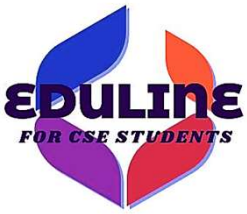
MODULE 1

CHAPTER 1 - INTRODUCTION TO DBMS

CO – Students will be able to summarize the DBMS concept and overview



Prepared By Mr. EBIN PM

DATABSE CONCEPT

- A database is a collection of data, typically describing the activities of one or more related organizations.
- For example, a university database might contain information about the following:
- Entities such as students, faculty, courses, and classrooms.
- Relationships between entities, such as students' enrolment in courses, faculty teaching courses, and the use of rooms for courses
- By data, we mean known facts that can be recorded and that have implicit meaning.

Prepared By Mr.EBIN PM EDULINE 2

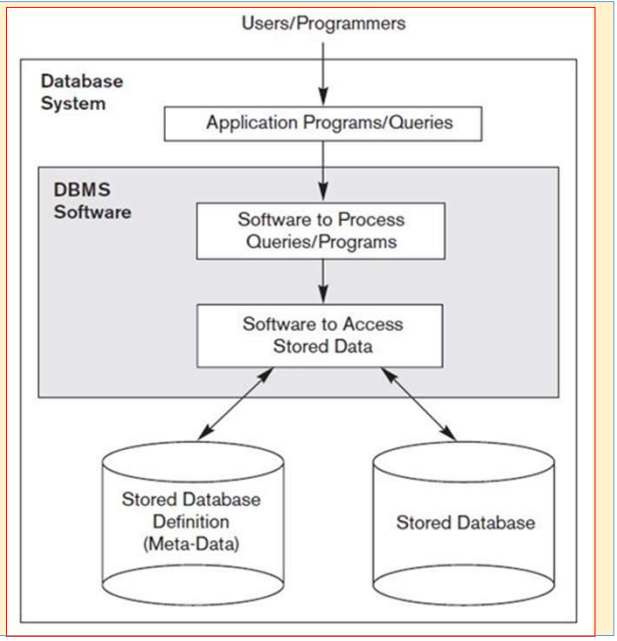
- For example, consider the names, telephone numbers, and addresses of the people you know.
- You may have recorded this data in an indexed address book or you may have stored it on a hard drive, using a Personal computer and software such as Microsoft Access or Excel.
- This collection of related data with an implicit meaning is a database.
- A database system is basically a computer based record keeping system.
- An example of a large commercial database is Amazon.com.

❖Data Base Management System (DBMS)

- A database management system (DBMS) is a collection of programs that enables users to create and maintain a database.
- The DBMS is a general-purpose software system that facilitates the processes of defining, constructing, manipulating, and sharing databases among various users and applications.
- Constructing the database is the process of storing the data on some storage medium that is controlled by the DBMS.
- Manipulating a database includes functions such as querying the database to retrieve specific data, updating the database to reflect changes, and generating reports from the data.
- Sharing a database allows multiple users and programs to access the database simultaneously.

- An application program accesses the database by sending queries or requests for data to the DBMS.
- A query typically causes some data to be retrieved .
- Other important functions provided by the DBMS include protecting the database and maintaining it over a long period of time.
- Protection includes system protection against hardware or software malfunction (or crashes) and security protection against unauthorized or malicious access.
- A typical large database may have a lifecycle of many years, so the DBMS must be able to maintain the database system by allowing the system to evolve as requirements change over time.

Fig: A simplified database system environment



❖FILE SYSTEMS VERSUS A DBMS

- In a typical file processing system, permanent records are stored in various files.
- A number of different application programs are written to extract records from and add records to the appropriate file.
- But this has a number of limitations and disadvantageous such as data redundancy, data inconsistency, unshareable data, unstandardized data, insecure data, incorrect data etc.
- By storing data in a DBMS, rather than as a collection of operating system files, we can use the DBMS's features to manage the data in a robust and efficient manner.

ADVANTAGES OF A DBMS

1. Controlling Redundancy

- Data redundancy means duplication of data.
- The redundancy in storing the same data multiple times leads to several problems. The storage space is wasted and files that represent the same data may become inconsistent.
- In the database approach, the views of different user groups are integrated during database design.
- Ideally, we should have a database design that stores each logical data item—such as a student's name or birth date—in only one place in the database. This is known as data normalization, and it ensures consistency and saves storage space.

2. Controlling Inconsistency

- An inconsistent database provides incorrect or conflicting information. By controlling redundancy the inconsistency is also controlled.

3. Data integrity and security

- If data is always accessed through the DBMS, the DBMS can enforce integrity constraints on the data.
- For example, before inserting salary information for an employee, the DBMS can check that the department budget is not exceeded. Also, the DBMS can enforce access controls that govern what data is visible to different classes of users. Data security refers to protection of data.

4. Facilitate sharing of data

- Individual pieces of data in the database may be shared among several different users in the sense that each of those users may have access to the same piece of data and each of them may use it for different purpose. When several users share the data, centralizing the administration of data can offer significant improvements.

5. Concurrent access and crash recovery

- A DBMS schedules concurrent accesses to the data in such a manner that users can think of the data as being accessed by only one user at a time.
- Further, the DBMS protects users from the effects of system failures.

6. Providing Backup and Recovery

- A DBMS must provide facilities for recovering from hardware or software failures.
- The **backup and recovery subsystem** of the DBMS is responsible for recovery.
- For example, if the computer system fails in the middle of a complex update transaction, the recovery subsystem is responsible for making sure that the database is restored to the state it was in before the transaction started executing.

7. Potential for Enforcing Standards

- The database approach permits the DBA to define and enforce standards among database users in a large organization.
- This facilitates communication and cooperation among various departments, projects, and users within the organization.
- Standards can be defined for names and formats of data elements, display formats, report structures, terminology, and so on.
- The DBA can enforce standards in a centralized database environment more easily than in an environment where each user group has control of its own data files and software.

8. Availability of Up-to-Date Information

- A DBMS makes the database available to all users.
- As soon as one user's update is applied to the database, all other users can immediately see this update.
- This availability of up-to-date information is essential for many transaction-processing applications, such as reservation systems or banking databases, and it is made possible by the concurrency control and recovery subsystems of a DBMS.

9. Economies of Scale

- The DBMS approach permits consolidation of data and applications, thus reducing the amount of wasteful overlap between activities of data-processing personnel in different projects or departments as well as redundancies among applications.
- This enables the whole organization to invest in more powerful processors, storage devices, or communication gear, rather than having each department purchase its own (lower performance) equipment.
- This reduces overall costs of operation and management.

❖DATABASE SCHEMA:

- A database schema is the skeleton structure that represents the logical view of the entire database.
- It defines how the data is organized and how the relations among them are associated.
- The overall design of the database is called Database Schema. Database schema is the skeleton of database.
- It is designed when the database doesn't exist at all.
- A database schema does not contain any data or information.

❖DATABASE INSTANCE:

- A database instance is a state of operational database with data at any given time.
- It contains a snapshot of the database. Database instances tend to change with time.
- The collection of information stored in the database at a particular moment is called an instance.

THREE SCHEMA ARCHITECTURE IN A DBMS

- A major purpose of a database system is to provide users with an abstract view of data.
- The system hides certain details of how the data are stored and maintained.
- Since many database system users are not computer trained, developers hide the complexity from users through several levels of abstraction.
- The data in a DBMS is described at three levels of abstraction, as illustrated in Figure.

- The database description consists of a schema at each of these three levels of abstraction: the **conceptual, physical**, and **external** schemas.
- A **data definition language (DDL)** is used to define the external and conceptual schemas.
- Information about the conceptual, external, and physical schemas is stored in the **system catalogs.**

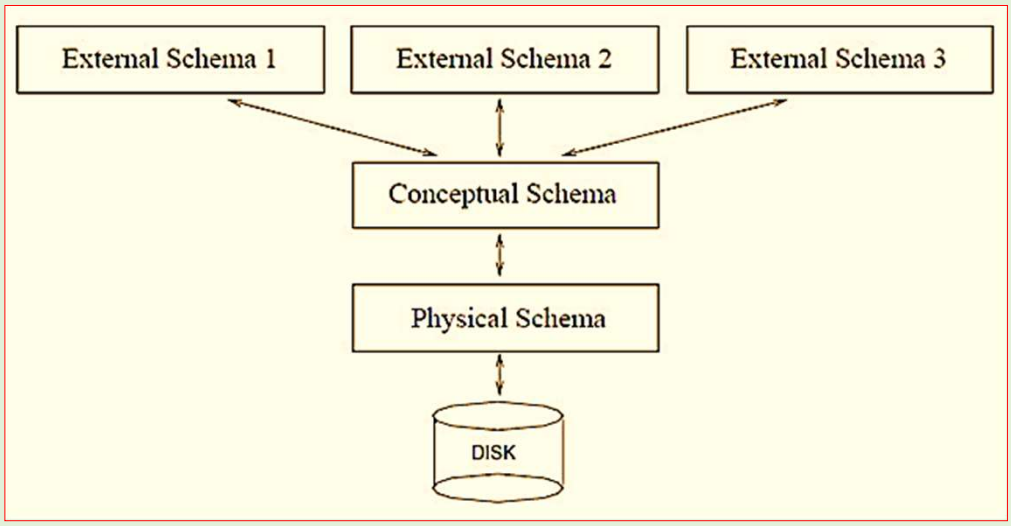


Fig: Levels of Abstraction (Three scheme architecture) in a DBMS

Prepared By Mr.EBIN PM, AP, IESCE EDULINE 19

❖CONCEPTUAL LEVEL

- The conceptual level (sometimes called the logical level) describes the stored data in terms of the data model of the DBMS.
- This level explains entire database.
- It hides all physical data storage detail from the user and focuses on relations, data types, operations, and constraints.
- Database programmer and Database administrator work at this level for creating functions, triggers, procedure, relations in the table.
- In a relational DBMS, the conceptual level describes all relations that are stored in the database.

Prepared By Mr.EBIN PM, AP, IESCE EDULINE 20

❖PHYSICAL LEVEL

- The physical level specifies additional storage details.
- Essentially, the physical level summarizes how the relations described in the conceptual schema are actually stored on secondary storage devices such as disks and tapes.
- It deals with data storage structure (B+ trees, Hash table) and data access way to access the data in the least time from the database.
- The user does not interact with this level.
- This level is written in DDL language.

❖EXTERNAL LEVEL

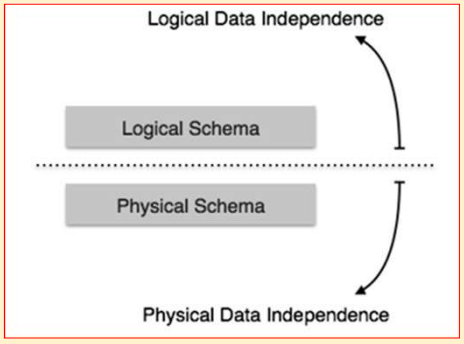
- External level allow data access to be customized (and authorized) at the level of individual users or groups of users.
- It is the closest interface to the user.
- With the help of GUI interface interact with the system without knowing which type of data stored and how the data stored in the database.
- Any given database has exactly one conceptual schema and one physical schema because it has just one set of stored relations, but it may have several external schemas, each tailored to a particular group of users.

DATA INDEPENDENCE

- A database system normally contains a lot of data in addition to users' data.
- For example, it stores data about data, known as metadata, to locate and retrieve data easily.
- It is rather difficult to modify or update a set of metadata once it is stored in the database.
- But as a DBMS expands, it needs to change over time to satisfy the requirements of the users.
- The ability to modify a schema definition in one level without affecting the schema definition in the next higher level is called Data Independence.

❖**Physical Data Independence:** It refers to the ability to modify a schema followed at the physical level without affecting the schema definition followed at the logical level.

❖**Logical Data Independence:** It refers to the ability to modify a schema followed at the logical level without affecting the schema definition followed at the view level.



DATA MODELS

- Data models define how the logical structure of a database is modelled.
- Data Models are fundamental entities to introduce abstraction in a DBMS.
- Data models define how data is connected to each other and how they are processed and stored inside the system.

1. Entity-Relationship Model

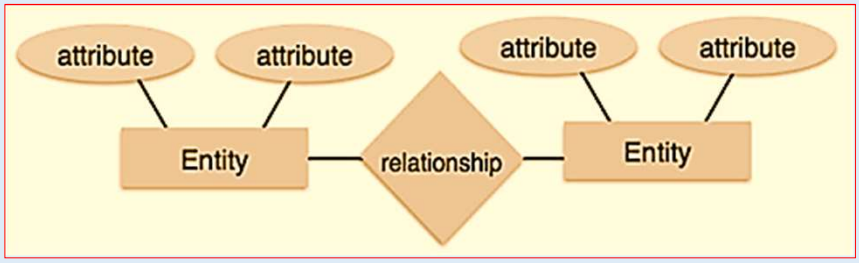
- Entity-Relationship (ER) Model is based on the notion of real-world entities and relationships among them.

- While formulating real-world scenario into the database model, the ER Model creates entity set, relationship set, general attributes, and constraints.
- ER Model is best used for the conceptual design of a database.
- ER Model is based on:

Entities and their attributes

Relationships among entities

Fig: E-R Diagram



❖**Entity**

- An entity in an ER Model is a real-world entity having properties called attributes.
- Every attribute is defined by its set of values called domain.
- For example, in a school database, a student is considered as an entity.
- Student has various attributes like name, age, class, etc.

❖**Relationship**

- The logical association among entities is called relationship.
- Relationships are mapped with entities in various ways.
- Mapping cardinalities define the number of association between two entities.

❖**Mapping cardinalities:**

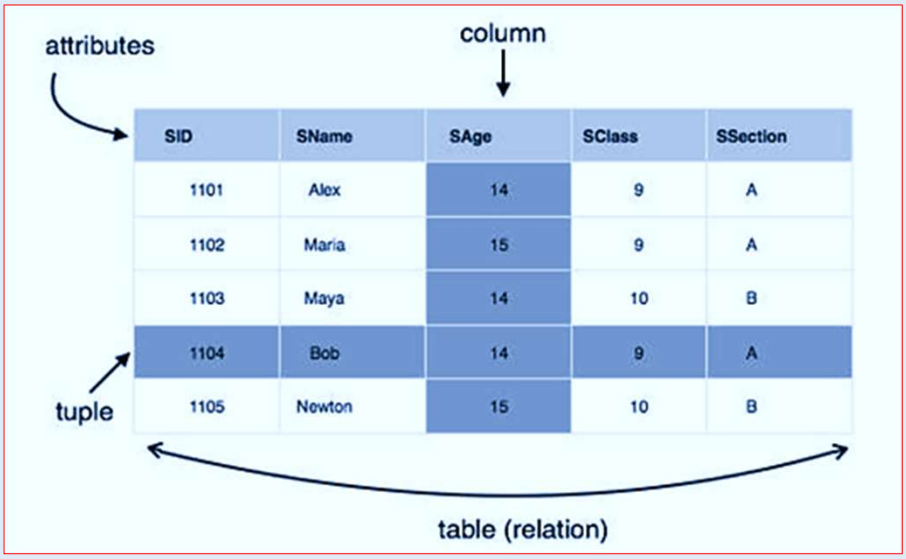
- one to one
- one to many
- many to one
- many to many

2. Relational Model

- The most popular data model in DBMS is the Relational Model.
- It is more scientific a model than others.
- This model is based on first-order predicate logic and defines a table as an n-ary relation.

- The main highlights of this model are:
  - Data is stored in tables called relations.
  - Relations can be normalized.
  - In normalized relations, values saved are atomic values.
  - Each row in a relation (called Tuples) contains a unique value.
  - Each column in a relation (called Attributes) contains values from a same domain.

Fig : Relational Model



PEOPLE WHO DEALWITH DATABASES

- Quite a variety of people are associated with the creation and use of databases.
- Obviously, there are database implementers, who build DBMS software, and end users who wish to store and use data in a DBMS.
- Database implementers work for vendors such as IBM or Oracle.
- End users come from a diverse and increasing number of fields.
- In addition to end users and implementers, two other classes of people are associated with a DBMS: application programmers and database administrators (DBAs).

- Database application programmers develop packages that facilitate data access for end users, who are usually not computer professionals, using the host or data languages and software tools that DBMS vendors provide.
- Such tools include report writers, spread sheets, statistical packages, etc.
- Application programmers can choose from many tools to develop user interfaces.
- Rapid Application Development (RAD) tools are tools that enable an application programmer to construct forms and reports with minimal programming effort.

- A personal database is typically maintained by the individual who owns it and uses it.
- However, corporate or enterprise-wide databases are typically important enough and complex enough that the task of designing and maintaining the database is entrusted to a professional called the database administrator.
- A database administrator (DBA) manages a DBMS for an enterprise.
- The DBA designs schemas, provide security, restores the system after a failure, and periodically tunes the database to meet changing user needs.

❖**The functions of DBA include:**

- Schema definition :
- Storage structure and access method definition
- Schema and physical organization modification
- Granting of authorization for data access
- Routine maintenance

❖**DISADVANTAGES OF DBMS**

- Problem associated with centralization
- Cost of software/hardware migration
- Complexity of backup and recovery
- Extra hardware may be required
- System is likely to be complex

DATABASE SYSTEM APPLICATIONS

- **Banking:** for storing customer information, accounts, loans and banking transactions.
- **Airlines:** For reservations and schedule information.
- **Universities:** For student information, course registration and grade.
- **Credit card transactions:** For purchase on credit card and generation of monthly statements.
- **Tele communications:** For keeping records for calls made, generating monthly bills, maintaining balances on prepaid calling cards etc.

- **Finance:** For storing information about holdings, sales and purchase of financial instruments such as stokes, bonds and for storing real time market data.
- **Sales:** For customer, product, and purchase info.
- **Manufacturing:** For management of the supply chain and for tracking production of items in factories and orders for items.
- **Human Resource:** For information about employee, salaries, payroll taxes, benefits and for generation of pay checks.

DATABASE LANGUAGES

1. Data Definition Language (DDL)

- These statements are used to define the database structure or schema.
- It defines data structure. In SQL, DDL is "a language that allows the Database Administrator or user to describe and name the entities, attributes, and relationships required for the application, together with any associated integrity and security constraints."

➢ **DDL includes the following commands:**

- CREATE - create table, to define an SQL relation.
- ALTER – To add attributes to an existing relation.

- DROP - To remove a relation from database.
- TRUNCATE- To remove all records from a table, including all spaces allocated for the records are removed

2. Data Manipulation Language (DML)

- DML is a family of computer languages used by computer programs and/or database users to insert, delete, and update data in a database. These are used for managing data within the schema.

➢ **DML includes the following verbs:**

- SELECT – Retrieve data from the database
- INSERT – Insert data in to a table

- UPDATE – Updates existing data with in a table
- DELETE – Deletes all records from a table, the space for the records remain.

3. Data Control Language (DCL)

➢Example of DCL statements are:

- GRANT – give user's access privileges to data base.
- REVOKE – withdraw access privileges given with the GRANT command.

ER MODEL

❖**ENTITY-RELATIONSHIP MODEL**

- The ER model defines the conceptual view of a database.
- It works around real world entities and the associations among them.
- At view level, the ER model is considered a good option for designing databases.
- The ER model is important primarily for its role in database design.
- It provides useful concepts that allow us to move from an informal description of what users want from their database to a more detailed and precise, description that can be implemented in a DBMS.
- E-R model employs three basic notions. They are

A. ENTITY

- An entity can be a real-world object, either animate or inanimate, that can be easily identifiable.
- For example, in a school database, students, teachers, classes, and courses offered can be considered as entities. All these entities have some attributes or properties that give them their identity.
- An entity set is a collection of similar types of entities.
- An entity set may contain entities with attribute sharing similar values. For example, a Students set may contain all the students of a school; likewise a Teachers set may contain all the teachers of a school from all faculties. Entity sets need not be disjoint.

B. ATTRIBUTES

- Entities are represented by means of their properties called attributes.
- All attributes have values. For example, a student entity may have name, class, and age as attributes.
- There exists a domain or range of values that can be assigned to attributes.
- The set of permitted values for an attribute is called the Domain of that attribute.
- For example, a student's name cannot be a numeric value. It has to be alphabetic. A student's age cannot be negative, etc.
- Each entity can be described by a set of (attribute, data value) pair.

❖ ***Types of Attributes***

- **Simple attribute:** Simple attributes are atomic values, which cannot be divided further. For example, a student's phone number is an atomic value of 10 digits.
- **Composite attribute:** Composite attributes are made of more than one simple attribute. For example, a student's complete name may have first_name and last_name.
- **Derived attribute:** 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, average_salary in a department should not be saved directly in the database, instead it can be derived. For another example, age can be derived from data_of_birth.

- **Single-value attribute:** Single-value attributes contain single value. For example: Social_Security_Number.
- **Multi-value attribute:** Multi-value attributes may contain more than one value. For example, a person can have more than one phone number, email_address, etc.

C. RELATIONSHIP

- The association among entities is called a relationship. For example, an employee works_at a department, a student enrolls in a course. Here, Works_at and Enrolls are called relationships.
- **Relationship Set:** A set of relationships of similar type is called a relationship set. Like entities, a relationship too can have attributes. These attributes are called **descriptive attributes**.
- **Degree of Relationship:** The number of participating entities in a relationship defines the degree of the relationship.

✓Binary = degree 2

✓Ternary = degree 3

✓n-ary = degree n

ENTITY-SET AND KEYS

- Key is an attribute or collection of attributes that uniquely identifies an entity among entity set. For example, the roll_number of a student makes him/her identifiable among students.

➢**Super Key:** A set of attributes (one or more) that collectively identifies an entity in an entity set.

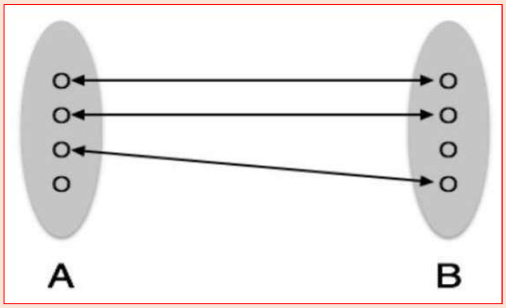
➢**Candidate Key:** A minimal super key is called a candidate key. An entity set may have more than one candidate key.

➢**Primary Key:** A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

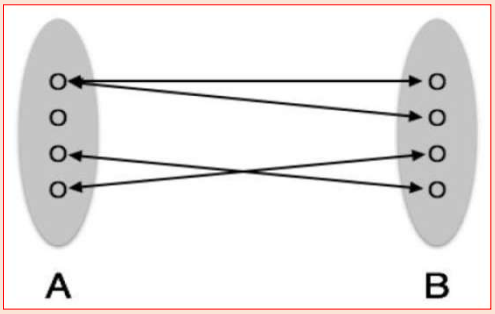
MAPPING CARDINALITIES

- Cardinality defines the number of entities in one entity set, which can be associated with the number of entities of other set via relationship set.

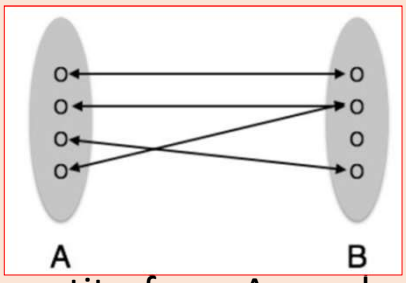
➢ **One-to-one:** One entity from entity set A can be associated with at most one entity of entity set B and vice versa.



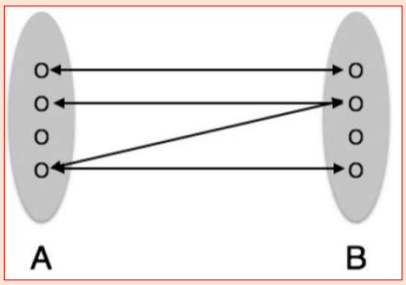
➢ **One-to-many:** One entity from entity set A can be associated with more than one entities of entity set B, however an entity from entity set B can be associated with at most one entity.



➢ **Many-to-one:** More than one entities from entity set A can be associated with at most one entity of entity set B, however an entity from entity set B can be associated with more than one entity from entity set A.



➢**Many-to-many:** One entity from A can be associated with more than one entity from B and vice versa

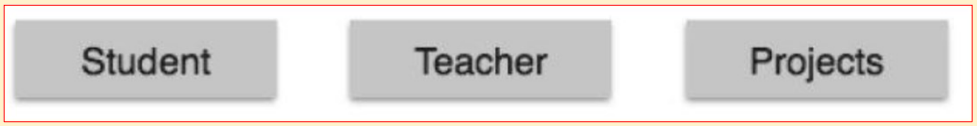


E-R DIAGRAM REPRESENTATION

- Let us now learn how the ER Model is represented by means of an ER diagram. Any object, for example, entities, attributes of an entity, relationship sets, and attributes of relationship sets, can be represented with the help of an ER diagram.

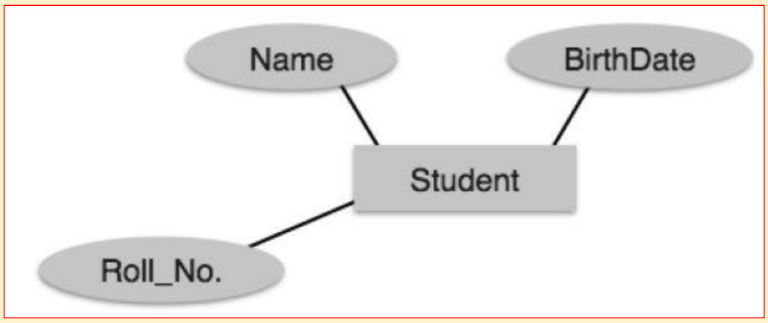
➢**ENTITY**

- Entities are represented by means of **rectangles**. Rectangles are named with the entity set they represent.



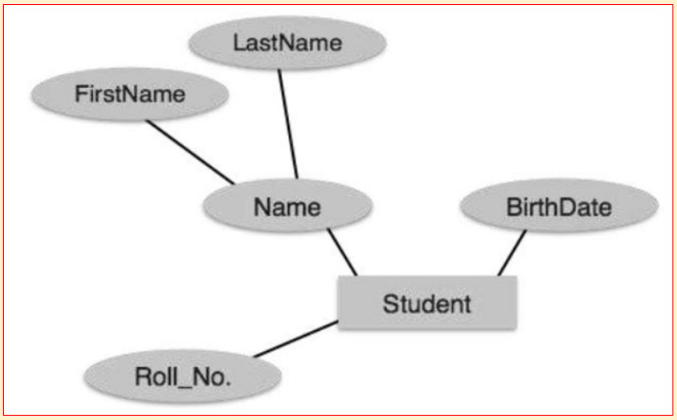
➢ATTRIBUTES

- Attributes are the properties of entities. Attributes are represented by means of **ellipses**. Every ellipse represents one attribute and is directly connected to its entity (rectangle).



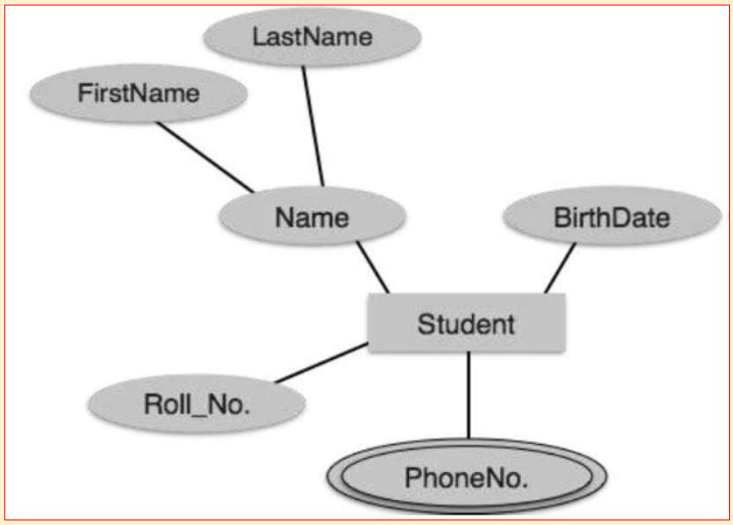
Prepared By Mr.EBIN PM, AP, CU EDULINE 51

- If the attributes are composite, they are further divided in a tree like structure. Every node is then connected to its attribute. That is, composite attributes are represented by ellipses that are connected with an ellipse.



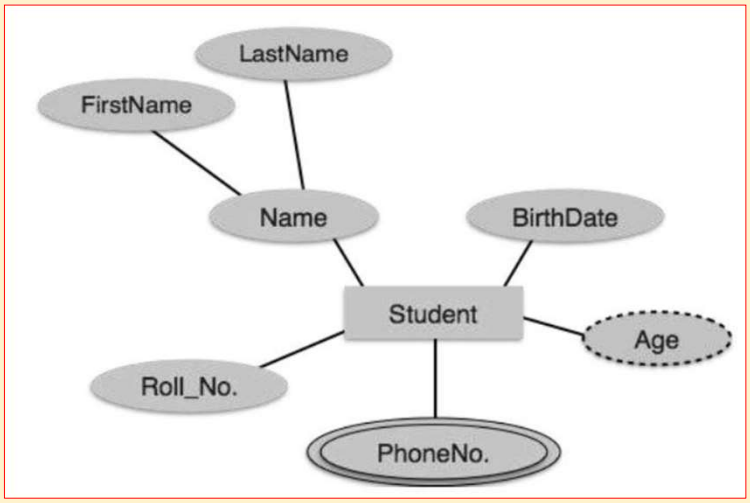
Prepared By Mr.EBIN PM, AP, CU EDULINE 52

- Multivalued attributes are depicted by double ellipse.



Prepared By Mr.EBIN PM, AP, CU EDULINE 53

- Derived attributes are depicted by dashed ellipse.



Prepared By Mr.EBIN PM, AP, CU EDULINE 54

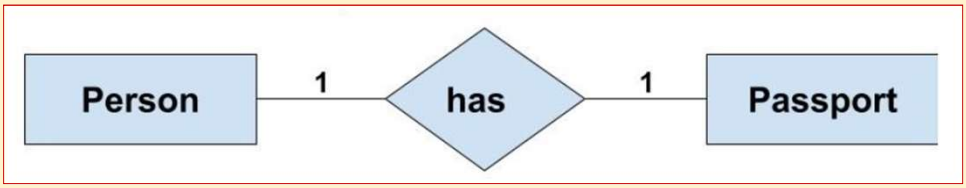
➢RELATIONSHIP

- 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.

❖*Binary Relationship and Cardinality*

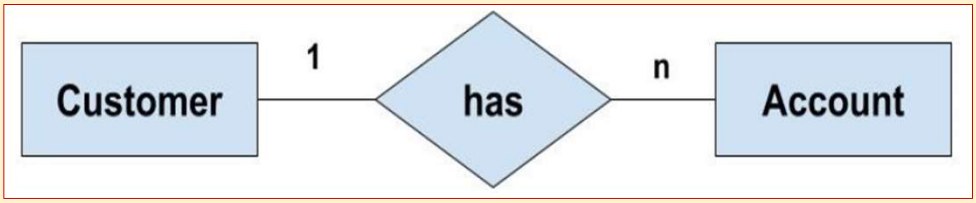
- A relationship where two entities are participating is called a binary relationship.
- Cardinality is the number of instance of an entity from a relation that can be associated with the relation.

- **One-to-one:** When only one instance of an entity is associated with the relationship, it is marked as '1:1'. The following image reflects that only one instance of each entity should be associated with the relationship. It depicts one-to-one relationship.



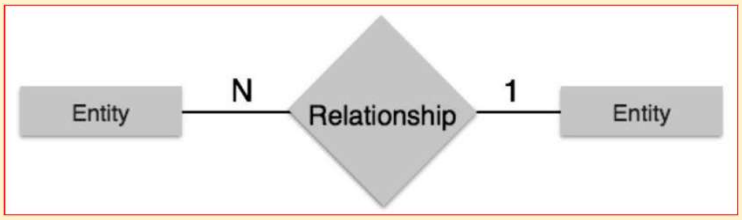
- For example, If there are two entities ‘Person’ (Id, Name, Age, Address)and ‘Passport’(Passport_id, Passport_no). So, each person can have only one passport and each passport belongs to only one person.

- **One-to-many:** When more than one instance of an entity is associated with a relationship, it is marked as '1:N'.
- For example, If there are two entity type 'Customer' and 'Account' then each 'Customer' can have more than one 'Account' but each 'Account' is held by only one 'Customer'. In this example, we can say that each Customer is associated with many Account. So, it is a one-to-many relationship.



- **Many-to-one:** When more than one instance of entity is associated with the relationship, it is marked as 'N:1'.

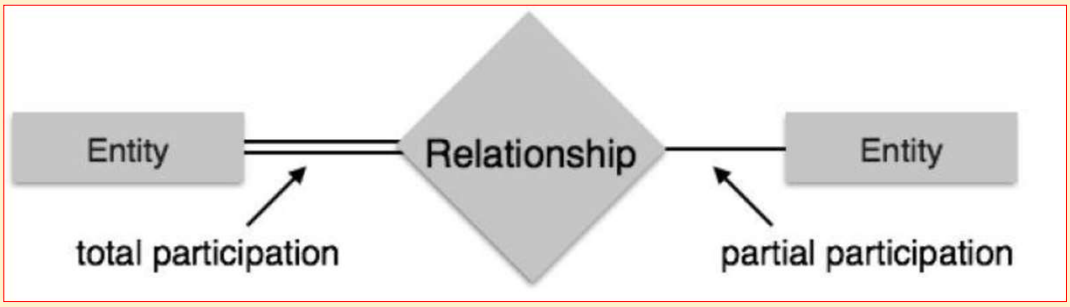
Eg: many Account is associated with one Customer then we can say that it is a many-to-one relationship.



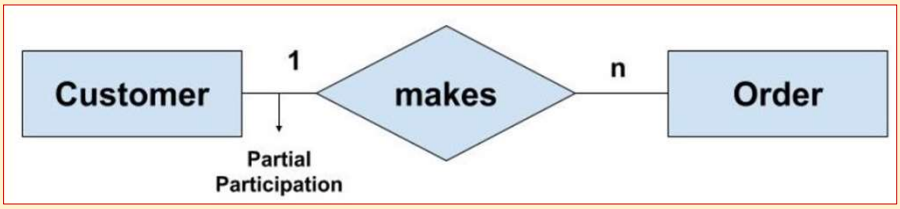
- **Many-to-many:** The following image reflects that more than one instance of an entity on the left and more than one instance of an entity on the right can be associated with the relationship. It depicts many-to-many relationship.
- Example: If there are two entity type 'Customer' and 'Product' then each customer can buy more than one product and a product can be bought by many different customers.

➢PARTICIPATION CONSTRAINTS

- **Total Participation:** Each entity is involved in the relationship. Total participation is represented by double lines.
- **Partial Participation:** Not all entities are involved in the relationship. Partial participation is represented by single lines.



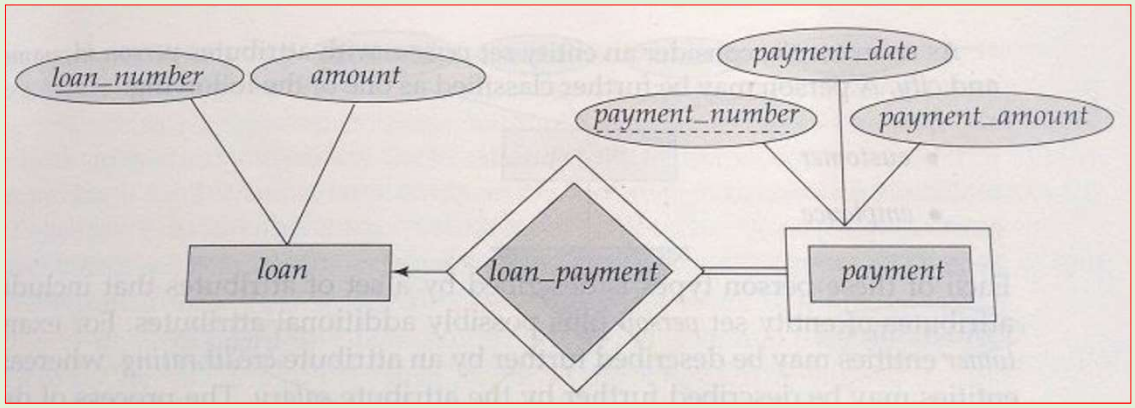
Example: We have two entity type 'Customer' and 'Order'. Then there can be 'Customer' who have not done any order. So, here there is partial participation of the entity in the relationship.



WEAK ENTITIES

- An entity set may not have sufficient attribute to form a primary key is termed a weak entity set.
- An entity set that has a primary key is termed as **strong entity** set.
- For a weak entity set to be meaningful, it must be associated with another entity set, called the **identifying** or **owner entity set**.
- The weak entity set is said to be "existence dependent" on the identifying entity set.
- The relationship associating the weak entity set with the identifying entity set is called the "identifying relationship".

- A weak entity set does not have a primary key.
- The discriminator of a weak entity set is also called the **partial key** of the entity set.
- The primary key of a weak entity set is formed by the primary of the identifying entity set, plus the weak entity set's discriminator.
- In the case of the entity set payment, its primary key is {loan _number, payment_number} where loan_number is the primary key of the identifying entity set, namely loan, and payment_number distinguishes payment entities within the same loan.



- The weak entities have total participation constraint in its identifying relationship with owner identity.
- Weak entity always has total participation but Strong entity may not have total participation
- Weak entities are represented with double rectangular box in the ER Diagram and the identifying relationships are represented with double diamond.
- Partial Key attributes are represented with dotted lines.
- In the above ER Diagram, 'Payment' is the weak entity. 'Loan Payment' is the identifying relationship and 'Payment Number' is the partial key.

DEGREE OF RELATIONSHIP

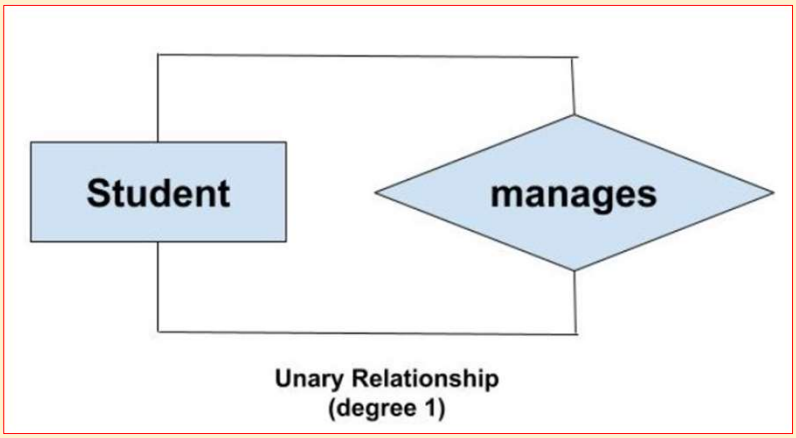
- The degree of a relationship is the number of entity types that participate(associate) in a relationship.
- By seeing an E-R diagram, we can simply tell the degree of a relationship i.e the number of an entity type that is connected to a relationship is the degree of that relationship .
- Based on the number of entity types that are connected we have the following degree of relationships:

➢ **Unary**
➢ **Binary**
➢ **Ternary**
➢ **N-ary**

❖Unary (degree 1)

- A unary relationship exists when both the participating entity type are the same. When such a relationship is present we say that the degree of relationship is 1.
- For example , Suppose in a classroom, we have many students who belong to a particular club-like dance club, basketball club etc. and some of them are club leads. So, a particular group of student is managed by their respective club lead. Here, the group is formed from students and also, the club leads are chosen from students. So, the 'Student' is the only entity participating here. We can represent this relationship using the E-R diagram as follows:

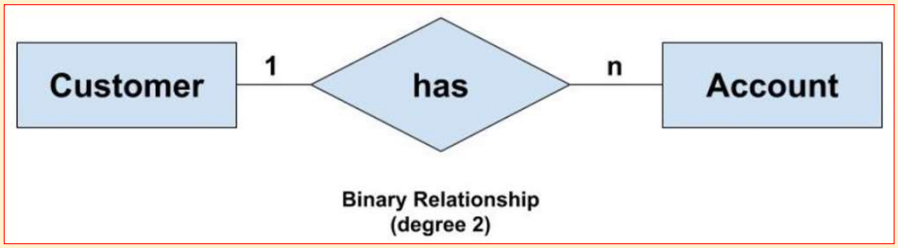
- There can be only one entity type in a relationship and the minimum degree of a relationship can be one.



Unary Relationship
(degree 1)

❖Binary (degree 2)

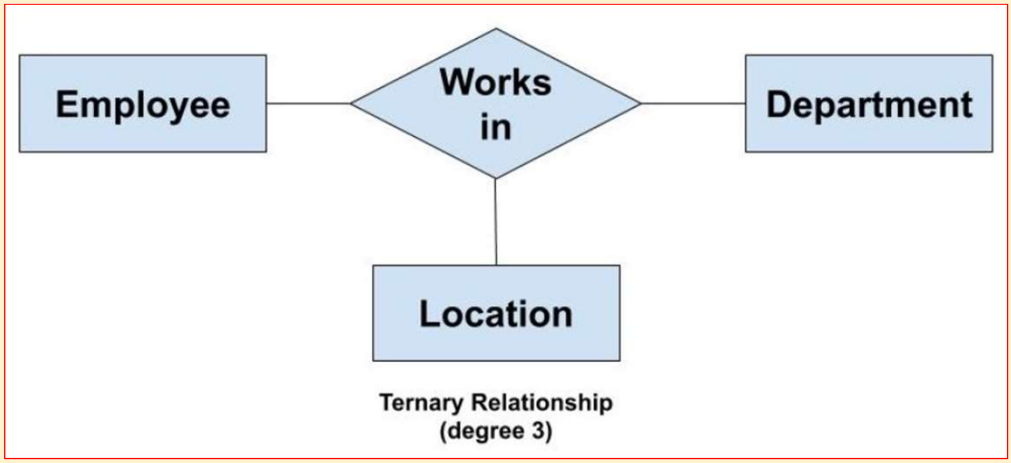
- A binary relationship exists when exactly two entity type participates.
- When such a relationship is present we say that the degree is 2.
- This is the most common degree of relationship.
- It is easy to deal with such relationship as these can be easily converted into relational tables.



Binary Relationship
(degree 2)

❖Ternary(degree 3)

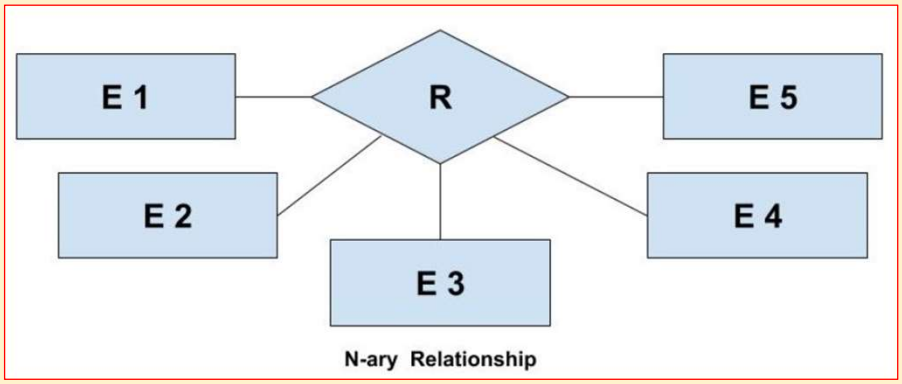
- A ternary relationship exists when exactly three entity type participates.
- When such a relationship is present we say that the degree is 3.
- As the number of entity increases in the relationship, it becomes complex to convert them into relational tables.
- For example, We have three entity type 'Employee', 'Department' and 'Location'. The relationship between these entities are defined as an employee works in a department, an employee works at a particular location. So, we can see we have three entities participating in a relationship so it is a ternary relationship. The degree of this relation is 3.



Ternary Relationship
(degree 3)

❖N-ary (n degree)

- An N-ary relationship exists when 'n' number of entities are participating. So, any number of entities can participate in a relationship. There is no limitation to the maximum number of entities that can participate.



N-ary Relationship