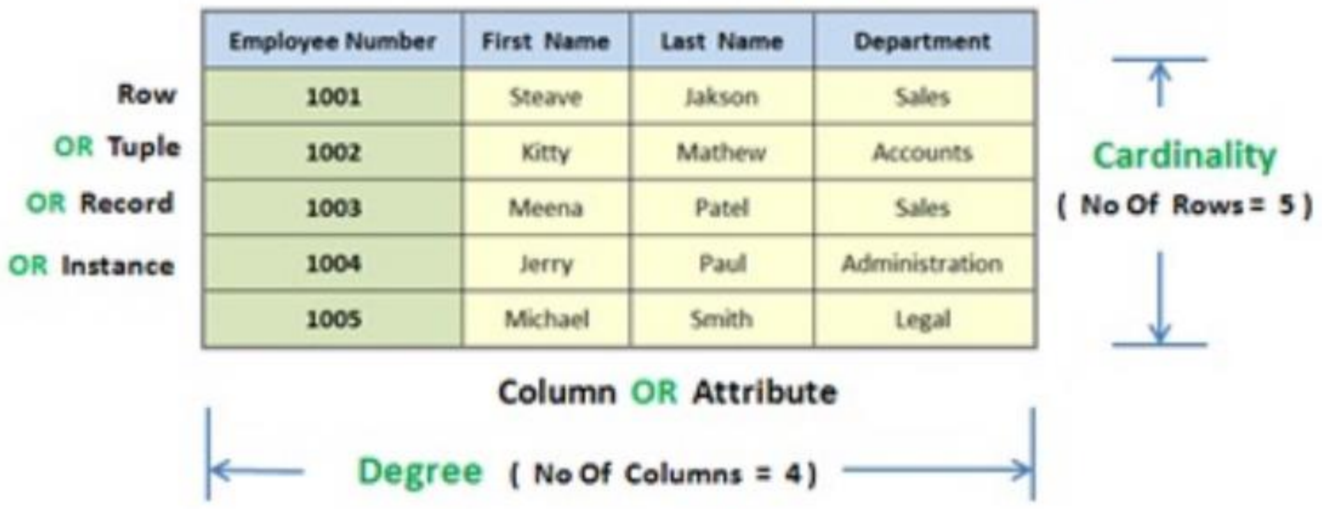


Table also called Relation
Primary Key
Domain
Ex: NOT NULL
NIET GN
Tuple OR Row
Total # of rows is Cardinality
Column OR Attributes
Total # of column is Degree

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

Table - Employee

A Table is also referred as Entity Type OR Entity Class



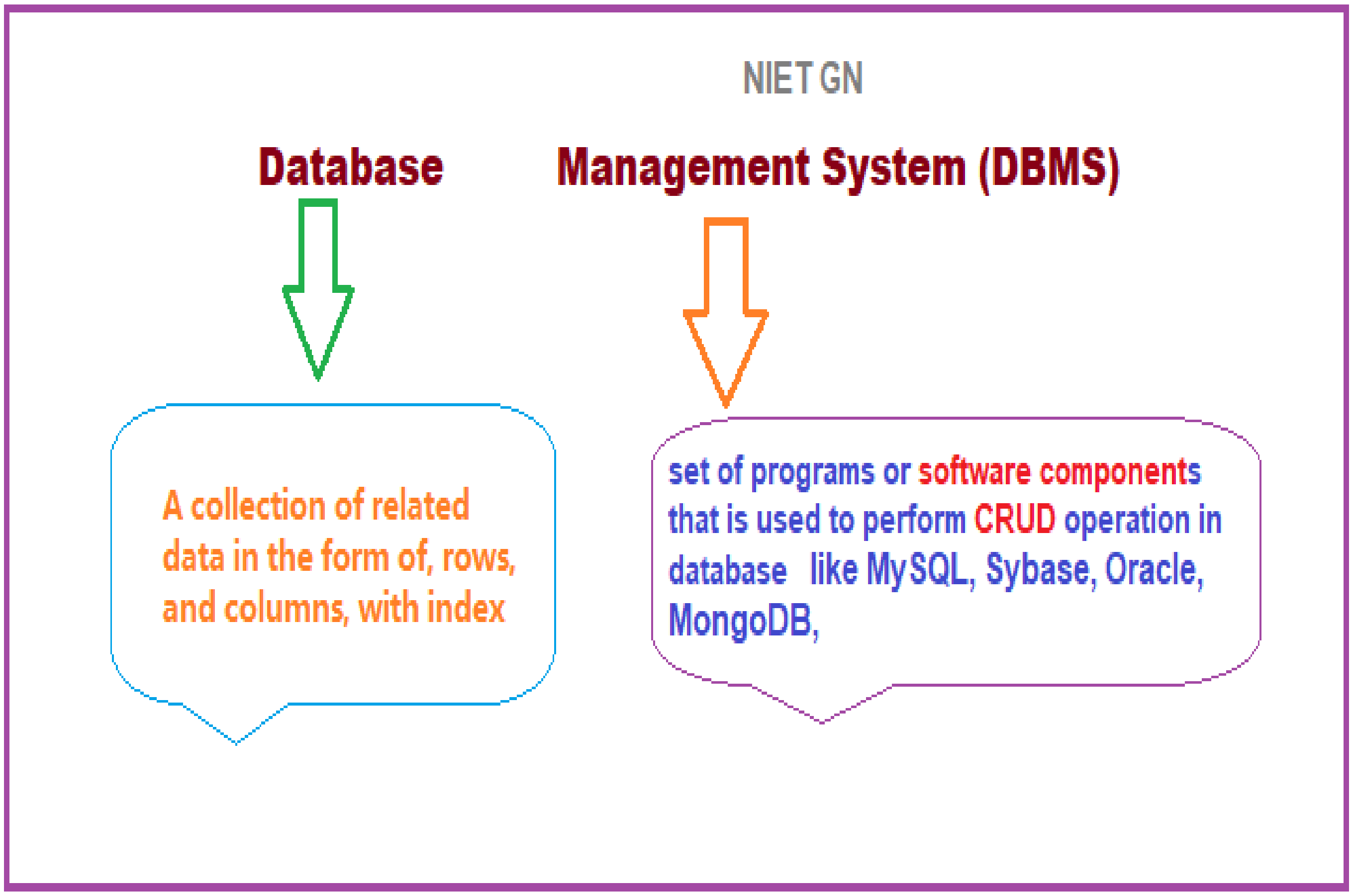
Employee Number	First Name	Last Name	Department
1001	Steave	Jakson	Sales
1002	Kitty	Mathew	Accounts
1003	Meena	Patel	Sales
1004	Jerry	Paul	Administration
1005	Michael	Smith	Legal

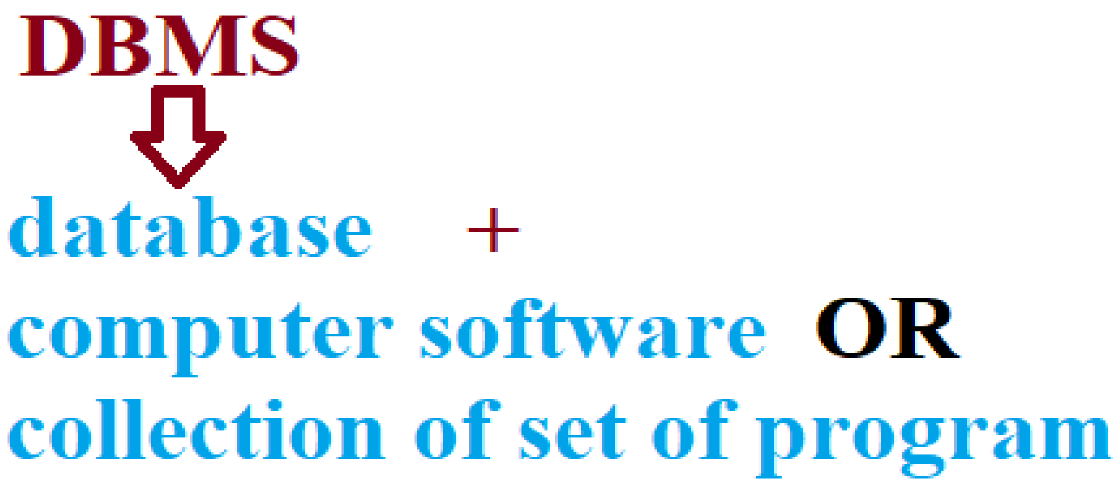
Each Row in a Employee Table is Record for Employee

Each Row in a Employee Table is also referred as Instance of Employee Type (Employee Class)

Employee Table = { Set Of Employee Records }

DBMS DATABASE MANAGEMENT SYSTEM


NIET GN
Database
Management System (DBMS)
A collection of related data in the form of, rows, and columns, with index
set of programs or software components that is used to perform CRUD operation in database like MySQL, Sybase, Oracle, MongoDB,



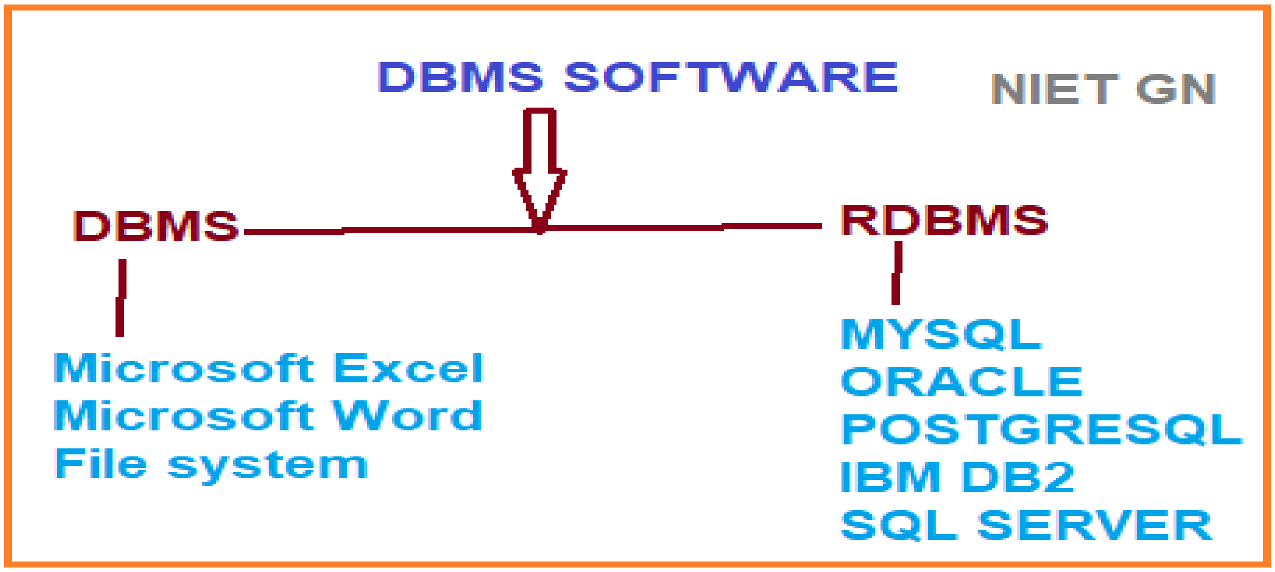
- **DBMS Database management system.** Is collection of programs / system software which helps users for managing the data in the databases.

- It provides an interface to end-users for **Creating inserting**, **updating**, **deleting accessing, sharing** the data in the database among various users and application

- **DBMS** provides security and protection to the data in the databases. If there is more than one uses the same database, then the system also maintains the consistency of the data.

Popular DBMS Software

- **DBMS are**: Oracle, MS SQL Server, SQLite, dBase, MySQL, IBM DB2, Microsoft Access, PostgreSQLetc.



WHY USE DBMS

- To develop software applications in less time.
- Creation of a database.
- Retrieval of information from the database.
- Updating the database.
- Managing a database.
- For data integrity and security.
- For concurrent access to data, recovery and backup from crashes.

- To use user-friendly declarative query language.

Applications of Database Management System

- Banking:
- Universities:
- Airlines:
- Railways:
- Sales:
- Manufacturing:
- Telecommunication:

Characteristics of DBMS

- DBMS should be able to store any kind of data in a database.
- Support ACID properties (atomicity, consistency, isolation, durability).
- One or more users to access the database at the same time.
- DBMS Provide the Backup and recovery for the future use
- Allow users to protect and security their data from damage or loss.

What is RDBMS

RDBMS stands for **Relational Database Management Systems**. It is basically a program that allows us to create, delete, and update and stores retrieves data in a tabular form of rows and columns a relational database.

It is a smaller subset of DBMS which was designed by E.F Codd in the 1970s. The major DBMS

Example SQL Server, Oracle, MySQL, MariaDB, and SQLite. are all based on the principles of relational DBMS.

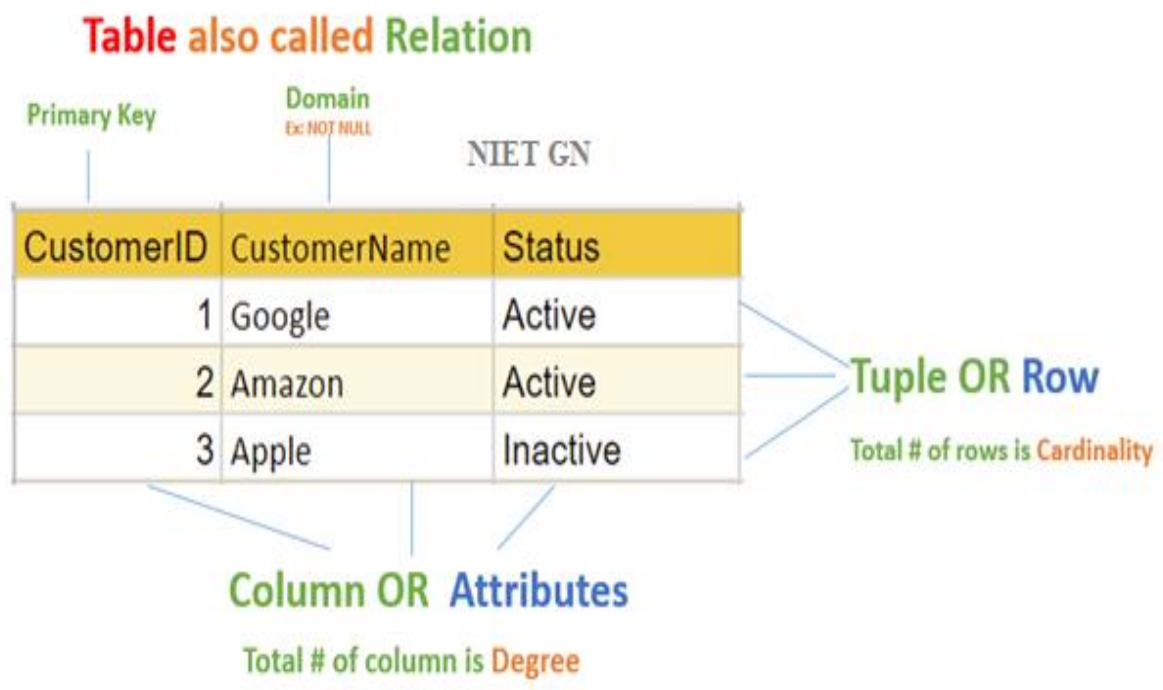


Table also called Relation
Primary Key
Domain
Ex: NOT NULL
NIET GN
Tuple OR Row
Total # of rows is Cardinality
Column OR Attributes
Total # of column is Degree

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

RDBMS - Relational Database Model - Table (DB Entity)

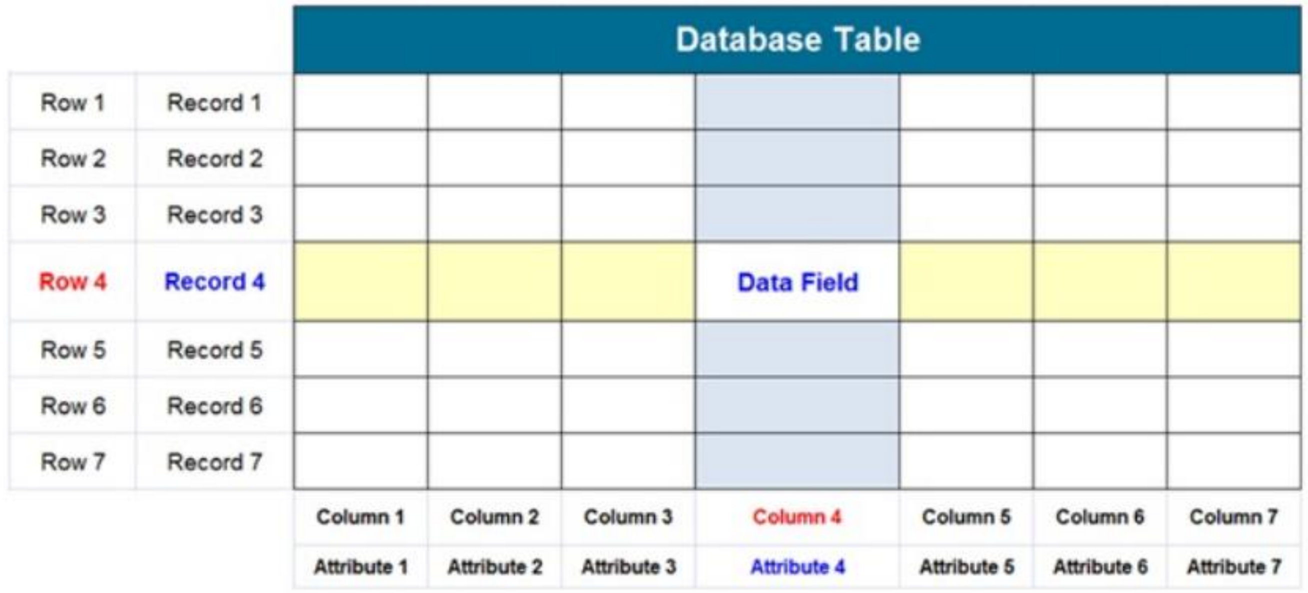


Table Column ---> Attribute , Table Row - Record , Tuple

Intersection Of Table Row And Column ---> Data Field

Best Practices for creating a Relational Model

- Data must be stored in tabular form in DB file, organized in the form of rows and columns.

- Each row of table is called record/tuple. Collection of such records is known as the cardinality of the table
- Each column of the table is called an attribute/field. Collection of such columns is called the Degree of the table.
- No two records of the DB table can be same. to identify each record uniquely.
- Cells of the table should hold a single value
- Each column should be given a unique name
- The data are always saved in rows and columns
- To retrieve the information, the indexes are used
- Database tables also allow NULL values, table are not filled or are missing, any values it becomes a NULL value, which is not equivalent to zero. (NOTE: Primary key cannot have a NULL value).

Advantages of RDBMS

- Easy to manage: Each table can be independently manipulated without affecting others.
- Security: It is more secure consisting of multiple levels of security. Access of data shared can be limited.
- Flexible: Updating of data can be done at a single point
- Users: RDBMS supports client-side architecture storing multiple users together.
- Facilitates storage and retrieval of large amount of data.
- Easy Data Handling:
- Data fetching is faster because of relational architecture.
- Data redundancy or duplicity is avoided due to keys, indexes, and normalization principles.
- Data consistency is ensured because RDBMS is based on ACID properties for data transactions (Atomicity Consistency Isolation Durability).

Difference between RDBMS and DBMS

Database Management System (DBMS) is a software that is used to define, create and maintain a database and provides controlled access to the data.

Relational Database Management System (RDBMS) is an advanced version of a DBMS.

No.	**DBMS**	**RDBMS**
1)	DBMS applications store **data as file**.	RDBMS applications store **data in a tabular form**.
2)	In DBMS, data is generally stored in either a hierarchical form or a navigational form.	In RDBMS, the tables have an identifier called primary key and the data values are stored in the form of tables.
3)	**Normalization is not** present in DBMS.	**Normalization is** present in RDBMS.
4)	DBMS does **not apply any security** with regards to data manipulation.	RDBMS **defines the integrity constraint** for the purpose of ACID (Atomocity, Consistency, Isolation and Durability) property.
5)	DBMS uses file system to store data, so there will be **no relation between the tables**.	RDBMS, data values are stored in the form of tables, so a **relationship** between these data values will be stored in the form of a table as well.
6)	DBMS has to provide some uniform methods to access the stored information.	RDBMS system supports a tabular structure of the data and a relationship between them to access the stored information.
7)	DBMS **does not support distributed database**.	RDBMS **supports distributed database**.

8)	DBMS is meant to be for small organization and **deal with small data**. it supports **single user**.	RDBMS is designed to **handle large amount of data**. it supports **multiple users**.
9)	Examples of DBMS are file systems, **xml** etc.	Example of RDBMS are **mysql**, **postgre**, **sql server**, **oracle** etc.

DBMS Architecture

A **Database Architecture** is a representation of DBMS design. It helps to design, develop, implement, and maintain the database management system

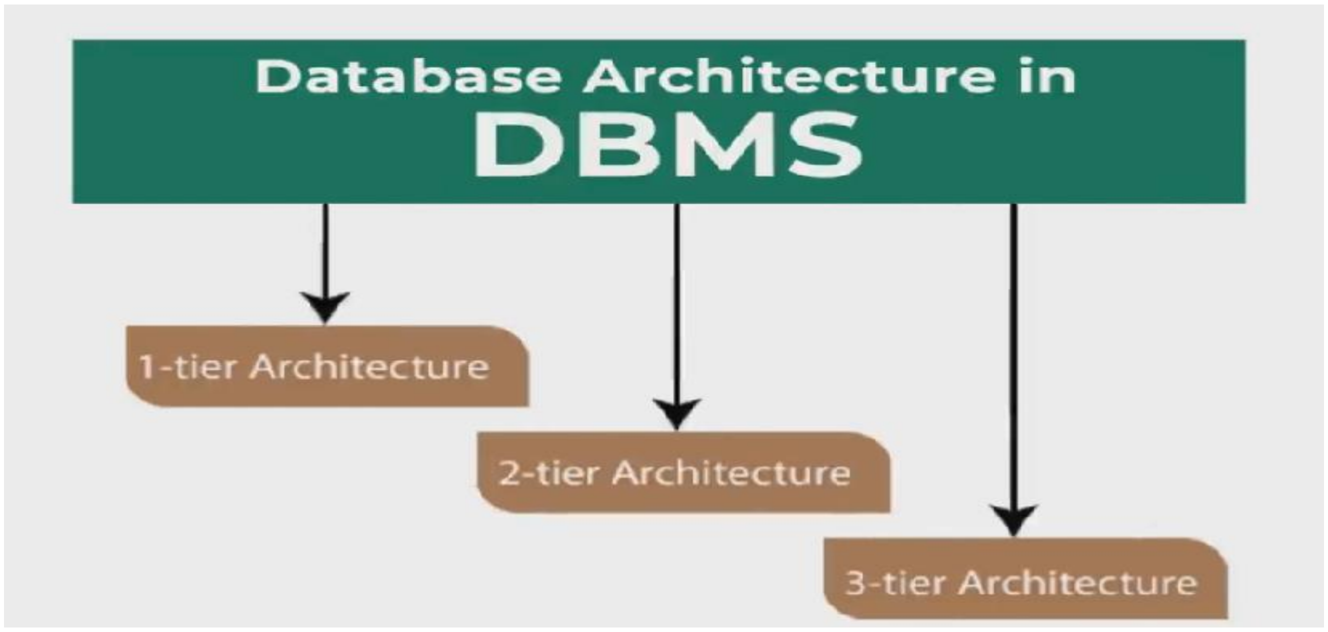
The DBMS design its architecture. The basic client/server architecture is used to deal with a large number of PCs, web servers, database servers and other components that are connected with networks.

The client/server architecture consists of many PCs and a workstation which are connected via the network.

Types of DBMS Architecture

1. Single tier architecture
2. Two tier architecture
3. Three tier architecture

Types of DBMS Architecture



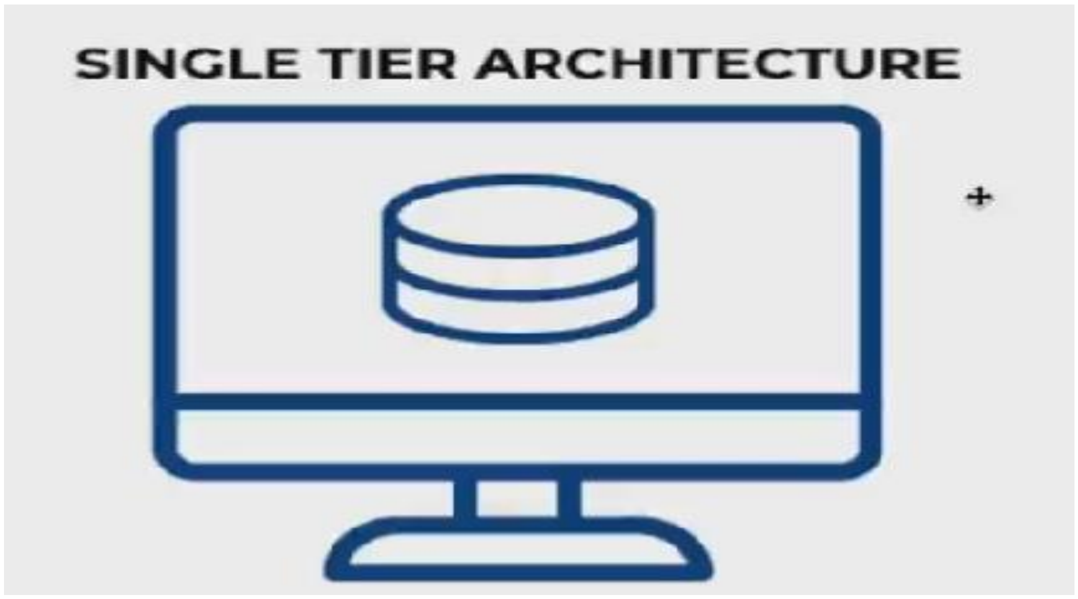
Database architecture can be seen as a single tier or multi-tier. But logically, database architecture is of two types like:

2-tier architecture and **3-tier architecture**.

1-Tier Architecture

- In this architecture, the database is directly available to the user. It means the user can directly sit on the DBMS and uses it.
- Any changes done here will directly be done on the database itself. It doesn't provide a handy tool for end users.
- The 1-Tier architecture is used for development of the local application, where programmers can directly communicate with the database for the quick response.

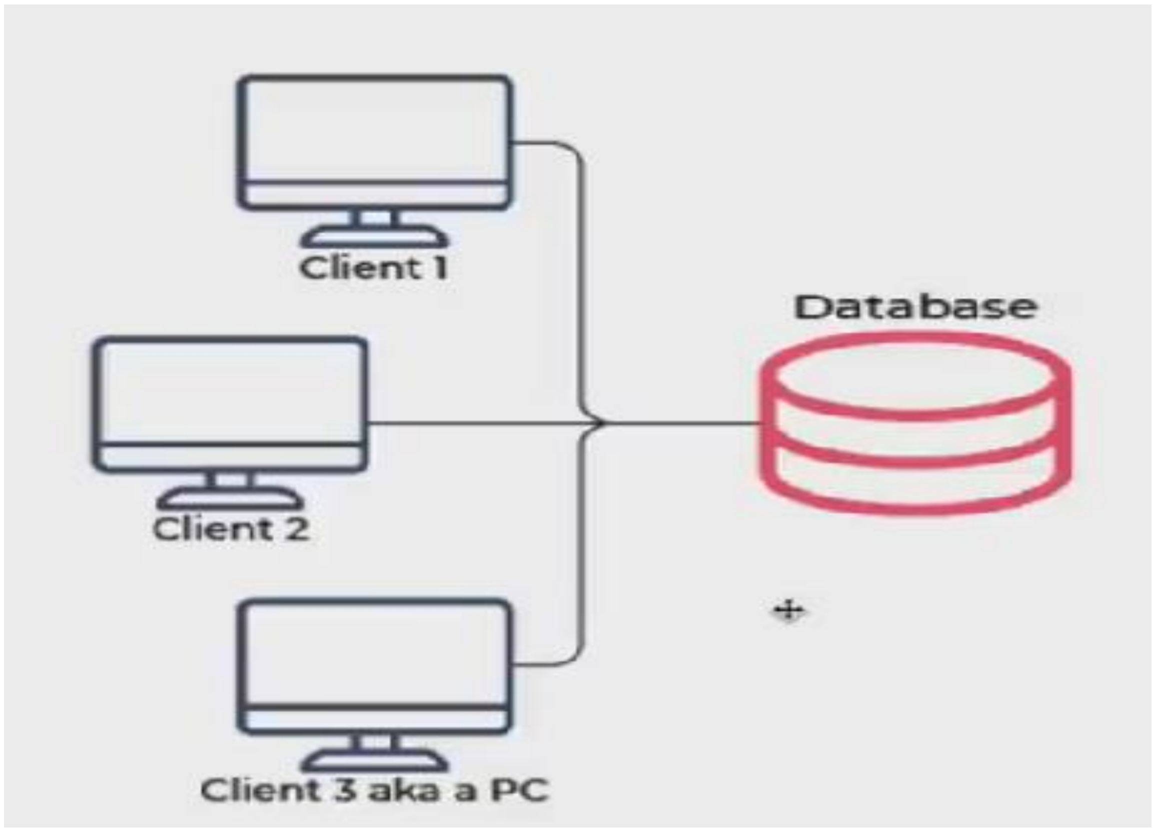
  Example   Normal RDBM soft in pc Oracle, MySQL study of student database



2-Tier Architecture

- The 2-Tier architecture is same as basic client-server. In the two-tier architecture, applications on the client end can directly communicate with the database at the server side. For this interaction, API's like: **ODBC**, **JDBC** are used.
- The user interfaces and application programs are run on the client-side.
- To communicate with the DBMS, client-side application establishes a connection with the server side.

JSE-**Standalone application** AWT Swing **JDBC Shopping Mall Billing,**



3-Tier Architecture

- The 3-Tier architecture contains an other layer between the client and server. In this architecture, client can't directly communicate with the server.
- The application on the client-end interacts with an application server which further communicates with the database system.
- End user has no idea about the existence of the database beyond the application server. The database also has no idea about any other user beyond the application.
- The 3-Tier architecture is used in case of large web application.

J2EE-WEB application on the server internet www Enterprise Application IRCTC

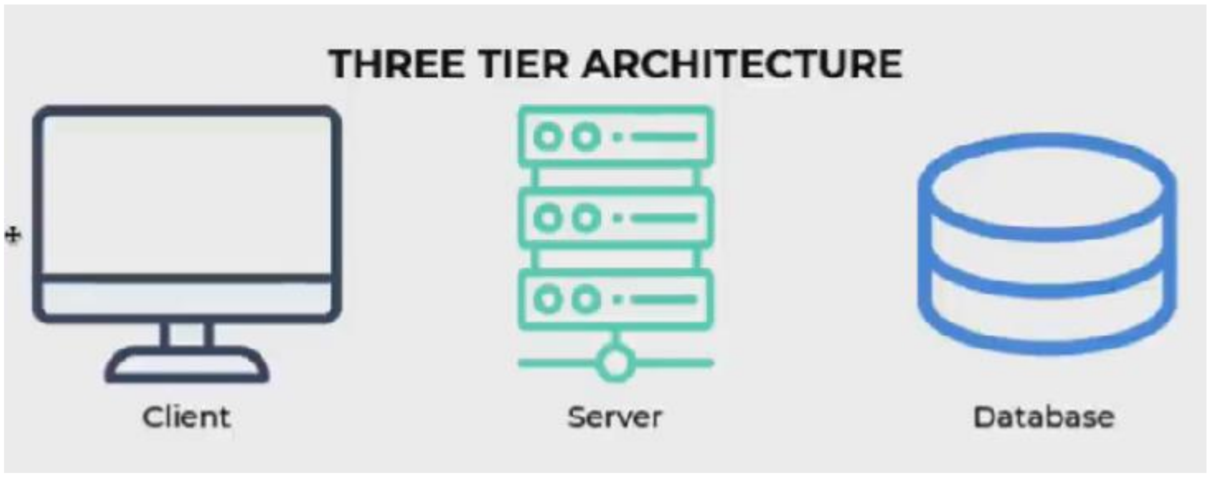
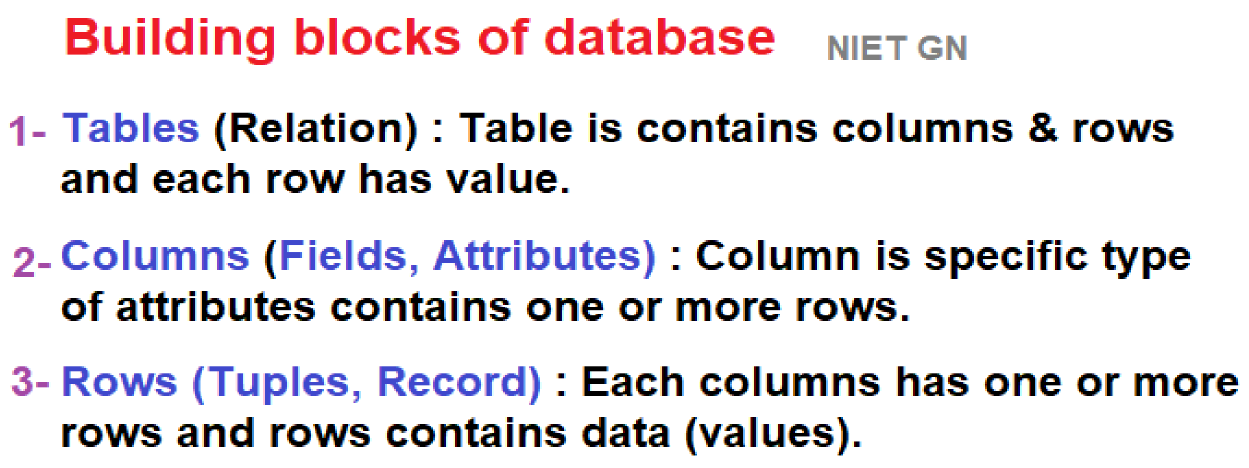


Table in database



Building blocks of database NIET GN

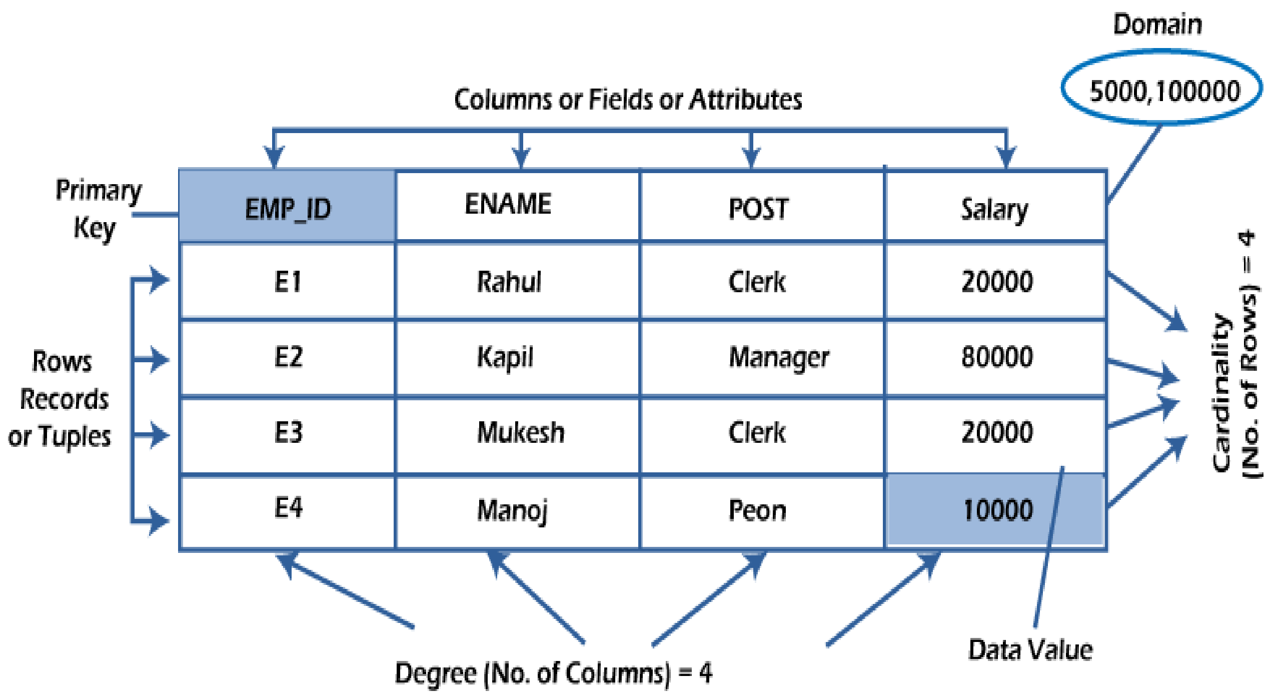
1- Tables (Relation) : Table is contains columns & rows and each row has value.

2- Columns (Fields, Attributes) : Column is specific type of attributes contains one or more rows.

3- Rows (Tuples, Record) : Each columns has one or more rows and rows contains data (values).

4 Degree: The total number of **Attributes (column) Fields** is called the degree.

5 Cardinality: Total number of **Tuple** (rows) Records


Domain
5000,100000
Columns or Fields or Attributes
Primary Key
Rows Records or Tuples
Cardinality (No. of Rows) = 4
Degree (No. of Columns) = 4
Data Value

EMP_ID	ENAME	POST	Salary
E1	Rahul	Clerk	20000
E2	Kapil	Manager	80000
E3	Mukesh	Clerk	20000
E4	Manoj	Peon	10000

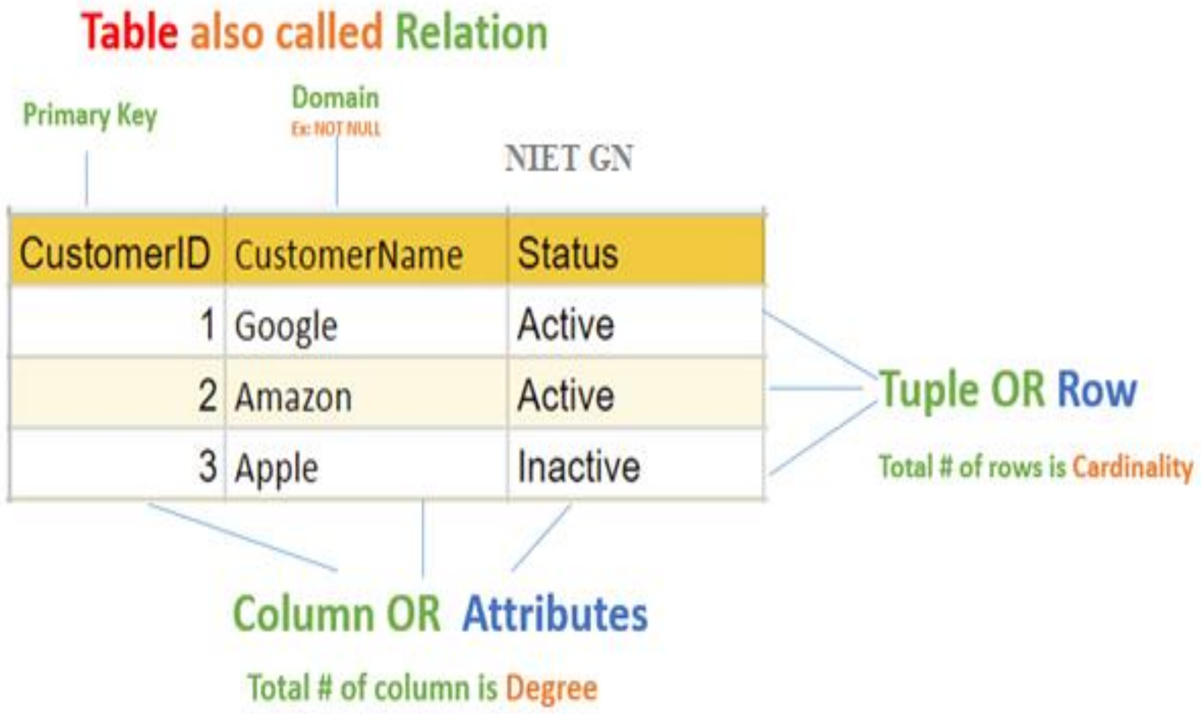


Table also called Relation
Primary Key
Domain
Ex: NOT NULL
NIET GN
Tuple OR Row
Total # of rows is Cardinality
Column OR Attributes
Total # of column is Degree

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

1 **A table has two properties rows and columns. Rows represent records and columns represent attributes**.

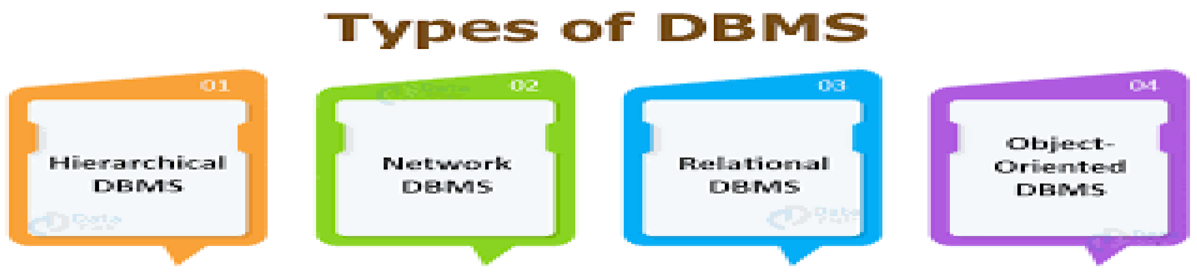
1. **Attribute:** Each column in a Table. Attributes are the properties which define a relation. e.g., Student_Rollno, NAME, etc.

2. **Tuple** – It is nothing but a single row of a table, which contains a single record.

DBMS Database Models

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. While the **Relational Model** is the most widely used database model,

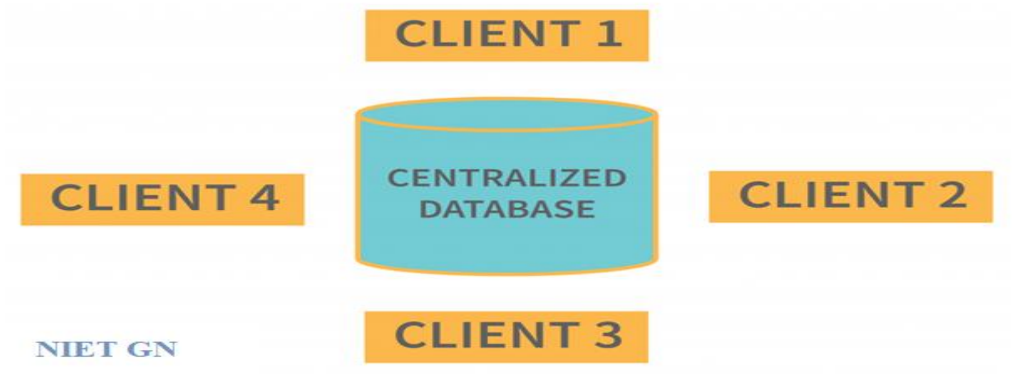
Types of Databases Models in DBMS



- Centralized Database
- Hierarchical databases
- Network databases
- Relational databases
- Object-oriented databases
- NoSQL databases
- Personal Database
- Cloud Database
- Distributed Database
- Entity-relationship Model

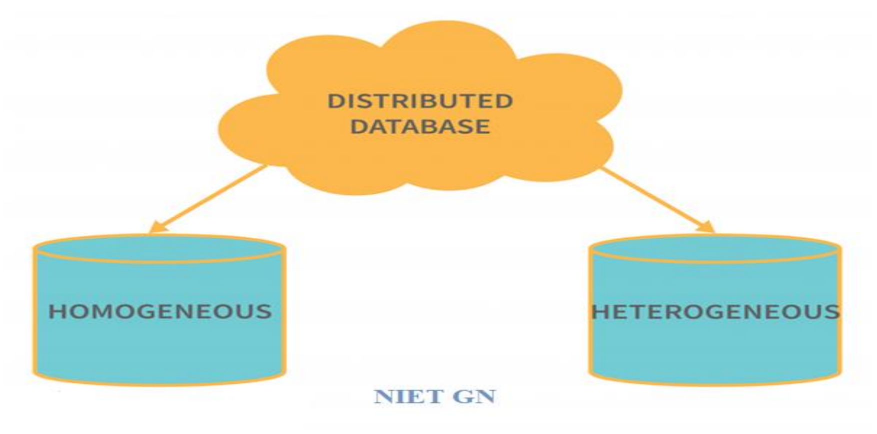
Centralized Database

A centralized database is kept, managed, and accessed from a single place. This type of database is operated, modified, and managed from that location itself. You might ask, what is this location? This location is any database system or a centralized computer system. An internet connection is required to reach the centralized location (LAN, WAN, etc.). The centralized database is mainly used by institutions or organizations to maintain the data related to their operations. An example of a centralized database is a primary library that carries a central database of each libraryin a college/university



Distributed Database

distributed database is a type of database that consists of multiple databases that are connected and are spread across different physical locations. Data stored in many physical places can be managed independently of one another. A computer network is used to communicate between the many physical sites.. **Examples** of the Distributed database are Apache **Cassandra, HBase, Ignite,** etc.



- **Homogeneous DDB:** Those database systems which execute on the same operating system and use the same application process and carry the same hardware devices.
- **Heterogeneous DDB:** Those database systems which execute on different operating systems under different application procedures, and carries different hardware devices

Relational Database table

This database is based on the relational data model, which stores data in the table and form of rows(tuple) and columns(attributes),

A relational database uses SQL for storing, manipulating,

Each table in the database carries a key that makes the data unique from others.

Examples of Relational databases are **MySQL, Microsoft SQL Server, Oracle,** etc.

NoSQL Database

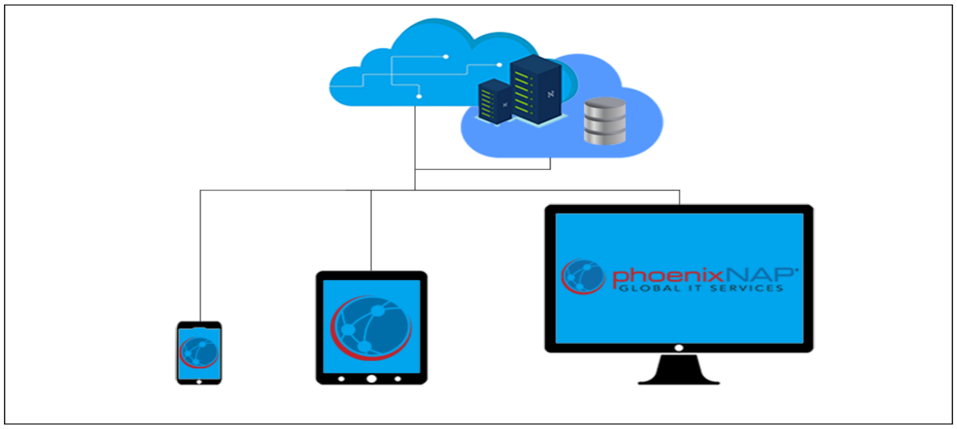
Non-SQL/Not Only SQL is a type of database that is used for storing a wide range of data sets. It is not a relational database as it stores data not only in tabular form but in several different ways NoSQL presented a wide variety of database technologies in response to the demands. We can further divide a NoSQL database

1. Cosmos DB
2. ArangoDB
3. Couchbase Server
4. CouchDB
5. Amazon DocumentDB
6. MongoDB, CouchBase

Cloud Database

A type of database where data is stored in a virtual environment and executes over the cloud computing platform. It provides users with various cloud computing services (SaaS, PaaS, IaaS, etc.) for accessing the database. There are numerous cloud platforms, but the best options are:

- Amazon Web Services (AWS)
- Microsoft Azure
- Oracle Autonomous Database
- Google Cloud SQL etc.

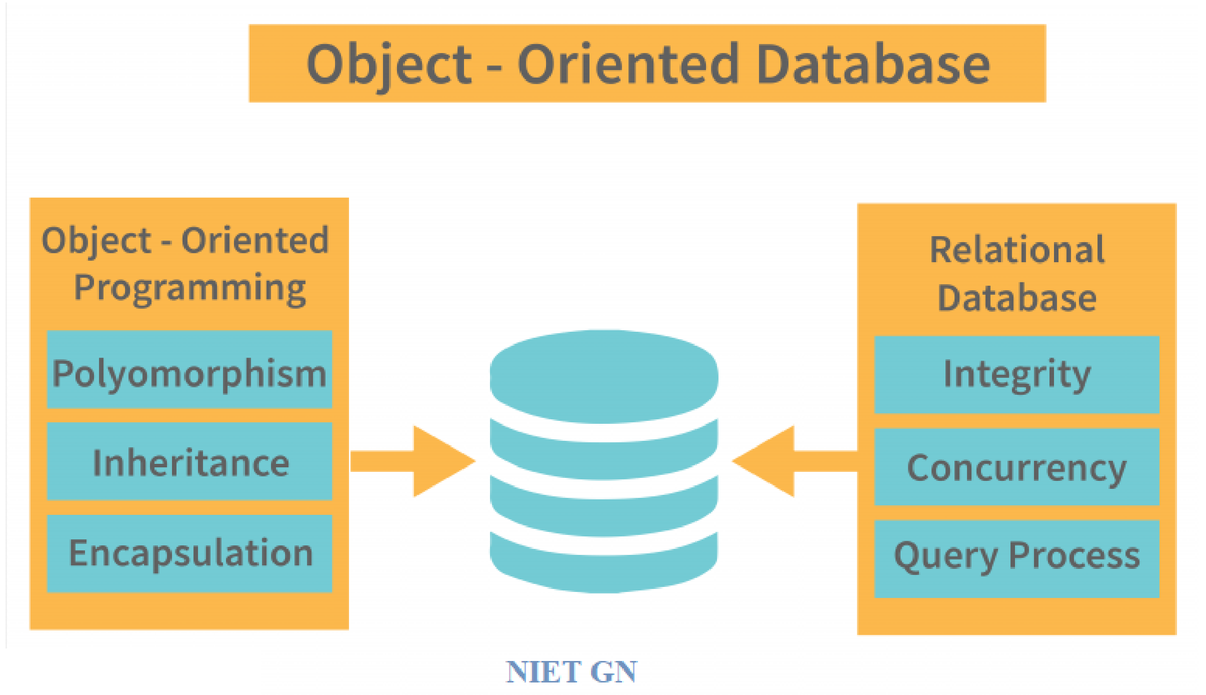


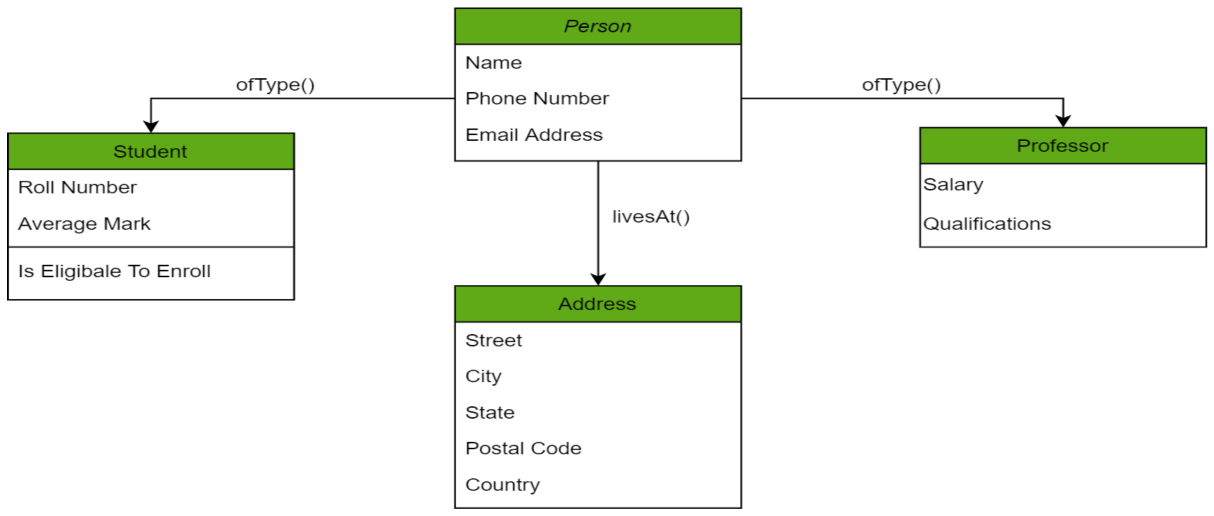
Object-oriented Databases

The type of database that uses the object-based data model approach for storing data in the database system. such as java and C++, can be saved in a relational database using object-oriented programming languages,

Using class is Logical Entity blueprint of the Objects

Object is real word Entity it has physically Existence The data is represented and stored as objects used in the object-oriented programming language, **PostgreSQL**

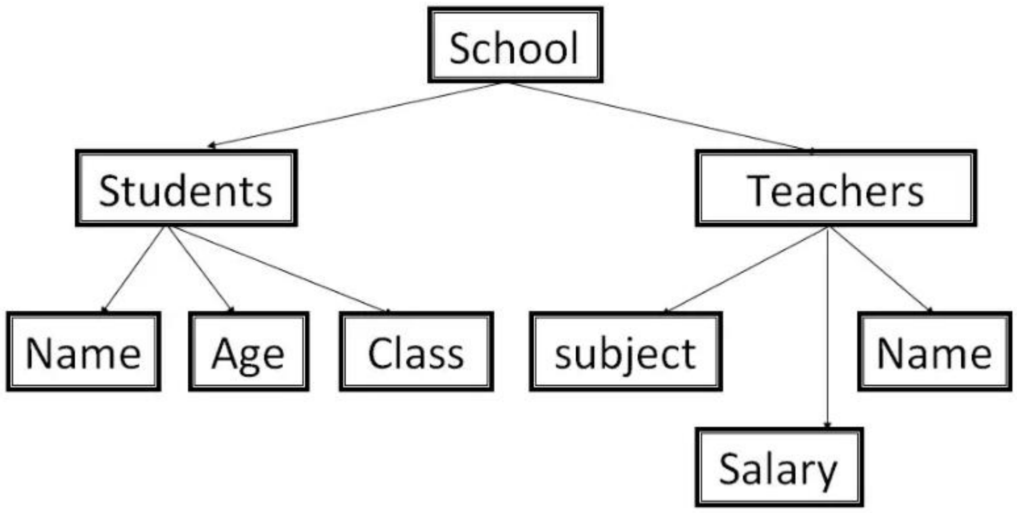




Hierarchical Databases

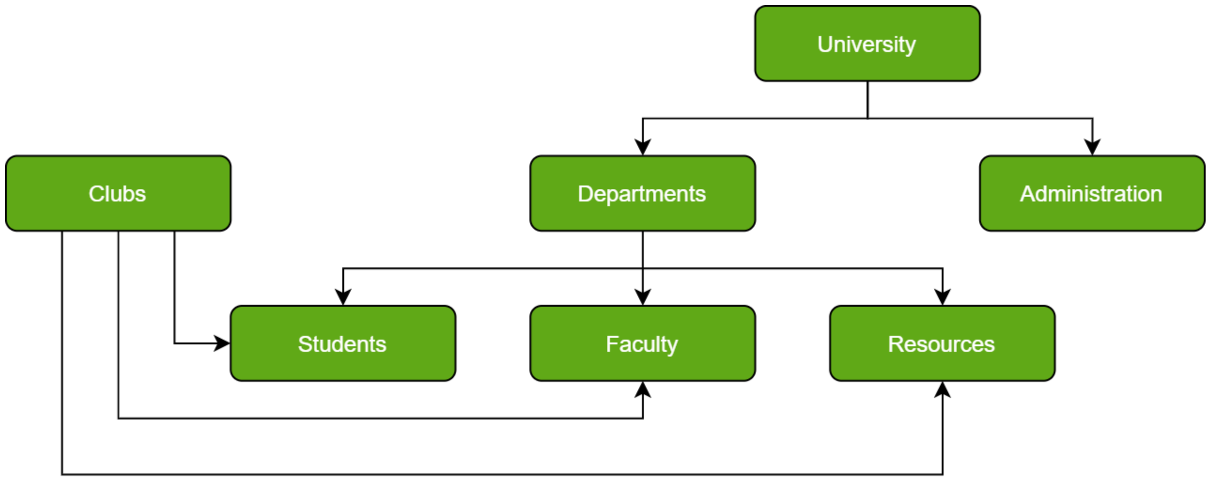
It is the type of database that stores data in the form of parent-children relationship nodes. Here, it organizes data in a tree-like structure. Data get stored in the form of records that are connected via links. Each child record in the tree will contain only one parent. On the other hand, each parent record can have multiple child records.

Hierarchical Database Model



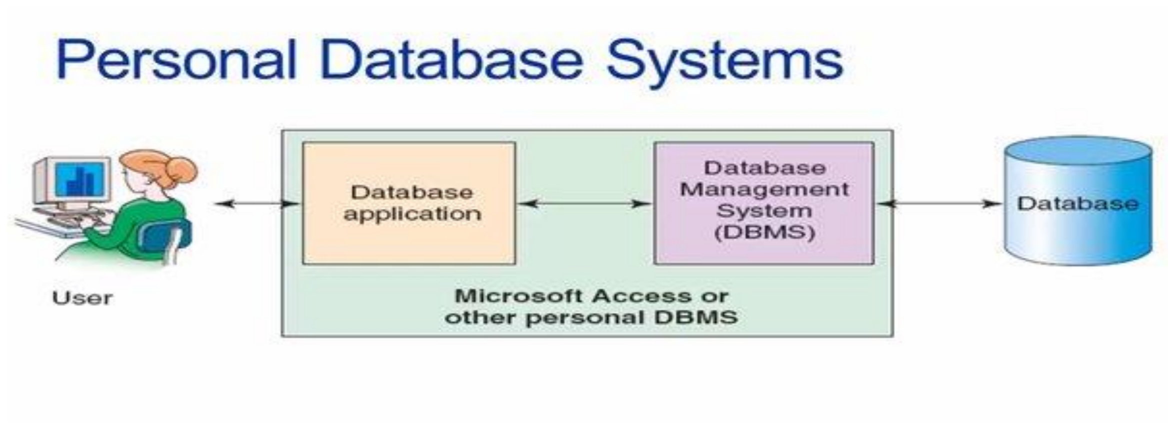
Network Databases

It is the database that typically follows the network data model. Here, the representation of data is in the form of nodes connected via links between them. Unlike the hierarchical database, it allows each record to have multiple children and parent nodes to form a generalized graph structure.



Personal Database

Collecting and storing data on the user's system defines a Personal Database. This database is basically designed for a single user

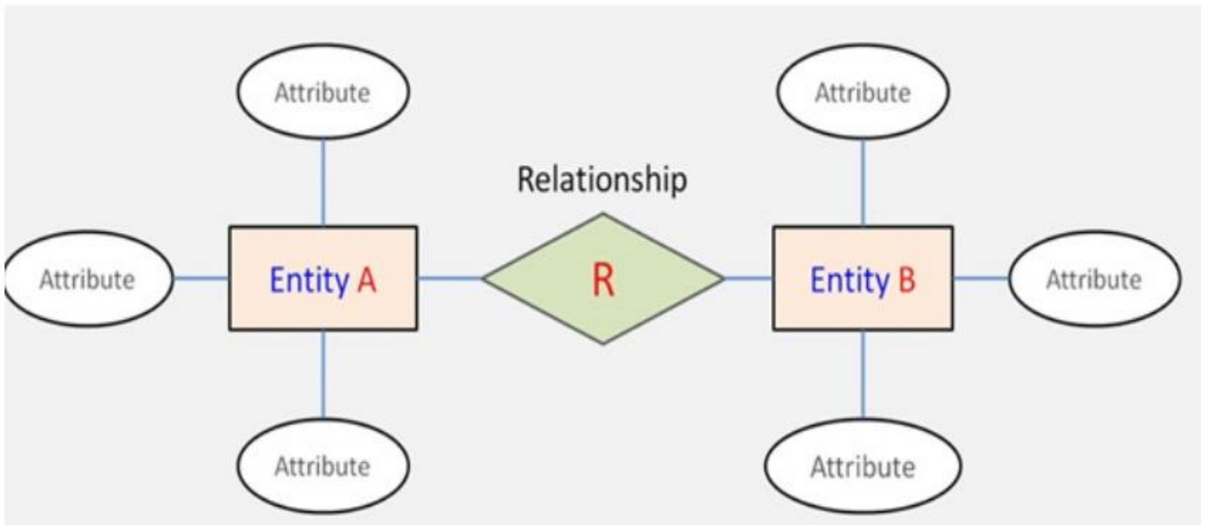

Personal Database Systems
User
Database application
Database Management System (DBMS)
Database
Microsoft Access or other personal DBMS

ER (Entity (Objects) Relationship) Diagram in DBMS
Graphical representation of data in the form of **Rectangles entities, Ellipses attributes, and Diamonds relationships**.

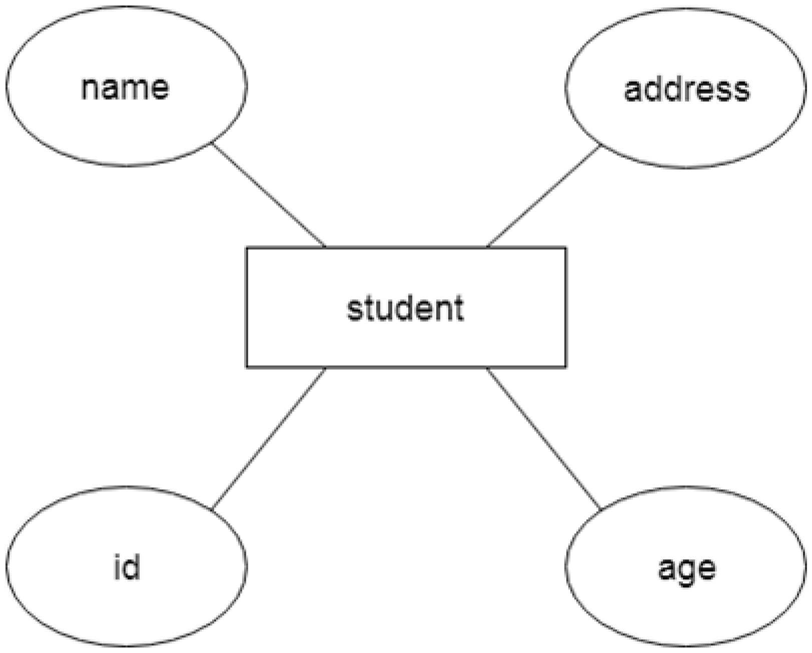
Entity Relationship Diagram in DBMS is a blueprint and pictorially Representation sketch or structural design of a database. The entities, their attributes, and showing the relationships between them, an ER diagram the logical structure of databases.

It allows users to get a preview of the logical structure of the database

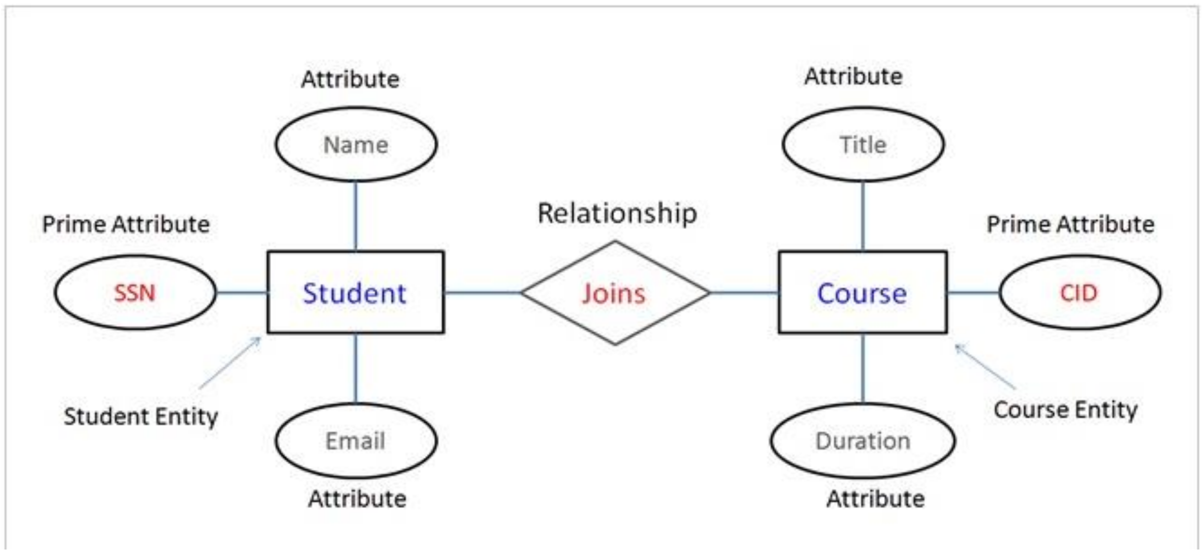
ER diagrams are created based on three basic concepts: **entities, attributes, and relationships.**



For example, suppose we design a school database. In this database, the **student** will be an entity with attributes like address, name, id, age, etc. The **address** can be another entity with attributes like city, street name, pin code, etc and there will be a relationship between them.

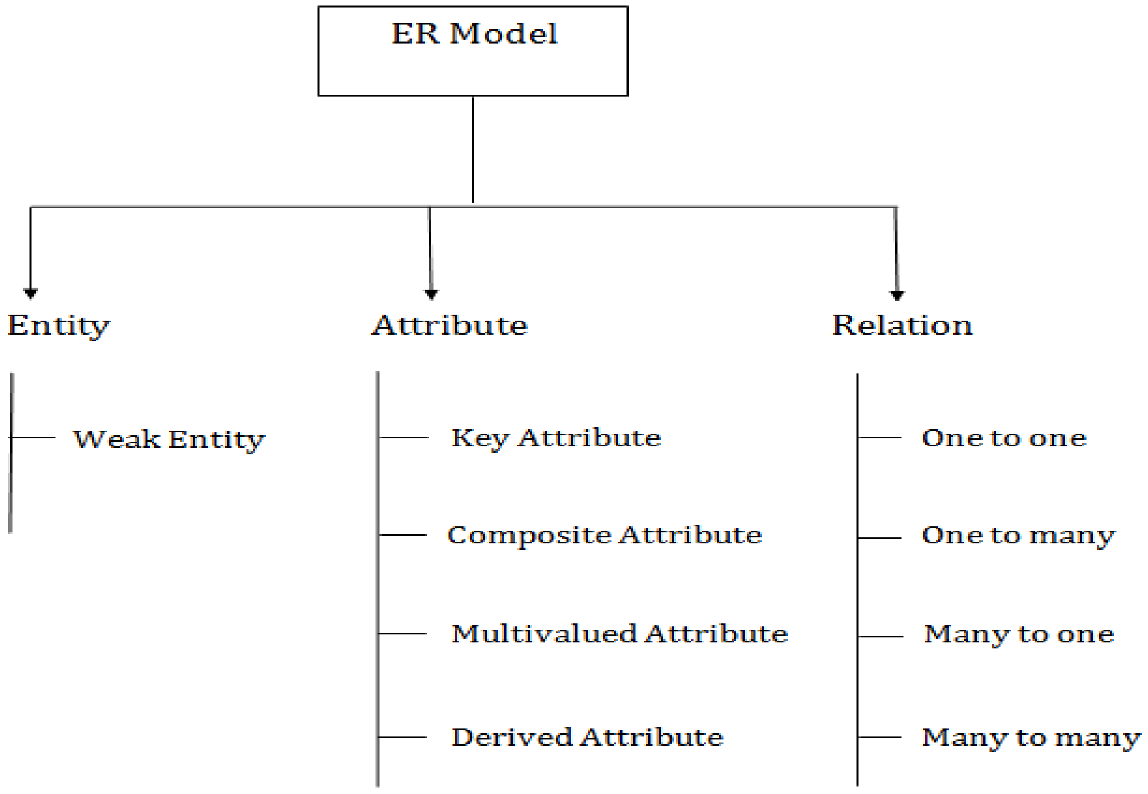


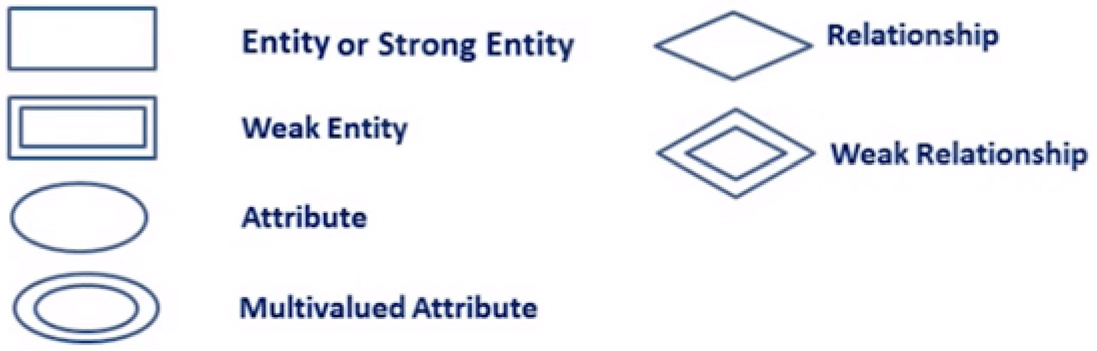
Entity Relationship Diagram (ERD)



Components of the ER Diagram

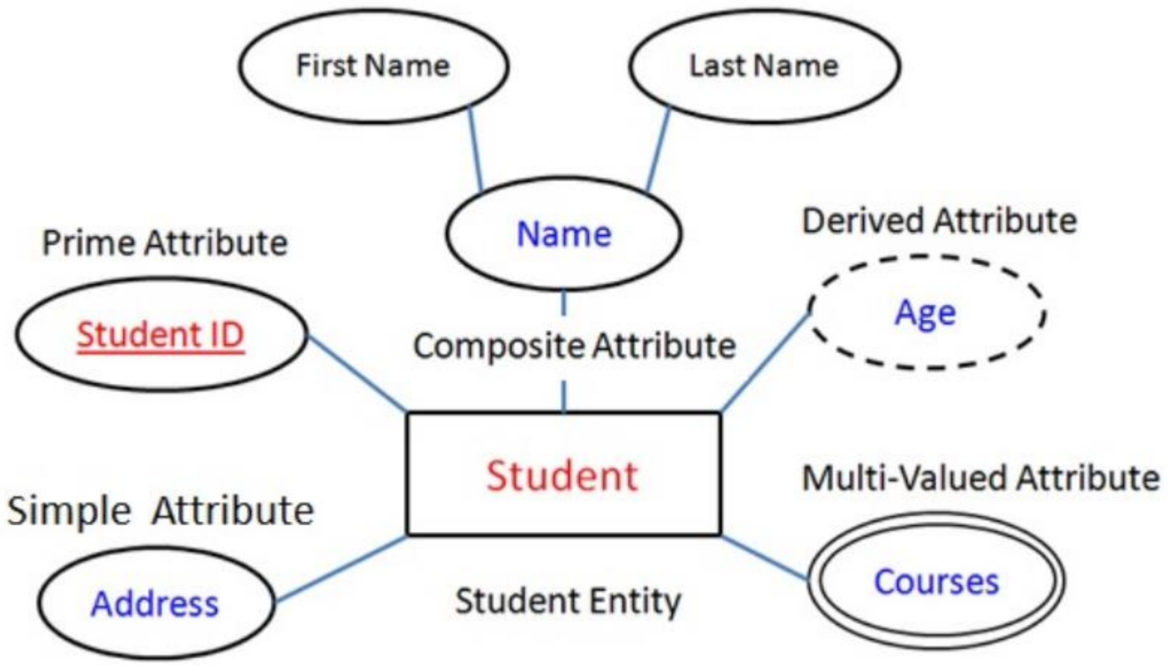
- Entities
- Attributes
- Relationships





Components of E-R Model

1-Rectangles	▭	Represents entity
2-Ellipse	⬭	Represents attributes
3-Diamonds	◇	Represents relationship among entities
4-Lines	—	link attributes to entities and entity sets to relationship



Components of ER Diagram

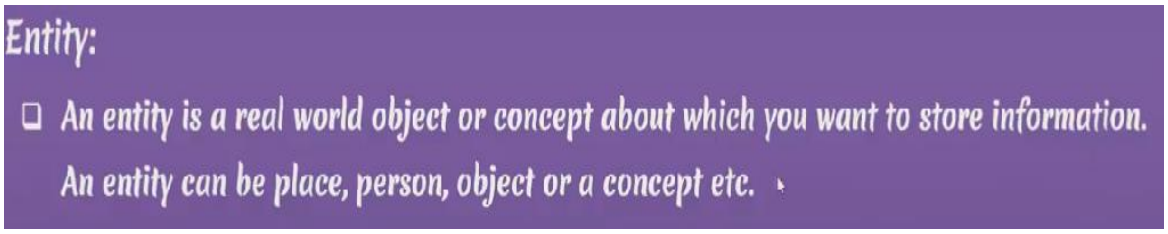
You base an ER Diagram on three basic concepts:

- Entities
- Weak Entity
- Attributes
    - Key Attribute
    - Composite Attribute
    - Multivalued Attribute
    - Derived Attribute
- Relationships
    - One-to-One Relationships
    - One-to-Many Relationships
    - Many-to-One Relationships
    - Many-to-Many Relationships

Symbols Used in ER Diagrams

- **Rectangles:** This Entity Relationship Diagram symbol represents entity types
- **Ellipses :** Symbol represent attributes
- **Diamonds:** This symbol represents relationship types
- **Lines:** It links attributes to entity types and entity types with other relationship types
- **Primary key:** attributes are underlined
- **Double Ellipses:** Represent multi-valued attributes

ENTITY (Object)



Entity:

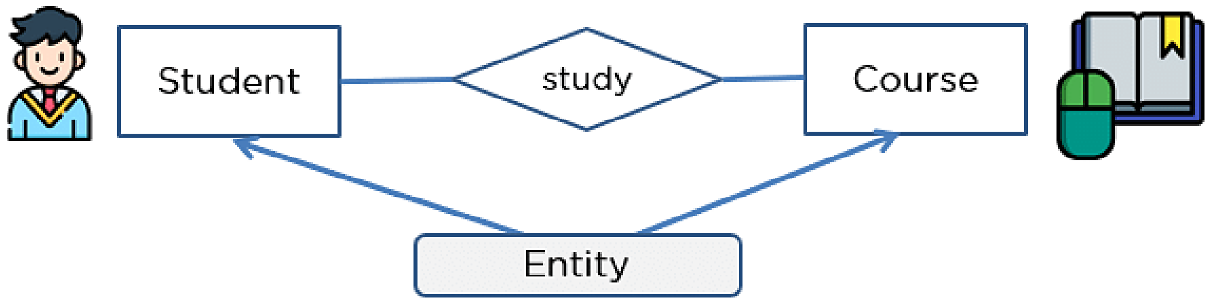
- An entity is a real world object or concept about which you want to store information. An entity can be place, person, object or a concept etc.

An entity can be a real-world object, person or place. In the ER diagram, an entity can be represented as rectangles.

An Entity may be an object with a physical existence person, car, house, or employee – or it may be an object with a conceptual existence – a company, a job, or a university course.

Examples of entities:

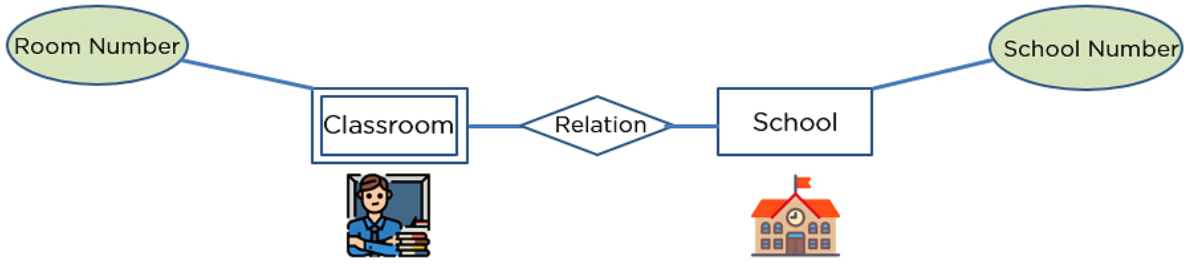
- **Person:** Employee, Student, Patient
- **Place:** Store, Building
- **Object:** Machine, product, and Car
- **Event:** Sale, Registration, Renewal
- **Concept:** Account, Course



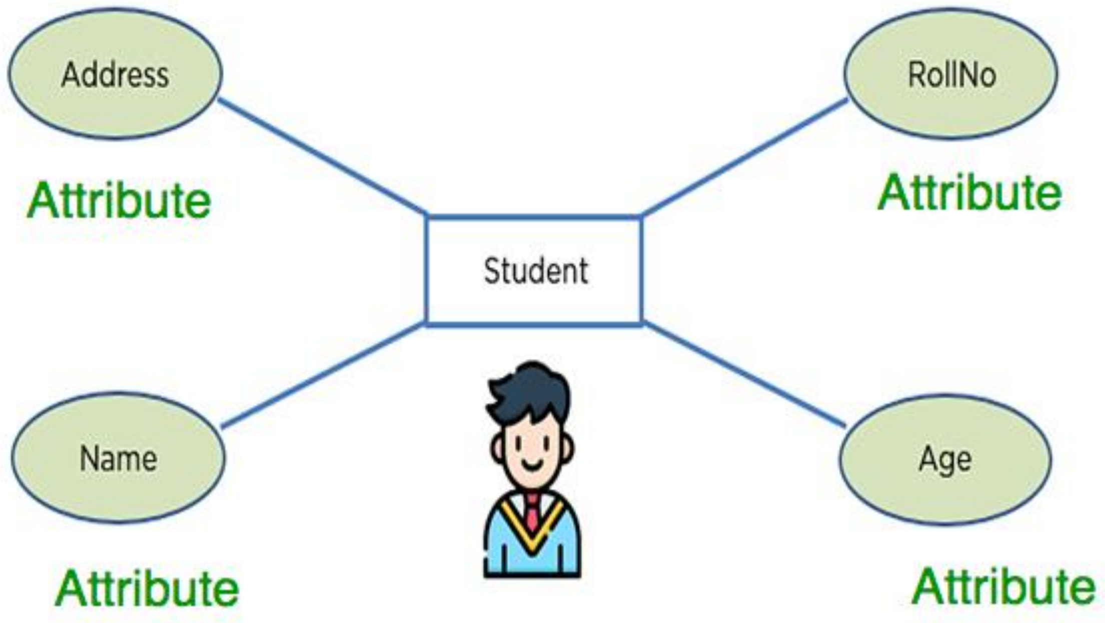
Weak Entity

An entity that makes reliance over another entity is called a weak entity

You showcase the weak entity as a double rectangle in ER Diagram.

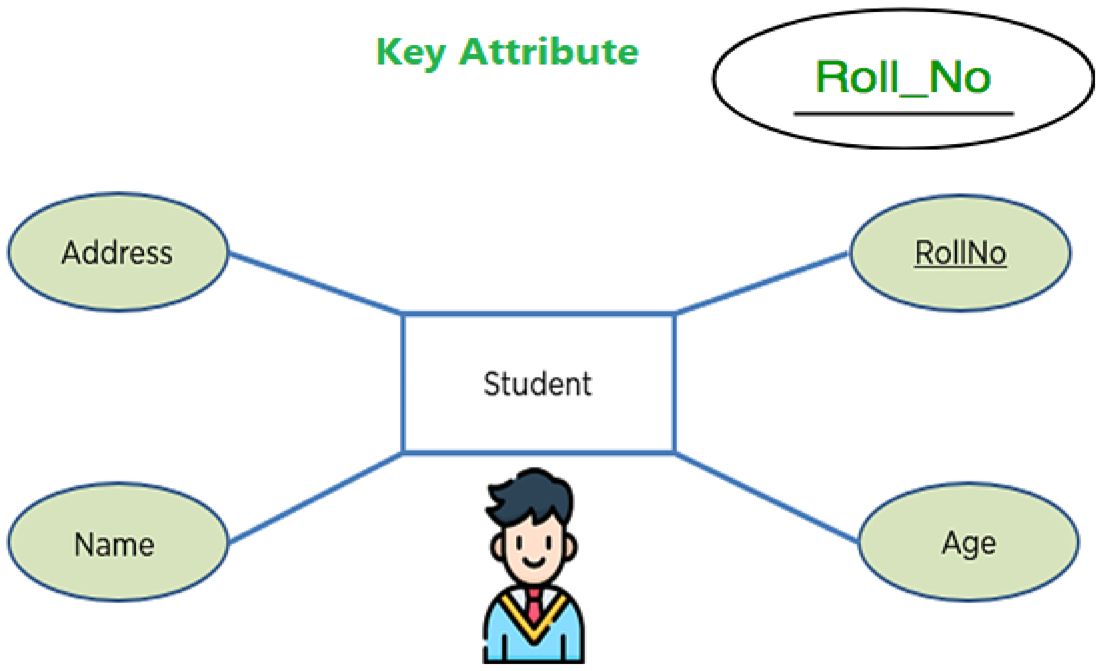


Attribute(s):
Attributes are the **properties that define the entity type**.
 example, Roll_No, Name, DOB, Age, Address, Mobile_No are the attributes

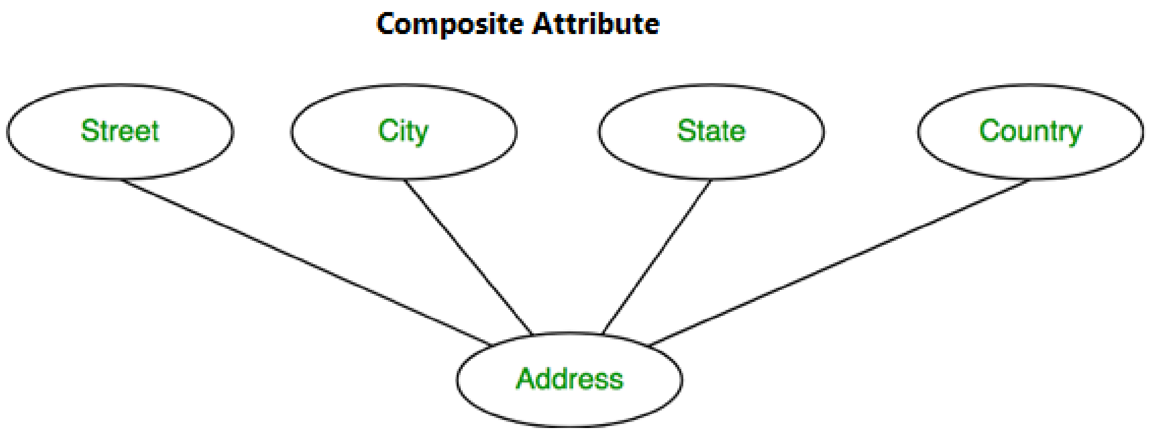


Key Attribute – Key attribute uniquely identifies an entity from an entity set.
The attribute which **uniquely identifies each entity** in the entity set is **called key attribute.**

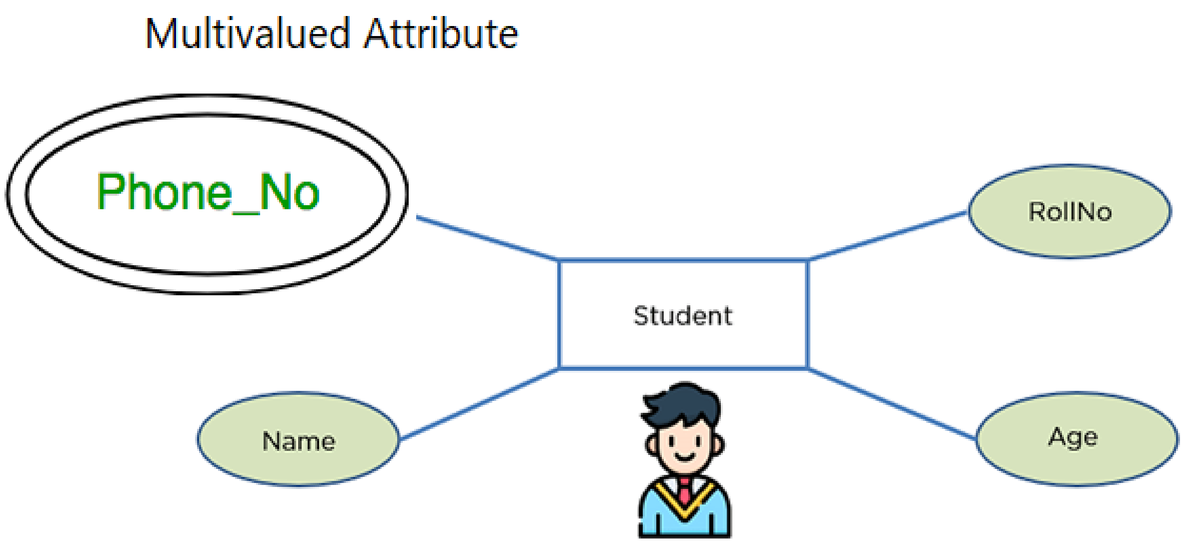
Example, Roll_No will be unique for each student.



Composite Attribute –
An attribute **composed of many other attribute** is called as composite attribute. For example, Address attribute of student Entity type consists of Street, City, State, and Country.



3. Multivalued Attribute –
An attribute consisting **more than one value** for a given entity. For example, Phone_No (can be more than one for a given student).



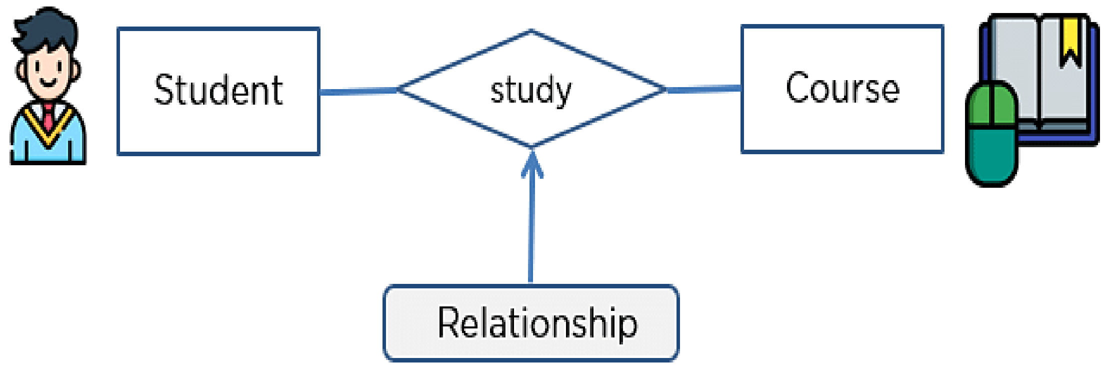
Relationship

The association among entities is called a relationship

Relationships are represented by **diamond-shaped box**.

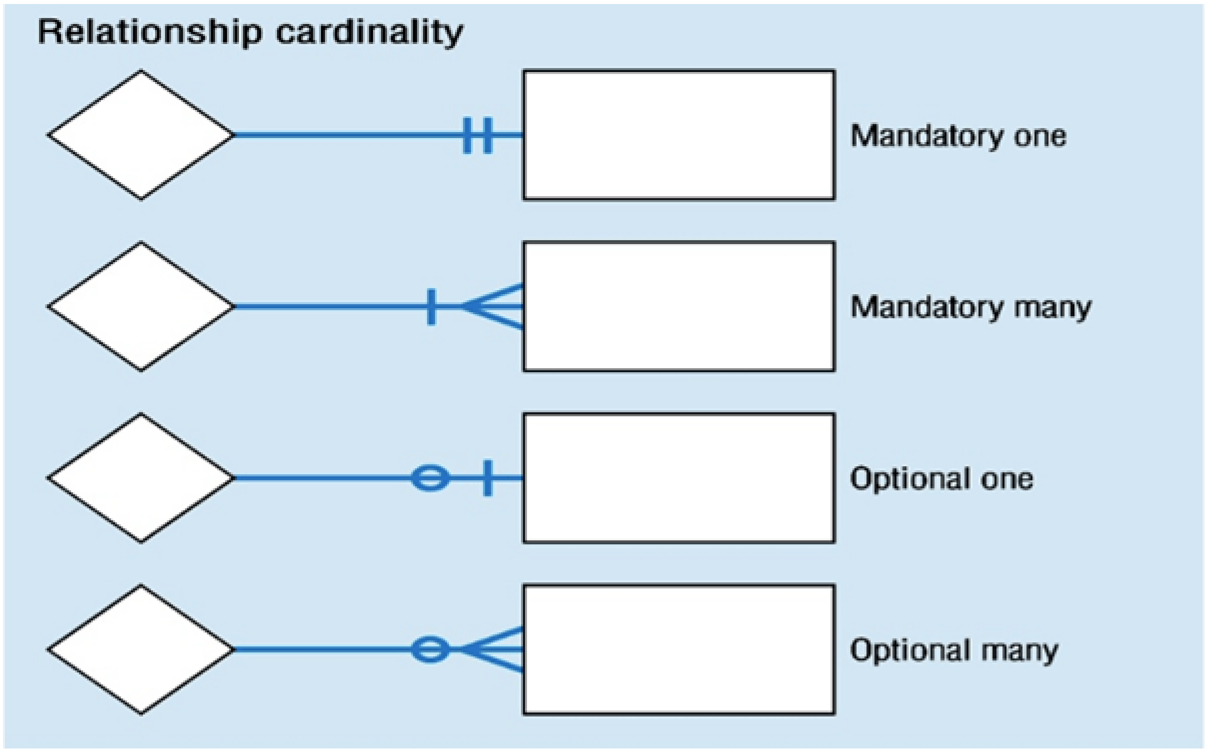
Name of the relationship is written inside the diamond-box. All the entities (rectangles) participating in a relationship, are connected to it by a line.

A relationship type represents the **association between entity types**. For example, Student and Course.ER diagram, the relationship type is represented by a **diamond** and **connecting** the entities with **lines**.



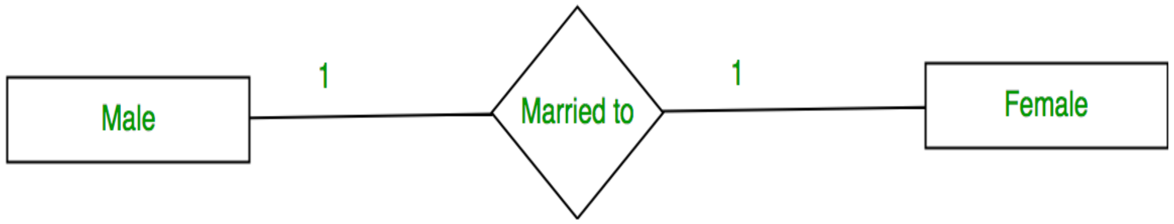
- One-to-One Relationships
- One-to-Many Relationships

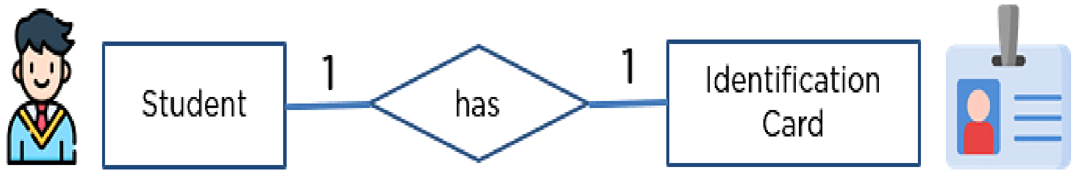
- May to One Relationships
- Many-to-Many Relationships



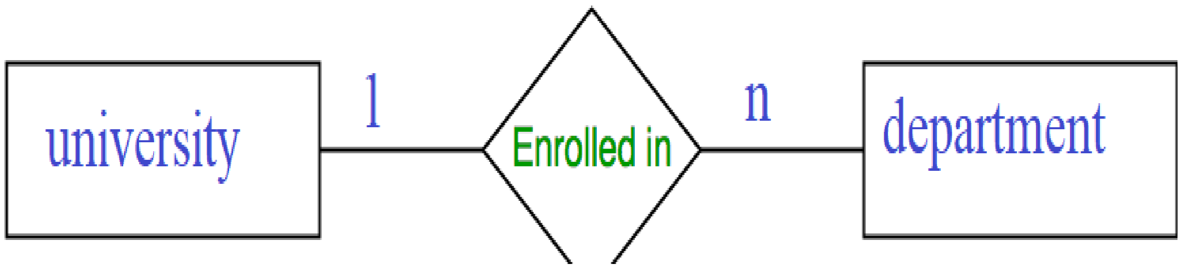
1 One-to-one male can marry one female and a female can marry one male. So the relationship will be one-to-one.

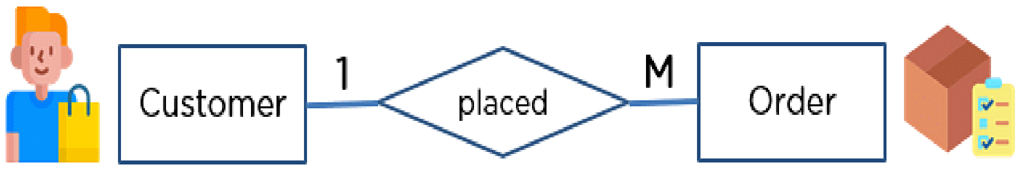
the total number of tables that can be used in this is **2**.



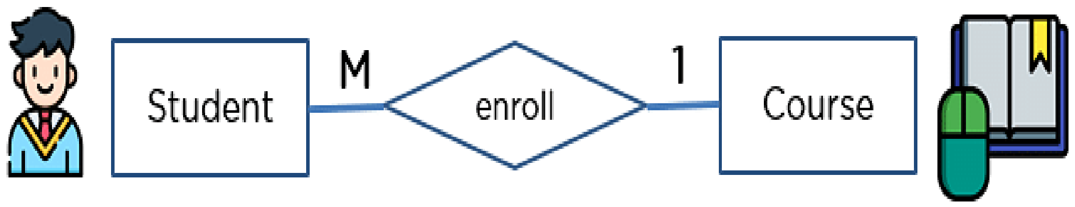


2 **one-to-many** When a single element of an entity is associated with more than one element of another entity, it is called a one-to-many relationship

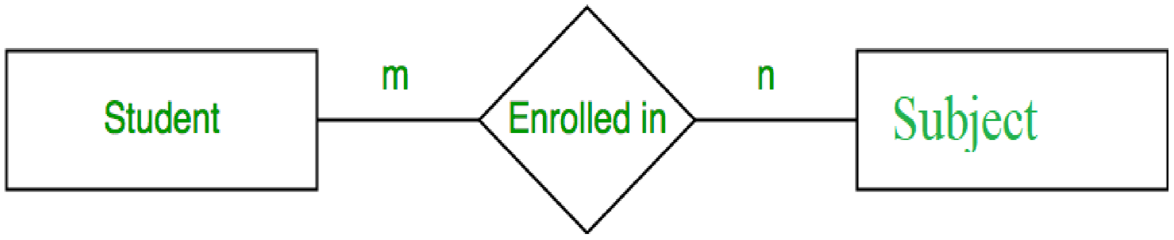


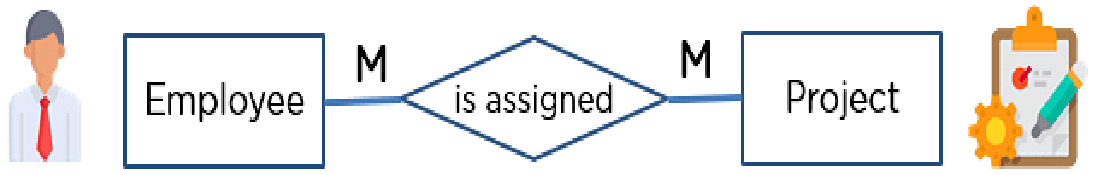


3. Many to one More then student can take only one course but one course can be taken by many students. So the cardinality will be n to 1. It means that for one course there can be n students

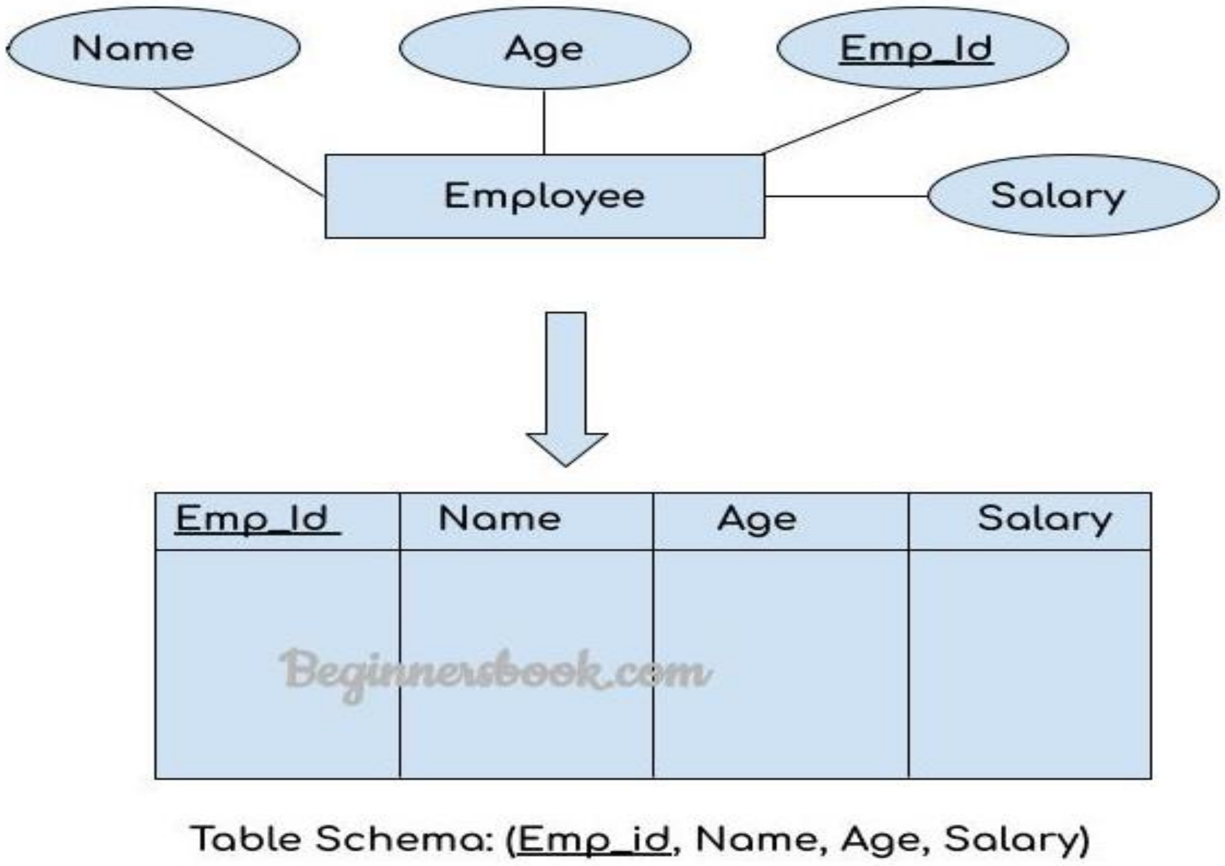


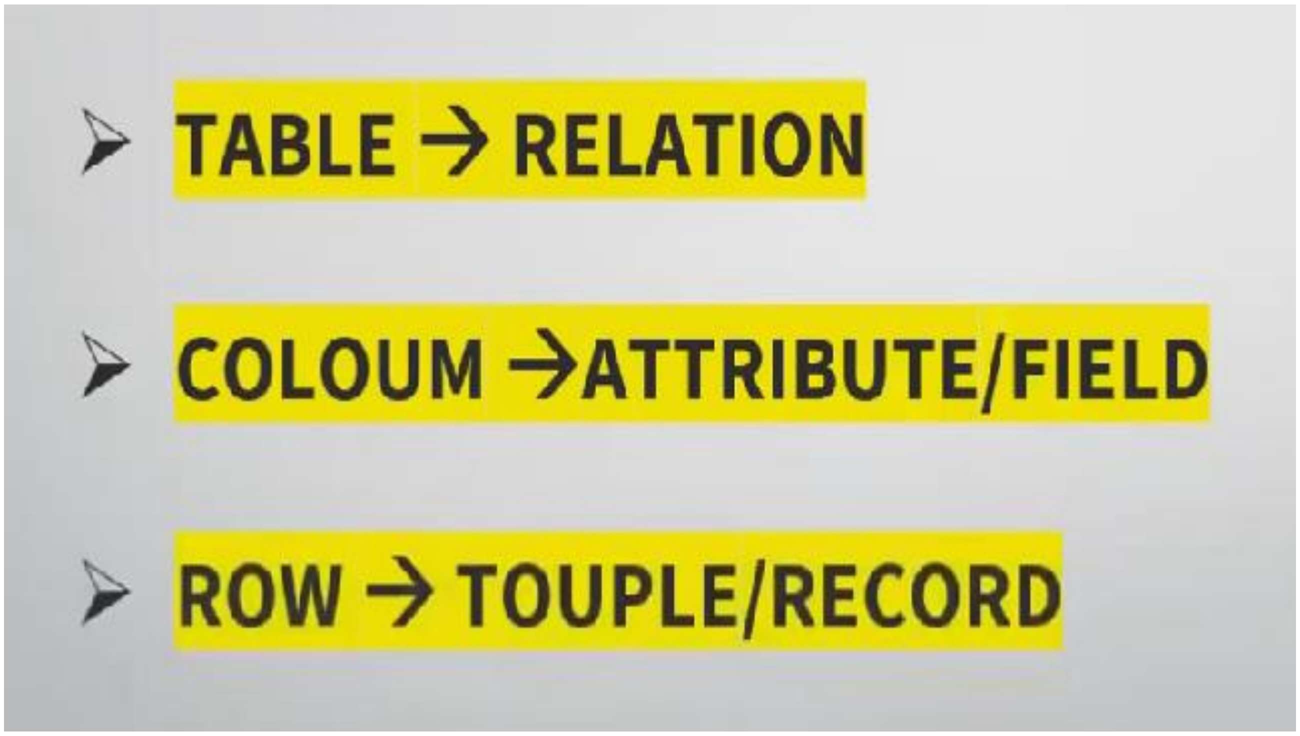
4- Many to many More then student can take more than one subject and one subject can be taken by many students. So the relationship will be many to many.





ER Diagram to Table Conversion





- TABLE → RELATION
- COLOUM →ATTRIBUTE/FIELD
- ROW → TOUPLE/RECORD

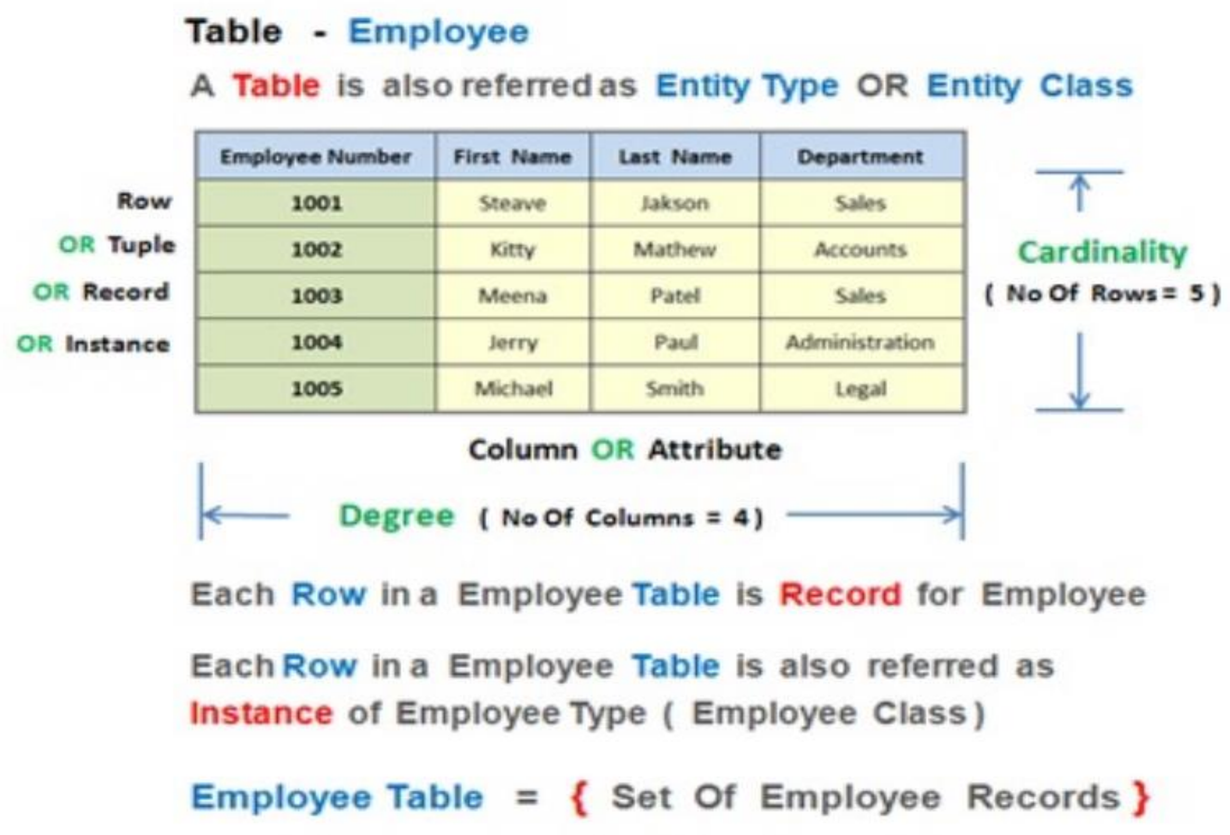


Table - Employee

A Table is also referred as Entity Type OR Entity Class

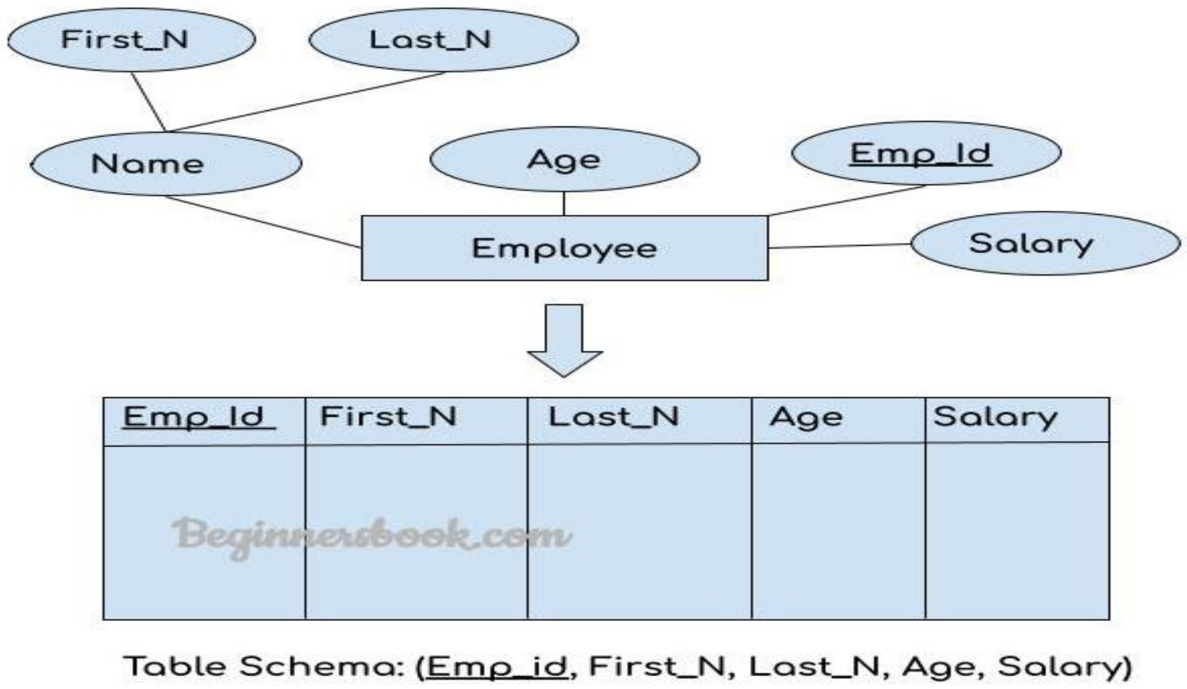
Employee Number	First Name	Last Name	Department
1001	Steave	Jakson	Sales
1002	Kitty	Mathew	Accounts
1003	Meena	Patel	Sales
1004	Jerry	Paul	Administration
1005	Michael	Smith	Legal

Each Row in a Employee Table is Record for Employee

Each Row in a Employee Table is also referred as Instance of Employee Type (Employee Class)

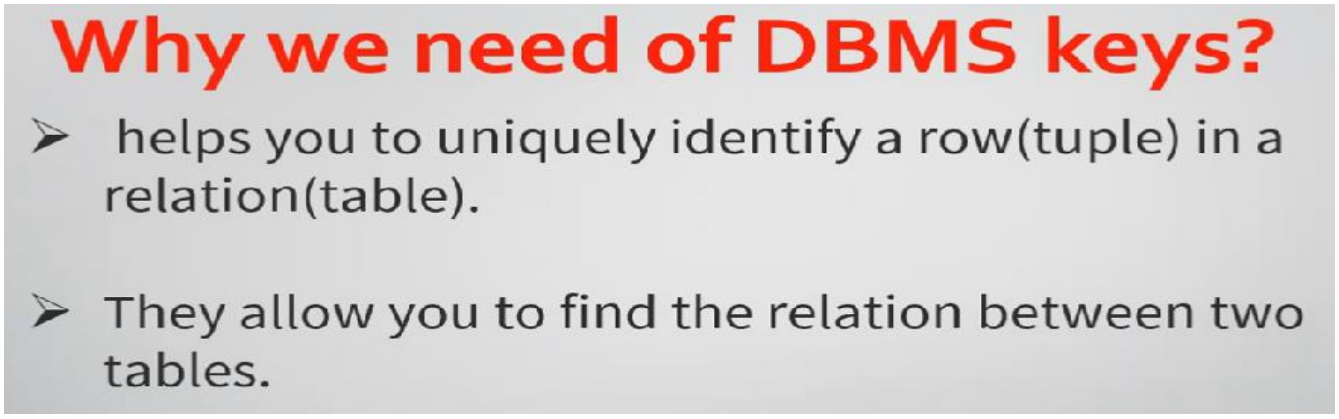
Employee Table = { Set Of Employee Records }

ER Diagram to Table Conversion



Conclusion

ER Diagram in DBMS is widely used to describe the conceptual design of databases. It helps both users and database developers to preview the structure of the database before implementing the database.



Why we need of DBMS keys?

- helps you to uniquely identify a row(tuple) in a relation(table).
- They allow you to find the relation between two tables.

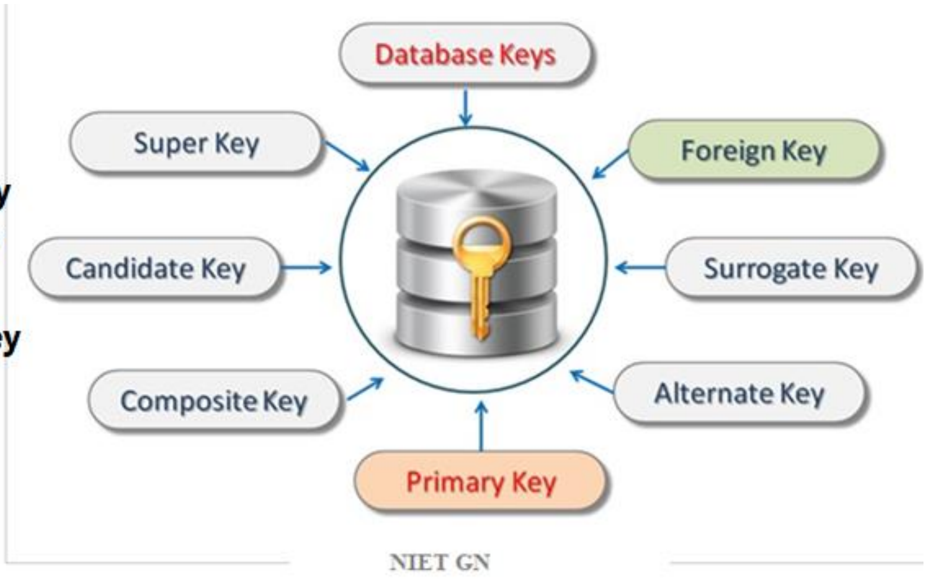
Keys in DBMS

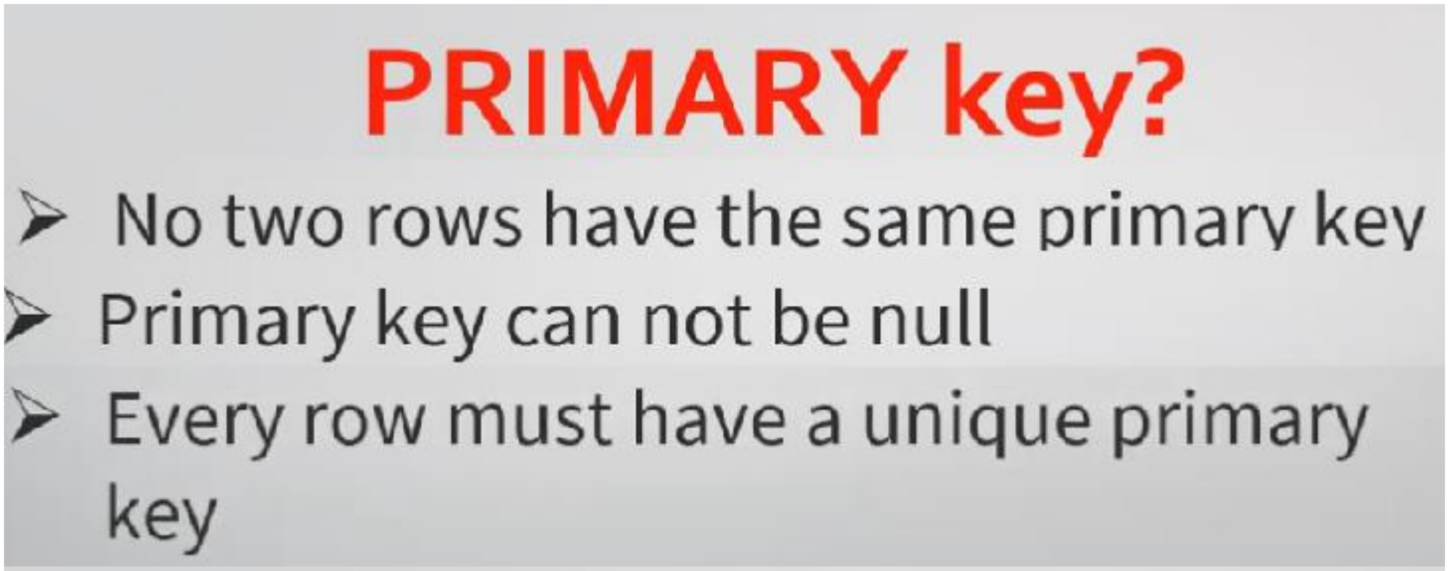
It is used to uniquely identify any record or row of data from the table. It is also used to establish and identify relationships between table

Different types of Keys in DBMS?

Keys are of seven broad types in DBMS:

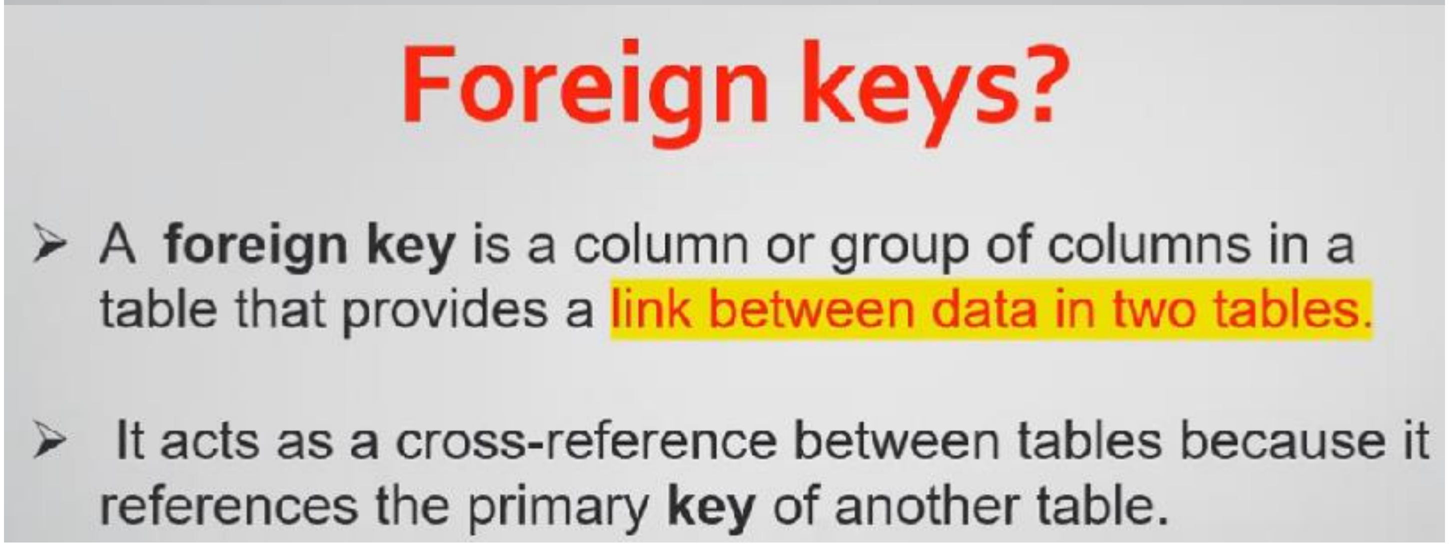
- **Primary Key**
- **Super Key**
- **Candidate Key**
- **Alternate Key**
- **Foreign Key**
- **Composite Key**
- **Unique Key**





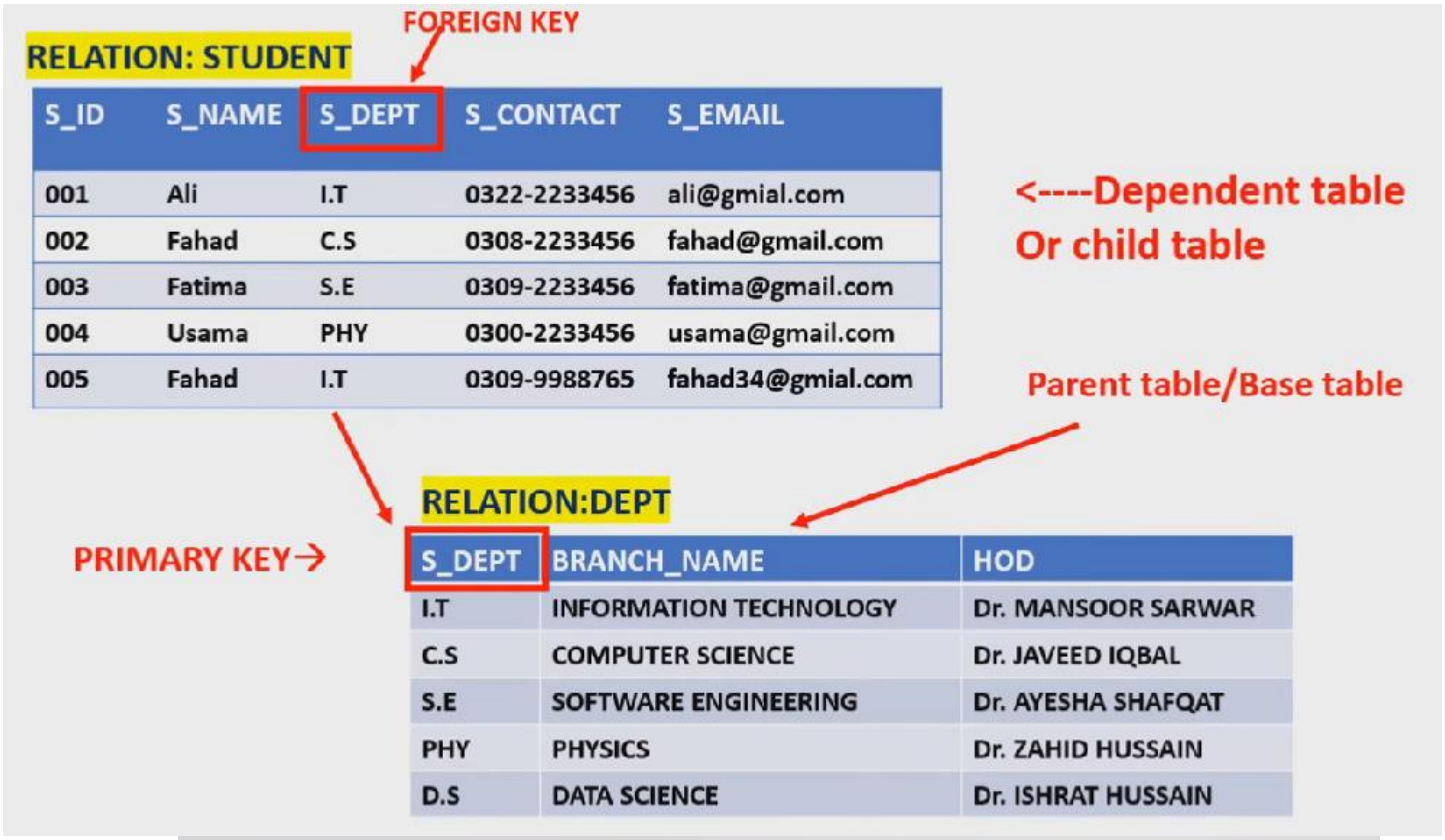
PRIMARY key?

- No two rows have the same primary key
- Primary key can not be null
- Every row must have a unique primary key



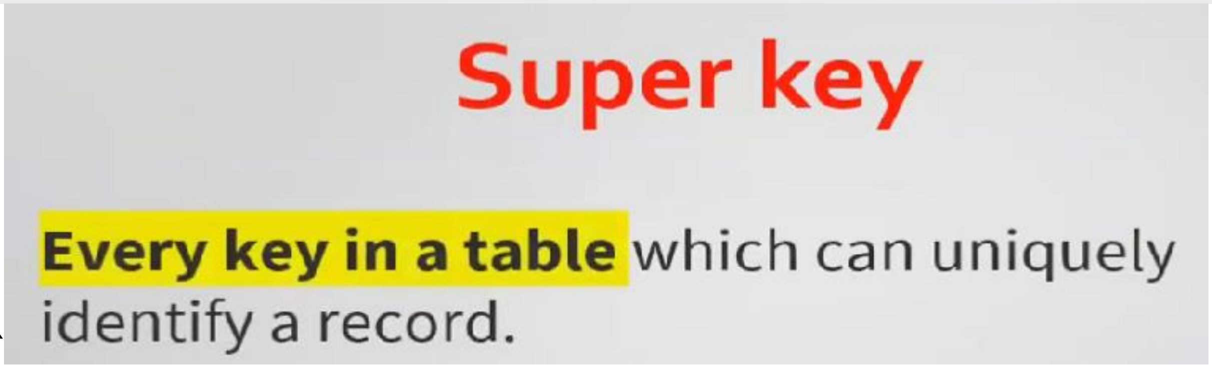
Foreign keys?

- A **foreign key** is a column or group of columns in a table that provides a link between data in two tables.
- It acts as a cross-reference between tables because it references the primary **key** of another table.



S_ID	S_NAME	S_DEPT	S_CONTACT	S_EMAIL
001	Ali	I.T	0322-2233456	ali@gmial.com
002	Fahad	C.S	0308-2233456	fahad@gmail.com
003	Fatima	S.E	0309-2233456	fatima@gmail.com
004	Usama	PHY	0300-2233456	usama@gmail.com
005	Fahad	I.T	0309-9988765	fahad34@gmial.com

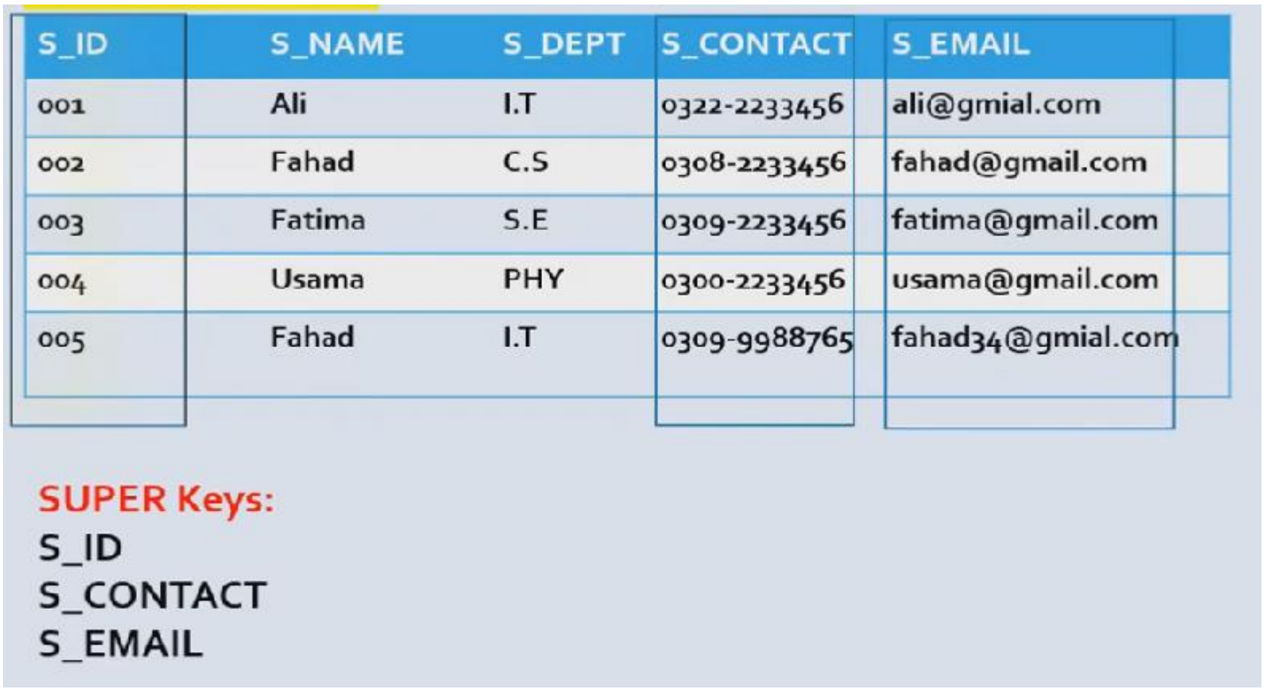
S_DEPT	BRANCH_NAME	HOD
I.T	INFORMATION TECHNOLOGY	Dr. MANSOOR SARWAR
C.S	COMPUTER SCIENCE	Dr. JAVEED IQBAL
S.E	SOFTWARE ENGINEERING	Dr. AYESHA SHAFQAT
PHY	PHYSICS	Dr. ZAHID HUSSAIN
D.S	DATA SCIENCE	Dr. ISHRAT HUSSAIN



Super key

Every key in a table which can uniquely identify a record.

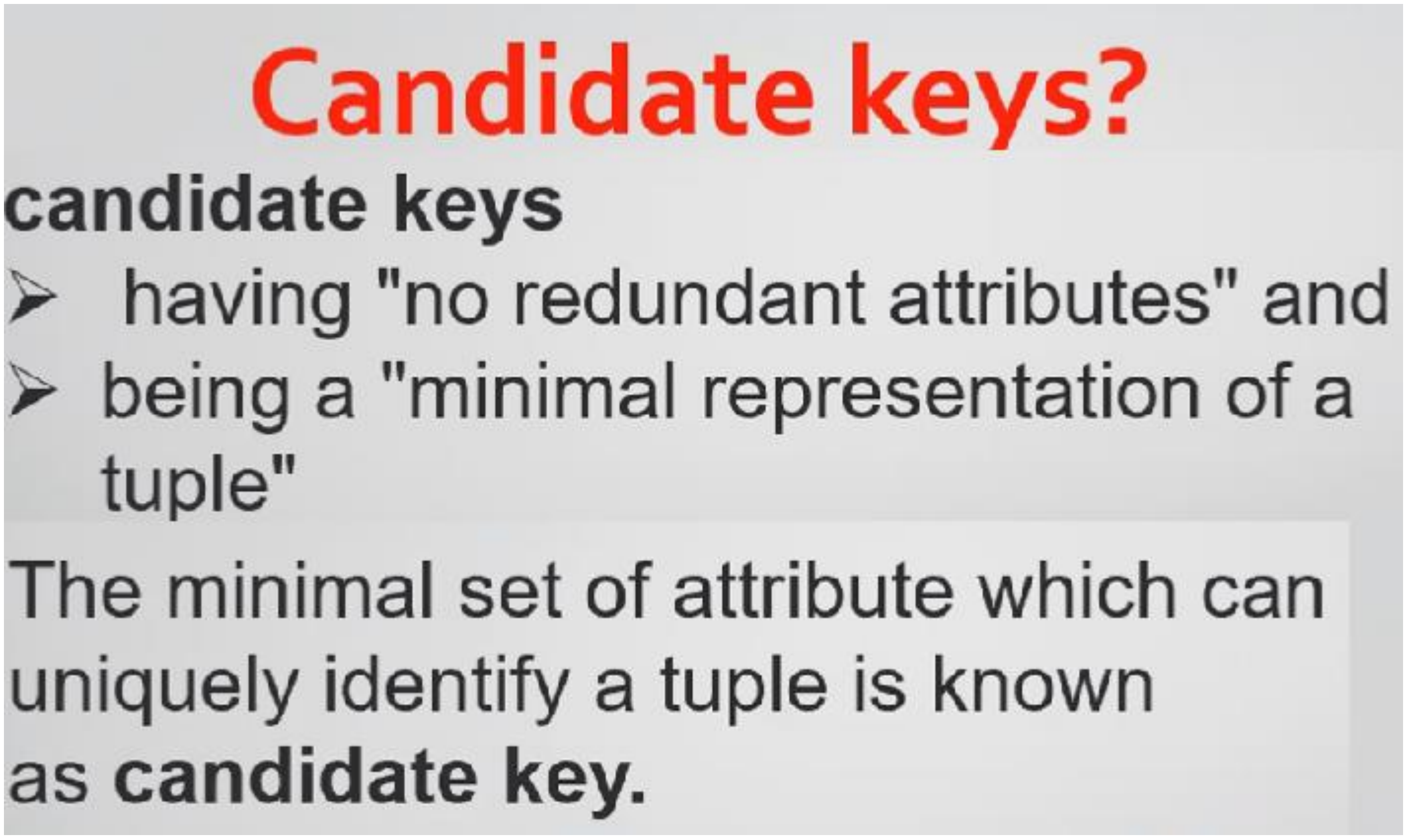
Super Key is defined as a set of attributes within a table that can uniquely identify each record within a table. Super Key is a superset of Candidate key.



S_ID	S_NAME	S_DEPT	S_CONTACT	S_EMAIL
001	Ali	I.T	0322-2233456	ali@gmial.com
002	Fahad	C.S	0308-2233456	fahad@gmail.com
003	Fatima	S.E	0309-2233456	fatima@gmail.com
004	Usama	PHY	0300-2233456	usama@gmail.com
005	Fahad	I.T	0309-9988765	fahad34@gmial.com

SUPER Keys:
S_ID
S_CONTACT
S_EMAIL

(student_id, name), now name of two students can be same, but their student_id can't be same hence this combination can also be a key.

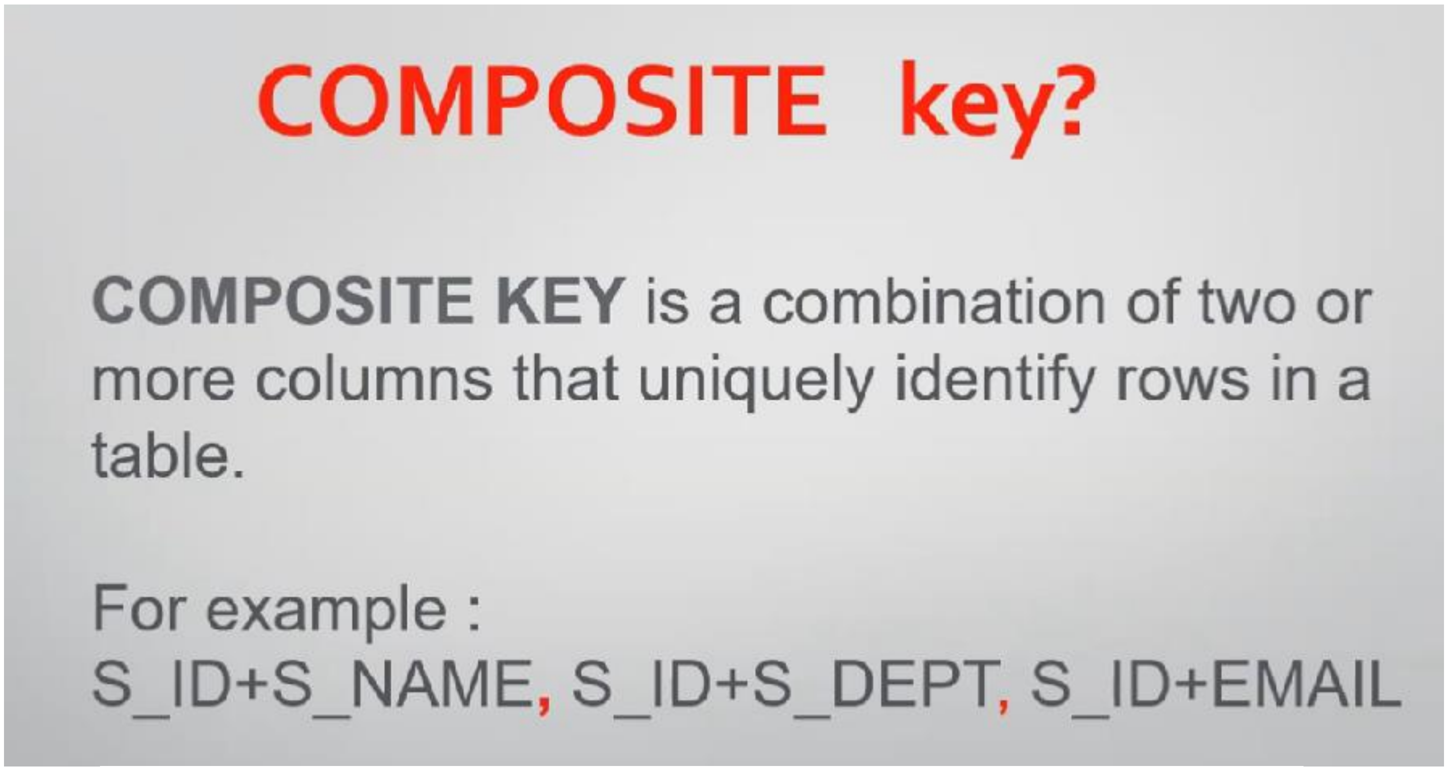


Candidate keys?

candidate keys

- having "no redundant attributes" and
- being a "minimal representation of a tuple"

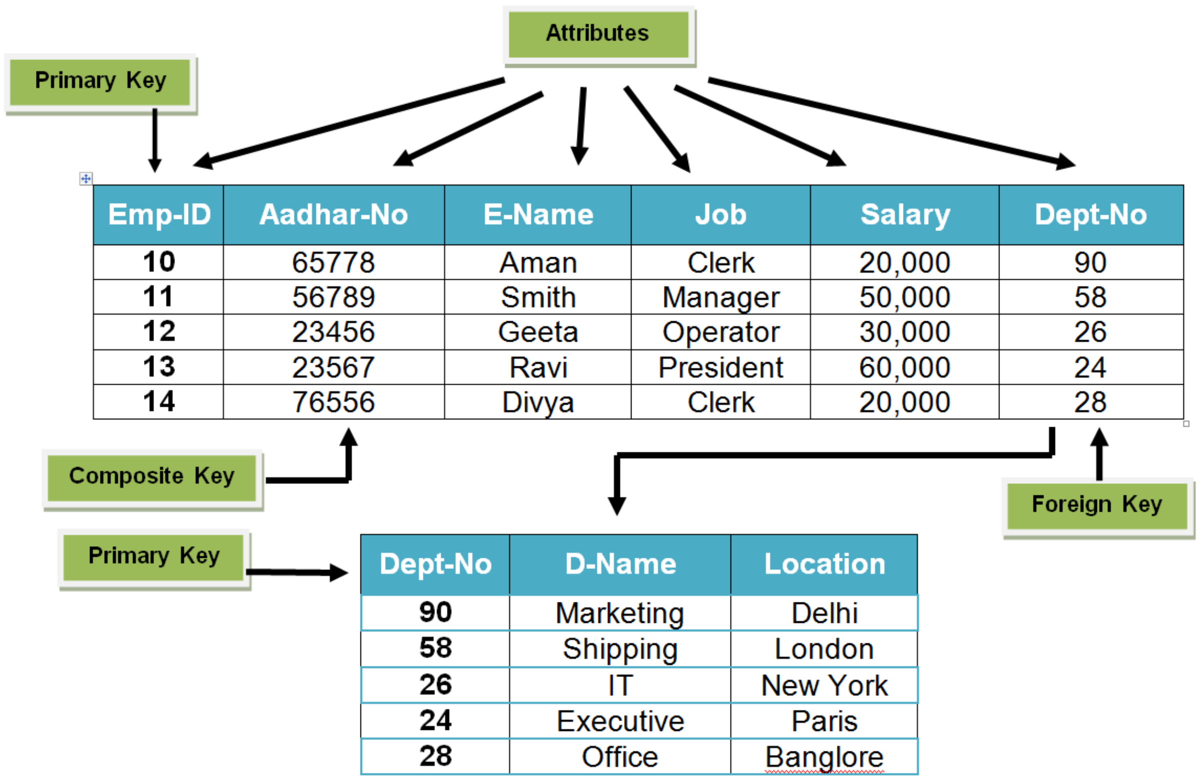
The minimal set of attribute which can uniquely identify a tuple is known as **candidate key.**



COMPOSITE key?

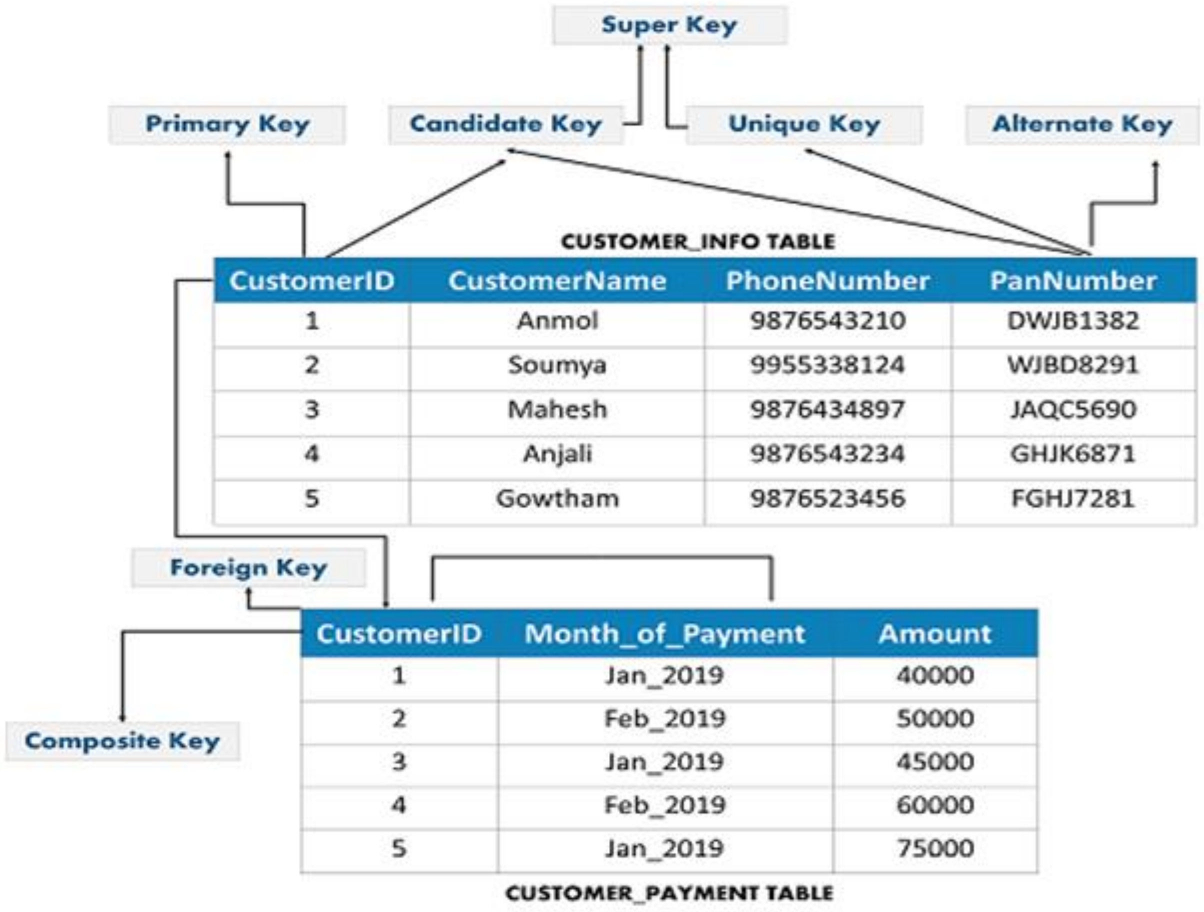
COMPOSITE KEY is a combination of two or more columns that uniquely identify rows in a table.

For example :
S_ID+S_NAME, S_ID+S_DEPT, S_ID+EMAIL



Emp-ID	Aadhar-No	E-Name	Job	Salary	Dept-No
10	65778	Aman	Clerk	20,000	90
11	56789	Smith	Manager	50,000	58
12	23456	Geeta	Operator	30,000	26
13	23567	Ravi	President	60,000	24
14	76556	Divya	Clerk	20,000	28

Dept-No	D-Name	Location
90	Marketing	Delhi
58	Shipping	London
26	IT	New York
24	Executive	Paris
28	Office	Banglore



CUSTOMER_INFO TABLE

CustomerID	CustomerName	PhoneNumber	PanNumber
1	Anmol	9876543210	DWJB1382
2	Soumya	9955338124	WJBD8291
3	Mahesh	9876434897	JAQC5690
4	Anjali	9876543234	GHJK6871
5	Gowtham	9876523456	FGHJ7281

CustomerID	Month_of_Payment	Amount
1	Jan_2019	40000
2	Feb_2019	50000
3	Jan_2019	45000
4	Feb_2019	60000
5	Jan_2019	75000

CUSTOMER_PAYMENT TABLE

Primary Key

primary key in DBMS, primary key is a column of a table or a set of columns that helps to identify every record present in that table uniquely. The Primary Key can’t be a duplicatethe primary Key cannot have the same values repeating for any row

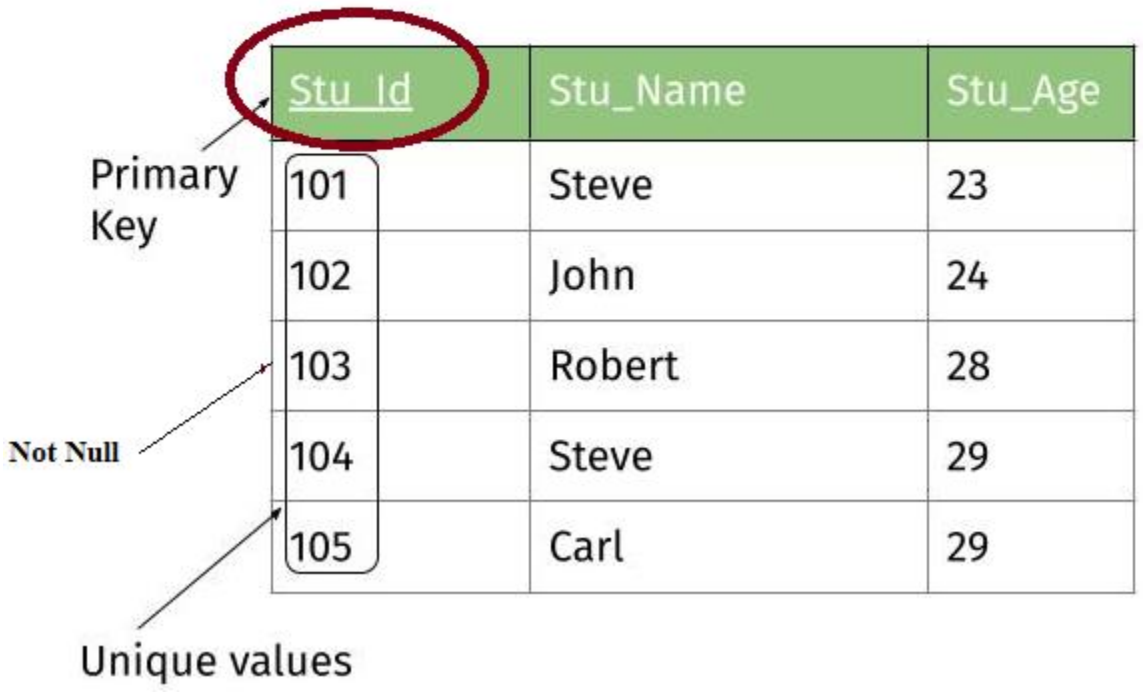
It uniquely identifies each record of the table

the primary key field cannot have a NULL value,

the table can’t have the same primary key value

It has no duplicate values, it has unique values

Example:Table Name: STUDENTS Stu_id is a Primary Key.


Primary Key
Not Null
Unique values

Stu_Id	Stu_Name	Stu_Age
101	Steve	23
102	John	24
103	Robert	28
104	Steve	29
105	Carl	29

Unique

The value of primary key should be unique for each row of the table. The column(s) that makes the key **cannot contain duplicate values.**
Non Null

The attribute(s) that is marked as primary key is not allowed to have null values.

Super key

A super key is a set of one or more attributes (columns), which can uniquely identify a row in a table.

Super keys: The above table has following super keys. All of the following sets of super key are able to uniquely identify a row of the employee table.

Table: Employee

Emp_SSN	Emp_Number	Emp_Name
123456789	226	Steve
999999321	227	Ajeet
888997212	228	Chaitanya
777778888	229	Robert

- {Emp_SSN}
- {Emp_Number}
- {Emp_SSN, Emp_Number}
- {Emp_SSN, Emp_Name}
- {Emp_SSN, Emp_Number, Emp_Name}
- {Emp_Number, Emp_Name}

How candidate key is different from super key?

Candidate keys are selected from the set of super keys, the only thing we take care while selecting candidate key is: It should not have any redundant attribute. That's the reason they are also termed as minimal super key.

Candidate Keys: As I mentioned in the beginning, a candidate key is a minimal super key with no redundant attributes. The following two set of super keys are chosen from the above sets as there are no redundant attributes in these sets.

- {Emp_SSN}
- {Emp_Number}
- Only these two sets are candidate keys as all other sets are having redundant attributes that are not necessary for unique identification.

Let’s take an example to understand this:
Table: Employee

```
Emp_SSN     Emp_Number        Emp_Name
---------------------------
123456789   226    Steve
999999321   227    Ajeet
888997212   228    Chaitanya
777778888   229    Robert
```

Foreign key in DBMS

Definition: Foreign keys are the columns of a table that points to the primary key of another table. They act as a cross-reference between tables.

In the relational databases, a foreign key is a field or a column that is used to establish a link between two tables.

First table:

S_Id	LastName	FirstName	CITY
1	MAURYA	AJEET	ALLAHABAD
2	JAISWAL	RATAN	GHAZIABAD
3	ARORA	SAUMYA	MODINAGAR

Second table:

O_Id	OrderNo	S_Id
1	99586465	2
2	78466588	2
3	22354846	3
4	57698656	1

- The "S_Id" column in the "Students" table is the PRIMARY KEY in the "Students" table.
- The "S_Id" column in the "Orders" table is a FOREIGN KEY in the "Orders" table.

Unique Key in SQL

A unique key is a set of one or more than one fields/columns of a table that uniquely identify a record in a database table.

You can say that it is little like primary key but it can accept only one null value and it cannot have duplicate values.

There is an automatically defined unique key constraint within a primary key constraint.

SQL UNIQUE KEY constraint on CREATE TABLE:

If you want to create a UNIQUE constraint on the "S_Id" column when the "students" table is created, use the following SQL syntax:

SQL Server / Oracle / MS Access:

```
CREATE TABLE students
(
S_Id int NOT NULL UNIQUE,
```

4. LastName **varchar** (255) NOT NULL,
5. FirstName **varchar** (255),
6. City **varchar** (255)
7.)

What is Normalization in DBMS?

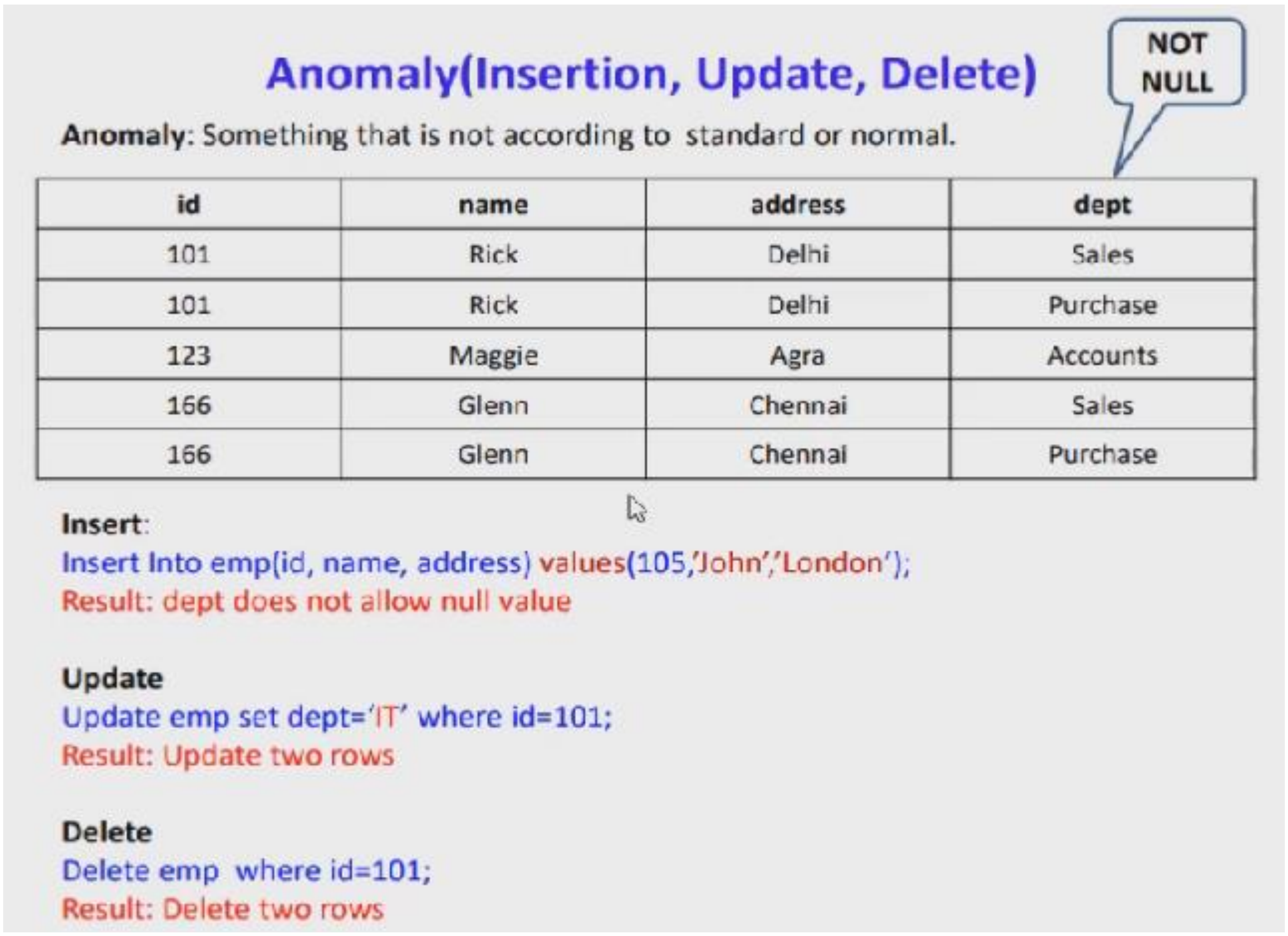
normalization is technique for database design organizing the data inside the database in such a way that we can ignore data redundancy, insertion update & deletion

It can be seen as a process used for minimizing redundancy from a relation or set of relations.

normalization is technique for database design bigger table divide into small table and create relationship between tables

- Normalization is the process of organizing the data in the database.
- Normalization is used to minimize the redundancy from a relation or set of relations.
- Normalization divides the larger table into smaller and links them using relationships.

Example =NOT WELL format table show many problems in table



Anomaly(Insertion, Update, Delete)

Anomaly: Something that is not according to standard or normal.

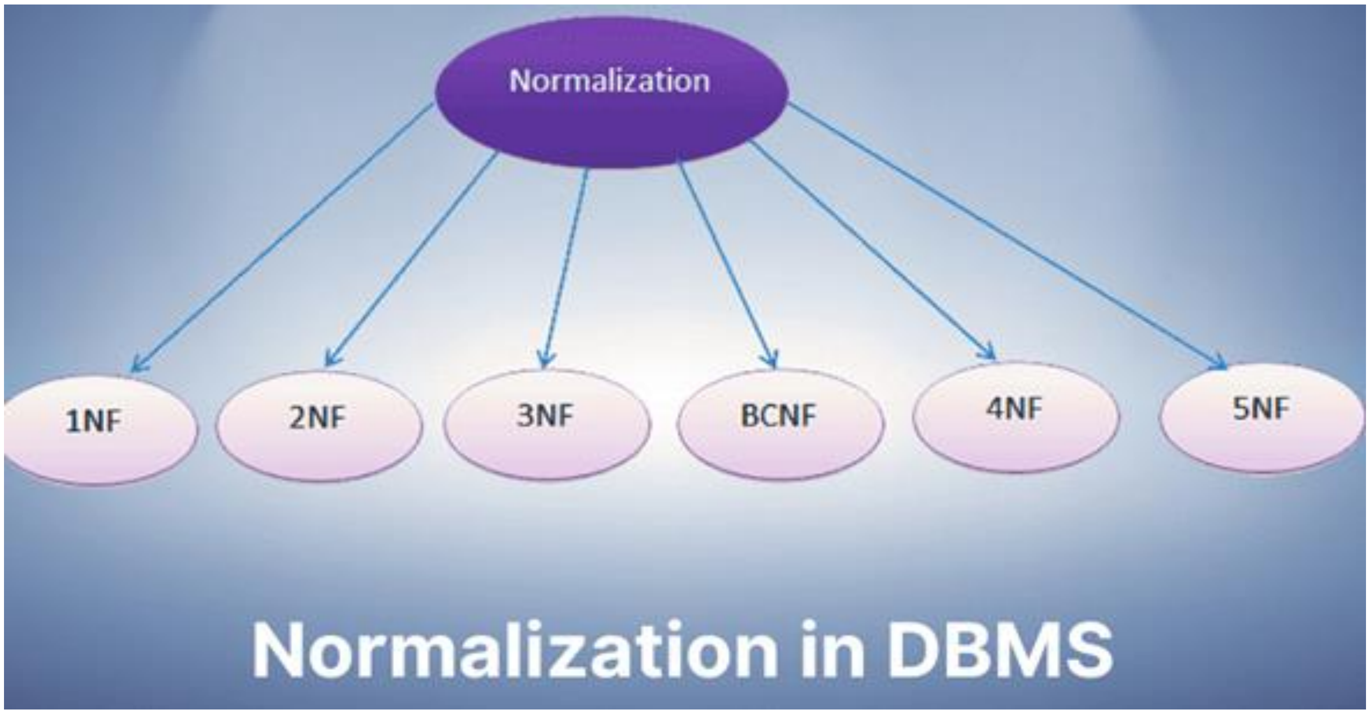
id	name	address	dept
101	Rick	Delhi	Sales
101	Rick	Delhi	Purchase
123	Maggie	Agra	Accounts
166	Glenn	Chennai	Sales
166	Glenn	Chennai	Purchase

Insert:
Insert Into emp(id, name, address) values(105,'John','London');
Result: dept does not allow null value

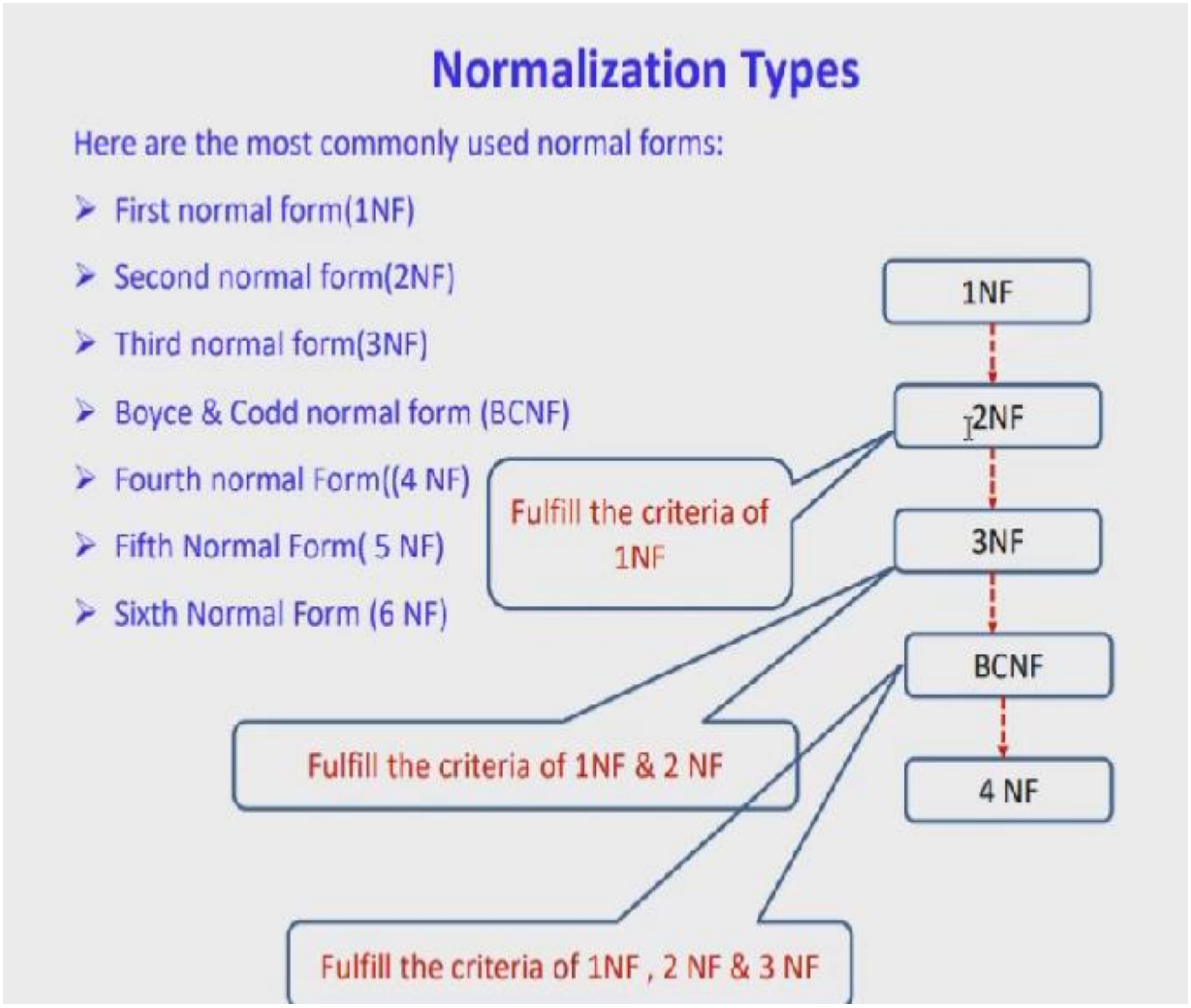
Update
Update emp set dept='IT' where id=101;
Result: Update two rows

Delete
Delete emp where id=101;
Result: Delete two rows

Types of Normal Forms:



1. **First normal form (1NF)**
2. **Second normal form (2NF)**
3. **Third normal form (3NF)**
4. **Boyce-Codd Normal Form(BCNF)**
5. **Fourth normal form (4NF)**
6. **Fifth normal form (5NF)**



Normalization Types

Here are the most commonly used normal forms:

- First normal form(1NF)
- Second normal form(2NF)
- Third normal form(3NF)
- Boyce & Codd normal form (BCNF)
- Fourth normal Form((4 NF)
- Fifth Normal Form(5 NF)
- Sixth Normal Form (6 NF)

1NF
2NF
3NF
BCNF
4 NF
Fulfill the criteria of 1NF
Fulfill the criteria of 1NF & 2 NF
Fulfill the criteria of 1NF , 2 NF & 3 NF

First normal form in DBMS

A relation is said to be in **1NF (first normal form)**, if it doesn't contain any multi-valued attribute.

1NF if each attribute contains only **atomic(single) value only**.

Each record needs to be unique.

Example: Relation EMPLOYEE is not in 1NF because of multi-valued attribute EMP_PHONE.

EMPLOYEE table:

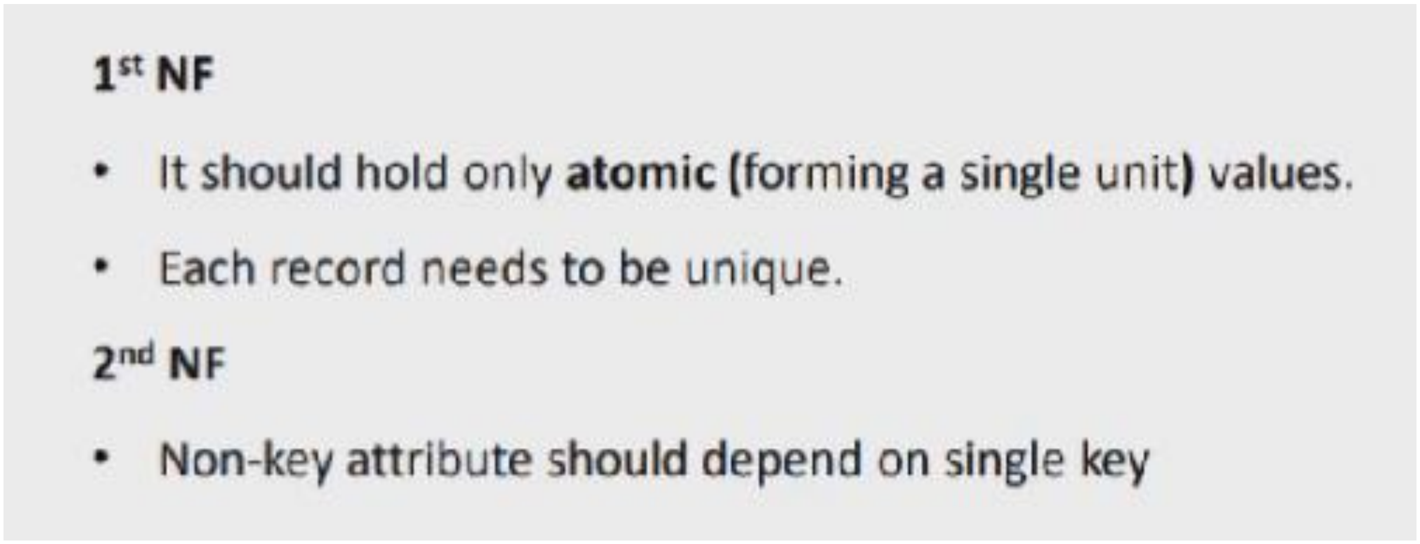
EMP_ID	EMP_NAME	EMP_PHONE	EMP_STATE
14	John	7272826385, 9064738238	UP
20	Harry	8574783832	Bihar
12	Sam	7390372389, 8589830302	Punjab

The decomposition of the EMPLOYEE table into 1NF has been shown below:

EMP_ID	EMP_NAME	EMP_PHONE	EMP_STATE
14	John	7272826385	UP
14	John	9064738238	UP
20	Harry	8574783832	Bihar
12	Sam	7390372389	Punjab
12	Sam	8589830302	Punjab

Second Normal Form (2NF).

- In the 2NF, relational must be in 1NF.
- In the second normal form, all non-key attributes are fully functional dependent on the primary key



1st NF

- It should hold only **atomic (forming a single unit)** values.
- Each record needs to be unique.

2nd NF

- Non-key attribute should depend on single key

Example: Let's assume, a school can store the data of teachers and the subjects they teach. In a school, a teacher can teach more than one subject.

TEACHER table

TEACHER_ID	SUBJECT	TEACHER_AGE
25	Chemistry	30
25	Biology	30
47	English	35
83	Math	38
83	Computer	38

TEACHER_DETAIL table:

TEACHER_ID	TEACHER_AGE
25	30
47	35
83	38

TEACHER_SUBJECT table:

TEACHER_ID	SUBJECT
25	Chemistry
25	Biology
47	English
83	Math
83	Computer

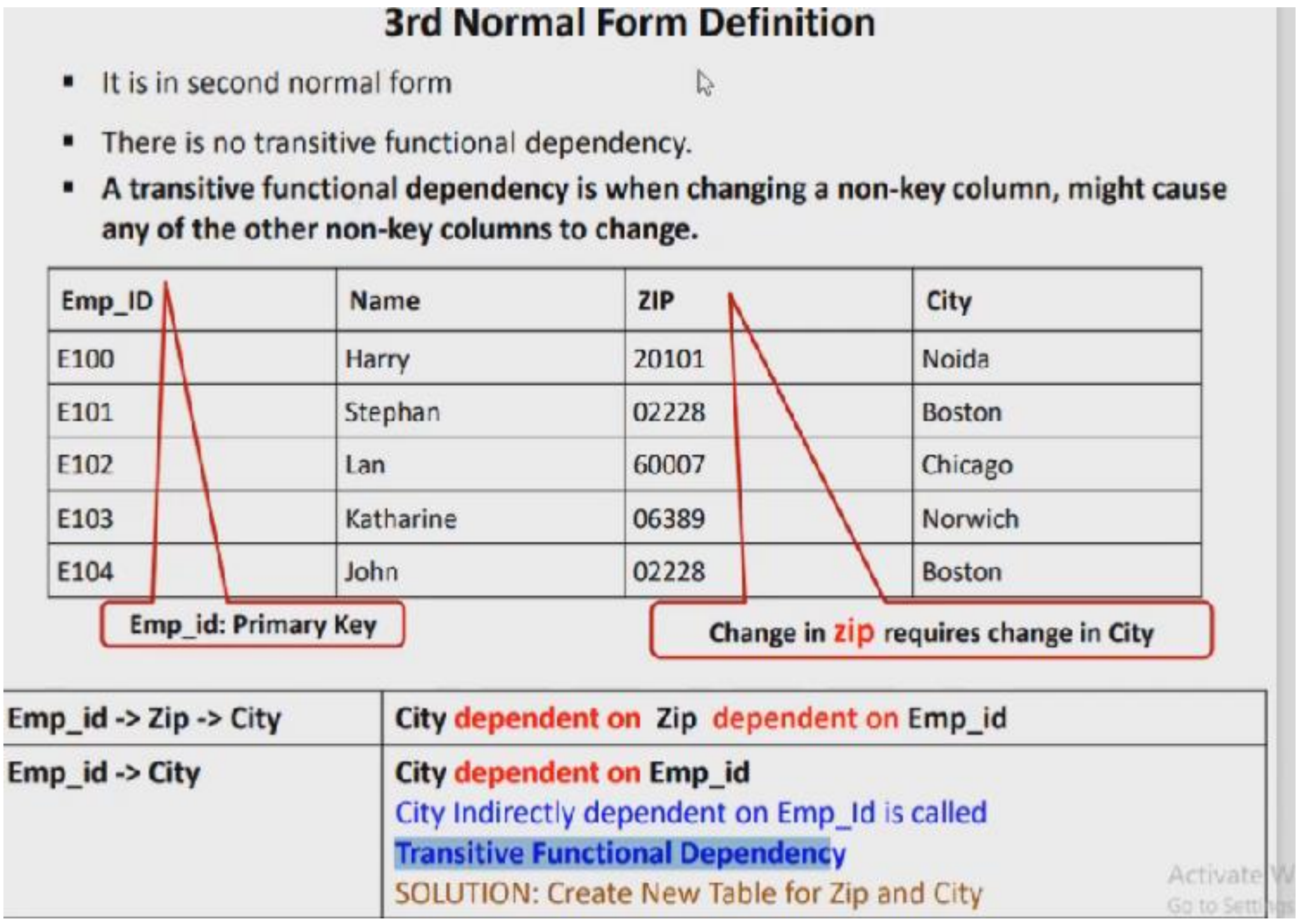
In the given table, non-prime attribute TEACHER_AGE is dependent on TEACHER_ID which is a proper subset of a candidate key. That's why it violates the rule for 2NF.

To convert the given table into 2NF, we decompose it into two tables:

Third Normal Form (3NF)

- A relation will be in 3NF if it is in 2NF and not contain any transitive partial dependency..
- If there is no transitive dependency for non-prime attributes, then the relation must be in third normal form.

A relation is in third normal form if it holds atleast one of the following conditions for every non-trivial function dependency X → Y.



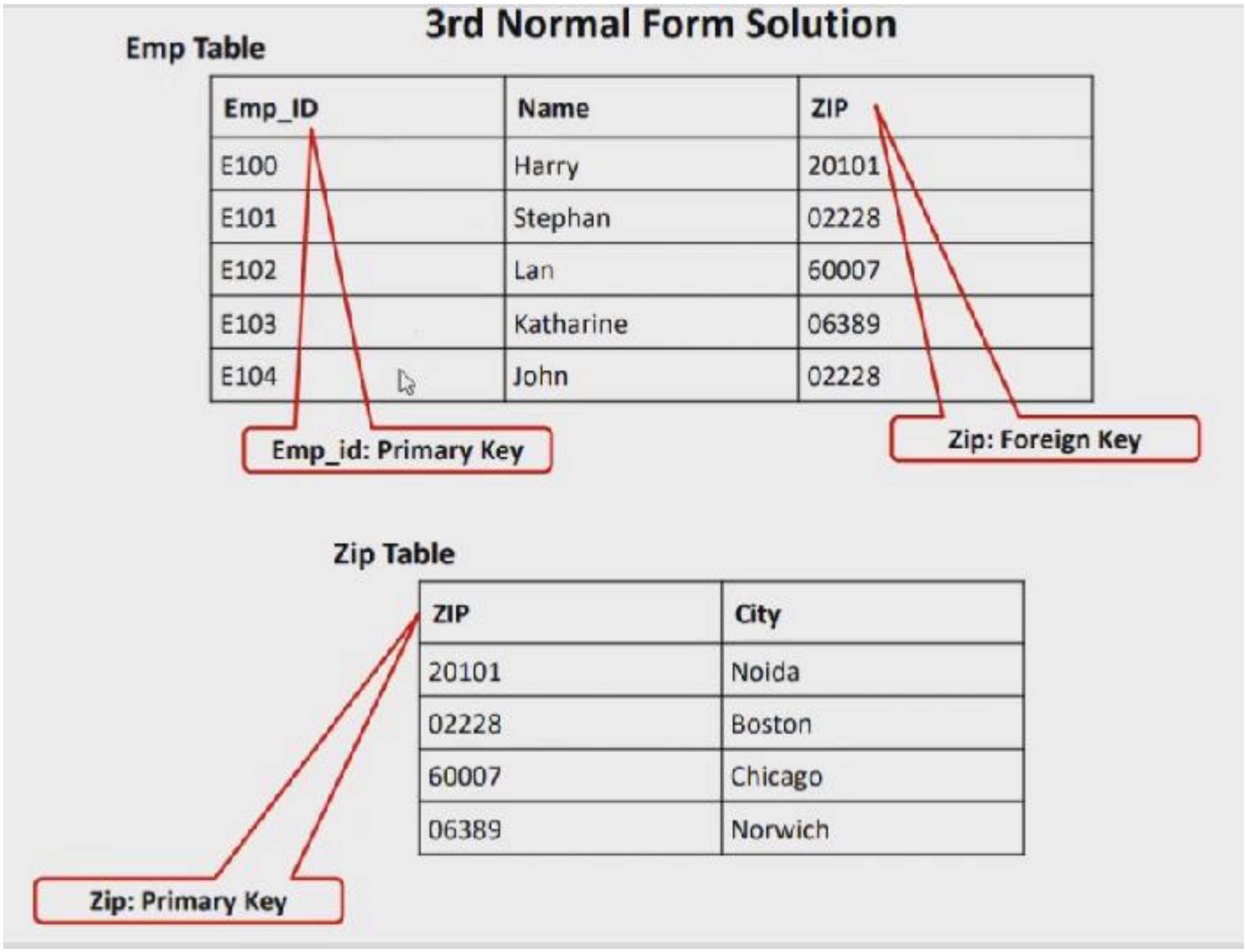
3rd Normal Form Definition

- It is in second normal form
- There is no transitive functional dependency.
- **A transitive functional dependency is when changing a non-key column, might cause any of the other non-key columns to change.**

Emp_ID	Name	ZIP	City
E100	Harry	20101	Noida
E101	Stephan	02228	Boston
E102	Lan	60007	Chicago
E103	Katharine	06389	Norwich
E104	John	02228	Boston

Emp_id -> Zip -> City	City dependent on Zip dependent on Emp_id
Emp_id -> City	City dependent on Emp_id City Indirectly dependent on Emp_Id is called Transitive Functional Dependency SOLUTION: Create New Table for Zip and City

Solution dived this table into two table with key relation



3rd Normal Form Solution

Emp Table

Emp_ID	Name	ZIP
E100	Harry	20101
E101	Stephan	02228
E102	Lan	60007
E103	Katharine	06389
E104	John	02228

Zip Table

ZIP	City
20101	Noida
02228	Boston
60007	Chicago
06389	Norwich

Emp_id pk and zip using as fk

Zip is pk with city attributes

Example:

EMPLOYEE_DETAIL table:

EMP_ID	EMP_NAME	EMP_ZIP	EMP_STATE	EMP_CITY
222	Harry	201010	UP	Noida
333	Stephan	02228	US	Boston

444	Lan	60007	US	Chicago
555	Katharine	06389	UK	Norwich
666	John	462007	MP	Bhopal

Super key in the table above:

1. {EMP_ID}, {EMP_ID, EMP_NAME}, {EMP_ID, EMP_NAME, EMP_ZIP}....so o
n

Candidate key: {EMP_ID}

Non-prime attributes: In the given table, 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). It violates the rule of third normal form.

That's why we need to move the EMP_CITY and EMP_STATE to the new <EMPLOYEE_ZIP> table, with EMP_ZIP as a Primary key.

EMPLOYEE table:

EMP_ID	EMP_NAME	EMP_ZIP
222	Harry	201010
333	Stephan	02228
444	Lan	60007

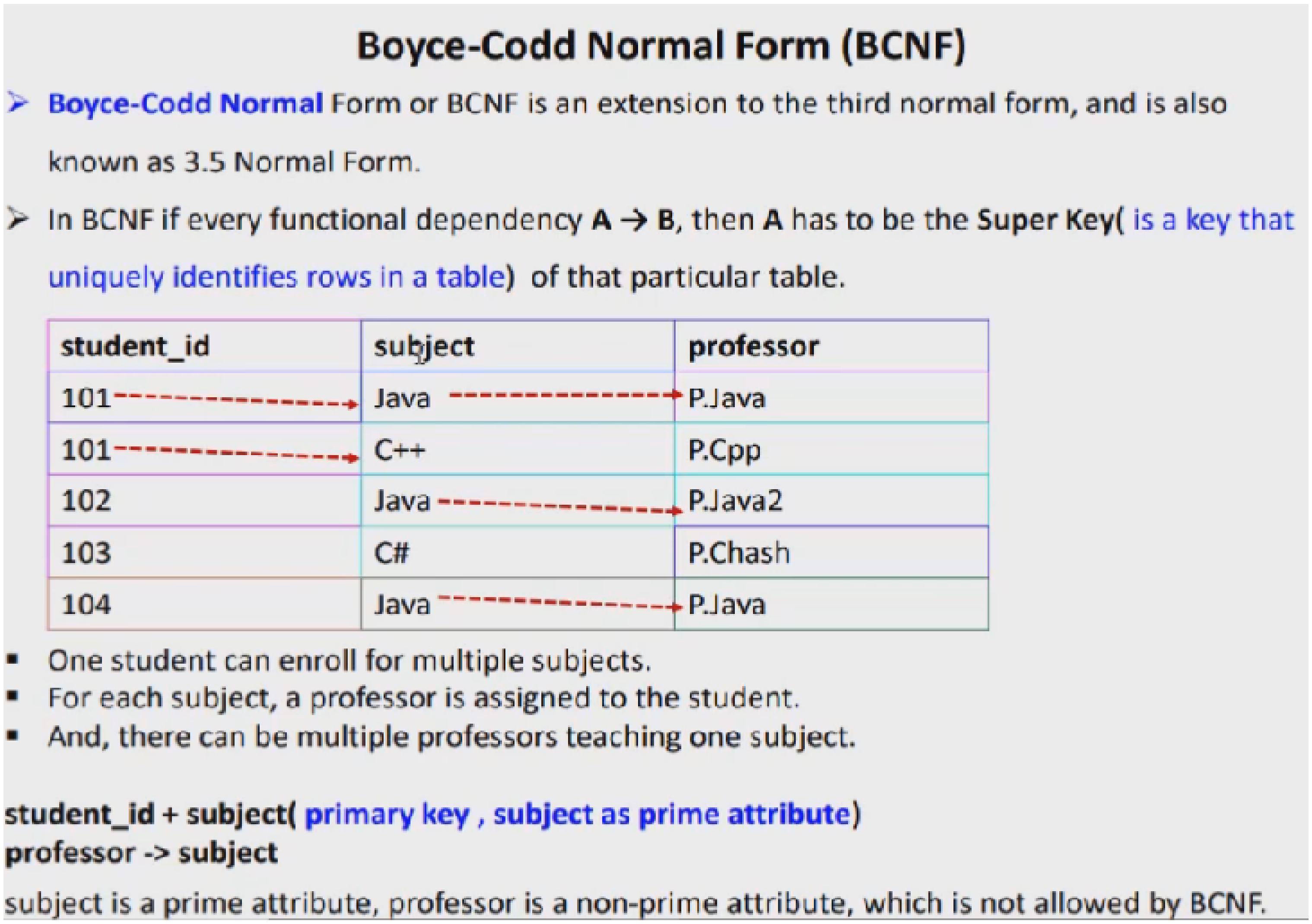
555	Katharine	06389
666	John	462007

EMPLOYEE_ZIP table:

EMP_ZIP	EMP_STATE	EMP_CITY
201010	UP	Noida
02228	US	Boston
60007	US	Chicago
06389	UK	Norwich
462007	MP	Bhopal

Boyce Codd normal form (BCNF)

- BCNF is the advance version of 3NF. It is stricter than 3NF.
- A table is in BCNF if every functional dependency X → Y, X is the super key of the table.
- For BCNF, the table should be in 3NF, and for every FD, LHS is super key.



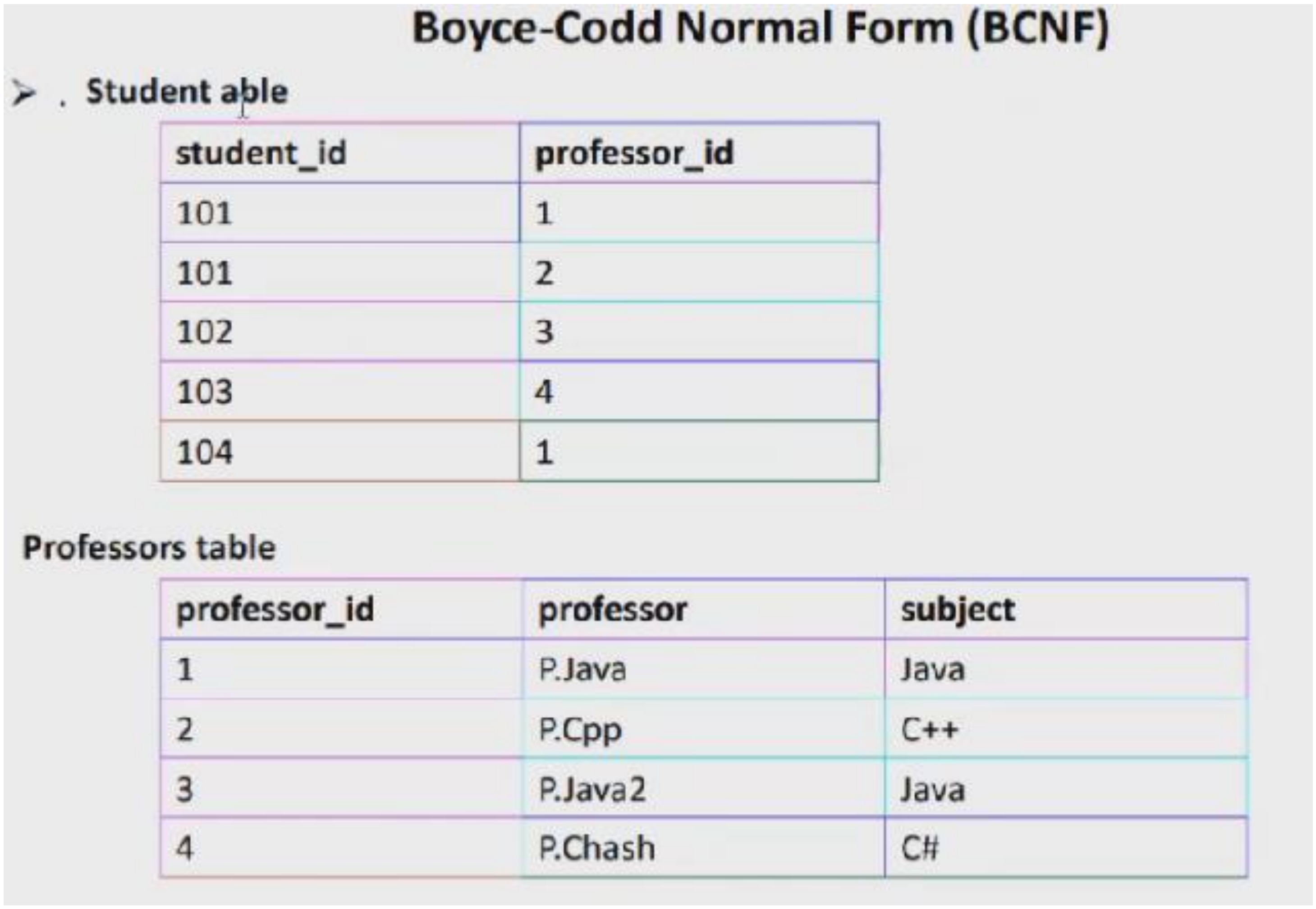
Boyce-Codd Normal Form (BCNF)

- **Boyce-Codd Normal** Form or BCNF is an extension to the third normal form, and is also known as 3.5 Normal Form.
- In BCNF if every functional dependency **A → B**, then **A** has to be the **Super Key(** is a key that uniquely identifies rows in a table) of that particular table.

student_id	**subject**	**professor**
101	Java	P.Java
101	C++	P.Cpp
102	Java	P.Java2
103	C#	P.Chash
104	Java	P.Java

- One student can enroll for multiple subjects.
- For each subject, a professor is assigned to the student.
- And, there can be multiple professors teaching one subject.

student_id + subject(primary key , subject as prime attribute)
professor -> subject
subject is a prime attribute, professor is a non-prime attribute, which is not allowed by BCNF.


Boyce-Codd Normal Form (BCNF)

- . Student able

student_id	professor_id
101	1
101	2
102	3
103	4
104	1

Professors table

professor_id	professor	subject
1	P.Java	Java
2	P.Cpp	C++
3	P.Java2	Java
4	P.Chash	C#

Example: Let's assume there is a company where employees work in more than one department.

EMPLOYEE table:

EMP_ID	EMP_COUNTRY	EMP_DEPT	DEPT_TYPE	EMP_DEPT_NO
264	India	Designing	D394	283
264	India	Testing	D394	300
364	UK	Stores	D283	232
364	UK	Developing	D283	549

In the above table Functional dependencies are as follows:

1. EMP_ID → EMP_COUNTRY
2. EMP_DEPT → {DEPT_TYPE, EMP_DEPT_NO}

Candidate key: {EMP-ID, EMP-DEPT}

The table is not in BCNF because neither EMP_DEPT nor EMP_ID alone are keys.

To convert the given table into BCNF, we decompose it into three tables:

EMP_COUNTRY table:

EMP_ID	EMP_COUNTRY
264	India
264	India

EMP_DEPT table:

EMP_DEPT	DEPT_TYPE	EMP_DEPT_NO
Designing	D394	283
Testing	D394	300
Stores	D283	232
Developing	D283	549

EMP_DEPT_MAPPING table:

EMP_ID	EMP_DEPT
D394	283

D394	300
D283	232
D283	549

Functional dependencies:

1. EMP_ID → EMP_COUNTRY
2. EMP_DEPT → {DEPT_TYPE, EMP_DEPT_NO}

Candidate keys:

For the first table: EMP_ID
For the second table: EMP_DEPT
For the third table: {EMP_ID, EMP_DEPT}

Now, this is in BCNF because left side part of both the functional dependencies is a key.

Normal Form	Description
1NF	A relation is in 1NF if it contains an atomic value.
2NF	A relation will be in 2NF if it is in 1NF and all non-key attributes are fully functional dependent on the primary key.
3NF	A relation will be in 3NF if it is in 2NF and no transition dependency exists.
BCNF	A stronger definition of 3NF is known as Boyce Codd's normal form.

4NF	A relation will be in 4NF if it is in Boyce Codd's normal form and has no multi-valued dependency.
5NF	A relation is in 5NF. If it is in 4NF and does not contain any join dependency, joining should be lossless.

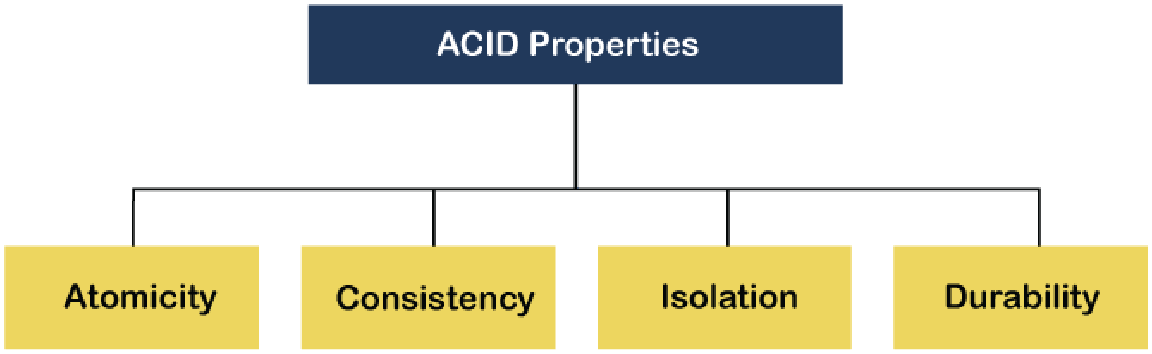
Advantages of Normalization

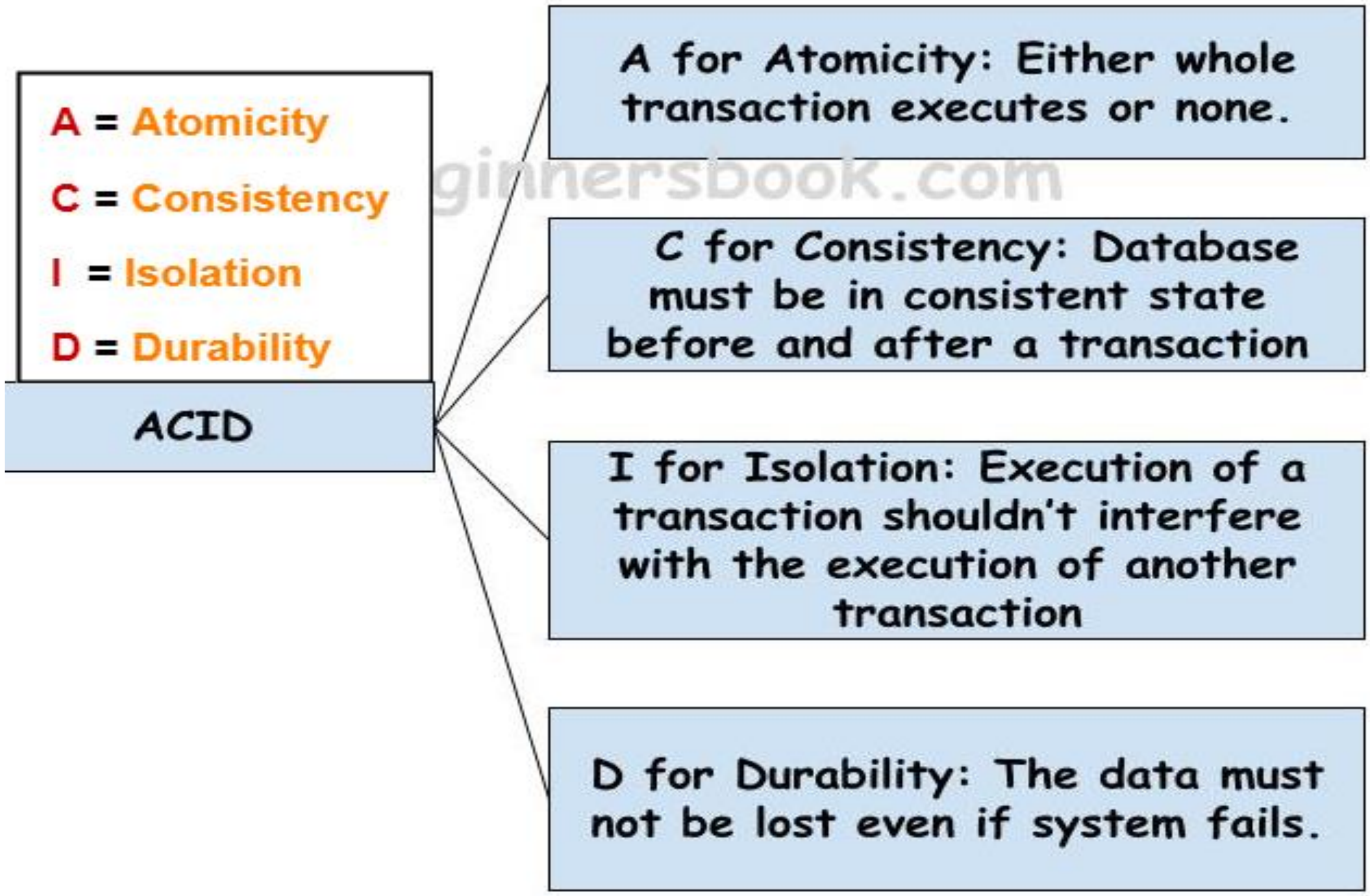
- Normalization helps to minimize data redundancy.
- Data consistency within the database.
- Much more flexible database design.
- Enforces the concept of relational integrity.

ACID Properties in DBMS

The ACID properties are meant for the transaction that goes through a different group of tasks, and there we come to see the role of the ACID properties

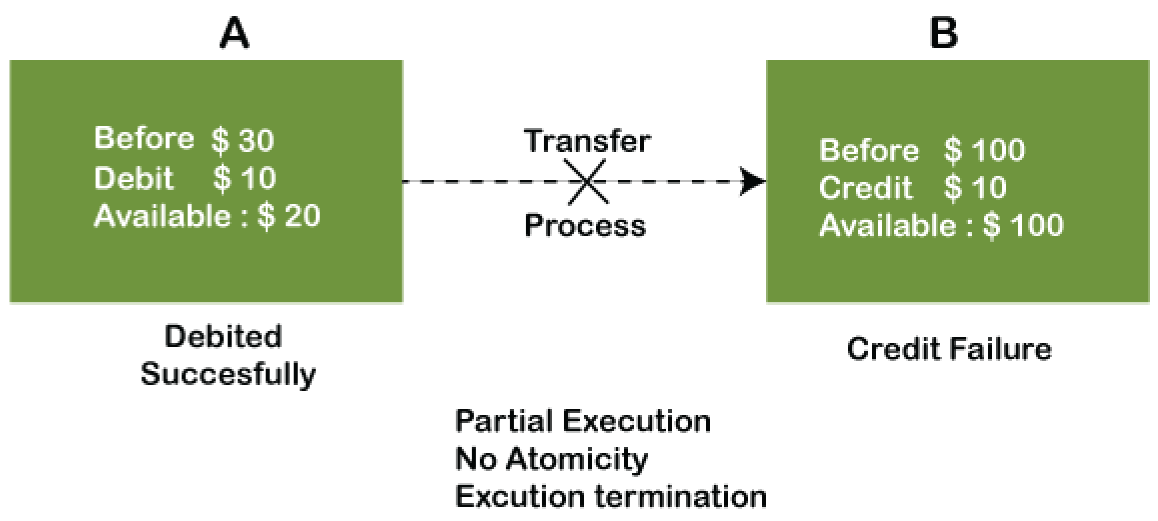
ACID Properties



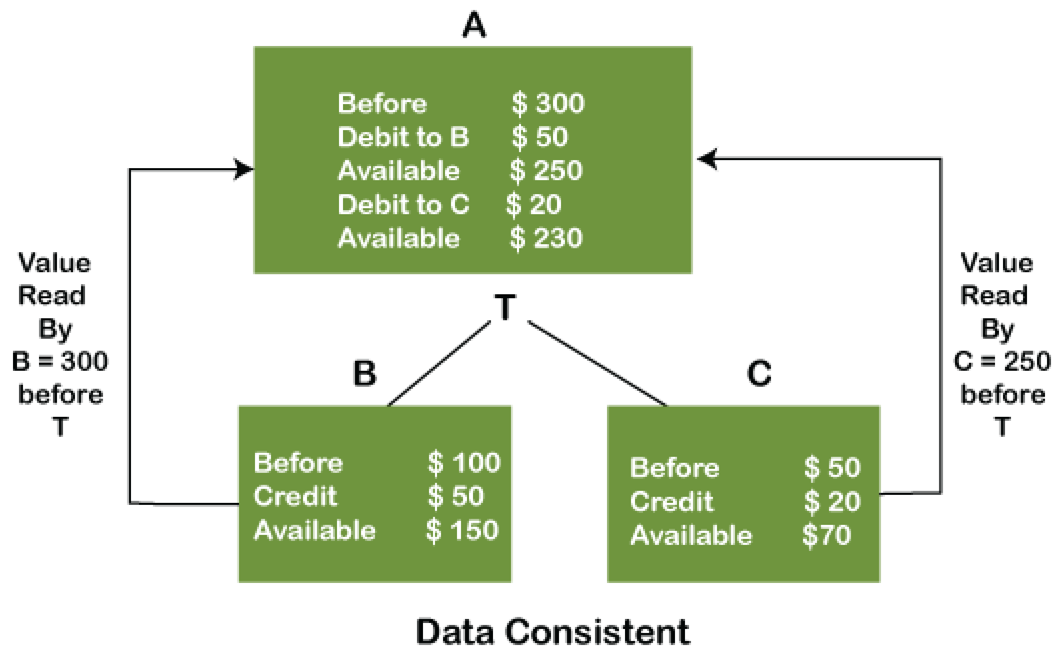

A = Atomicity
C = Consistency
I = Isolation
D = Durability
ACID
A for Atomicity: Either whole transaction executes or none.
C for Consistency: Database must be in consistent state before and after a transaction
I for Isolation: Execution of a transaction shouldn't interfere with the execution of another transaction
D for Durability: The data must not be lost even if system fails.

ACID Properties are used for maintaining the integrity of database during transaction processing. ACID in DBMS stands for **A**tomicity, **C**onsistency, **I**solation, and **D**urability.

Atomicity: A transaction is a single unit of operation. You either execute it entirely or do not execute it at all. There cannot be partial execution.

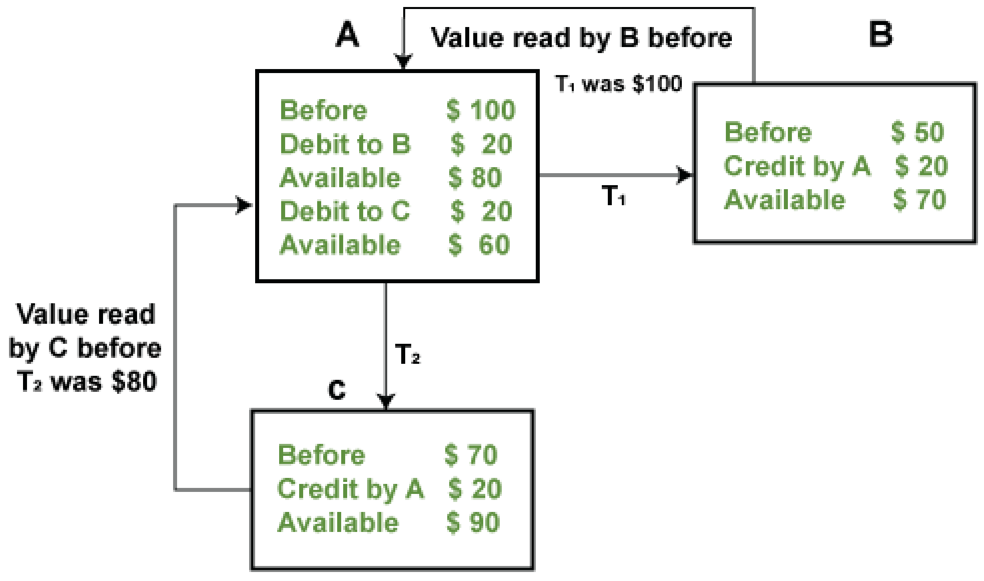


- **Consistency:** Once the transaction is executed, it should move from one consistent state to another.



-
- **Isolation:** Transaction should be executed in isolation from other transactions (no Locks). During concurrent transaction execution,

intermediate transaction results from simultaneously executed transactions should not be made available to each other. (Level 0,1,2,3)



Isolation - Independent execution of T_1 & T_2 by A

•

Durability: · After successful completion of a transaction, the changes in the database should persist. Even in the case of system failures.