

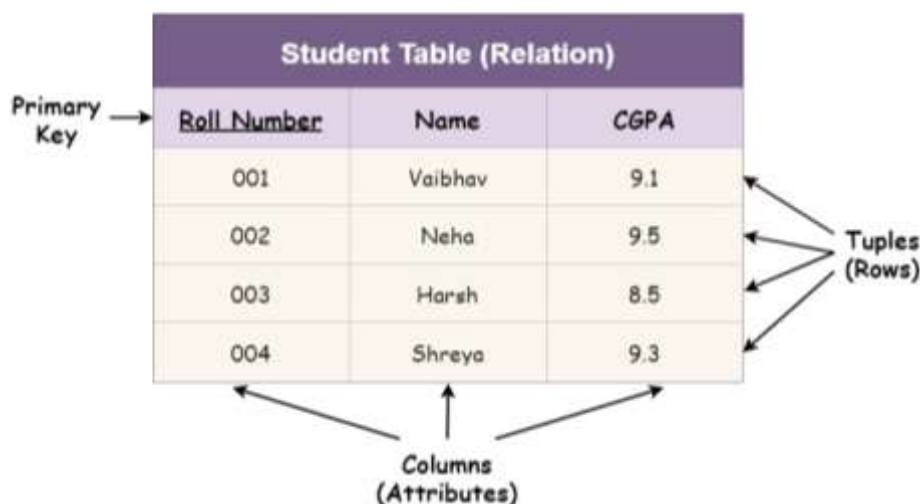
What is data model? Explain 2 data models in brief

- Underlying the structure of a database is called as the data model. It is a collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints.
- A data model provides a way to describe the design of a database at the physical, logical, and view levels.
- There are a number of different data models, The data models can be classified into four different categories:
 1. Relational Model.
 2. Entity-Relationship Model.
 3. Object-Based Data Model.
 4. Semistructured Data Model.

Relational Model :

- The relational model uses a collection of tables to represent both data and the relationships among those data.
- A relational model uses tables for representing data and in-between relationships. Each table has multiple columns, and each column has a unique name. Tables are also known as relations.
- Each row is known as a [tuple](#) (a tuple contains all the data for an individual record) while each column represents an attribute.
- **scenario to understand the relational model:**

Relational Model in DBMS



Consider a case where you wish to store the name, the CGPA attained, and the roll number of all the students of a particular class.






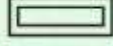
- Any given row of the relation indicates a student i.e., the row of the table describes a real-world entity.
- The columns of the table indicate the attributes related to the entity. In this case, the roll number, CGPA, and the name of the student.

Entity-Relationship Data Model:

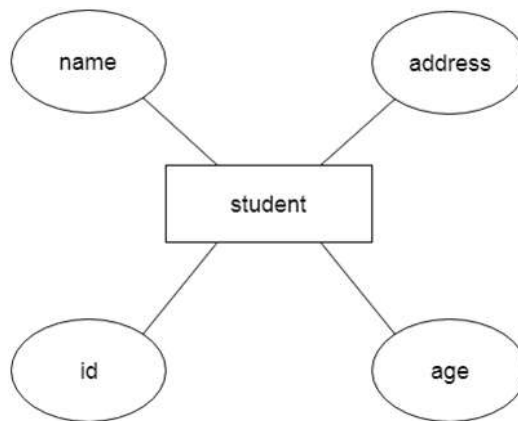
- An ER model is the logical representation of data as objects and relationships among them.
- An entity is a “thing” or “object” in the real world that is distinguishable from other objects.
- The Entity Relationship Diagram explains the relationship among the entities present in the database. ER models are used to model real-world objects like a person, a car, or a company and the relation between these real-world objects.

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:

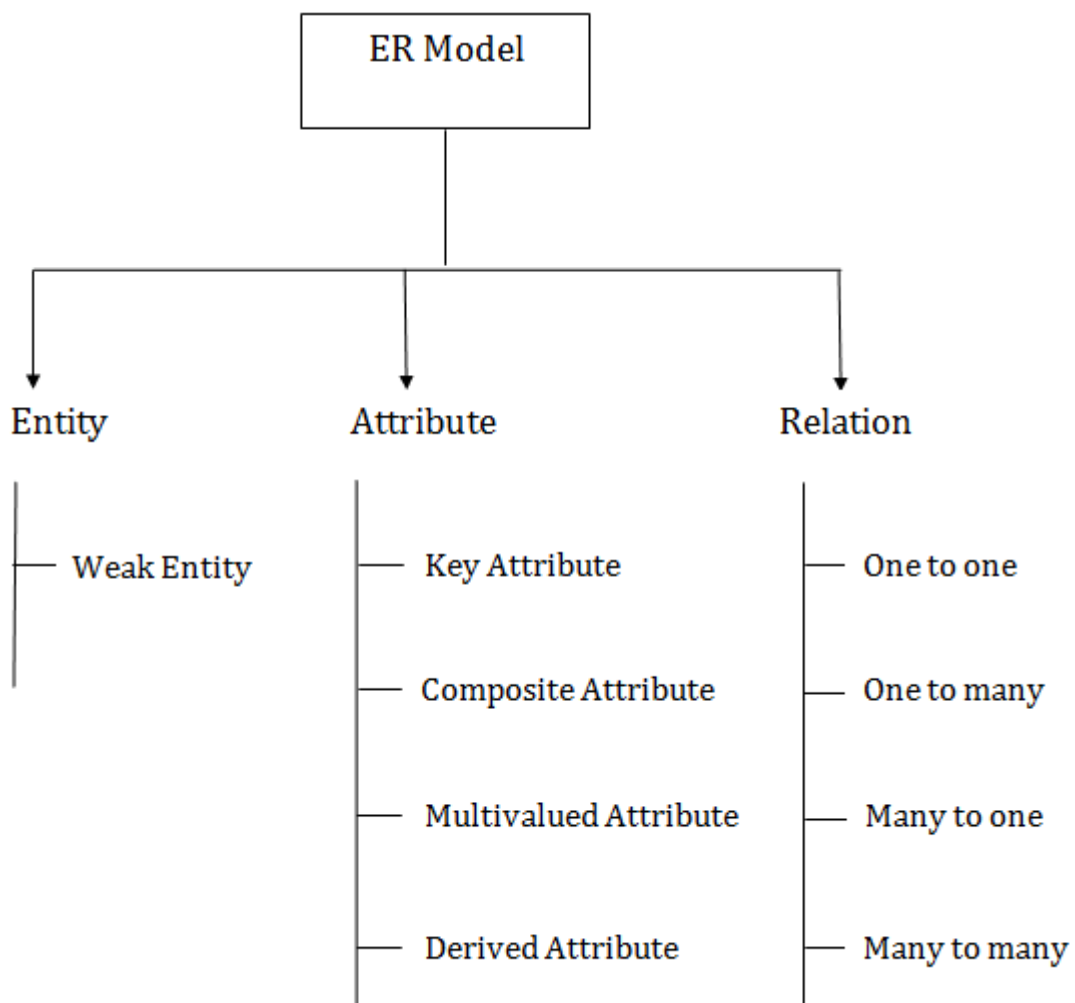
Figures	Symbols	Represents
Rectangle		Entities in ER Model
Ellipse		Attributes in ER Model
Diamond		Relationships among Entities
Line		Attributes to Entities and Entity Sets with Other Relationship Types
Double Ellipse		Multi-Valued Attributes
Double Rectangle		Weak Entity

- In ER modeling, the database structure is portrayed as a diagram called an entity-relationship diagram.
- Consider the example ,



- **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 :





Define and Differentiate between Super Key, Candidate Key and Primary Key. Give appropriate example

- Keys are one of the basic requirements of a relational database model. It is widely used to identify the tuples(rows) uniquely in the table. We also use keys to set up relations amongst various columns and tables of a relational database.

Super Key :

- A superkey is a set of one or more attributes that, taken collectively, allow to identify uniquely a tuple in the relation.
- The role of the super key is simply to identify the tuples of the specified table in the database. It is the superset where the candidate key is a part of the super key only.
- So, all those attributes in a table that is capable of identifying the other attributes of the table in a unique manner are all super keys.
- A superkey may contain extraneous attributes. A relation can have at most one super key.

Let's consider an **EMPLOYEE_DETAIL** table example where we have the following attribute:

1. **Emp_SSN:** The SSN number is stored in this field.
2. **Emp_Id:** An attribute that stores the value of the employee identification number.
3. **Emp_name:** An attribute that stores the name of the employee holding the specified employee id.
4. **Emp_email:** An attribute that stores the email id of the specified employees.

The **EMPLOYEE_DETAIL** table is given below,



Emp_SSN	Emp_Id	Emp_name	Emp_email
11051	01	John	john@email.com
19801	02	Merry	merry@email.com
19801	03	Riddle	riddle@email.com
41201	04	Cary	cary@email.com

So, from the above table, we conclude the following set of the super keys :

Set of super keys obtained
{ Emp_SSN }
{ Emp_Id }
{ Emp_email }
{ Emp_SSN, Emp_Id }
{ Emp_Id, Emp_name }
{ Emp_SSN, Emp_Id, Emp_email }
{ Emp_SSN, Emp_name, Emp_Id }

- These all are the set of super keys which, together or combining with other prime attributes, can identify a table uniquely.

Candidate key :

- A candidate key is an attribute or set of attributes that can uniquely identify a tuple.
- A **candidate key** is a part of a key known as **Super Key**, where the super key is the super set of all those attributes that can uniquely identify a table.
- Although the purpose of both candidate and the primary key is the same, that is to uniquely identify the tuples, and then also they are different from each other. It is because, in a table, we can have one or more than one candidate key, but we can create only one primary key for a table.
- Thus, from the number of obtained candidate keys, we can identify the appropriate primary key. However, if there is only one candidate key in a table, then it can be considered for both key constraints.
- Example of Candidate Key : The **EMPLOYEE_DETAIL** table is given below

Emp_SSN	Emp_Id	Emp_name	Emp_email
11051	01	John	john@email.com
19801	02	Merry	merry@email.com
19801	03	Riddle	riddle@email.com
41201	04	Cary	cary@email.com

- Now, from the previous sets of super keys, we can conclude the candidate keys. In order to pick up the candidate keys, the best way is to analyze and form the primary keys as much as we can.
- So, we need to identify those sets from the super key sets that alone can identify the whole table, or we can say the other attributes of the table. Thus, the result is :

Candidate Keys :

Emp_SSN

Emp_Id

Emp_email

Primary Key :

- We shall use the term primary key to denote a candidate key that is chosen by the database designer as the principal means of identifying tuples within a relation.
- A Primary Key is the minimal set of attributes of a table that has the task to uniquely identify the rows, or we can say the tuples of the given particular table.
- An entity can contain multiple keys. The key which is most suitable from those lists becomes a primary key.

Properties of a Primary Key:

- A relation can contain only one primary key.

- A primary key may be composed of a single attribute known as single primary key or more than one attribute known as composite key.
 - A primary key is the minimum super key.
 - The data values for the primary key attribute should not be null.
 - Attributes which are part of a primary key are known as Prime attributes.
 - Primary key is always chosen from the possible candidate keys.
- **For example:** When we store the registration details of the students in the database, we find the registration number field unique and assign the primary key to the field

Below is the table named **STUDENT_DETAILS**, where Roll_no, Name, and Marks are the specified attributes of it.

STUDENT_DETAILS			
	Roll_no	Name	Marks
Primary Key	101	X	34
	102	Y	46
	103	Z	94

Consider a two-dimensional array of $n*m$ size that is to be used in your favorite programming language. Using the array as an example demonstrate the difference between:

(a) the three levels of data abstraction

(b) schema and instance.

Data Abstraction:

Data abstraction involves hiding the complexity of data structures and operations to simplify interactions with the system.

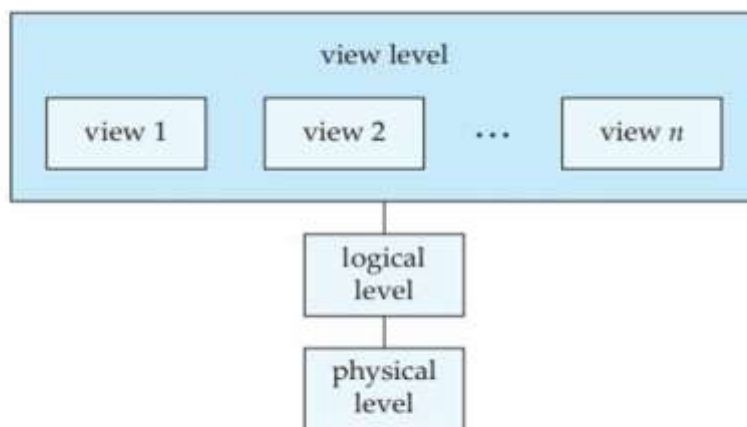


Figure 1.1 The three levels of data abstraction.

1. Physical Level:

- Describes how the data are actually stored in the system.
- Involves low-level details of storage structures.
- **Example:**
 - In a programming language, a customer record might be represented as a block of consecutive storage locations.
 - Database Management Systems (DBMS) hide these storage details from programmers.

2. Logical Level:

- Describes what data are stored in the database and the relationships among them.
- Focuses on type definitions and interrelationships of record types.
- **Example:**
 - In a programming language, a logical record might be defined with fields like name, street, and phone for a customer.
 - Programmers and Database Administrators (DBAs) work at this level of abstraction.

3. View Level:

- Describes only part of the entire database, providing a customized view for users.
- Users interact with application programs that hide data details.
- **Example:**
 - University clerks might have access to a specific view of student data but not the account section.
 - Offers security mechanisms to control access to database parts.

Instances and Schemas:

- **Instances:**

Instances refer to the collection of information stored in the database at a particular moment.

- **Example:**

- If a database table "student" has 100 records today, then today's instance of the database has 100 records.
 - Tomorrow, if another 100 records are added, the instance of the database will have 200 records.

- **Schema:**

Schema represents the overall design of the database, including its logical structure.



- **Example:**

- A database schema might consist of information about customers, accounts, and their relationships.
- It corresponds to variable declarations and type definitions in a program.

- **Types of Schema:**

- **Physical Schema:** Describes the database design at the physical level.
- **Logical Schema:** Describes the database design at the logical level.
- **Subschemas:** Represent different views of the database, allowing different users to see customized views.

- **Physical Data Independence:**

- Application programs exhibit physical data independence if they don't depend on the physical schema.

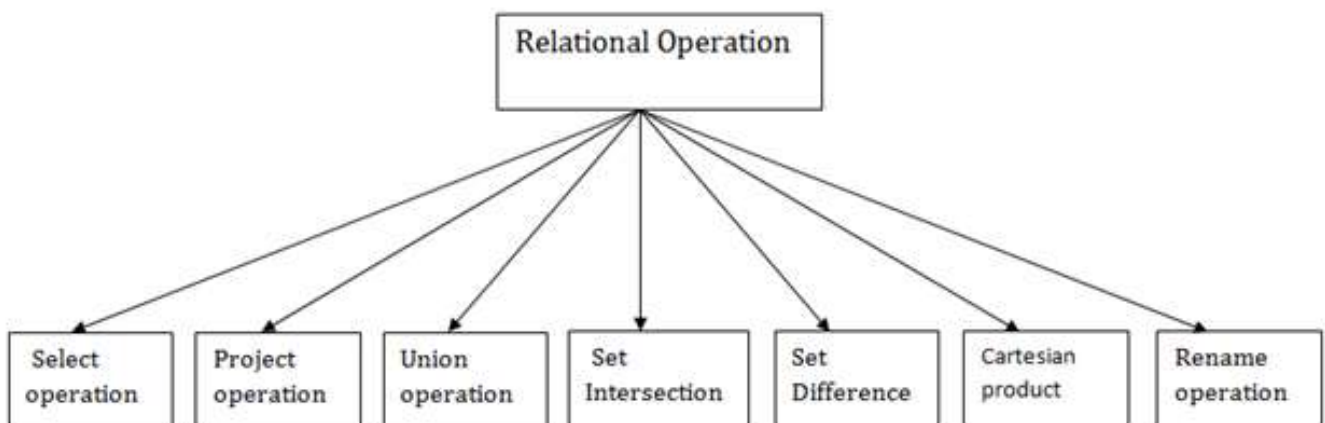
Changes in the physical schema should not require rewriting application programs



List the basic operators of Relational Algebra. Explain any 3 operators with respective syntax and example.

- Relational Algebra is a procedural query language that takes relations as an input and returns relations as an output. There are some basic operators which can be applied in relation to producing the required results

Types of Relational operation



Selection operation :

- Selection operator is used to selecting tuples from a relation based on some condition.
- Only those tuples that fall under certain conditions are selected. The selection operator denoted by sigma σ .
- Following is the syntax of selection operation :

$\sigma(\text{condition})(\text{relation_name})$

Notation: $\sigma p(r)$

Where:

σ is used for selection prediction

r is used for relation

p is used as a propositional logic formula which may use connectors like: AND OR and NOT. These relational can use as relational operators like $=, \neq, \geq, <, >, \leq$.

- For example: LOAN Relation :**

BRANCH_NAME	LOAN_NO	AMOUNT
Downtown	L-17	1000
Redwood	L-23	2000
Perryride	L-15	1500
Downtown	L-14	1500
Mianus	L-13	500
Roundhill	L-11	900
Perryride	L-16	1300

Input:

1. σ BRANCH_NAME="perryride" (LOAN)

Output:

BRANCH_NAME	LOAN_NO	AMOUNT
Perryride	L-15	1500
Perryride	L-16	1300

Project Operation :

- The projection eliminates all attributes of the input relation but those mentioned in the projection list. The projection method defines a relation that contains a vertical subset of Relation.

- This helps to extract the values of specified attributes to eliminates duplicate values. (pi) symbol is used to choose attributes from a relation.
- This operator helps you to keep specific columns from a relation and discards the other columns.
- Following is the syntax of projection operation :

$\pi_{(Column\ 1, Column\ 2, \dots, Column\ n)}(Relation\ Name)$

Example: CUSTOMER RELATION

NAME	STREET	CITY
Jones	Main	Harrison
Smith	North	Rye
Hays	Main	Harrison
Curry	North	Rye
Johnson	Alma	Brooklyn
Brooks	Senator	Brooklyn

Input:

$\pi_{NAME, CITY}(CUSTOMER)$

Output:

NAME	CITY
Jones	Harrison
Smith	Rye
Hays	Harrison

Curry	Rye
Johnson	Brooklyn
Brooks	Brooklyn

Union Operation:

- Suppose there are two tuples R and S. The union operation contains all the tuples that are either in R or S or both in R & S.
- It eliminates the duplicate tuples. It is denoted by U.

Notation: $R \cup S$

union operation must hold the following condition:

- R and S must have the attribute of the same number.
- Duplicate tuples are eliminated automatically.

Example: DEPOSITOR RELATION

CUSTOMER_NAME	ACCOUNT_NO
Johnson	A-101
Smith	A-121
Mayes	A-321
Turner	A-176
Johnson	A-273
Jones	A-472
Lindsay	A-284

BORROW RELATION

CUSTOMER_NAME	LOAN_NO
Jones	L-17
Smith	L-23
Hayes	L-15
Jackson	L-14
Curry	L-93
Smith	L-11
Williams	L-17

Input:

1. π CUSTOMER_NAME (BORROW) \cup π CUSTOMER_NAME (DEPOSITOR)

Output:

CUSTOMER_NAME
Johnson
Smith
Hayes
Turner
Jones
Lindsay
Jackson
Curry



Williams

Mayes



List different types of users in Database environment. Explain the role played by each of the listed users.

- Any person who uses a database and avails benefits from the database is known as database user in DBMS.
- Database users in DBMS can **access the database and retrieve the data** from the database using applications and interfaces provided by the **Database Management System (DBMS)**.
- Database users in DBMS can be categorized based on their interaction with the databases. According to the tasks performed by the database users on the databases, we can categorize them into seven categories as follows:
 1. Database Administrators (DBA)
 2. Database Designers
 3. System Analysts
 4. Application Programmers / Back-End Developers
 5. Naive Users / Parametric Users
 6. Sophisticated Users
 7. Casual Users / Temporary Users

Database Administrators (DBA) :

- Database Administrator (DBA) is a person/team who defines the schema and also controls the 3 levels of database.
- The DBA will then create a new account id and password for the user if he/she need to access the database.
- Database Administrators (DBAs) have full control of the database and they are sometimes known as the **super-users** of the database.
- DBA is also responsible for providing security to the database and he allows only the authorized users to access/modify the data base.

Database Designers :

- Database Designers are the users in DBMS who design and create the [structure of the database](#) including **triggers, indexes, schemas, entity relationships, tables, constraints**, etc. which complete the database.
- Database designers try to gather information depending upon the requirements related to the database like the **layout, looks, database**

functioning, costing, technologies to be used & implementation techniques.

- **Database Designers** are the type of database users in DBMS who are responsible for **implementing the overall design** of the database.
- They decide which form of data needs to be stored, what kind of relations exist among different entities of the database, what will be the type of attributes etc.

System Analysts :

- System Analyst is a user who analyzes the requirements of parametric end users. They check whether all the requirements of end users are satisfied.
- Analyzing **feasibility, economic and technical aspects** are some of the major responsibilities for a system analyst in DBMS.
- Sometimes, they are also responsible for the design, structure & functioning of the database.
- They usually check and gather all the necessary information related to the database, and if needed, they can change or **update the final layout** of the database as per requirements

Application Programmers / Back-End Developers :

- Application Programmers also referred as System Analysts or simply Software Engineers, are the back-end programmers who writes the code for the application programs.
- They are the computer professionals. These programs could be written in Programming languages such as Visual Basic, Developer, C, FORTRAN, COBOL etc.
- Application programmers design, debug, test, and maintain set of programs called “canned transactions” for the Naive (parametric) users in order to interact with database.

Naive Users / Parametric Users :

- Naive users also known as Parametric End users, don't have any knowledge of DBMS but still frequently use the database applications to get the desired results.

- With the help of the interface provided by the [DBMS applications](#), **Naive users** mostly use the database to fill in or retrieve the information (view level of the database).
- Naive users don't need to be aware of the presence of the database system as they can interact with the database with the help of **menu-driven** application interface.
- In simple terms, **Naive / Parametric End users** work directly on developed applications to access the database indirectly for getting the desired results.

Sophisticated Users :

- Sophisticated users are the type of database users in DBMS who know DBMS (DDL & DML commands) and are familiar with the database.
- Sophisticated users can be business analysts, engineers, scientists, system analysts, etc.
- Sophisticated users can develop or access their database applications according to the requirements, without actually writing the program code for it.
- **These users are also known as SQL programmers** as they can interact with the database directly using SQL queries using query processors.

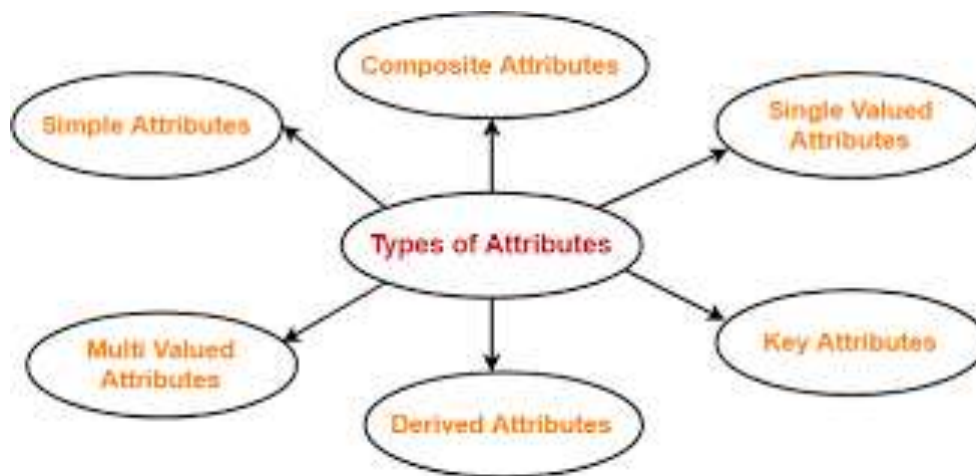
Casual Users / Temporary Users :

- Casual users also known as temporary users, are the type of database users in DBMS who frequently or occasionally use the database services.
- Whenever these users try to access the database, they want all the information sorted in place.
- Casual/Temporary users have little knowledge about DBMS and each time they try to access the database, they require new information.
- For example: **High-level management** people are casual users who have little knowledge about DBMS and hence, they can access the database to either fill in new information or retrieve existing results.

List and explain different types of attributes with appropriate examples for each.

- An attribute is a property or characteristic of an entity. An entity may contain any number of attributes.
- In relational databases, we have tables, and each column contains some entity that has some attributes, so all the entries for that column should strictly follow the attribute of the entity.

The different types of attributes are as follows –



Simple Attributes :

- Simple attributes are those that cannot be further divided into sub-attributes.
- For example, A student's roll number of a student or the employee identification number. These attributes are usually assigned by an organization and are used to identify an individual within that organization uniquely.
- Databases and other systems often use simple attributes to track and manage information.

Composite Attribute :

- Composite attributes are those that are made up of the composition of more than one attribute.
- When any attribute can be divided further into more sub-attributes, then that attribute is called a composite attribute.

- For example, in a student table, we have attributes of student names that can be further broken down into first name, middle name, and last name. So the student name will be a composite attribute.

Single-valued Attribute :

- 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.
- For example, the DOB of a student can be a single valued attribute. Another example is gender because one person can have only one gender.

Multi-valued Attribute:

- 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.

Derived Attribute:

- Derived attributes are also called stored attributes. When one attribute can be derived from the other attribute, then it is called a derived attribute. We can do some calculations on normal attributes and create derived attributes.
- For example, the age of a student can be a derived attribute because we can get it by the DOB of the student.
- Another example can be of working experience, which can be obtained by the date of joining of an employee.

Complex Attribute:

- If any attribute has the combining property of multi values and composite attributes, then it is called a complex attribute.



- It means if one attribute is made up of more than one attribute and each attribute can have more than one value, then it is called a complex attribute.
- For example, if a person has more than one office and each office has an address made from a street number and city. So the address is a composite attribute, and offices are multi valued attributes, So combining them is called complex attributes.

Key Attribute:

- Those attributes which can be identified uniquely in the relational table are called key attributes.
- DBMS's key attributes are used to uniquely identify each row in a table. Usually, there is more than one key attribute in a table (primary key and foreign key).
- For example, a student is a unique attribute
- For example: In a table of employees, the employee ID would be the primary key, while the manager ID would be the foreign key.



List and Explain the Drawbacks of using file systems to store data

- Before the origin of Computer Systems or before their heavy usage, the data were used to be stored in files manually. This means that the same data could have been present multiple times in a single institution.
- The traditionally used “File Systems” were nothing but a manual way of storing data as “Files”.

Disadvantages of the Traditional File Systems

Distinguished and Isolated Data:

- Imagine a user needs information that is not possible to be provided using a single file, multiple different files were required, which are situated in different departments.
- So all the employees first need to manually and carefully check each of the files in each department and find the relationship between them to decipher the information that the user wants.

Data Duplication / Data Redundancy :

- Since each application has its own data file, the same data may have to be recorded and stored in many files. For example, personal file and payroll file, both contain data on employee name, designation etc.
- The result is unnecessary duplicate or redundant data items. This redundancy requires additional or higher storage space, costs extra time and money, and requires additional efforts to keep all files up-to-date.
- Storing the same data multiple times not only wastes resources in every machine but also is costly to maintain and wastes time.

Dependence on Data :

- Files and information were stored in a certain specific format in files which is hard coded by programmers in languages like C/C++, COBOL, etc.
- So if any of the file's format changes then the programmers need to update the code every time and the format of every piece of data stored in that file will be changed, which is a rigorous task for programmers.

- For example, in payroll application, the file may be organized on employee records sorted on their last name, which implies that accessing of any employee's record has to be through the last name only.

Data representation is challenging from the user's perspective –

- Data stored in files must be presented in a clear and understandable manner for every user.
- Programmers often face challenges in navigating through different systems to find connections between data and merge them into a more human-readable format.
- Effective communication between programmers and users is essential to ensure that the represented data meets the users' understanding and requirements.

Different file types :

- The variations in file structures based on programming languages can lead to compatibility challenges.
- Data stored in one format may not be easily accessible or compatible with systems utilizing a different programming language, requiring additional effort for conversion or integration.
- Managing multiple file types adds complexity to the maintenance process.

Data Protection :

- Data protection was very less due to different reasons like Data Redundancy, manual storing of data, easy access of confidential data by unauthorized parties, etc.
- Data redundancy and the decentralized nature of traditional file systems amplify the risk of data breaches.
- Without adequate security measures in place, sensitive data becomes vulnerable to breaches and unauthorized use.

Lack of Data Integration :



- The existence of independent data files obstructs users from efficiently accessing information for ad hoc queries.
- Complicated programs often need to be developed to retrieve data from multiple files. This requirement increases the complexity of data retrieval processes.
- In such a case complicated programs have to be developed to retrieve data from every file or the users have to manually collect the required information

Program Dependence :

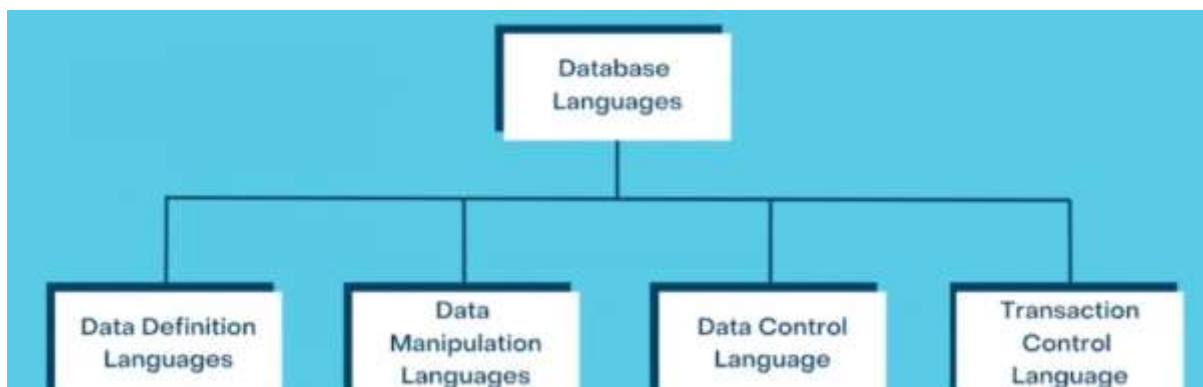
- The reports produced by the file processing system are program dependent, which means if any change in the format or structure of data and records in the file is to be made, the programs have to be modified correspondingly.
- Also, a new program will have to be developed to produce a new report.



List the types of database languages. Explain each type with appropriate example

- A DBMS has appropriate languages and interfaces to express database queries and updates.
- Database languages can be used to read, store and update the data in the database.
- These languages allow users to complete tasks such as controlling access to data, defining and updating data and searching for information within the database management system (DBMS).

Different types of DBMS languages are as follows –



Data Definition Language (DDL) :

- **DDL** stands for **Data Definition Language**. It is used to define database structure or pattern.
- It is used to create schema, tables, indexes, constraints, etc. in the database.
- Using the DDL statements, you can create the skeleton of the database.
- Data definition language is used to store the information of metadata like the number of tables and schemas, their names, indexes, columns in each table, constraints, etc.

Here are some tasks that come under DDL:

- **Create:** It is used to create objects in the database.
- **Alter:** It is used to alter the structure of the database.



- **Drop:** It is used to delete objects from the database.
- **Truncate:** It is used to remove all records from a table.
- **Rename:** It is used to rename an object.
- **Comment:** It is used to comment on the data dictionary.

These commands are used to update the database schema that's why they come under Data definition language.

Data Manipulation Language (DML) :

DML stands for **Data Manipulation Language**. It is used for accessing and manipulating data in a database. It handles user requests.

Here are some tasks that come under DML:

- **Select:** It is used to retrieve data from a database.
- **Insert:** It is used to insert data into a table.
- **Update:** It is used to update existing data within a table.
- **Delete:** It is used to delete all records from a table.
- **Merge:** It performs UPSERT operation, i.e., insert or update operations.
- **Call:** It is used to call a structured query language or a Java subprogram.
- **Explain Plan:** It has the parameter of explaining data.
- **Lock Table:** It controls concurrency.

Data Control Language (DCL) :

- **DCL** stands for **Data Control Language**. It is used to retrieve the stored or saved data.
- The DCL execution is transactional. It also has rollback parameters.

(But in Oracle database, the execution of data control language does not have the feature of rolling back.)

Here are some tasks that come under DCL:

- **Grant:** It is used to give user access privileges to a database.





- **Revoke:** It is used to take back permissions from the user.

There are the following operations which have the authorization of Revoke:

CONNECT, INSERT, USAGE, EXECUTE, DELETE, UPDATE and SELECT

Transaction Control Language (TCL):

TCL is used to run the changes made by the DML statement. TCL can be grouped into a logical transaction.

Here are some tasks that come under TCL:

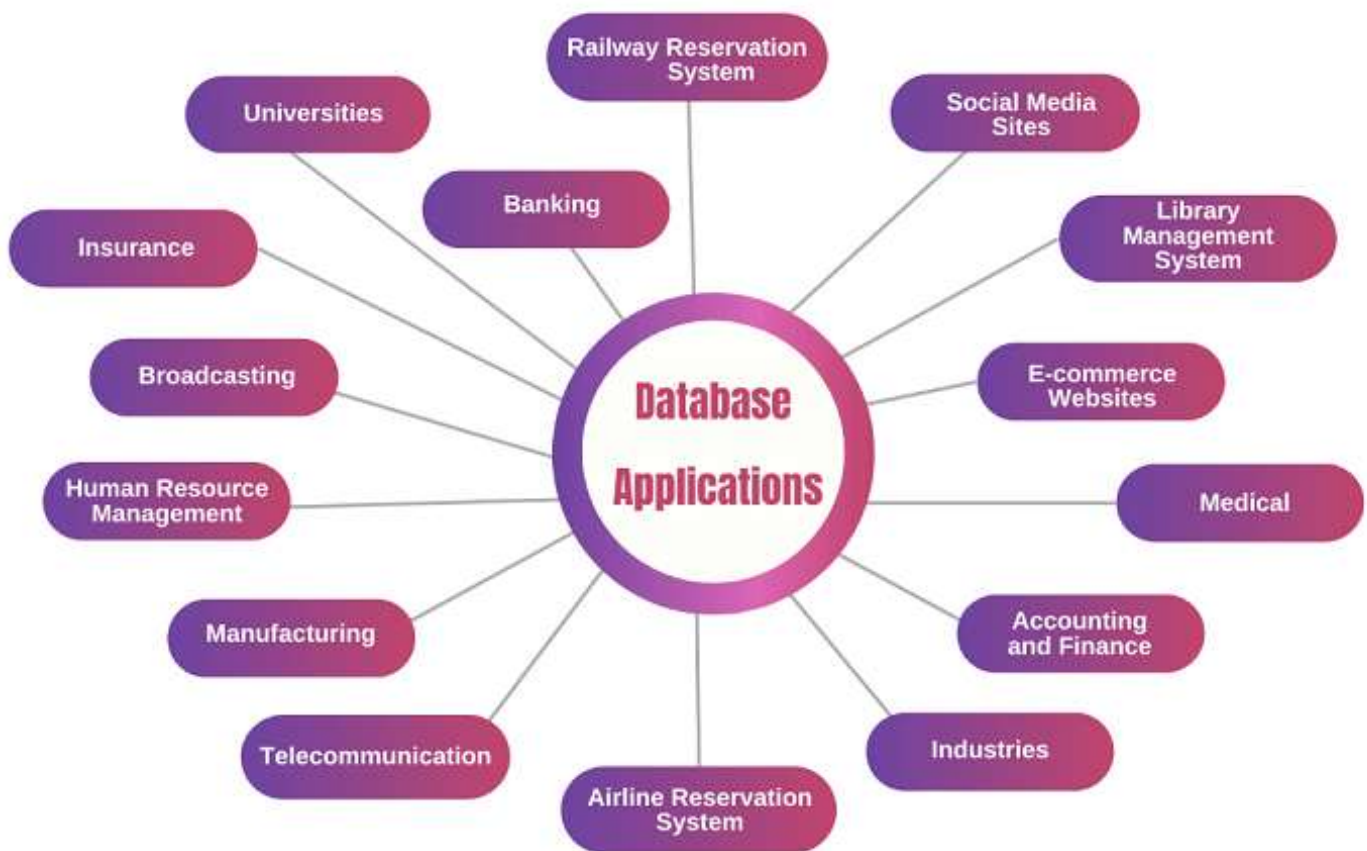
- **Commit:** It is used to save the transaction on the database.
- **Rollback:** It is used to restore the database to original since the last Commit.



Define Database. List and explain the applications of Database.

- A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a [database management system \(DBMS\)](#).
- A database-management system (DBMS) is a collection of interrelated data and a set of programs to access those data.

Databases are widely used. Here are some representative applications :



Universities:

- Universities utilize databases to store vast amounts of student, teacher, and course information securely.
- This ensures easy retrieval of data, maintaining comprehensive records for future reference.

Banking:

- In the banking sector, databases manage extensive customer data, including transactions, balances, and financial records.
- By ensuring systematic organization and security, databases facilitate efficient banking operations and customer service.

Railway Reservation System:

- Railway reservation systems rely on databases to handle passenger information, bookings, and train schedules effectively.
- This ensures smooth operations and provides passengers with accurate and up-to-date information.

Social Media Sites:

- Social media platforms store millions of user accounts and multimedia content within databases.
- Databases enable scalable and secure storage, facilitating seamless user interactions and content management.

Library Management System:

- Libraries leverage databases to maintain comprehensive records of books, issuances, and returns.
- This ensures efficient management of library resources and easy access to information for patrons.

E-commerce Websites:

- E-commerce platforms manage vast amounts of customer, product, and transaction data through databases.
- Databases play a crucial role in ensuring secure transactions, accurate inventory management, and personalized customer experiences.

Medical:

- In the medical field, databases store patient records, prescriptions, and treatment histories securely.



- This facilitates efficient patient care, enables accurate diagnosis, and ensures compliance with regulatory requirements.

Accounting and Finance:

- Databases in accounting and finance manage financial transactions, invoices, and statements effectively.
- They enable accurate financial reporting, analysis, and decision-making by organizing data in a structured manner.

Industries:

- Industries rely on databases to manage customer data, sales records, and production information efficiently.
- Databases facilitate streamlined operations, supply chain management, and data-driven decision-making.

Airline Reservation System:

- Airline reservation systems store passenger details, flight schedules, and booking information in databases.
- This ensures smooth operations, accurate seat allocation, and timely communication with passengers.

Telecommunication:

- Telecommunication companies manage customer accounts, call details, and network usage data through databases.
- Databases enable efficient billing, network management, and personalized service offerings.

Manufacturing:

- Manufacturing companies utilize databases to manage product details, supply chain data, and production schedules.
- Databases ensure efficient inventory management, quality control, and timely delivery of goods.

Human Resource Management:



- HR departments store employee information, payroll details, and benefits administration records within databases.
- This facilitates efficient HR operations, talent management, and compliance with employment regulations.

Broadcasting:

- Broadcasting organizations store subscriber information, event schedules, and recording details in databases.
- Databases enable efficient content distribution, audience management, and programming scheduling.

Insurance:

- Insurance companies manage policy details, user information, and claims records through databases.
- Databases ensure accurate policy management, claims processing, and personalized customer service.









Explain the ER Model in details.

- The entity-relationship (E-R) data model uses a collection of basic objects, called **entities**, and **relationships** among these objects.
- An entity is a “thing” or “object” in the real world that is distinguishable from other objects. For example, each person is an entity, and bank accounts can be considered as entities.
- Entities are described in a database by a set of attributes. For example, the attributes dept name, building, and budget may describe one particular department in a university, and they form attributes of the department entity set.
- The ER data model specifies enterprise schema that represents the overall logical structure of a database graphically.
- ER models are used to model real-world objects like a person, a car, or a company and the relation between these real-world objects.

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 :

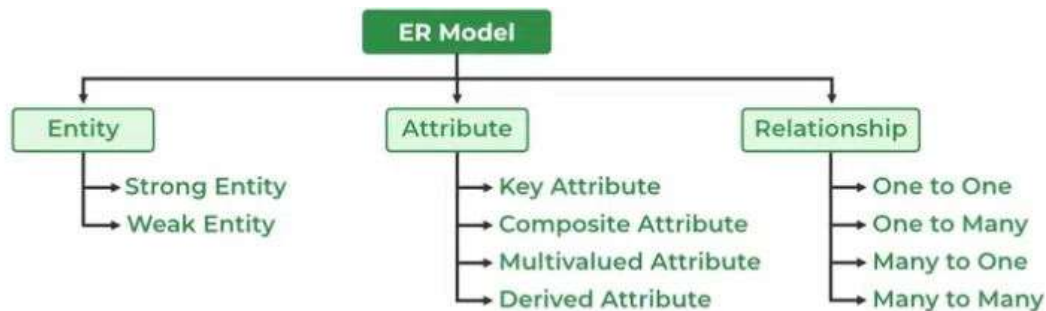
Figures	Symbols	Represents
Rectangle		Entities in ER Model
Ellipse		Attributes in ER Model
Diamond		Relationships among Entities
Line		Attributes to Entities and Entity Sets with Other Relationship Types
Double Ellipse		Multi-Valued Attributes
Double Rectangle		Weak Entity

- The overall logical structure (schema) of a database can be expressed graphically by an entity-relationship (E-R) diagram. There are several ways in which to draw these diagrams.
- ER diagrams provide the purpose of real-world modeling of objects which makes them intently useful.

- It gives a standard solution for visualizing the data logically.

Components of ER Diagram

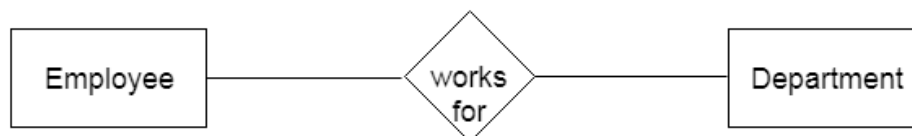
ER Model consists of Entities, Attributes, and Relationships among Entities in a Database System.



1. Entity:

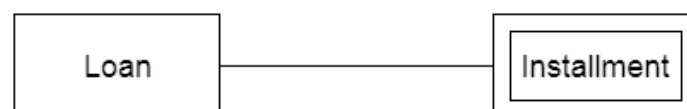
An entity may be any object, class, person or place. In the ER diagram, an entity can be represented as rectangles.

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



a. Weak Entity

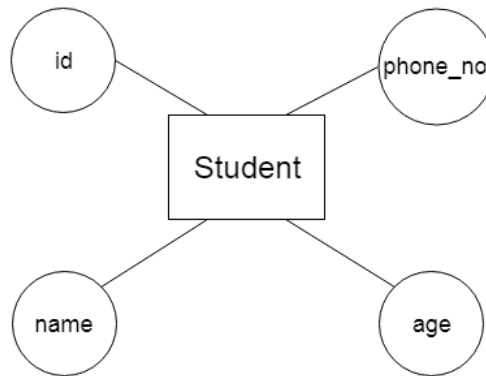
An entity that depends on another entity called a weak entity. The weak entity doesn't contain any key attribute of its own. The weak entity is represented by a double rectangle.



2. Attribute

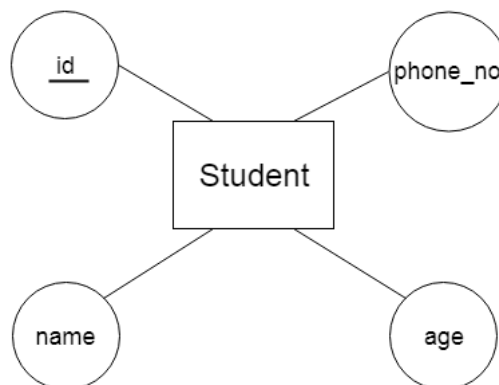
The attribute is used to describe the property of an entity. Eclipse is used to represent an attribute.

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



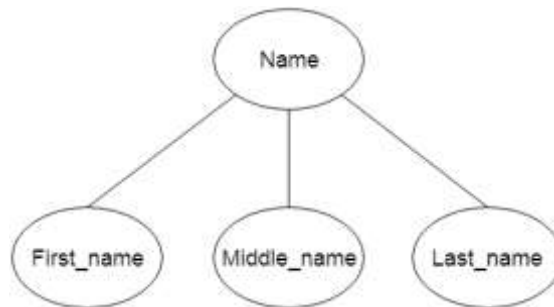
a. Key Attribute

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

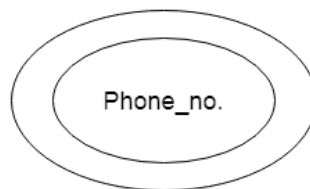
An attribute that composed of many other attributes is known as a composite attribute. The composite attribute is represented by an ellipse, and those ellipses are connected with an ellipse.



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.

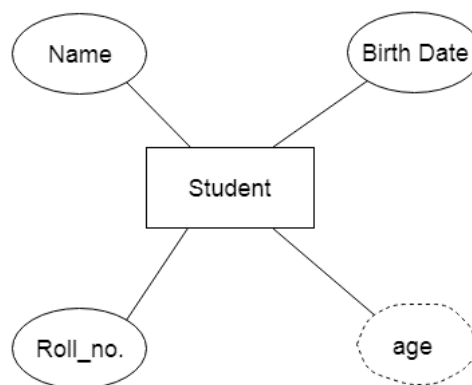
For example, a student can have more than one phone number.



d. Derived Attribute

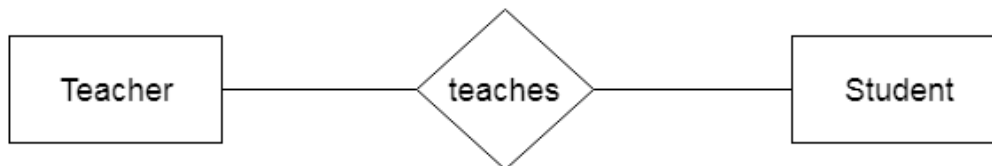
An attribute that can be derived from other attribute is known as a derived attribute. It can be represented by a dashed ellipse.

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.

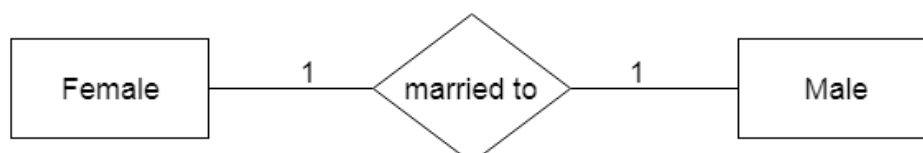


Types of relationship are as follows:

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.

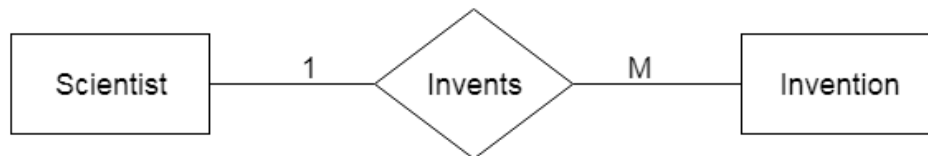
For example, A female can marry to one male, and a male can marry to one female.



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.

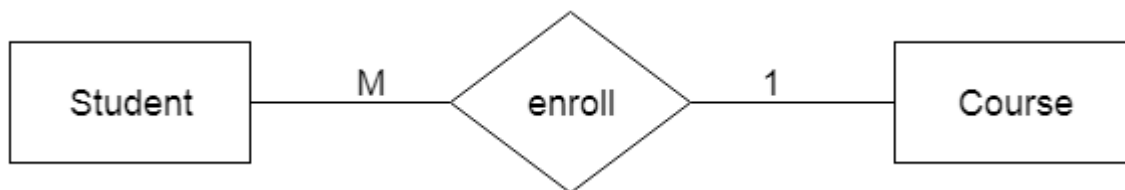
For example, Scientist can invent many inventions, but the invention is done by the only specific scientist.



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.

