DATABASE MANAGEMENT SYSTEM

JNTUH – R22 **CSE** **II Yr –II Sem**

UNIT – I

Data: Data refers to a collection of raw facts, unprocessed facts. It can be in various formats such as text, numbers, images, videos, audio, or any other type of digital or analog representation.

Ex: 25, Suresh, Hyderabad, etc

Information: Information refers to the processed data, and presented in a meaningful way to convey a specific message

Ex: the age of Suresh is 25

Database: A database is an organized collection of related data stored and accessed electronically through the use of a database management system.

Small databases can be stored on a file system, while large databases are hosted on computer clusters or cloud storage.

Meta Data: Metadata represents data about data. Metadata enriches the data with information, which makes it easier to discover, use and manage. In other words, metadata provides information about the characteristics, attributes, and properties of data.

Definition of DBMS

DBMS: DBMS Stands for **Database Management System**. It refers to a System / Software / Collection of programs that enables the creation, organization, and management of databases. A DBMS provides a structured and efficient way to store, retrieve, update, and manipulate large volumes of data.

Functionalities of DBMS:

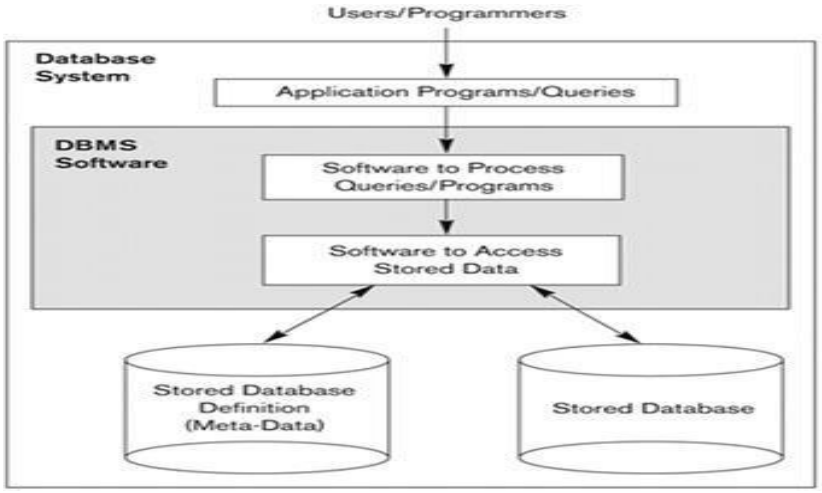
- **Define**: Specifying the data type, structures and constraints for the data to be stored.
- **Construct**: Process of storing data on some storage medium.
- **Manipulate**: Querying the database to retrieve specific data, updating database, and generating reports.
- **Share:** Allows multiple users and programs to access the database concurrently.
- **Protection**: Protection of the database from unauthorized access or from hardware and software failures and also maintenance of the database for long period of the time.

Properties of DBMS:

1. A database **represents** some aspects of the **real world** (mini world)
2. A database is a **logically coherent** collection of data with some inherent meaning.
3. A database is **designed, built** and **populated** with data for a **specific purpose**.

Database Environment:

A database system environment refers to the surroundings and components in which a database system operates. It includes various elements that interact with the database system to facilitate data storage, retrieval, management, and processing.

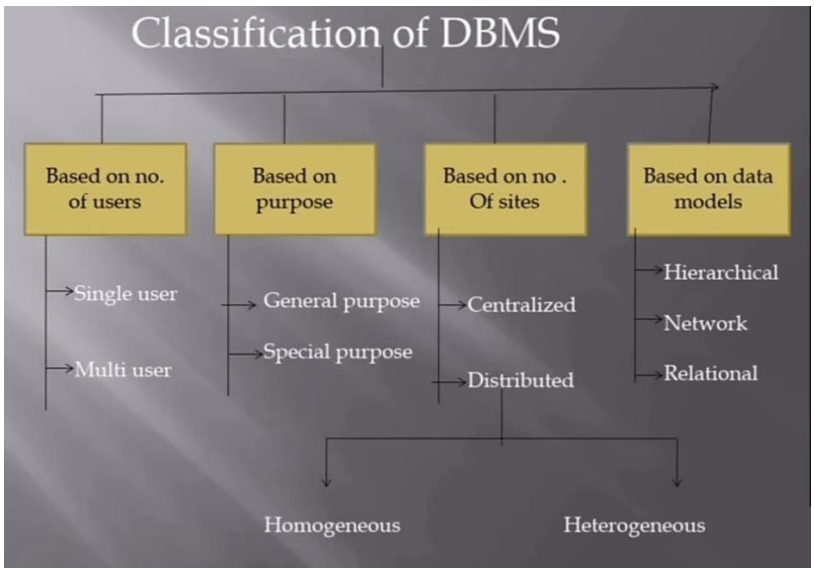


Here are some key components of a database system environment:

- Hardware
- Operating System
- Database Management System (DBMS)
- Database
- Application Programs
- Users
- Network
- Backup and Recovery Systems
- Security Infrastructure
- Data Integration and ETL Tools

Classification of DBMS

Database Management Systems (DBMS) can be classified into several categories based on different criteria. Here are some common classifications of DBMS:



Historical Perspective:

1. In the early 1960s, the first general-purpose DBMS, designed by **Charles Bachman**, was called the Integrated Data Store (IDS) which was based on network data model
2. In the late 1960s, IBM developed the Information Management System (IMS) DBMS. Which was based on hierarchical data model
3. In 1970, relational data model was developed by **Dr. E.F. Codd**. Representing data in the form of relations (tables)
4. In 1976, Peter Chan developed Entity Relationship(ER) model widely used in database design
5. In 1980, SQL (Structured Query Language) became standard.IBM launched DB2.
6. In 1990, Oracle, IBM DB2, and PostgreSQL introduced ORDBMS capabilities. Now various companies are developing and maintaining the different DBMS software.

Vendors	Products
Oracle Corporation	Oracle Database MySQL Oracle NoSQL Database Oracle TimesTen In-Memory Database Oracle Berkeley DB
Microsoft Corporation	Microsoft SQL Server Azure SQL Database Azure Cosmos DB Access (Desktop Database System)
IBM Corporation	IBM Db2 IBM Informix IBM Cloudant (NoSQL DB) IBM IMS (Hierarchical DB)
Amazon Web Services (AWS)	Amazon Aurora Amazon DynamoDB Amazon Redshift
SAP SE	SAP HANA SAP ASE (formerly Sybase ASE) SAP IQ (formerly Sybase IQ)

Database / DBMS Applications: Database Management Systems (DBMS) are widely used to build various types of database applications. Here are some common types of applications that make use of DBMS:

1. Universities
2. Banking
3. Railway Reservation System
4. Social Media Sites
5. Library Management System
6. E-commerce Websites
7. Medical
8. Accounting and Finance
9. Industries
10. Airline Reservation System
11. Telecommunication
12. Manufacturing
13. Human Resource Management
14. Broadcasting
15. Insurance

File based system versus a DBMS:

Basis	DBMS Approach	File System Approach
Meaning	DBMS is a collection of data. In DBMS, the user is not required to write the procedures.	The file system is a collection of data. In this system, the user has to write the procedures for managing the database.
Sharing of data	Due to the centralized approach, data sharing is easy.	Data is distributed in many files, and it may be of different formats, so it isn't easy to share data.
Data Abstraction	DBMS gives an abstract view of data that hides the details.	The file system provides the detail of the data representation and storage of data.
Security and Protection	DBMS provides a good protection mechanism.	It isn't easy to protect a file under the file system.
Cost	The database system is expensive to design.	The file system approach is cheaper to design.
Data Redundancy and Inconsistency	Due to the centralization of the database, the problems of data redundancy and inconsistency are controlled.	In this, the files and application programs are created by different programmers so that there exists a lot of duplication of data which may lead to inconsistency.
Structure	The database structure is complex to design.	The file system approach has a simple structure.
Data Independence	In this system, Data Independence exists,	In the File system approach, there exists no Data Independence.
Integrity Constraints	Integrity Constraints are easy to apply.	Integrity Constraints are difficult to implement in file system.
Flexibility	The changes are more easily with a database approach.	less flexibility
Examples	Oracle, SQL Server, Sybase etc.	Cobol, C++ etc.

What is DBA? What are the roles and responsibilities of DBA?

DBA stands for Database Administrator. A person who has central control over the database system, DBA may be a single or group of persons who are responsible for implementing the database system within an Organization. DBA is only the guardian of the organization but not the owner.

Roles and responsibilities of DBA

1. Responsible for Design, Implementation and maintenance of database
2. Software installation and maintenance
3. Data Extraction, Transformation and loading
4. Specialized in data handling
5. Database backup & recovery
6. Security
7. Authentication
8. Performance monitoring
9. Data storage
10. Integrity constraint

Advantage and disadvantage of DBMS:

Advantages of DBMS:

1. **Data Centralization**: DBMS allows for the centralization of data, meaning that data is stored in a single location and can be accessed by multiple users and applications. This reduces data redundancy and inconsistency.
2. **Data Consistency and Integrity**: DBMS enforces integrity constraints and rules on the data to ensure consistency.
3. **Data Security**: DBMS offers security features to protect sensitive data. It allows for user authentication, authorization, and access control, ensuring that only authorized users can access and modify the data.
4. **Data Sharing and Collaboration**: DBMS enables concurrent access to the database by multiple users and applications.
5. **Data Independence**: This allows for modifications in the physical storage without affecting the logical view, providing data independence.

Disadvantages of DBMS:

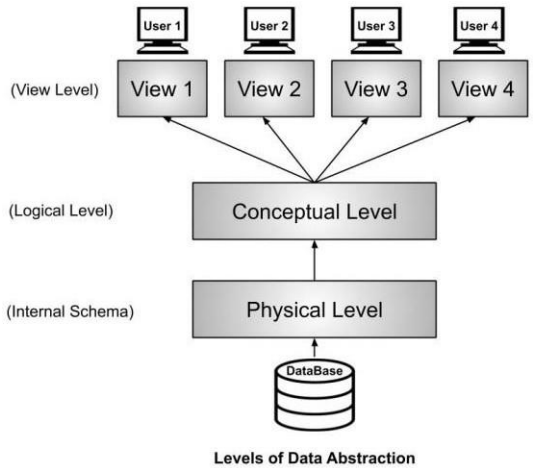
1. **Complexity**: Implementing and managing a DBMS can be complex and requires specialized knowledge and expertise.
2. **Cost**: Acquiring and maintaining a DBMS can be expensive.
3. **Performance Overhead**: DBMS introduces overhead in terms of processing and resource utilization.
4. **Single Point of Failure**: If the DBMS experiences a failure or downtime, it can disrupt access to the entire database, affecting all applications and users dependent on it.
5. **Learning Curve**: Users and developers need to learn the specific query languages and programming interfaces supported by the DBMS.

Data Abstraction and its phases (levels):

Data Abstraction is a process of hiding unwanted or irrelevant details from the end user. Data abstraction allows users to interact with the database without needing to understand the underlying complexities. It provides a conceptual framework that separates the logical view of the data from its physical storage. This separation enables users to focus on the essential aspects of the data and its manipulation, rather than being concerned with the technical details.

There are typically three levels of data abstraction in a DBMS:

1. **External Level (also known as the View Level):** This level represents the individual user's view of the database. It defines how each user or application sees and interacts with the data. Users can define their own customized views of the data by specifying the desired data fields, their relationships, and the operations they want to perform. The external level provides a simplified and tailored view of the database for different users, which enhances data security and privacy.
2. **Conceptual Level (also known as the Logical Level):** This level represents the overall logical structure and organization of the entire database. It focuses on the relationships between different data elements and provides a global view of the data. The conceptual level defines the entities, attributes, and their relationships using a data model (such as the Entity-Relationship Model or the Relational Model). It acts as an intermediary between the external and internal levels and ensures data independence, where changes in the physical storage or organization of data do not affect the external views.
3. **Internal Level (also known as the Physical Level):** This level represents the physical storage and implementation details of the database on the underlying hardware. It describes how the data is stored, indexed, and accessed by the DBMS. The internal level deals with low-level details such as data storage structures, file organization, and access methods. It optimizes data storage and retrieval for efficient performance and supports the logical and external levels by providing the necessary data manipulation capabilities.



Levels of Data Abstraction

Data Independence in DBMS:

Data independence is the ability to modify the scheme without affecting the programs and the application to be rewritten. Data is separated from the programs, so that the changes made to the data will not affect the program execution and the application.

We know the main purpose of the three levels of data abstraction is to achieve data independence. If the database changes and expands over time, it is very important that the changes in one level should not affect the data at other levels of the database. This would save time and cost required when changing the database.

There are two levels of data independence based on three levels of abstraction. These are as follows –

- Physical Data Independence
- Logical Data Independence

Physical Data Independence:

Physical Data Independence means changing the physical level without affecting the logical level or conceptual level. Using this property, we can change the storage device of the database without affecting the logical schema.

The changes in the physical level may include changes using the following –

- A new storage device like magnetic tape, hard disk, etc
- A new data structure for storage
- A different data access method or using an alternative files organization technique.
- Changing the location of the database.

Logical Data Independence:

Logical view of data is the user view of the data. It presents data in the form that can be accessed by the end user.

Codd's Rule of Logical Data Independence says that users should be able to manipulate the Logical View of data without any information of its physical storage. Software or the computer program is used to manipulate the logical view of the data.

The changes in the logical level may include −

- Change the data definition.
- Adding, deleting, or updating any new attribute, entity or relationship in the database.

Data Models:

Data models describe how a database's logical structure is represented. In a database management system, data models are essential for introducing abstraction. Data models specify how data is linked to one another, as well as how it is handled and stored within the system.

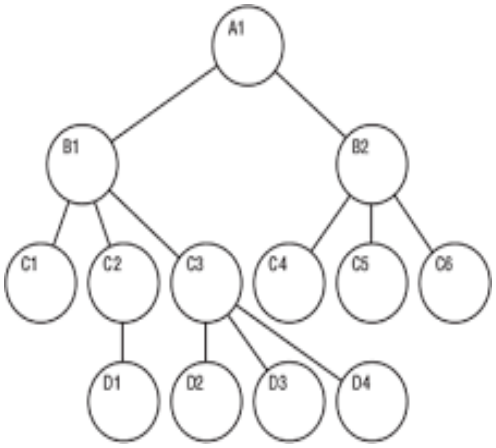
A data model is a conceptual representation of how data is organized, stored, and accessed within a database. It defines the structure, constraints, and relationships of the data.

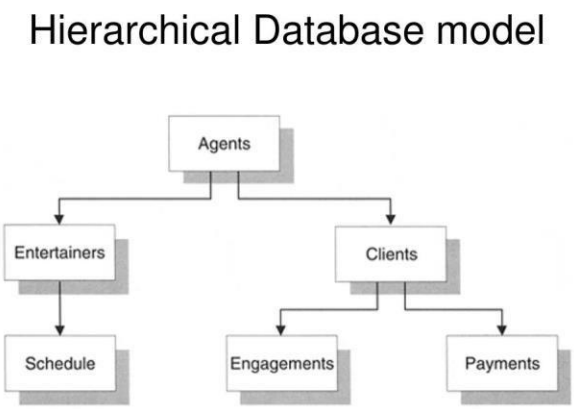
There are several types of data models commonly used in DBMS

1. Hierarchical Data Model
2. Network Data Model
3. Entity-Relationship (ER) Model
4. Relational Data Model
5. Object-Oriented Data Model

1. Hierarchical Data Model:

The hierarchical model organizes data in a tree-like structure, where each record has a parent-child relationship. It consists of nodes connected by parent-child relationships, forming a hierarchy.





Advantages of the Hierarchical Model:

1. **Simplicity**: The hierarchical model is relatively simple to understand and implement.
2. **Efficiency**: Retrieving data using the hierarchical model can be efficient, especially when navigating through parent-child relationships.
3. **Data Integrity**: The hierarchical model enforces data integrity by maintaining strict parent-child relationships. It ensures that each child record has a single parent, preventing anomalies and inconsistencies in the data.

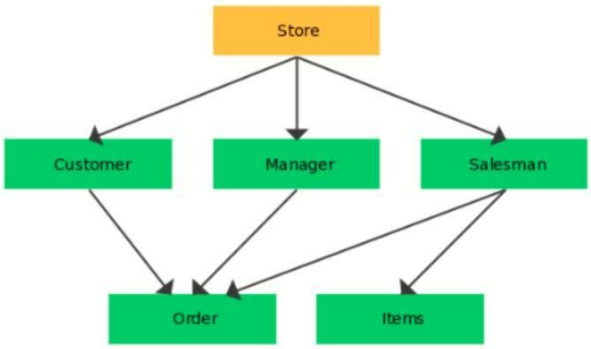
Disadvantages of the Hierarchical Model:

1. **Lack of Flexibility**: Drawbacks of the hierarchical model is its lack of flexibility. The strict one-to-many parent-child relationship can be limiting when dealing with complex or evolving data structures.
2. **Data Redundancy**: In the hierarchical model, data redundancy can occur since the same data might need to be duplicated across multiple parent-child relationships

2. **Network Data Model:**

Data Organization in the network model is just like a graph rather than a tree, a child record can have any number of parent records

This was the most widely used model before the relational database model was introduced



Advantages of Network Model :

1. This model is very simple and easy to design like the hierarchical data model.
2. This model is capable of handling multiple types of relationships which can help in modeling real-life applications, for example, 1: 1, 1: M, M: N relationships.
3. In this model, we can access the data easily, and also there is a chance that the application can access the owner's and the member's records within a set.

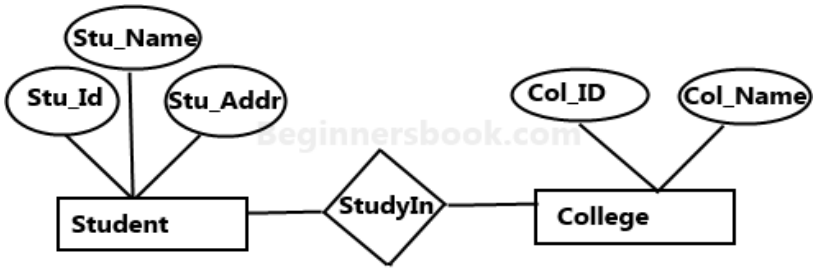
Disadvantages of Network Model:

1. The schema or the structure of this database is very complex in nature as all the records are maintained by the use of pointers.
2. There's an existence of operational anomalies as there is a use of pointers for navigation which further leads to complex implementation.
3. The design or the structure of this model is not user-friendly.

3. **Entity-Relationship (ER) Model**

The ER diagram was proposed by Peter Chen in 1976 as a visual tool to represent the ER model. ER model in DBMS is the high-level data model. It stands for the Entity- relationship model and is used to represent a logical view of the system from a data perspective. It works around real-world entities and the associations among them.

ER model makes use of **ER diagrams**, which are the diagrams sketched to design a database. ER diagrams are built on **three** basic concepts: **entities, attributes,** and **relationships** between them. An ER diagram in DBMS defines **entities, associated attributes**, and **relationships between entities**. This helps visualize the logical structure of the database.



Sample E-R Diagram

Advantages of the ER Model:

1. **Simplicity**: The ER model uses a simple and intuitive graphical representation, which makes it easy to understand and communicate the database structure to stakeholders.
2. **Clear representation of relationships**: The ER model provides a clear depiction of the relationships between entities, such as one-to-one, one-to-many, and many-to-many relationships. This helps in understanding the data dependencies and designing the database accordingly.
3. **Database design**: The ER model assists in the logical design phase of database development. It helps in identifying entities, attributes, and relationships, which serve as a foundation for creating a well-structured database schema.

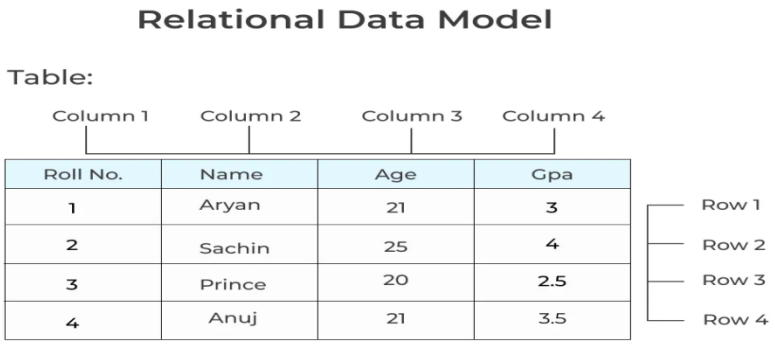
Disadvantages of the ER Model:

1. **Lack of support for implementation details**: The ER model is a high-level conceptual model and does not provide details about implementation aspects such as data types, indexing, or physical storage considerations.
2. **Limited support for behavior modeling**: The ER model primarily focuses on the static structure of the database and does not explicitly capture the dynamic behavior or operations performed on the data. It lacks constructs to represent actions, triggers, or procedures.

4. Relational data model

The relational data model is a conceptual model used to organize and structure data in a database. It was introduced by Edgar F. Codd in the early 1970s and has since become the most widely used database model

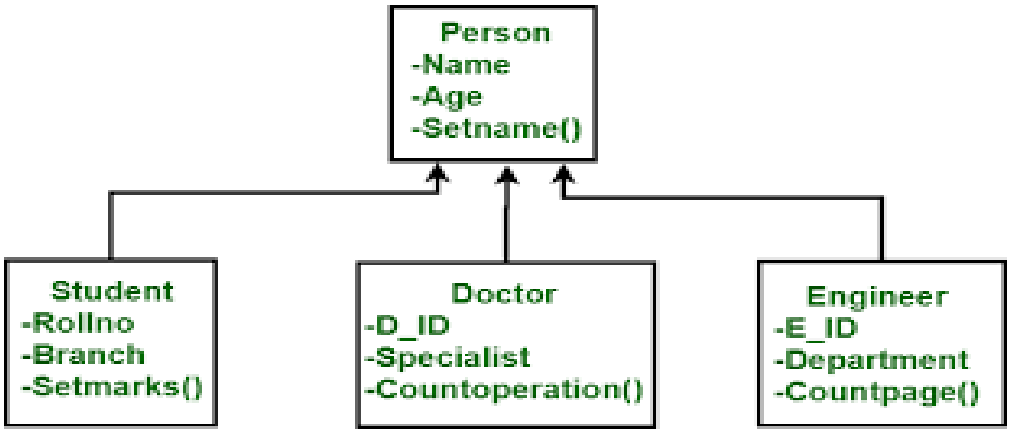
The relational model uses a collection of tables to represent data with columns and rows. Tables are also known as relations.



5. Object-oriented data model:

The object-oriented data model is a data model that represents data in the form of objects. Objects are instances of classes and represent real-world entities.

Both data and the data relationships are stored into a single structure that's known as an object in the object-oriented data model (or OODM).

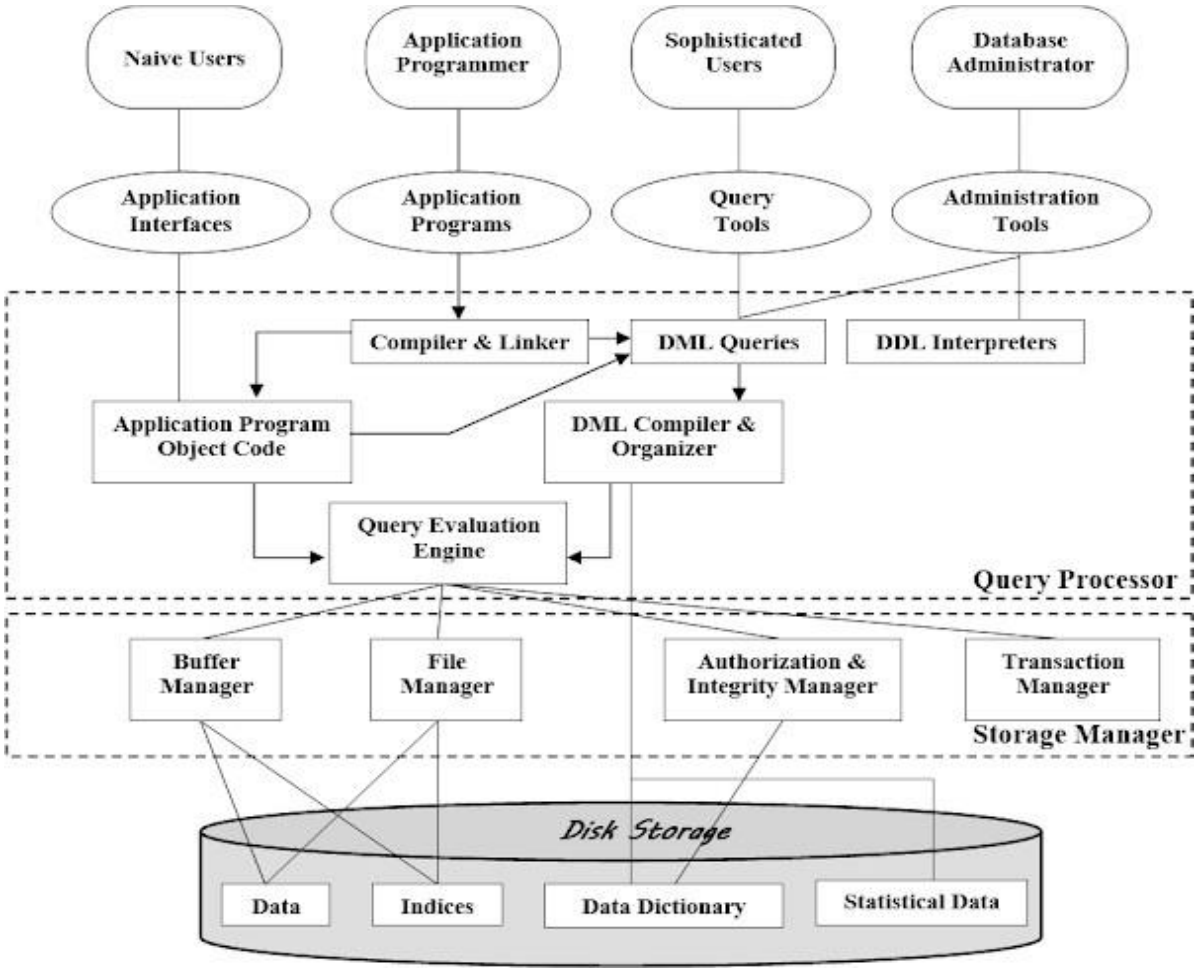


STRUCTURE OF DBMS

DBMS (Database Management System) acts as an interface between the user and the database. The user requests the DBMS to perform various operations (insert, delete, update and retrieval) on the database. The components of DBMS perform these requested operations on the database and provide necessary data to the users. The various components of DBMS are shown below:

Four Parts that make up the Database System are:

- Users
- Query Processor
- Storage Manager
- Disk Storage



Users:

Database Users: Users are differentiated by the way they expect to interact with the system

Application programmers:

- Application programmers are computer professionals who write application programs. Application programmers can choose from many tools to develop user interfaces.

Sophisticated users:

- Sophisticated users interact with the system without writing programs. Instead, they form their requests in a database query language.
- They submit each such query to a query processor, whose function is to break down DML statements into instructions that the storage manager understands.

Naïve users:

- Naive users are unsophisticated users who interact with the system by invoking one of the application programs that have been written previously.
- For example, a bank teller who needs to transfer Rs. 50/- from account A to account B invokes a program called transfer. This program asks the teller for the amount of money to be transferred, the account from which the money is to be transferred, and the account to which the money is to be transferred.

Database Administrator:

- Database administrator has full control on database. They may be a group of persons or a single person. They use Data Definition Language to pass on schemas and mapping constraints on database, which further passed to DDL processor.

Query Processor:

The query processor will accept query from user and solves it by accessing the database.

Parts of Query processor:

- **DDL interpreter**
  This will interprets DDL statements and fetch the definitions in the data dictionary.
- **DML compiler**
  a. This will translates DML statements in a query language into low level instructions that the query evaluation engine understands.
  b. A query can usually be translated into any of a number of alternative evaluation plans for same query result DML compiler will select best plan for query optimization.
- **Query evaluation engine**
  This engine will execute low-level instructions generated by the DML compiler on DBMS.

The storage manager components include:

- **Authorization and integrity manager**: Checks for integrity constraints and authority of users to access data.
- **Transaction manager**: Ensures that the database remains in a consistent state although there are system failures.
- **File manager**: Manages the allocation of space on disk storage and the data structures used to represent information stored on disk.
- **Buffer manager**: It is responsible for retrieving data from disk storage into main memory. It enables the database to handle data sizes that are much larger than the size of main memory.

Disk Storage: The Disk Storage in the Structure of DBMS represents the space where data is stored. It has the following components:

- **Files**: These are responsible for storing the data.
- **Data Dictionary**: It is the repository that maintains the information of the database object and maintains the metadata.
- **Indices**: These are the keys that are used for faster retrieval of data.

Database Design: Database design is the organization of data according to a database model. Properly designed databases are easy to maintain, improves data consistency. The database design process can be divided into six steps.

1. Requirement analysis
2. Conceptual database design
3. Logical database design
4. Schema refinement
5. Physical database design
6. Application and security design

1. Requirement analysis

- ❖ It is necessary to understand what data need to be stored in the database, what applications must be built, what are all those operations that are frequently used by the system.
- ❖ The requirement analysis is an informal process and it requires proper communication with user groups.
- ❖ There are several methods for organizing and presenting information gathered in this step. Some automated tools can also be used for this purpose.

2. Conceptual database design

- ❖ The information gathered, is used to develop a high-level description of the data to be stored in the database
- ❖ This is a steps in which E-R Model i.e. Entity Relationship model is built.
- ❖ The goal of this design is to create a simple description of data that matches with the requirements of users.

3. Logical database design

- ❖ This is a step in which ER model in converted to relational database schema, sometimes called as the logical schema in the relational data model.

4. Schema refinement

- ❖ In this step, relational database schema is analyzed to identify the potential problems and to refine it.
- ❖ The schema refinement can be done with the help of normalizing and restructuring the relations.

5. Physical database design

- ❖ The design of database is refined further.
- ❖ This step may simply involve building indexes on tables and clustering tables, redesign of parts of the database schema obtained from the earlier design steps.

6. Application and security design

- ❖ Using design methodologies like UML(Unified Modeling Language) try to address the complete software design of the database can be accomplished.
- ❖ The role of each entity in every process must be reflected in the application task.
- ❖ For each role, there must be the provision for accessing and prohibiting some part of database.
- ❖ Thus some access rules must be enforced on the application(which is accessing the database) to protect the security features.

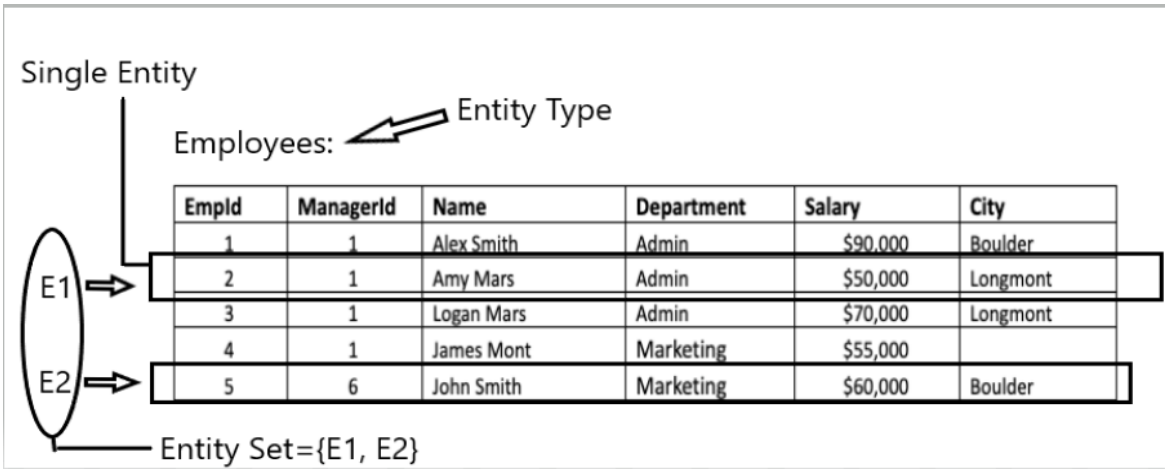
Entity: An entity refers to a real-world object or concept about which we want to collect and store the data that is being represented in the database. It may be a living thing or nonliving thing.

Example: STUDENT, EMPLOYEE, PRODUCT, CUSTOMER, ORDER, etc.

- In ERD's entities are represented using a rectangular box
- Entity name should be a noun in singular form
- Entity names are represented in all capital letters

Entity Type: It refers to the category that a particular entity belongs to.

Entity set: The group of related entities of an entity type which share all common attributes is called entity set

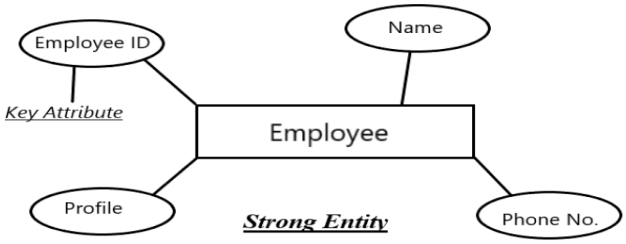


EmpId	ManagerId	Name	Department	Salary	City
1	1	Alex Smith	Admin	$90,000	Boulder
2	1	Amy Mars	Admin	$50,000	Longmont
3	1	Logan Mars	Admin	$70,000	Longmont
4	1	James Mont	Marketing	$55,000	
5	6	John Smith	Marketing	$60,000	Boulder

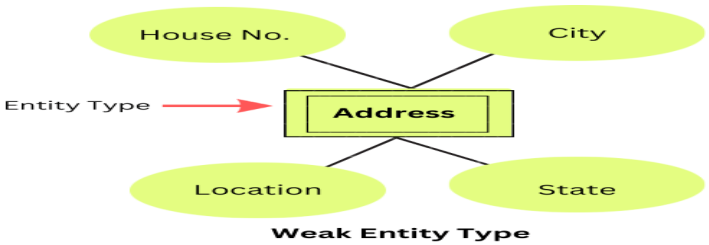
It can be classified into two types

1. Strong Entity
2. Weak Entity

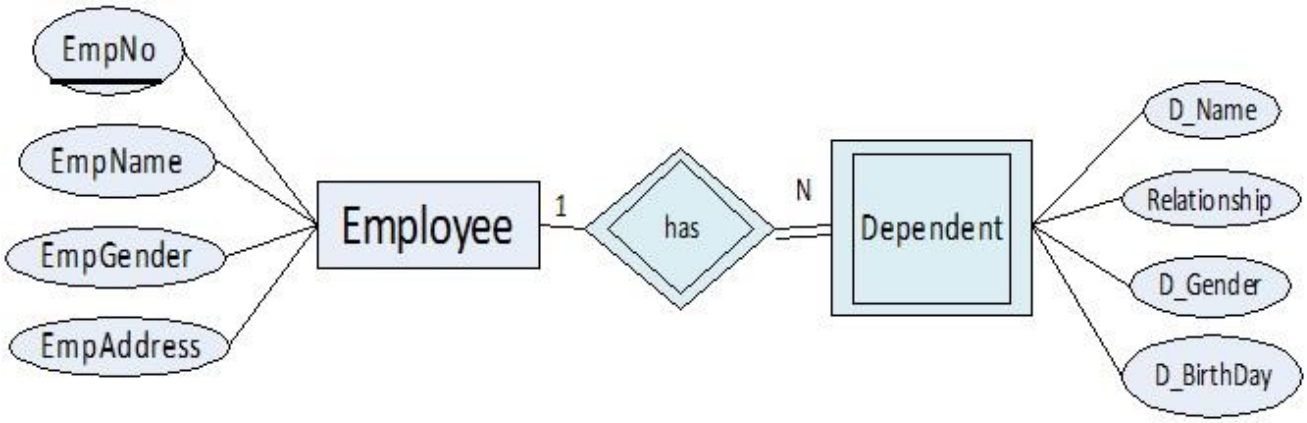
1. Strong Entity: A strong entity is a standalone entity that does not require support from any other entity. A strong entity is an entity type that has a key attribute (primary key), and it is represented by a single rectangle in the ER diagram.



2. Weak Entity: Contrary to a strong entity, a weak entity is dependent and cannot exist independently. It requires the support of a strong entity to validate its existence. Weak entity type doesn't have a key attribute and it is represented by a double rectangle in the ER diagram.



Example:

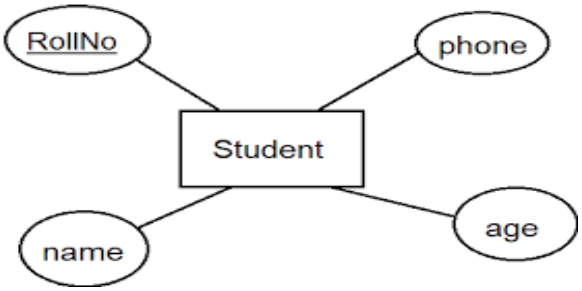


In the above example **Employee** is a strong entity and **Dependent** is a weak entity

Attribute: An attribute is a property, characteristic or feature of an entity. An entity may contain any number of attributes.

Example: Student has attributes like name, age, roll number, and many more.

- In an Entity-Relation model, attributes are represented in an oval or ellipse shape. 
- Attribute names are represented with initial capital letter and remaining small



Types of attributes:

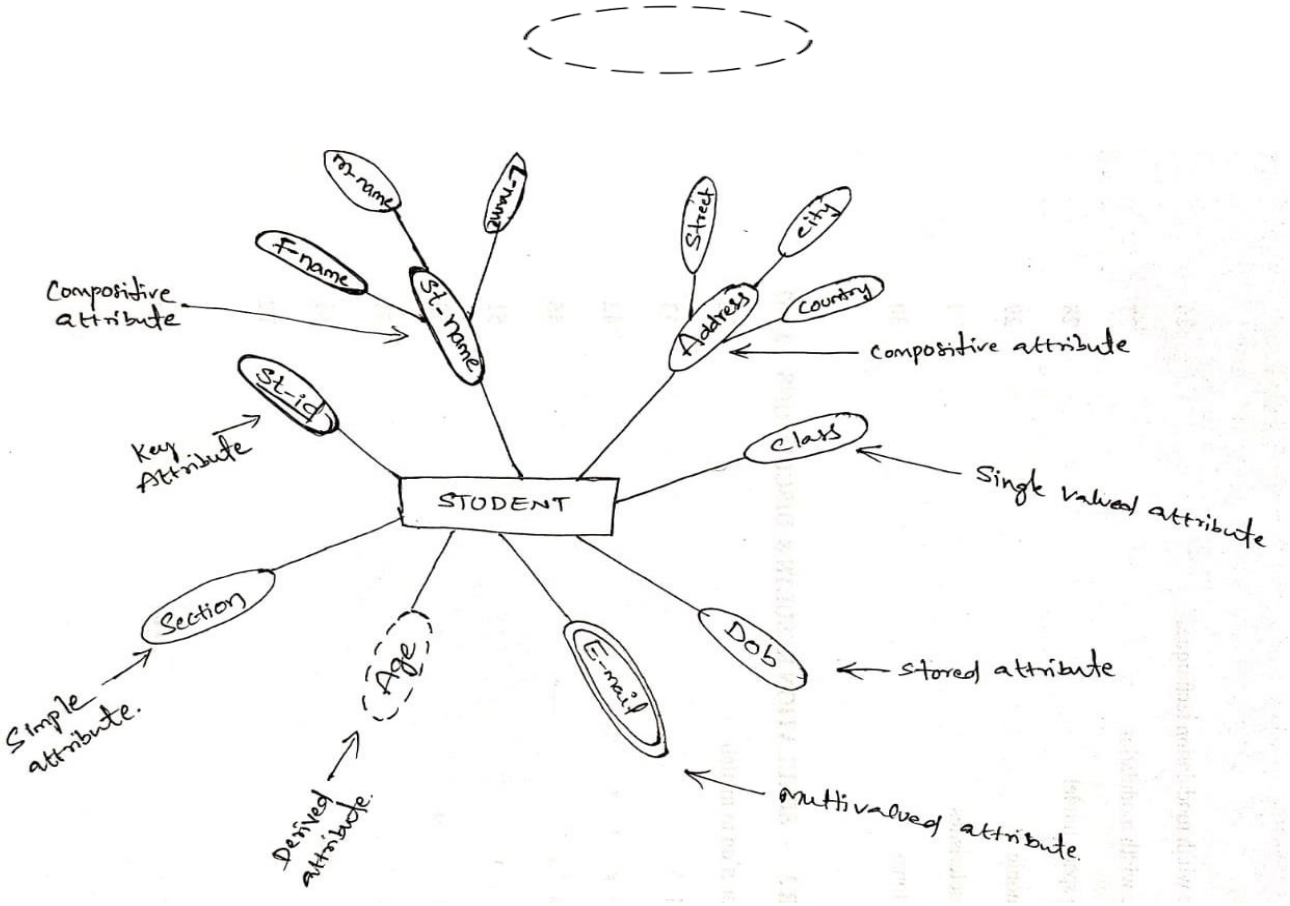
There are several types of attributes in DBMS, each with its unique characteristics and applications. Following are some of the types of attributes-

1. Key Attributes
2. Simple Attributes
3. Composite Attributes
4. Single-valued attributes
5. Multi-valued attributes
6. Stored attribute
7. Derived attribute

1. **Key Attributes:**
    - Attributes that define the originality of the entity,
    - Key attributes are those attributes that can uniquely identify the entity in the entity set.
    - In a student database, for example, the roll number can be the key property that uniquely identifies each student.
    - The key attribute is represented using an oval with the attribute name **underlined (_____)** inside the oval.
2. **Simple Attributes:**
    - An attribute which can not be diveded to yield additional attributes/component is called simple attributes (or) An attribute that cannot be further subdivided into components is a simple attribute
    - For example, in a student table, the branch attribute cannot be further divided.
3. **Composite Attributes:**
    - An attribute which can be diveded to yield additional new attributes/component is called Composite attributes (or) An attribute that can be split into new attributes/component is called a composite attribute
    - The address can be further split into house number, street number, city, state, country, and pin code, the name can also be split into first name middle name, and last name.
4. **Single-valued attributes:**
    - Those attributes which can have exactly one value are known as single valued attributes. They contain singular values, so more than one value is not allowed.
    - Example: The age of a student. Another example is gender because one person can have only one gender.
5. **Multi-valued attributes:**
    - Those attributes which can have more than one entry or which contain more than one value are called multi valued attributes.
    - In the Entity Relationship (ER) diagram, we represent the multi valued attribute by **double oval** representation. 
    - For example, one person can have more than one phone number, so that it would be a multi valued attribute. Another example is the hobbies of a person because one can have more than one hobby.
6. **Stored attribute:**
    - Stored attribute is an attribute which supplies value for another attribute is called stored attribute
    - Stored attributes represent data that remains constant and fixed for an entity instance. They provide the foundation for deriving derived attributes.
    - Example: DOB(Date of birth) is the stored attribute. We can find the Age (attribute value) by using DOB (attribute value) i.e. DOB supplies values to Age, so DOB is stored attribute.

7. Derived attribute

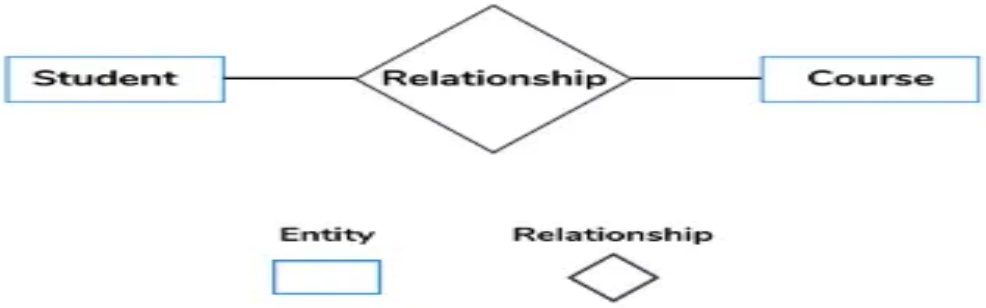
- Derived attributes are those attributes which can be derived from other attribute(s).
- Derived attributes are the attributes that do not exist in the physical database, but their values are derived from other attributes present in the database.
- For example, age of the student should not be saved directly in the database, instead it can be derived from Date of Birth.
- In the ER diagram, the **dashed oval** represents the derived attribute



Relationship:

A relationship is the meaningful assossiation among the instanes of two or more entity types / same entity type. Relationships define how data in different tables are connected or related to each other.

- In ERD relationships are reprasented using Rhambus /Diamond shape.
- Relationship name shoud be a verb.



Relationship Set:

The group of (or) the no.of relationships participated in a relationship type is called relatinship set.

(Or)

A collection of various relationships that belongs to the same relationship type is called a relationship set.

Degree of Relationship:

A degree of relationship represents the number of entity types that are associated with a relationship.

Cardinality of relationship:

Cardinality of a relationship can be defined as the no.of entities of an entity type having the assosiation with no.of entities of the another entity type participated in the relationship

(Or)

The cardinality in DBMS refers to the number of entities to which another entity can be linked via a given relation set. It shows the number of times instances of one thing are connected to another.

Relationship types:

In database management systems (DBMS), there are several types of relationships that can exist between entities in a database. Here are the main types of relationships

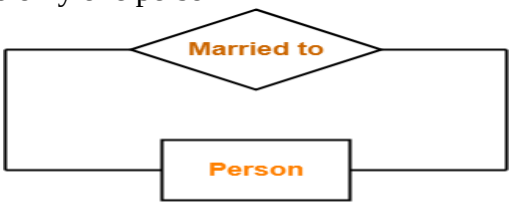
- Based on the degree relationships are classified into four types
  1. Unary Relationship
  2. Binary Relationship
  3. Ternary Relationship
  4. n-ary Relationship
- Based on the cordinality relationships are classified into four types
  1. One - to - One Relationship
  2. One - to - Many Relationship
  3. Many - to - One Relationship
  4. Many - to - Many Relationship

Based on the degree relationships are classified into four types:

1. **Unary Relationship**: A unary relationship, also called recursive, is one in which a relationship exists between occurrences of the same entity set. In this relationship, the primary and foreign keys are the same, but they represent two entities with different roles.

   The degree of relationship is 1

   Ex: One person is married to only one person



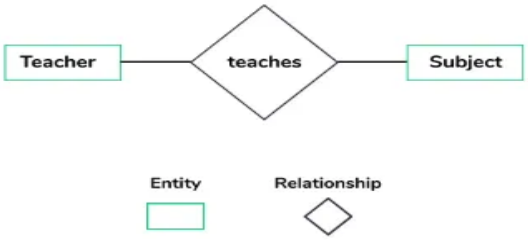
Unary Relationship Set

2. **Binary Relationship:**

   A Binary Relationship is the relationship between two different Entities.

   The degree of relationship is 2

   Ex:

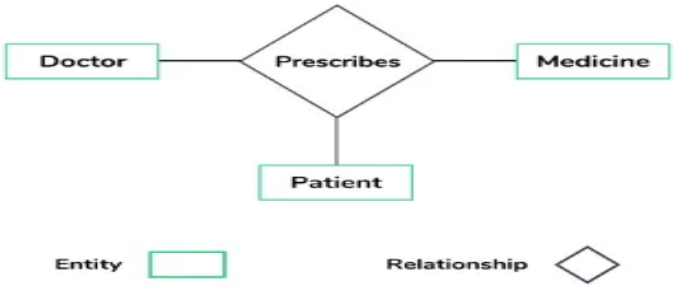


3. **Ternary Relationship:**

   A Ternary Relationship is the relationship between three different Entities.

   The degree of relationship is 3

   Ex:

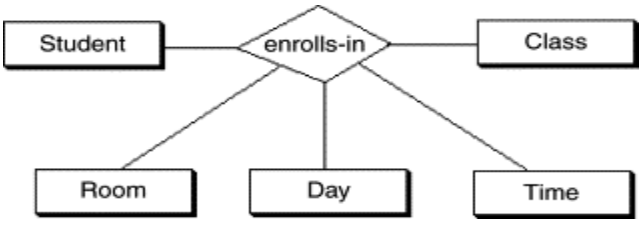


4. **N-ary Relationship:**

   A n-ary Relationship is the relationship between ‘n’ no.of different Entities.

   The degree of relationship is n

   Ex:



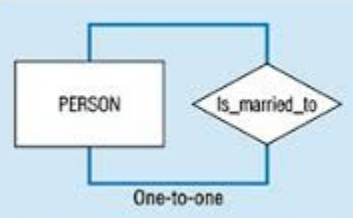
   The degree of the above n-ary relationship is 5

Based on the cordinality relationships are classified into four types

1. One - to - One Relationship (1:1):

It is used to create a relationship between two tables in which a single row of the first table can only be related to one and only one records of a second table. Similarly, the row of a second table can also be related to anyone row of the first table.

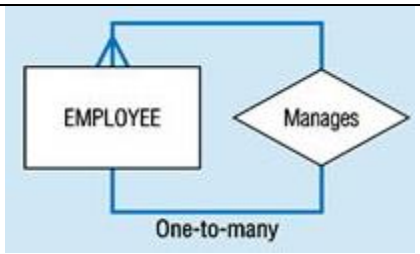
Ex:

Unary One - to - One Relationship	Binary One - to - One Relationship
	

2. One - to - Many Relationship(1:M):

It is used to create a relationship between two tables. Any single rows of the first table can be related to one or more rows of the second tables, but the rows of second tables can only relate to the only row in the first table.

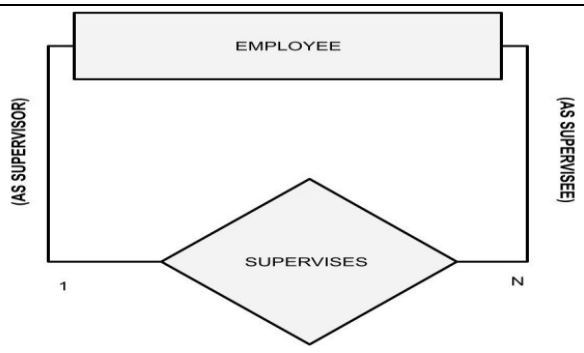
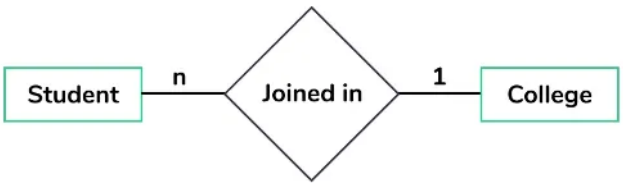
Ex:

Unary One - to - Many Relationship	Binary One - to - Many Relationship
	

3. Many - to - One Relationship (M:1):

It is reverse of one to many Relationship, Any single rows of the second table can be related to one or more rows of the first tables, but the rows of first tables can only relate to the only row in the second table.

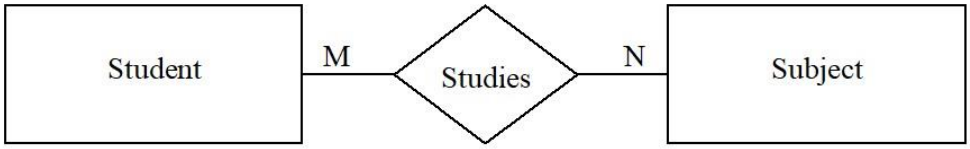
Ex:

Unary Many - to - One Relationship	Binary Many - to - One Relationshipss
	

4. **Many - to - Many Relationship (M:N):**

   A many-to-many relationship exists between the tables if a single record of the first table is related to one or more records of the second table and a single record in the second table is related to one or more records of the first table.

   Ex:



Relationship Participation:

Participation Constraints in database management refer to rules that determine the minimum and maximum participation of entities or relationships in a given relationship set.

There are two types of participation constraints

1. Total Participation
2. Partial Participation

1. **Total Participation**:

- It specifies that each entity in the entity set must compulsorily participate in at least one relationship instance in that relationship set. That is why, it is also called as mandatory participation.
- Total participation is represented using a double line between the entity set and relationship set.

2. **Partial Participation**:

- It specifies that each entity in the entity set may or may not participate in the relationship instance in that relationship set. That is why, it is also called as optional participation.
- Partial participation is represented using a single line between the entity set and relationship set.

- Double line between the entity set “Student” and relationship set “Enrolled in” signifies total participation. It specifies that each student must be enrolled in at least one course
- Single line between the entity set “Course” and relationship set “Enrolled in” signifies partial participation. It specifies that there might exist some courses for which no enrollments are made.

Additional Features of the ER Model:

"Enhanced Entity-Relationship / Extended Entity-Relationship " (EER) data modeling, an extension of the traditional ER model. EER models include additional constructs (Features) to represent more complex relationships and constraints.

EER model is a conceptual (or semantic) data model, capable of describing the data requirements for a new information system in a direct and easy to understand graphical notation.

There are several additional features or concepts that can enhance its expressiveness and usefulness in database design

1. Super type or Subtype / Super class or Subclass
2. Generalization and Specialization
3. Category or Union
4. Aggregation

1. Super type or Subtype / Super class or Subclass:

A supertype is a generalized entity that represents a group of one or more related entities. It contains common attributes and relationships that are shared among its subtypes.

Or

Supertype is an entity type that has got relationship (parent to child relationship) with one or more subtypes and it contains attributes that are common to its subtypes..

In a supertype/subtype entity structure: The top-level entity is referred to as the parent entity or the supertype entity. Each lower-level entity is referred to as a child entity or a subtype entity.

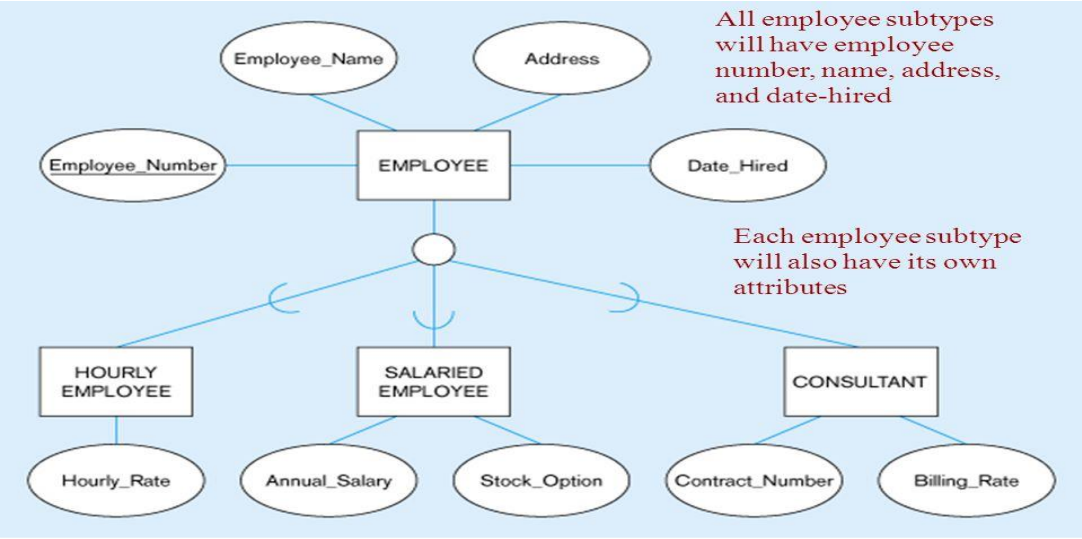


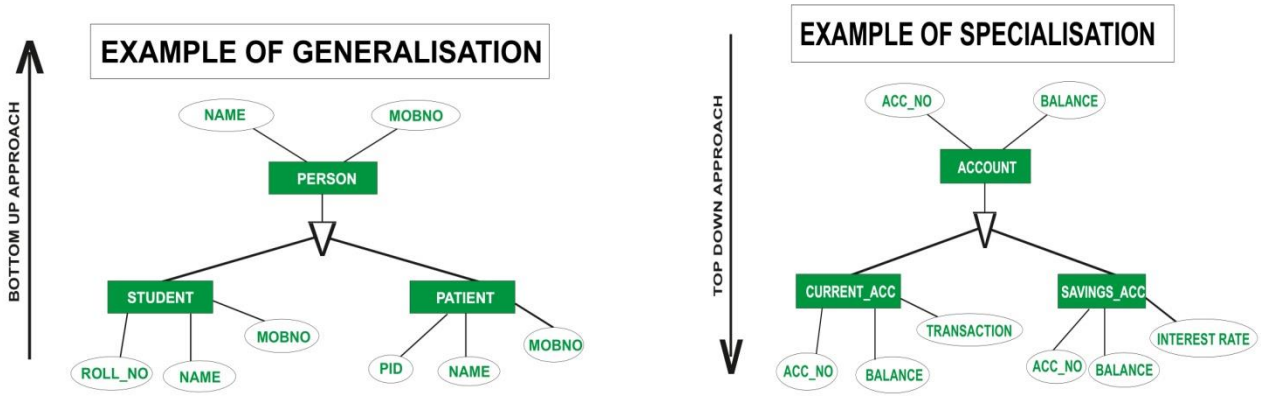
Figure – Employee supertype with three subtypes

- Super type and subtypes are connected through the connectors by using symbols (circle/triangle)
- On connecting line ‘U’ shape symbol inddicates the subtype is subset of super type

2. Generalization and Specialization:

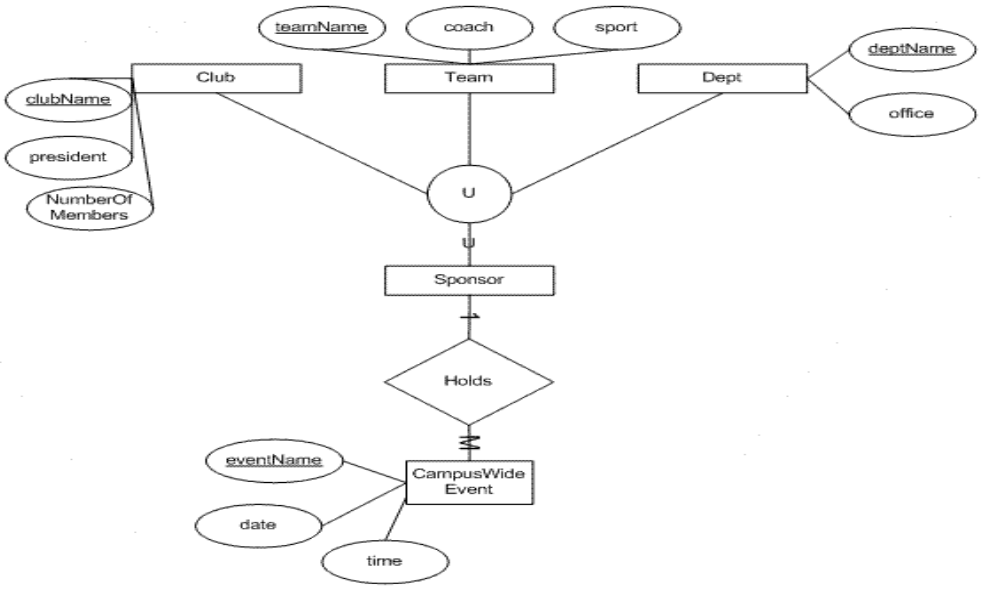
Generalization is the process of defining a general entity type from a set of more specialized entity types. It involves identifying common characteristics among multiple entity types and abstracting them into a higher-level entity type

Specialization is the opposite of generalization. It involves creating one or more specialized entity types from a more general entity type. Specialization allows you to define specific characteristics or attributes for certain subsets of entities that are distinct from the general entity type.



3. Category or Union:

EER model allows for the creation of a union type, which is a combination of two or more entity types. The union type can have attributes and relationships that are common to all the entity types that make up the union.



4. Aggregation:

Aggregation in DBMS (Database Management System) is the process of combining two or more entities to form a more meaningful new entity. When the entities do not make sense on their own, the aggregation process is used