

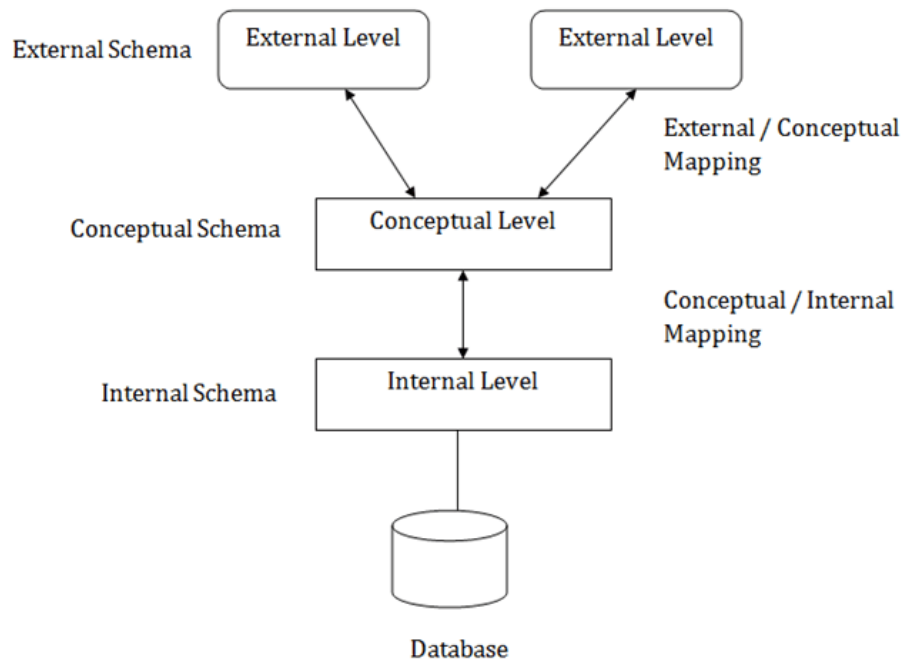
# UNIT 2:

## Database Architecture and Modeling

### Three Level Architecture of Database

- The three schema architecture is also called ANSI/SPARC architecture or three-level architecture.
- This framework is used to describe the structure of a specific database system.
- The three schema architecture is also used to separate the user applications and physical database.
- The three schema architecture contains three-levels. It breaks the database down into three different categories.

The three-schema architecture is as follows:



In the above diagram:

- It shows the DBMS architecture.
- Mapping is used to transform the request and response between various database levels of architecture.
- Mapping is not good for small DBMS because it takes more time.
- In External / Conceptual mapping, it is necessary to transform the request from external level to conceptual schema.
- In Conceptual / Internal mapping, DBMS transform the request from the conceptual to internal level.

## Objectives of Three schema Architecture

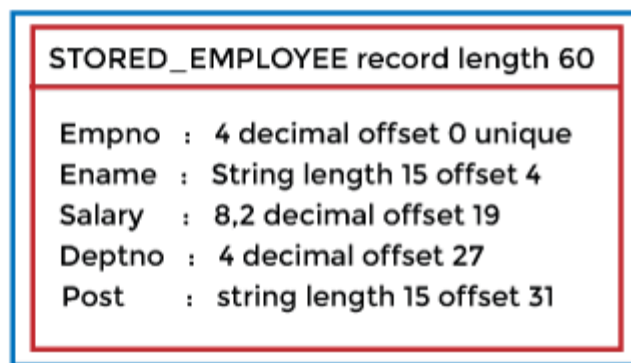
The main objective of three level architecture is to enable multiple users to access the same data with a personalized view while storing the underlying data only once. Thus it separates the user's view from the physical structure of the database. This separation is desirable for the following reasons:

- Different users need different views of the same data.
- The approach in which a particular user needs to see the data may change over time.

- The users of the database should not worry about the physical implementation and internal workings of the database such as data compression and encryption techniques, hashing, optimization of the internal structures etc.
- All users should be able to access the same data according to their requirements.
- DBA should be able to change the conceptual structure of the database without affecting the user's
- Internal structure of the database should be unaffected by changes to physical aspects of the storage.

## 1. Internal Level

Internal view



- The internal level has an internal schema which describes the physical storage structure of the database.
- The internal schema is also known as a physical schema.
- It uses the physical data model. It is used to define that how the data will be stored in a block.
- The physical level is used to describe complex low-level data structures in detail.

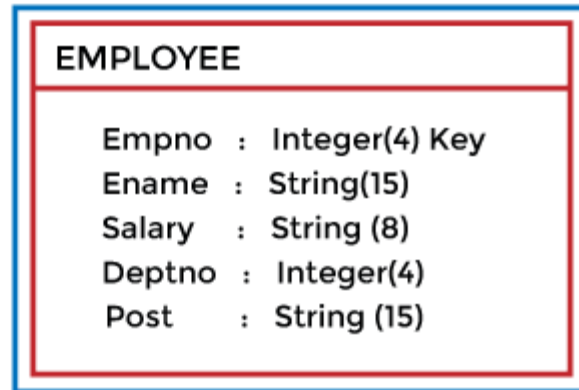
The internal level is generally is concerned with the following activities:

- Storage space allocations.  
For Example: B-Trees, Hashing etc.
- Access paths.  
For Example: Specification of primary and secondary keys, indexes, pointers and sequencing.
- Data compression and encryption techniques.

- Optimization of internal structures.
- Representation of stored fields.

## 2. Conceptual Level

Global view



- The conceptual schema describes the design of a database at the conceptual level. Conceptual level is also known as logical level.
- The conceptual schema describes the structure of the whole database.
- The conceptual level describes what data are to be stored in the database and also describes what relationship exists among those data.
- In the conceptual level, internal details such as an implementation of the data structure are hidden.
- Programmers and database administrators work at this level.

## 3. External Level

External View



- At the external level, a database contains several schemas that sometimes called as subschema. The subschema is used to describe the different view of the database.
- An external schema is also known as view schema.
- Each view schema describes the database part that a particular user group is interested and hides the remaining database from that user group.
- The view schema describes the end user interaction with database systems.

## Mapping between Views

The three levels of DBMS architecture don't exist independently of each other. There must be correspondence between the three levels i.e. how they actually correspond with each other. DBMS is responsible for correspondence between the three types of schema. This correspondence is called Mapping.

There are basically two types of mapping in the database architecture:

- Conceptual/ Internal Mapping
- External / Conceptual Mapping

### Conceptual/ Internal Mapping

The Conceptual/ Internal Mapping lies between the conceptual level and the internal level. Its role is to define the correspondence between the records and fields of the conceptual level and files and data structures of the internal level.

### External/ Conceptual Mapping

The external/Conceptual Mapping lies between the external level and the Conceptual level. Its role is to define the correspondence between a particular external and the conceptual view.

## Logical and Physical data models

Logical and physical data models are two different ways of representing the structure and relationships of data in a system. They are often used in system analysis, which is the process of understanding and designing how a system works and meets the requirements of its users. In this article, we will explain the differences between logical and physical data models, the advantages and disadvantages of each, and how to choose the most suitable one for your system analysis project.

### **What is a logical data model?**

A logical data model is a conceptual representation of the data and its meaning, without considering how it is stored or implemented in a specific system. It focuses on the entities, attributes, and relationships that are relevant to the business domain and the user needs. A logical data model is usually expressed in a graphical notation, such as an entity-relationship diagram (ERD), or a textual notation, such as a data dictionary. A logical data model is independent of any hardware, software, or data format.

### **What is a physical data model?**

A physical data model is a detailed representation of how the data is actually stored and manipulated in a specific system. It takes into account the technical aspects, such as the data types, formats, sizes, constraints, indexes, keys, and performance. A physical data model is usually expressed in a schema language, such as SQL, or a diagram that shows the tables, columns, and relationships of a database. A physical data model is dependent on the choice of hardware, software, and data format.

### **Why use logical and physical data models?**

Logical and physical data models serve different purposes and audiences in system analysis. A logical data model helps to capture the essential meaning and structure of the data, without being influenced by the implementation details. It is useful for communicating with the stakeholders, such as the users, managers, and analysts, who are interested in the business rules and requirements of the system. A

physical data model helps to optimize the data storage and manipulation, according to the technical specifications and constraints. It is useful for communicating with the developers, programmers, and administrators, who are responsible for building and maintaining the system.

### **How to choose between logical and physical data models?**

When deciding whether to use a logical or physical data model for your system analysis project, there is no definitive answer. It depends on the scope, complexity, and stage of your project, the availability and skills of your team, and the expectations and preferences of your stakeholders. Generally speaking, if you are in the early or intermediate stages of your project, you should start with a logical data model to validate and verify assumptions, identify inconsistencies or gaps, and facilitate collaboration with stakeholders. Once you have a clear and stable understanding of the data requirements and business logic of your system, then you should move to a physical data model to implement and test your system. If you want to have a comprehensive view of your system from both conceptual and technical perspectives, use both models. You can use one as a blueprint for creating or updating the other, as well as compare and contrast similarities and differences between them.

## **DBA(Database Administrator)-**

A [Database Administrator \(DBA\)](#) is an individual or person responsible for controlling, maintaining, coordinating, and operating a database management system. Managing,

securing, and taking care of the database systems is a prime responsibility. They are responsible and in charge of authorizing access to the database, coordinating, capacity, planning, installation, and monitoring uses, and acquiring and gathering software and hardware resources as and when needed. Their role also varies from configuration, database design, migration, security, troubleshooting, backup, and data recovery. Database administration is a major and key function in any firm or organization that is relying on one or more databases. They are overall commanders of the Database system.

### *Types of Database Administrator (DBA) :*

- **Administrative DBA –**  
Their job is to maintain the server and keep it functional. They are concerned with data backups, security, troubleshooting, replication, migration, etc.
- **Data Warehouse DBA –**  
Assigned earlier roles, but held accountable for merging data from various sources into the data warehouse. They also design the warehouse, with cleaning and scrubs data prior to loading.
- **Cloud DBA –**  
Nowadays companies are preferring to save their workpiece on cloud storage. As it reduces the chance of data loss and provides an extra layer of data security and integrity.
- **Development DBA –**  
They build and develop queries, stores procedure, etc. that meets firm or organization needs. They are par at programming.
- **Application DBA –**  
They particularly manage all requirements of application components that interact with the database and accomplish activities such as application installation and coordination, application upgrades, database cloning, data load process management, etc.
- **Architect –**  
They are held responsible for designing schemas like building tables. They work to build a structure that meets organizational needs. The design is further used by developers and development DBAs to design and implement real applications.
- **OLAP DBA –**  
They design and build multi-dimensional cubes for determination support or OLAP systems.
- **Data Modeler –**  
In general, a data modeler is in charge of a portion of a data architect's duties. A data modeler is typically not regarded as a DBA, but this is not a hard and fast rule.



- **Task-Oriented DBA –**  
To concentrate on a specific DBA task, large businesses may hire highly specialised DBAs. They are quite uncommon outside of big corporations. Recovery and backup DBA, whose responsibility it is to guarantee that the databases of businesses can be recovered, is an example of a task-oriented DBA. However, this specialism is not present in the majority of firms. These task-oriented DBAs will make sure that highly qualified professionals are working on crucial DBA tasks when it is possible.
- **Database Analyst –**  
This position doesn't actually have a set definition. Junior DBAs may occasionally be referred to as database analysts. A database analyst occasionally performs functions that are comparable to those of a database architect. The term "Data Administrator" is also used to describe database analysts and data analysts. Additionally, some businesses occasionally refer to database administrators as data analysts.

### *Importance of Database Administrator (DBA) :*

- Database Administrator manages and controls three levels of database internal level, conceptual level, and external level of Database management system architecture and in discussion with the comprehensive user community, gives a definition of the world view of the database. It then provides an external view of different users and applications.
- Database Administrator ensures held responsible to maintain integrity and security of database restricting from unauthorized users. It grants permission to users of the database and contains a profile of each and every user in the database.
- Database Administrators are also held accountable that the database is protected and secured and that any chance of data loss keeps at a minimum.
- Database Administrator is solely responsible for reducing the risk of data loss as it backup the data at regular intervals.

### *Role and Duties of Database Administrator (DBA) :*

- **Decides hardware –**  
They decide on economical hardware, based on cost, performance, and efficiency of hardware, and best suits the organization. It is hardware that is an interface between end users and the database.
- **Manages data integrity and security –**  
Data integrity needs to be checked and managed accurately as it protects and restricts data from unauthorized use. DBA eyes on relationships within data to maintain data integrity.
- **Database Accessibility –**  
Database Administrator is solely responsible for giving permission to access data

available in the database. It also makes sure who has the right to change the content.

- **Database design –**  
DBA is held responsible and accountable for logical, physical design, external model design, and integrity and security control.
- **Database implementation –**  
DBA implements DBMS and checks database loading at the time of its implementation.
- **Query processing performance –**  
DBA enhances query processing by improving speed, performance, and accuracy.
- **Tuning Database Performance –**  
If the user is not able to get data speedily and accurately then it may lose organization's business. So by tuning SQL commands DBA can enhance the performance of the database.

### ***Various responsibilities of Database Administrator (DBA) :***

- Responsible for designing overall database schema (tables & fields).
- To select and install database software and hardware.
- Responsible for deciding on access methods and data storage.
- DBA selects appropriate DBMS software like oracle, SQL server or MySQL.
- Used in designing recovery procedures.
- DBA decides the user access level and security checks for accessing, modifying or manipulating data.
- DBA is responsible for specifying various techniques for monitoring the database performance.
- DBA is responsible for operation managements.
- The operation management deals with the data problems which arises on day to day basis, and the responsibilities include are:
  1. Investigating if any error is been found in the data.
  2. Supervising of restart and recovery procedures in case of any event failure.
  3. Supervising reorganization of the databases.
  4. Controlling and handling all periodic dumps of data.

### ***Skills Required for DBA:***

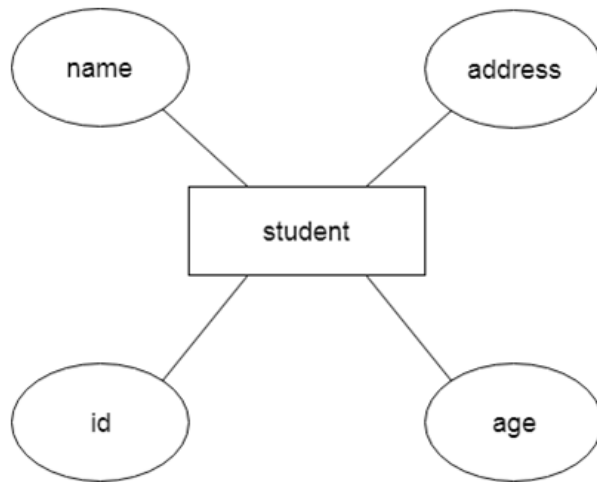
1. The various programming and soft skills are required to DBA are as follows,
  - Good communication skills
  - Excellent knowledge of databases architecture and design and RDBMS.
  - Knowledge of Structured Query Language (SQL).

2. In addition, this aspect of database administration includes maintenance of data security, which involves maintaining security authorization tables, conducting periodic security audits, investigating all known security breaches.
3. To carry out all these functions, it is crucial that the DBA has all the accurate information about the company's data readily on hand. For this purpose he maintains a data dictionary.
4. The data dictionary contains definitions of all data items and structures, the various schemes, the relevant authorization and validation checks and the different mapping definitions.
5. It should also have information about the source and destination of a data item and the flow of a data item as it is used by a system. This type of information is a great help to the DBA in maintaining centralized control of data.

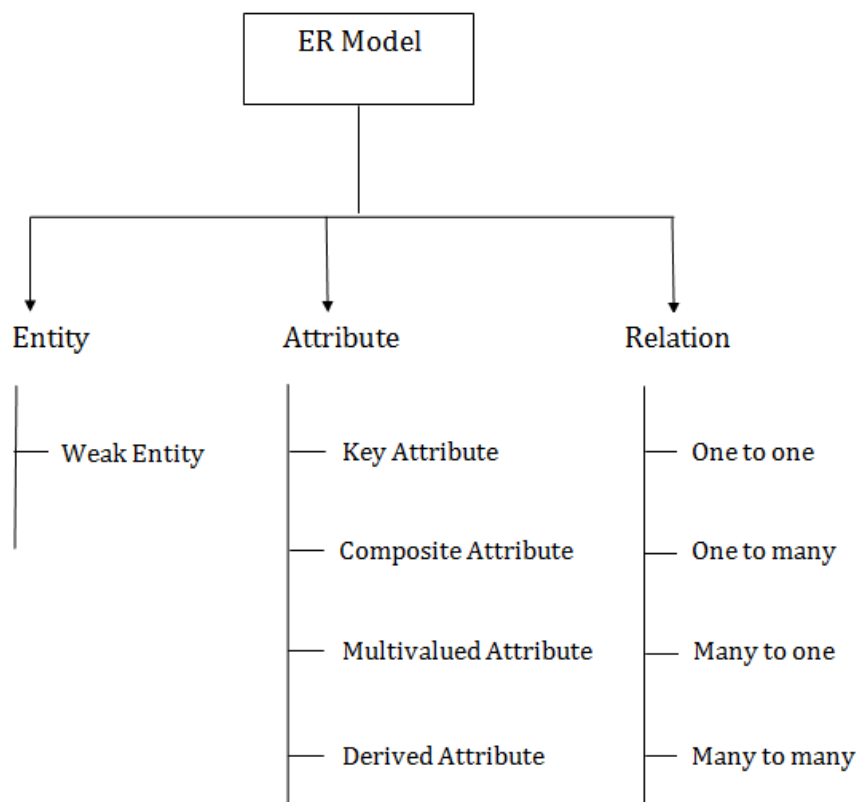
## ER (Entity Relationship) Diagram in DBMS

- ER model stands for an Entity-Relationship model. It is a high-level data model. This model is used to define the data elements and relationship for a specified system.
- It develops a conceptual design for the database. It also develops a very simple and easy to design view of data.
- In ER modeling, the database structure is portrayed as a diagram called an entity-relationship diagram.

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.



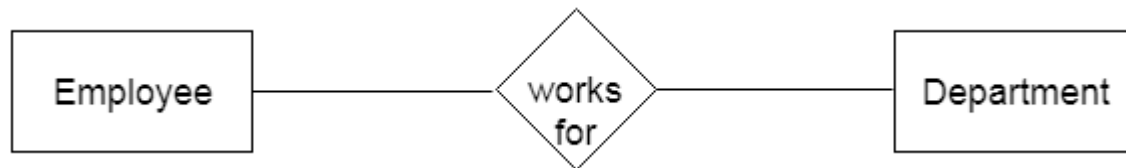
## Component of ER Diagram



### 1. Entity:

An entity may be any object, class, person or place. In the ER diagram, an entity can be represented as rectangles.

Consider an organization as an example- manager, product, employee, department etc. can be taken as an entity.



#### a. Weak Entity

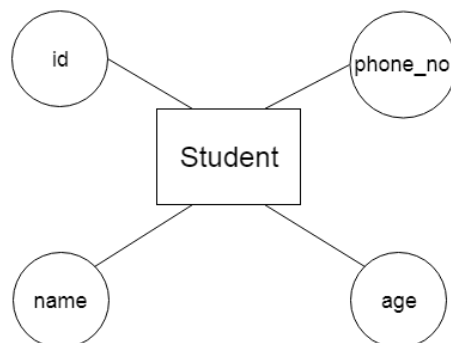
An entity that depends on another entity called a weak entity. The weak entity doesn't contain any key attribute of its own. The weak entity is represented by a double rectangle.



## 2. Attribute

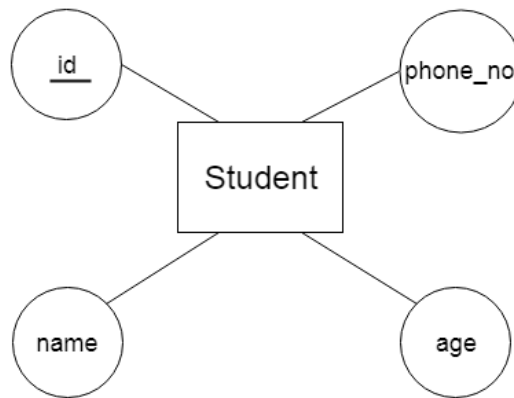
The attribute is used to describe the property of an entity. Eclipse is used to represent an attribute.

For example, id, age, contact number, name, etc. can be attributes of a student.



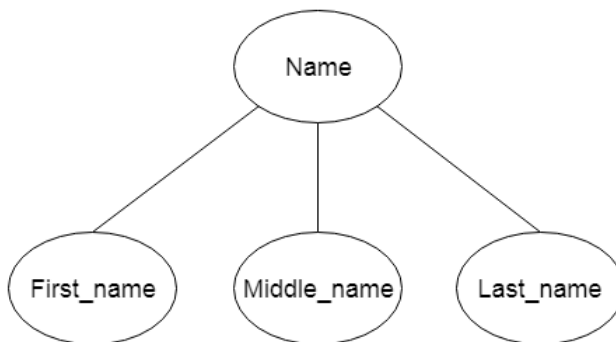
#### a. Key Attribute

The key attribute is used to represent the main characteristics of an entity. It represents a primary key. The key attribute is represented by an ellipse with the text underlined.



### b. Composite Attribute

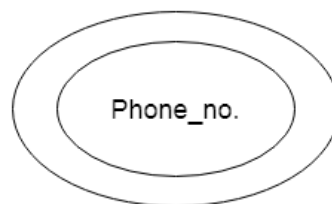
An attribute that composed of many other attributes is known as a composite attribute. The composite attribute is represented by an ellipse, and those ellipses are connected with an ellipse.



### c. Multivalued Attribute

An attribute can have more than one value. These attributes are known as a multivalued attribute. The double oval is used to represent multivalued attribute.

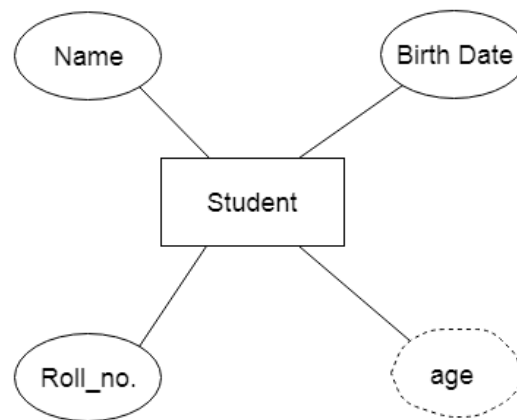
For example, a student can have more than one phone number.



### d. Derived Attribute

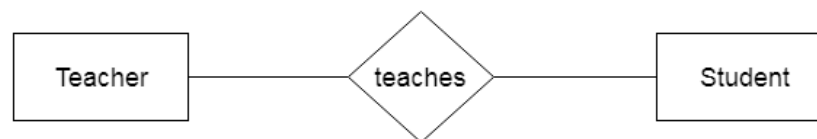
An attribute that can be derived from other attribute is known as a derived attribute. It can be represented by a dashed ellipse.

For example, A person's age changes over time and can be derived from another attribute like Date of birth.



### 3. Relationship

A relationship is used to describe the relation between entities. Diamond or rhombus is used to represent the relationship.



Types of relationship are as follows:

#### a. One-to-One Relationship

When only one instance of an entity is associated with the relationship, then it is known as one to one relationship.

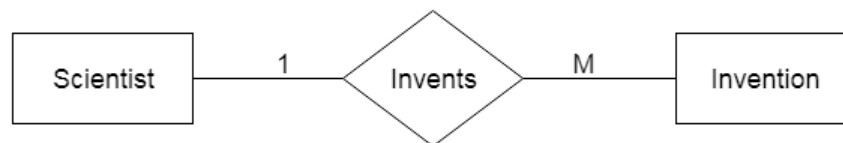
For example, A female can marry to one male, and a male can marry to one female.



#### b. One-to-many relationship

When only one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then this is known as a one-to-many relationship.

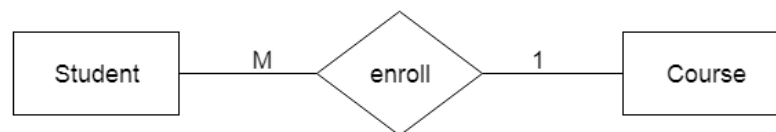
For example, Scientist can invent many inventions, but the invention is done by the only specific scientist.



#### c. Many-to-one relationship

When more than one instance of the entity on the left, and only one instance of an entity on the right associates with the relationship then it is known as a many-to-one relationship.

For example, Student enrolls for only one course, but a course can have many students.



#### d. Many-to-many relationship

When more than one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then it is known as a many-to-many relationship.

For example, Employee can assign by many projects and project can have many employees.






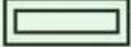




### Symbols Used in ER Model

ER Model is used to model the logical view of the system from a data perspective which consists of these symbols:

- Rectangles: Rectangles represent Entities in ER Model.
- Ellipses: Ellipses represent Attributes in ER Model.
- Diamond: Diamonds represent Relationships among Entities.
- Lines: Lines represent attributes to entities and entity sets with other relationship types.
- Double Ellipse: Double Ellipses represent *Multi-Valued Attributes*.
- Double Rectangle: Double Rectangle represents a Weak Entity.

Figures	Symbols	Represents
Rectangle		Entities in ER Model
Ellipse		Attributes in ER Model
Diamond		Relationships among Entities
Line		Attributes to Entities and Entity Sets with Other Relationship Types
Double Ellipse		Multi-Valued Attributes
Double Rectangle		Weak Entity

*Symbols used in ER Diagram*

## Superclass and Subclass in DBMS

### Superclass in DBMS:

**Class:** A set of objects that share a common structure and a common behaviour is called a class.

**Object:** In the ordinary sense, an object is something capable of being seen, touched or sensed. Putting it in layman's language, an object is something that has a fixed shape or well-defined boundary. If you look at your computer desk, you would notice several objects of varying descriptions. They may be connected to or related to adjacent objects.

For example, a keyboard may be attached to a PC, and it is also easy to see that different parts of the PC are distinct objects.

**Superclass:** A class supertype is an object class whose instances store attributes that are common to one or more class subtypes of the object.

**Inheritance:** It means the methods and or attributes defined in an object class that can be inherited or reused by another object class. Inheritance is always transitive that can be transferred. A class can inherit features from superclasses many levels away. Inheritance means that the behaviour and data associated with child classes are always an extension of the properties associated with the parent class.

### **Subclass in DBMS:**

A class subtype is an object whose instances inherit some common attributes from a class super type and then add other attributes that are unique to an instance of the subtype. A subclass must have all the properties of the parent class and other properties as well.

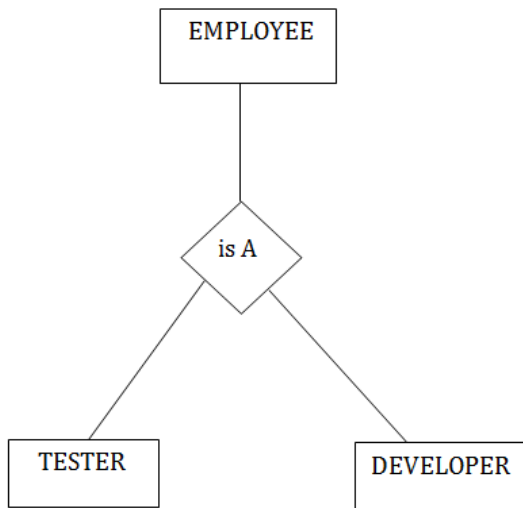
## **Attribute inheritance**

The attributes of a higher level entity set are inherited through a lower level entity set made by specialization-generalization hierarchy. Namely, all the attributes of the higher level entity set are as well the attributes of the lower level entity set.

## **Specialization**

- Specialization is a top-down approach, and it is opposite to Generalization. In specialization, one higher level entity can be broken down into two lower level entities.
- Specialization is used to identify the subset of an entity set that shares some distinguishing characteristics.
- Normally, the superclass is defined first, the subclass and its related attributes are defined next, and relationship set are then added.

For example: In an Employee management system, EMPLOYEE entity can be specialized as TESTER or DEVELOPER based on what role they play in the company.



## Generalization

- Generalization is like a bottom-up approach in which two or more entities of lower level combine to form a higher level entity if they have some attributes in common.
- In generalization, an entity of a higher level can also combine with the entities of the lower level to form a further higher level entity.
- Generalization is more like subclass and superclass system, but the only difference is the approach. Generalization uses the bottom-up approach.
- In generalization, entities are combined to form a more generalized entity, i.e., subclasses are combined to make a superclass.

For example, Faculty and Student entities can be generalized and create a higher level entity Person.

