Database Management System (DBMS)

V – SEM, CSE

Introduction to DBMS

What is Data ?

- Data is the plural of Datum which means a single piece of information.
- Data is defined as a known fact that can be recorded and that has implicit meaning.
- Data is a raw fact fro which the required information is produced.
- It can be numbers, letters, words, special symbols etc.
- Data by themselves has **NO MEANING**.

Example

- 5689912 is meaningless by itself since it does not signify anything.

- So, it is our data and not information.

- Other examples of data can be objects like documents, video segments, photographic images etc.

Information

- Information is defined as collection of related data that when put together, communicate meaningful and useful message to a recipient who uses it, to make decision or to interpret the data to get the meaning.

- We can say it is data which is converted into a more useful or intelligible form.

- Example: Marks obtained by students & their RollNo form **Data** but their Marksheet is **Information**.

Dimensions of Information

- Dimensions of Information are
- the syntactic,
- the semantic and
- the pragmatic

Traffic Signal Example

In the syntactic: – We differentiate the three colours red, yellow and green.

More sense in the semantic dimension. – The colours are linked to meanings. Red means stop, green means go.

Only in the pragmatic dimension does the traffic light become useable for the traffic. – Red means that the driver of a car must stop.

Knowledge

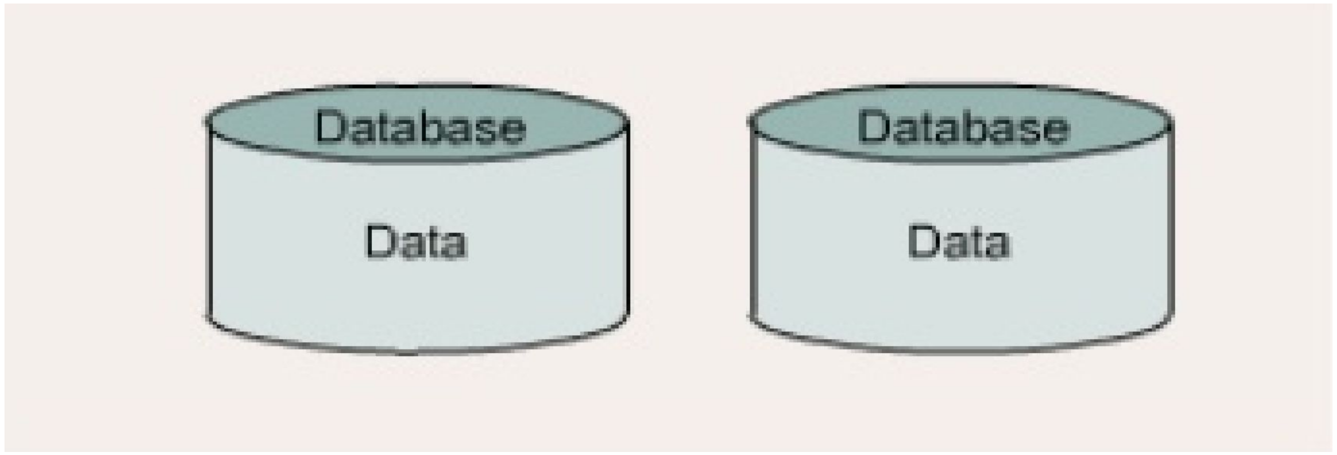
- The Information may be further processed to form Knowledge.

- Information containing wisdom is known as Knowledge.

$$\begin{tikzcd}\boxed{\text{Data}} \arrow[r, "\text{When processed}"] & \boxed{\text{Information}} \arrow[r, "\text{When processed}"] & \boxed{\text{Knowledge}} \arrow[r, "\text{When processed}"] & \boxed{\text{Intelligence}}\end{tikzcd}$$

Database

- It is defined as the collection of logically inter-related data and a description of this data, designed to meet the information needs of an organization.
- Example: A Dictionary, Telephone directory, Student Record Register.



Features of data in Database

1) Shared
2) Persistence
3) Validity
4) Security
5) Consistency
6) Non – Redundancy
7) Independence

Database Management System

- A database-management system (DBMS) is a collection of interrelated data and a set of programs to access those data.
- Database management system is a software which is used to manage the database. For example: MySQLDatabase management system is a software which is used to manage the database. For example: MySQL, Oracle, etc are a very popular commercial database which is used in different applications.
- DBMS provides an interface to perform various operations like database creation, storing data in it, updating data, creating a table in the database and a lot more.
- The primary goal of a DBMS is to provide a way to store and retrieve database information that is both **convenient and efficient**

Database systems

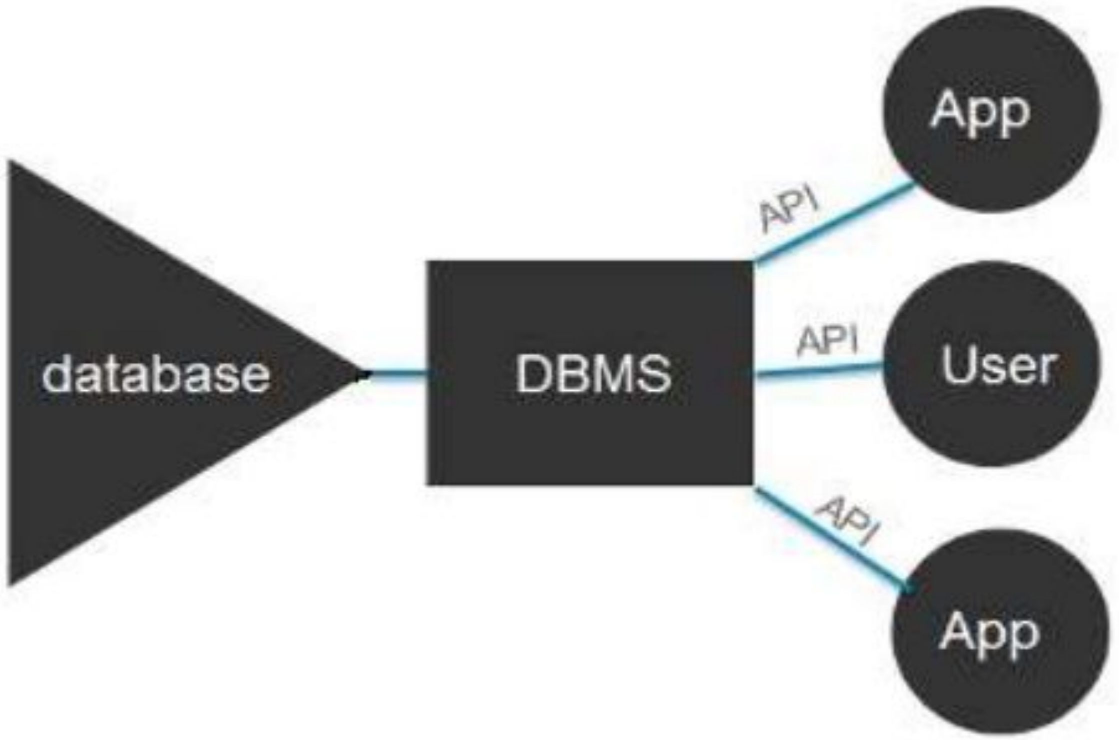
- Database systems are designed to manage large bodies of information. Management of data involves both defining structures for storage of information and providing mechanisms for the manipulation of information.
- In addition, the database system must ensure the safety of the information stored, despite system crashes or attempts at unauthorized access. If data are to be shared among several users, the system must avoid possible anomalous results

Database System Applications

- Banking: For customer information, accounts, and loans, and banking transactions.
- Airlines: For reservations and schedule information. Airlines were among the first to use databases in a geographically distributed manner—terminals situated around the world accessed the central database system through phone lines and other data networks.
- Universities: For student information, course registrations, and grades.

- Credit card transactions: For purchases on credit cards and generation of monthly statements.
- Telecommunication: For keeping records of calls made, generating monthly bills, maintaining balances on prepaid calling cards, and storing information about the communication networks.
- Finance: For storing information about holdings, sales, and purchases of financial instruments such as stocks and bonds.
- Sales: For customer, product, and purchase information.
- Manufacturing: For management of supply chain and for tracking production of items in factories, inventories of items in warehouses/stores, and orders for items.
- Human resources: For information about employees, salaries, payroll taxes and benefits, and for generation of paychecks.

Block Diagram of DBMS



Characteristics of DBMS

- It uses a digital repository established on a server to store and manage the information.
- It can provide a clear and logical view of the process that manipulates data.
- DBMS contains automatic backup and recovery procedures.
- It contains ACID properties which maintain data in a healthy state in case of failure.
- It can reduce the complex relationship between data.
- It is used to support manipulation and processing of data.

- It is used to provide security of data.
- It can view the database from different viewpoints according to the requirements of the user.

Advantages of DBMS

- **Controls database redundancy:** It can control data redundancy because it stores all the data in one single database file and that recorded data is placed in the database.
- **Data sharing:** In DBMS, the authorized users of an organization can share the data among multiple users.
- **Easily Maintenance:** It can be easily maintainable due to the centralized nature of the database system.

- **Reduce time:** It reduces development time and maintenance need.
- **Backup:** It provides backup and recovery subsystems which create automatic backup of data from hardware and software failures and restores the data if required.
- **Multiple user interface:** It provides different types of user interfaces like graphical user interfaces, application program interfaces

Disadvantages of DBMS

- **Cost of Hardware and Software:** It requires a high speed of data processor and large memory size to run DBMS software.
- **Size:** It occupies a large space of disks and large memory to run them efficiently.
- **Complexity:** Database system creates additional complexity and requirements.
- **Higher impact of failure:** Failure is highly impacted the database because in most of the organization, all the data stored in a single database and if the database is damaged due to electric failure or database corruption then the data may be lost forever.

History of DBMS

- **1960's-1970's:** The emergence of the first type of DBMS, the **hierarchical DBMS**. IBM had the first model, developed on IBM 360 and their (DBMS) was called IMS, originally it was written for the Apollo program. This type of DBMS was based on binary trees, where the shape was like a tree and relations were only limited between parent and child records. The benefits were numerous; less redundant data, data independence, security and integrity, which all lead to efficient searches. Nonetheless; there were some disadvantages such as; complex implementation, was hard to manage because of the absence of standards, which made it harder to handle many relationships

1960's-1970's: The emergence of the **network DBMS**. Charles Bachmann developed first DBMS at Honeywell, Integrated Data Store (IDS) then a group called CODASYL who is responsible for the creation of COBOL, had that system standardized. However; the CODASYL group invented what they call the "CODASYL APPROACH. Based on that approach many systems using network DBMS were developed for business use. In this model, each record can have multiple parents in comparison with one in the hierarchical DBMS. It is made of sets of relationships where a set represents a one to many relationship between the owner and the member. The main and unfortunate disadvantage was that the System was complex and there was difficulty in design and maintenance, it is believed that the ***Lack of structural independence*** was the main cause.

1970's- 1990's: The emergence of the relational DBMS on the hands of **Edgar Codd**. He worked at IBM, and he was unhappy with the navigational model of the CODASYL APPROACH. To him, a tool for searching, such as a search facility was very useful, and it was absent . In 1970, he proposed a new approach to database construction, which made the creation of a ***Relational DBMS*** intended for Large Shared Data Banks, possible and easy to grab (3). Moreover; This was a new system for entering data and working with big databases, where the idea was to use a table of records. All tables will be then linked by either one to one relationships, one to many, or many to many(2). when elements took space and were not useful, it was easy to remove them from the original table, and all the other "entries" in other tables linked to this record were removed.

Popular DBMS Software

Here, is the list of some popular DBMS system:

- **MySQL**
- **Microsoft Access**
- **Oracle**
- PostgreSQL
- dBASE
- FoxPro
- SQLite
- IBM DB2
- LibreOffice Base
- MariaDB
- Microsoft SQL Server etc.
- MongoDB

Flat File System

- File system is a collection of data. In this system, the user has to write the procedures for managing the database.
- File system provides the detail of the data representation and storage of data.
- File system doesn't have a crash mechanism, i.e., if the system crashes while entering some data, then the content of the file will lost.
- It is very difficult to protect a file under the file system.
- File system can't efficiently store and retrieve the data.
- In the File system, concurrent access has many problems like redirecting the file while other deleting some information or updating some information.

DBMS vs. File System

DBMS	File System
DBMS is a collection of data. In DBMS, the user is not required to write the procedures.	File system is a collection of data. In this system, the user has to write the procedures for managing the database.
DBMS gives an abstract view of data that hides the details.	File system provides the detail of the data representation and storage of data.
DBMS provides a crash recovery mechanism, i.e., DBMS protects the user from the system failure.	File system doesn't have a crash mechanism, i.e., if the system crashes while entering some data, then the content of the file will lost.
DBMS provides a good protection mechanism.	It is very difficult to protect a file under the file system.
DBMS contains a wide variety of sophisticated techniques to store and retrieve the data.	File system can't efficiently store and retrieve the data.
DBMS takes care of Concurrent access of data using some form of locking.	In the File system, concurrent access has many problems like redirecting the file while other deleting some information or updating some information.

Disadvantages of File System

1) Data redundancy and inconsistency: Redundancy leads to higher storage and access cost. In addition, it may lead to **data inconsistency**; that is, the various copies of the same data may no longer agree.

2) Difficulty in accessing data: Conventional file-processing environments do not allow needed data to be retrieved in a convenient and efficient manner. More responsive data-retrieval systems are required for general use.

3) Data isolation: Data are scattered in various files

4) Integrity problems: The data values stored in the database must satisfy certain types of **consistency constraints.**

5) Atomicity problems: Complete or None

6) Concurrent-access anomalies:

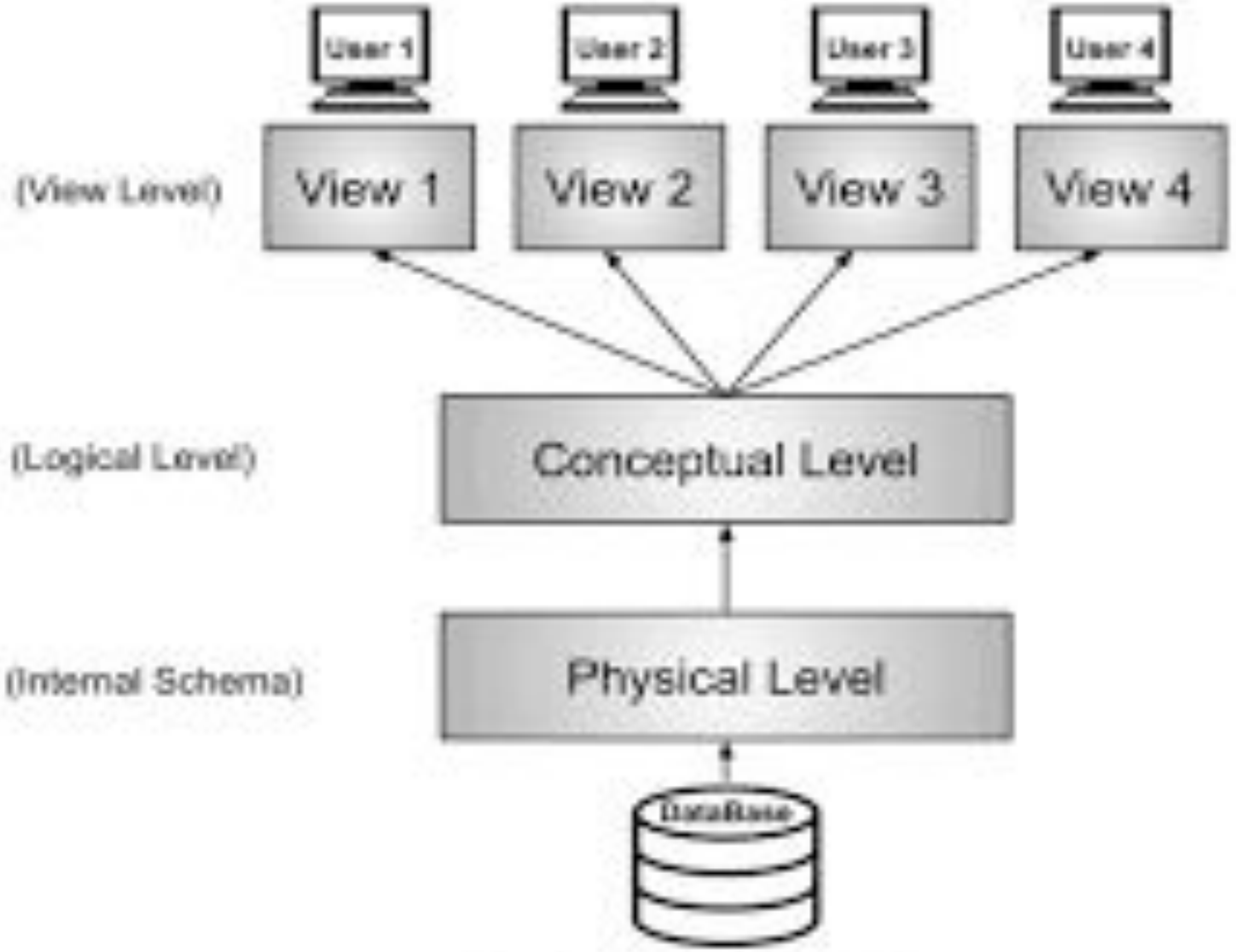
7) Security problems:

Disadvantages of DBMS

- Cost of Software
- Requires more space
- Backup facility required
- Complex backup and recovery operation
- Data can get corrupt

Data Abstraction

- It means to hide something.
- Database stored at complex level is kept hidden from the users at three different levels.
- The architecture of DBMS is divided into three general levels:
- 1) External or View Level
- 2) Conceptual or Logical Level
- 3) Internal or Physical Level



Levels of Data Abstraction

PHYSICAL LEVEL

- This is the lowest level of data abstraction.
- It describes how data is actually stored in database.
- We get the complex data structure details at this level.

LOGICAL LEVEL

- This is the middle level of the data abstraction architecture.
- It describes what data is stored in database.

VIEW LEVEL

- It is highest level of data abstraction.
- It describes the user interaction with database system.

Example:

We store a customer information in Customer table.

At Physical level, the records can be described as blocks of storage (bytes, GB, TB) in memory. These details are hidden from the programmers.

At Logical Level, these records can be described as the fields and attributes along with their data types, their relationship among each other can be logically implemented.

Programmers generally work at this level because they are aware of such things about database systems.

At View Level, user just interact with system with the help of GUI and enter details at the screen.

They are not ware of how the data is stored & what data is stored.

Roll_No	Student_Name	Section
1	Riya	A
2	Jai	A
30	Rati	A
144	Ram	B
82	Sita	B

STUDENT DETAILS TABLE

Roll_No, Student_Name, Section = **Attributes or fields**

Schema

- Design of a database is called as **SCHEMA.**
- **Schema is of three types:**

1) Physical Schema
2) Logical Schema
3) View Schema

Physical Schema

The design of database at Physical level is called Physical schema, how the data stored in blocks of storage is described at this level.

Logical Schema

- Design of database at logical level is called logical schema.
- Programmers and database administrator work at this level.
- At this level data can be described as certain types of data records which gets stored in data structures.

View Level

- Design of Database at the View Level is called View Schema.   It describes end user interaction with database systems.

Instance

- The data stored in database at a particular moment of time is called **instance** of database.

Data Independence

- Data Independence means that upper levels are unaffected by changes in lower levels.
- There are 2 types of Data Independence:

1) Physical Independence

2) Logical Independence

Logical Data Independence

It is data about database, that is it stores information about how data is managed.

Physical Data Independence

All schemas are logical and the actual data is stored in bit format on the disk.

Database Languages

- DDL (Data Definition Language)
- DML (Data Manipulation Language)
- DCL (Data Control Language)

DDL (Data Definition Language)

- It provides for the definition or description of database objects. It is basically used to define the structure of database.
- It is mainly used to create files, databases, tables within the databases.
- Example: Create, alter , drop, rename are DDL commands.

DML (Data Manipulation Language)

It is defined as a language that provides a set of operations to support the basic data manipulation operations on the data held in databases.

 It allows user to insert update, delete and retrieve data from the database.
 DML care of 02 types:
1) Procedural DML: It indicates the procedure of what and how to retrieve the data.
2) Non Procedural DML: It indicates the procedure of what to retrieve without concerning how to retrieve the data.

DCL (Data Control Language)
 DCL statement control access to data & the database using statements such as GRANT & REVOKE.
A privilege can be granted with GRANT and it can be removed with REVOKE.

WORKING OF DBMS

1. The users issues a query using a particular database language like SQL.
2. The passed query is presented to a query optimizer which use information about how the data is stored to produce an efficient execution plan for evaluating the query.
3. The DBMS accepts the users SQL commands and analyzes them.
4. The DBMS produces query evaluation plan i.e. the external schema for the user, the corresponding external/conceptual mapping and storage structure.
5. Therefore, an evaluation plan is a blueprint for evaluating a query. The DBMS executes this plans against the physical database and returns the results to the user.

Functions of DBA

1) Defining Conceptual Schema
2) Physical Database Design
3) Security & Integrity Checks
4) Give backup & Recovery Strategies
5) Granting access to users

Data Models

- It is defined as an integrated collection of concepts for describing & manipulating data, relationships between data & constraints on the data in an organization.
- Data model comprises of 03 components:

1) Structural Part: It consists of a set of rules according to which databases can be constructed.

2)Manipulative Part: It includes the operations that are used for updating or retrieving data from the databases & for changing the structure of the databases.

3) Integrity Rules: A set of rules which ensures that the data is accurate.

We will be studying 04 data models:

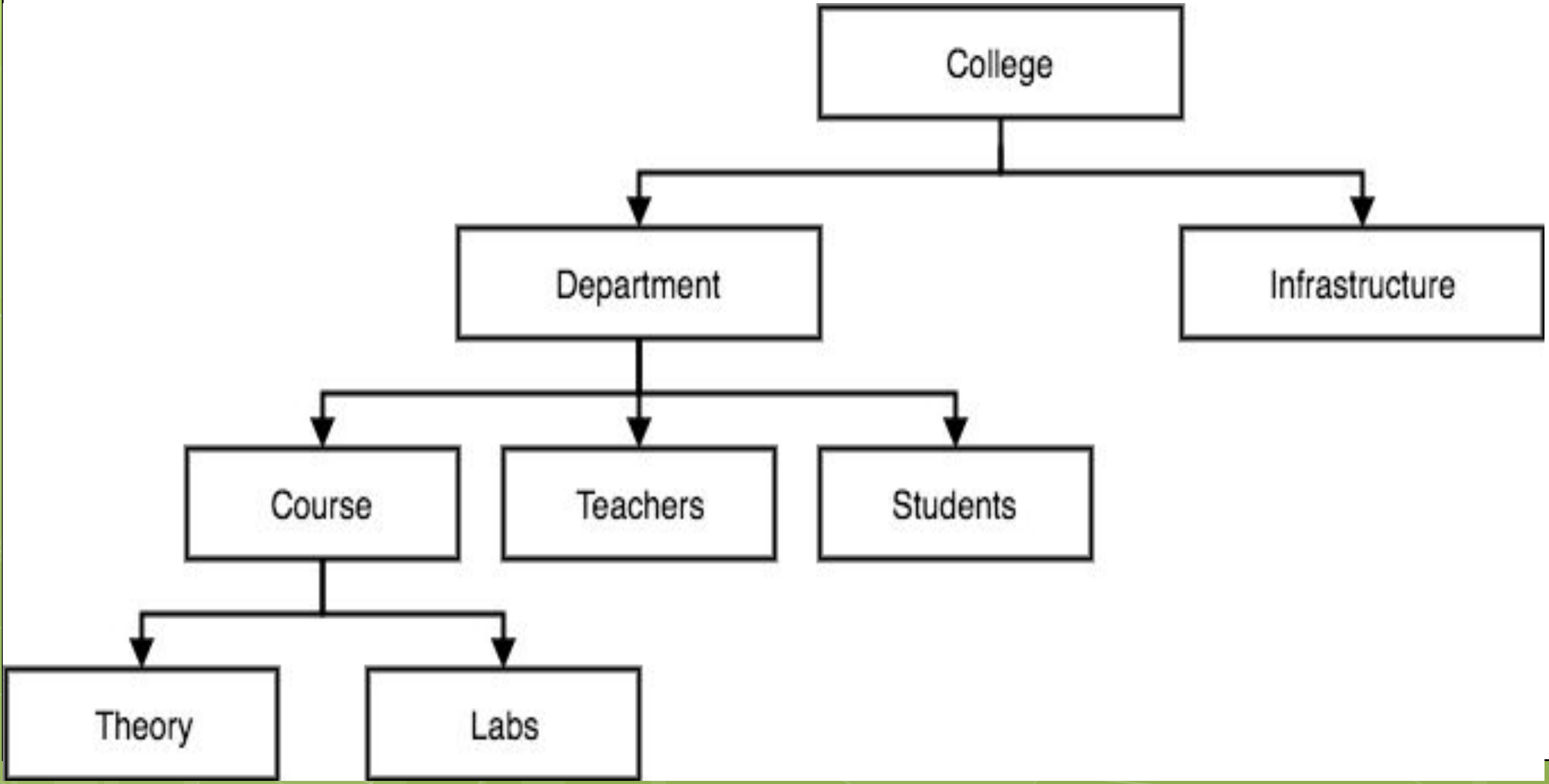
1)Hierarchical Model
2)Network Model
3)Relational Model
4)Entity Relational Model/ ER Model

1) **Hierarchical Model**

- This database model organizes data into a tree-like-structure, with a single root, to which all the other data is linked.
- The hierarchy starts from the **Root** data, and expands like a tree, adding child nodes to the parent nodes.
- In this model, a child node will only have a single parent node.
- This model efficiently describes many real-world relationships like index of a book, recipes etc.

- The hierarchical structure is used as the physical order of records in storage. One can access the records by navigating down through the data structure using pointers which are combined with sequential accessing.
- Therefore, the hierarchical structure is not suitable for certain database operations when a full path is not also included for each record.

- In hierarchical model, data is organized into tree-like structure with one-to-one & one-to-many relationship between two different types of data, for example, one department can have many courses, many professors and of-course many students.



2) Network Model

- This is an extension of the Hierarchical model. In this model data is organized more like a graph, and are allowed to have more than one parent node.
- In this database model data is more related as more relationships are established in this database model. Also, as the data is more related, hence accessing the data is also easier and fast. This database model was used to map many-to-many data relationships.

- **There are two fundamental concepts of a network model –**

- Records contain fields which need hierarchical organization.

Sets are used to define one-to-many relationships between records that contain one owner, many members.

A record may act as an owner in any number of sets, and a member in any number of sets.

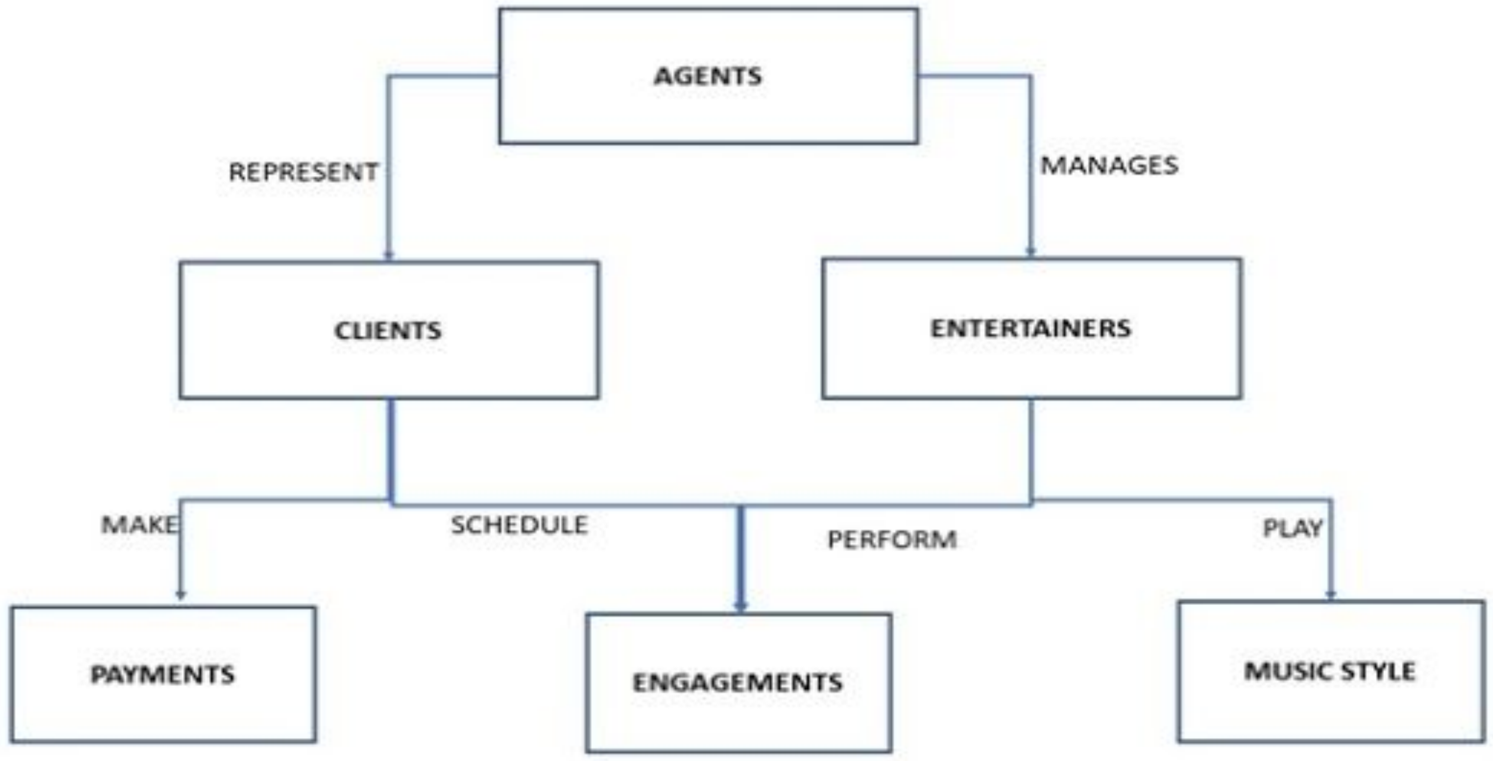
- A set is designed with the help of circular linked lists where one record type, the owner of the set also called as a parent, appears once in each circle, and a second record type, also known as the subordinate or child, may appear multiple times in each circle.

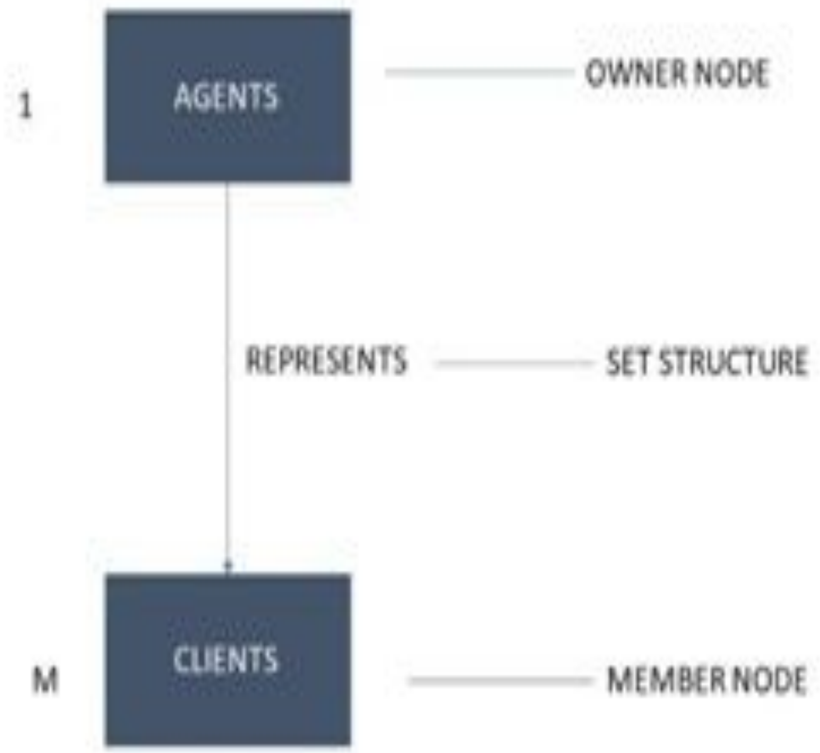
 A hierarchy is established between any two record types where one type (A) is the owner of another type (B). At the same time, another set can be developed where the latter set (B) is the owner of the former set (A). In this model, ownership is defined by the direction, thus all the sets comprise a general directed graph. Access to records is developed by the indexing structure of circular linked lists.

 The network model has the following major features −

1) It can represent redundancy in data more efficiently than that in the hierarchical model.
2) There can be more than one path from a previous node to successor node/s.
3) The operations of the network model are maintained by indexing structure of linked list (circular) where a program maintains a current position and navigates from one record to another by following the relationships in which the record participates.
4) Records can also be located by supplying key values.

The following diagram depicts a network model. An agent represents several clients and manages several entertainers. Each client schedules any number of engagements and makes payments to the agent for his or her services. Each entertainer performs several engagements and may play a variety of musical styles.




1
AGENTS
OWNER NODE
REPRESENTS
SET STRUCTURE
M
CLIENTS
MEMBER NODE

3) Relational Model

- Relational Model was proposed by E.F. Codd in 1970, to model data in the form of relations or tables.
- The model provides a simple concept of how users identify data.
- This model represents data in the form of two-dimensional tables.
- **RELATIONAL MODEL** represents the database as a collection of relations. A relation is nothing but a table of values. Every row in the table represents a collection of related data values. These rows in the table denote a real-world entity or relationship.
- The table name and column names are helpful to interpret the meaning of values in each row.
- The data are represented as a set of relations. In the relational model, data are stored as tables. However, the physical storage of the data is independent of the way the data are logically organized.

Relational Model Concepts

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

2) Tables: In the Relational model the, relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes.

3) Tuple: It is nothing but a single row of a table, which contains a single record.

4) Relation Schema: A relation schema represents the name of the relation with its attributes.

5) Degree: The total number of attributes which in the relation is called the degree of the relation.

6) Cardinality: Total number of rows present in the Table.

7) Column: The column represents the set of values for a specific attribute.
8)Relation instance: Relation instance is a finite set of tuples in the RDBMS system. Relation instances never have duplicate tuples.
9) Relation key : Every row has one, two or multiple attributes, which is called relation key.
10) Attribute domain: Every attribute has some pre-defined value and scope which is known as attribute domain

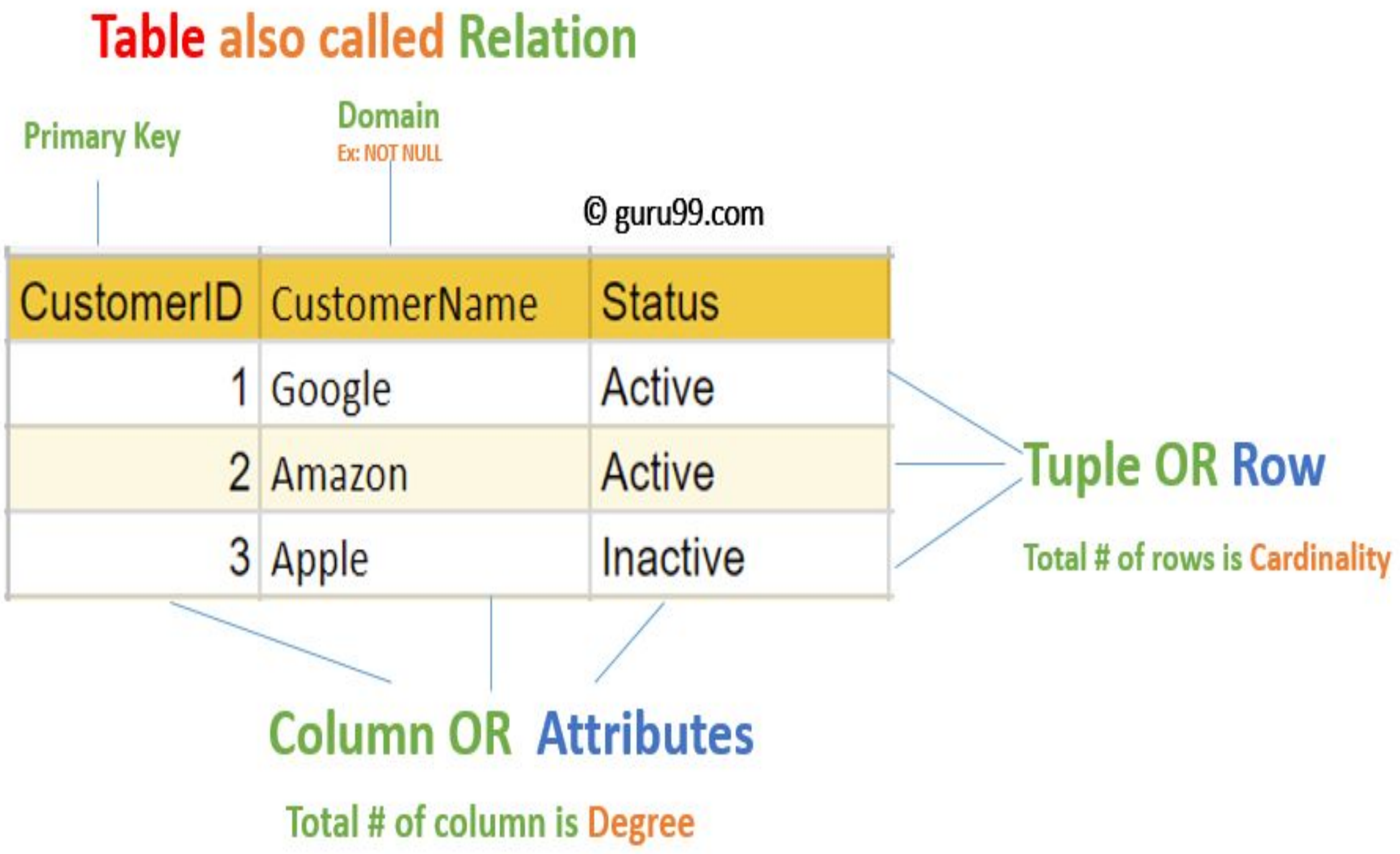


Table also called Relation
Primary Key
Domain
Ex: NOT NULL
© guru99.com
Tuple OR Row
Total # of rows is Cardinality
Column OR Attributes
Total # of column is Degree

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

CUSTOMER TABLE

Properties of Relational Model

1) Values are Atomic
2) Column values are of same kind
3) Each row is unique
4) The sequence of columns is insignificant
5) The sequence of rows is insignificant
6) Each column has unique name

Entity Relationship Model

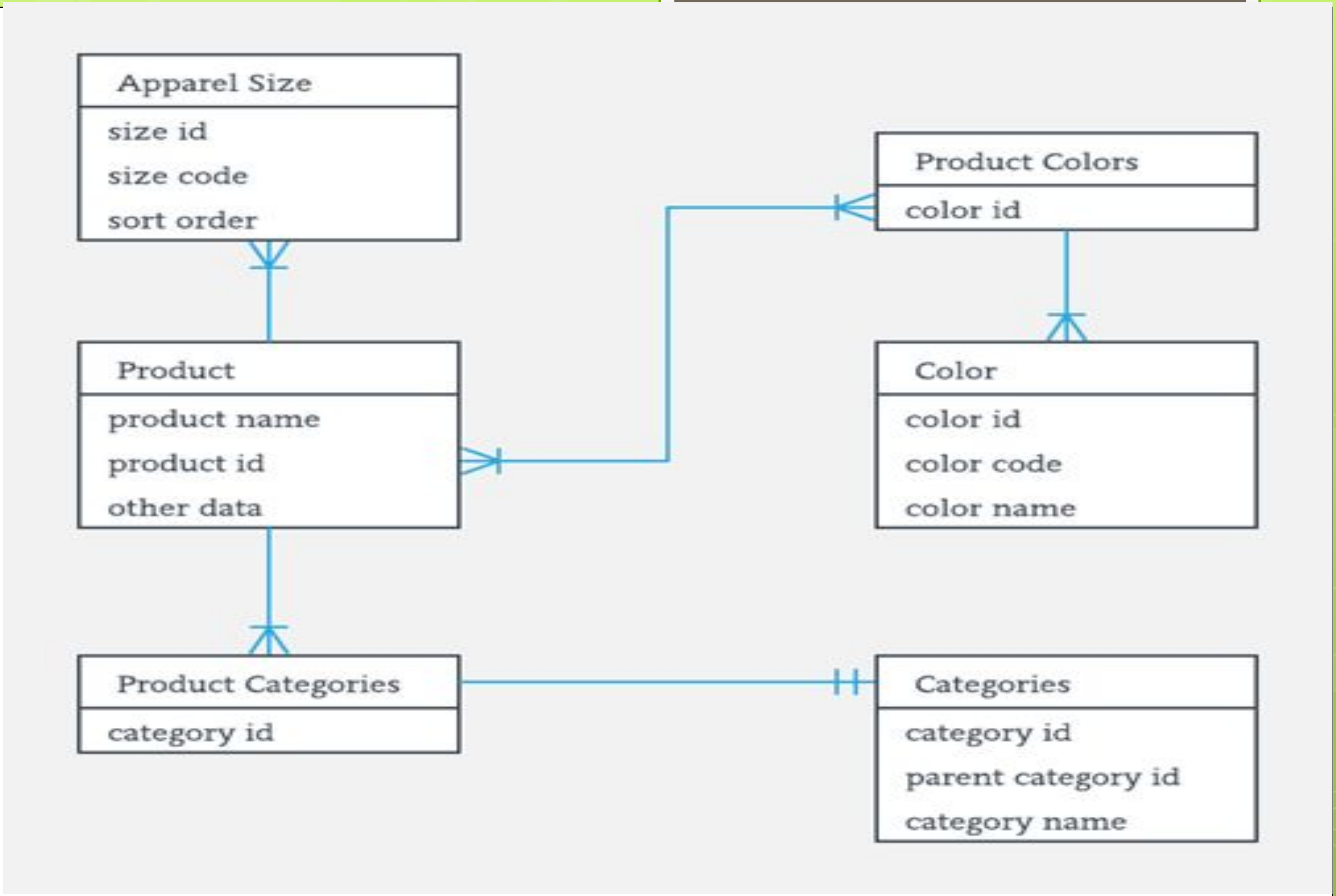
- **ENTITY RELATIONAL (ER) MODEL** is a high-level conceptual data model diagram. ER modeling helps you to analyze data requirements systematically to produce a well-designed database. The Entity-Relation model represents real-world entities and the relationship between them. It is considered a best practice to complete ER modeling before implementing your database.
- ER modeling helps you to analyze data requirements systematically to produce a well-designed database. So, it is considered a best practice to complete ER modeling before implementing your database.

History of ER models

ER diagrams are a visual tool which is helpful to represent the ER model. It was proposed by Peter Chen in 1971 to create a uniform convention which can be used for relational database and network. He aimed to use an ER model as a conceptual modeling approach.

What is ER Diagrams?

- **ENTITY-RELATIONSHIP DIAGRAM (ERD)** displays the relationships of entity set stored in a database. In other words, we can say that ER diagrams help you to explain the logical structure of databases. At first look, an ER diagram looks very similar to the flowchart. However, ER Diagram includes many specialized symbols, and its meanings make this model unique. The purpose of ER Diagram is to represent the entity framework infrastructure.



SAMPLE ER DIAGRAM

Facts about ER Diagram Model:

- ER model allows you to draw Database Design
- It is an easy to use graphical tool for modeling data
- Widely used in Database Design
- It is a GUI representation of the logical structure of a Database
- It helps you to identifies the entities which exist in a system and the relationships between those entities

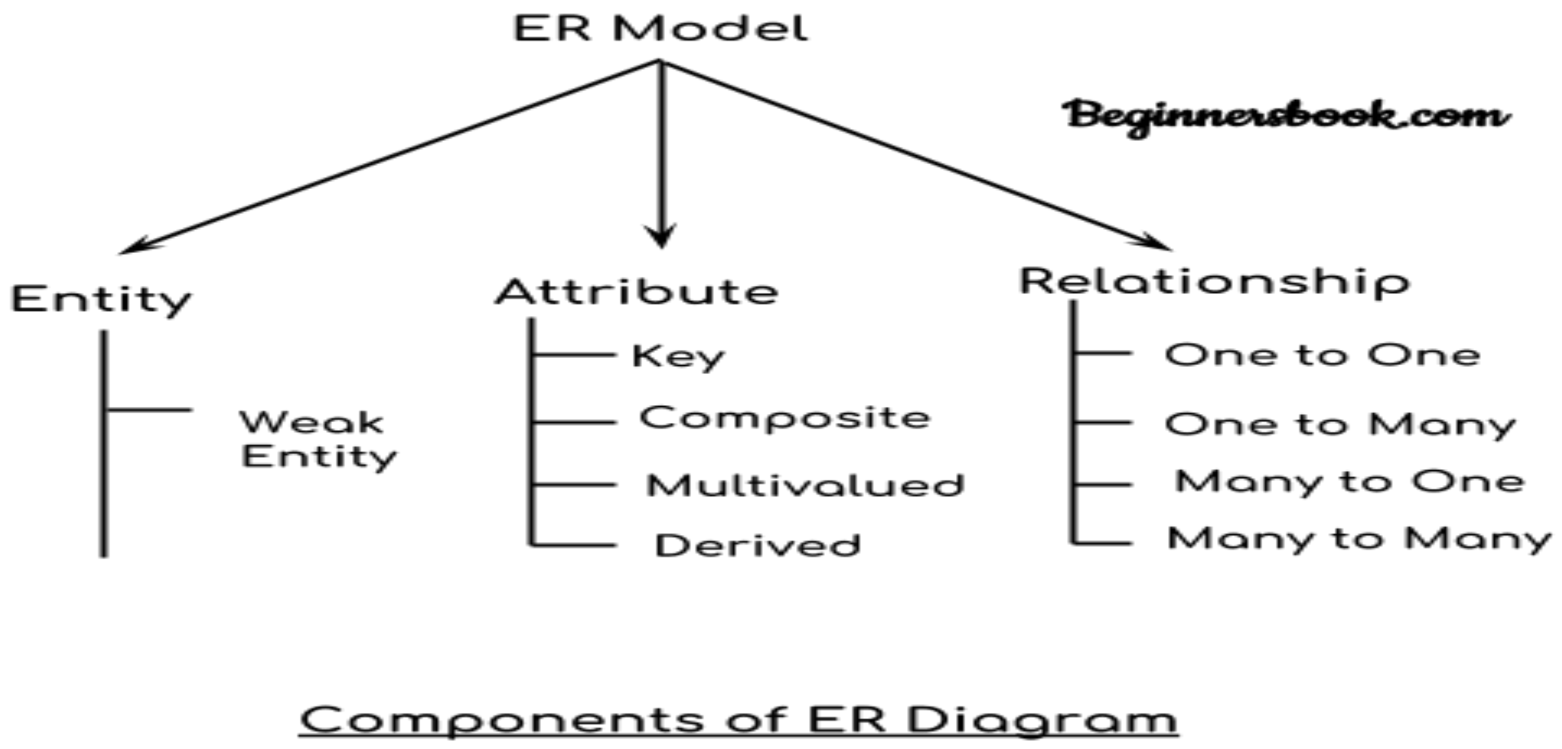
Why use ER Diagrams?

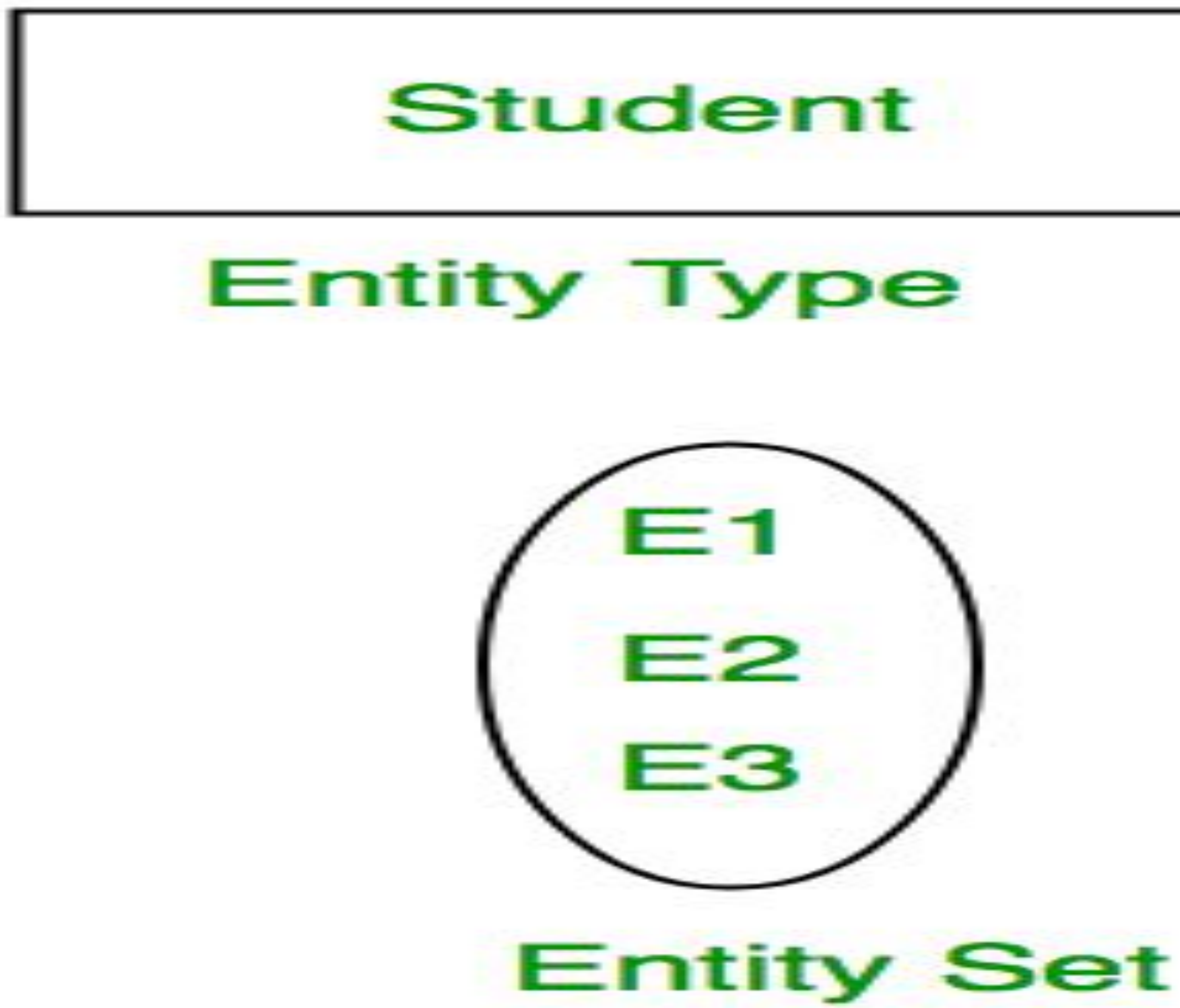
- Helps you to define terms related to entity relationship modeling
- Provide a preview of how all your tables should connect, what fields are going to be on each table
- Helps to describe entities, attributes, relationships
- ER diagrams are translatable into relational tables which allows you to build databases quickly
- ER diagrams can be used by database designers as a blueprint for implementing data in specific software applications
- The database designer gains a better understanding of the information to be contained in the database with the help of ERP diagram
- ERD is allowed you to communicate with the logical structure of the database to users.

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

Entity Type: An Entity is an object of Entity Type and set of all entities is called as entity set. e.g.; E1 is an entity having Entity Type Student and set of all students is called Entity Set. In ER diagram, Entity Type is represented as

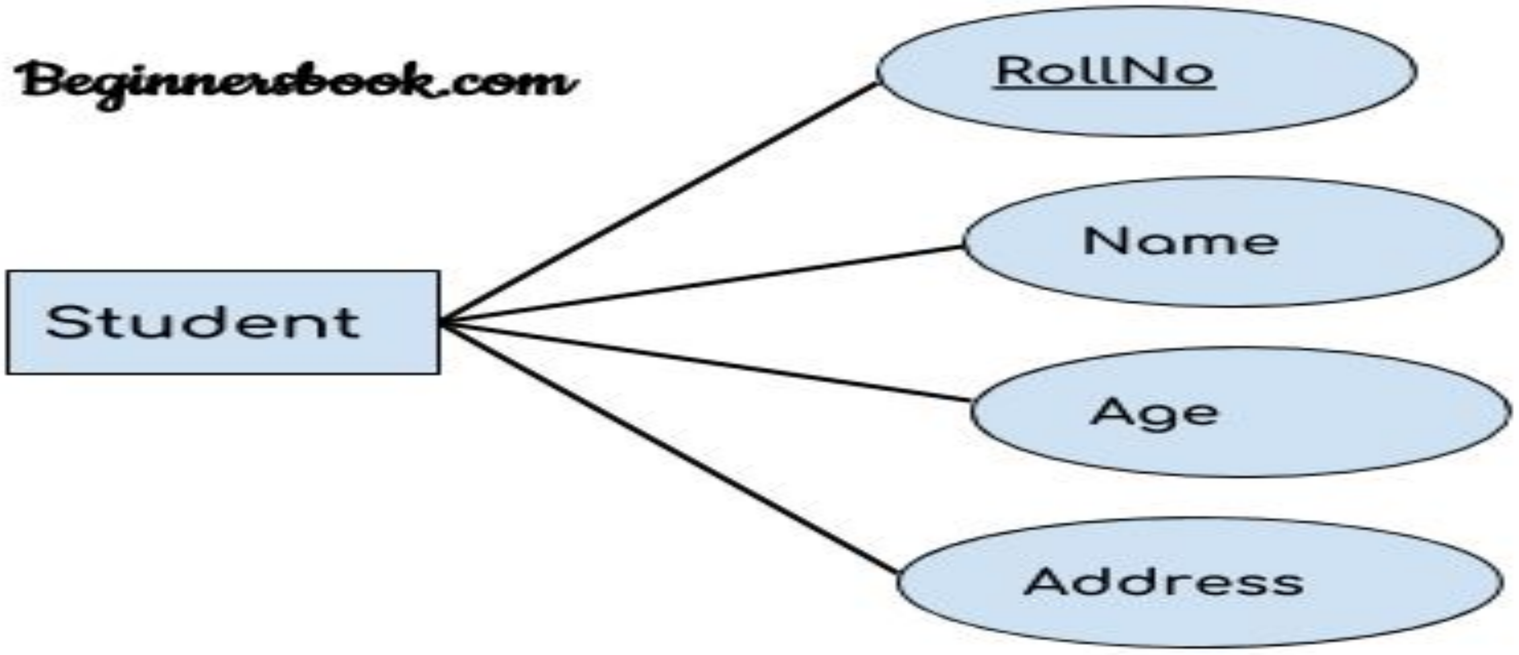
Components of the ER Diagram




Student
Entity Type
E1
E2
E3
Entity Set

1. Key attribute:

A key attribute can uniquely identify an entity from an entity set. For example, student roll number can uniquely identify a student from a set of students. Key attribute is represented by oval same as other attributes however the text of key attribute is underlined.



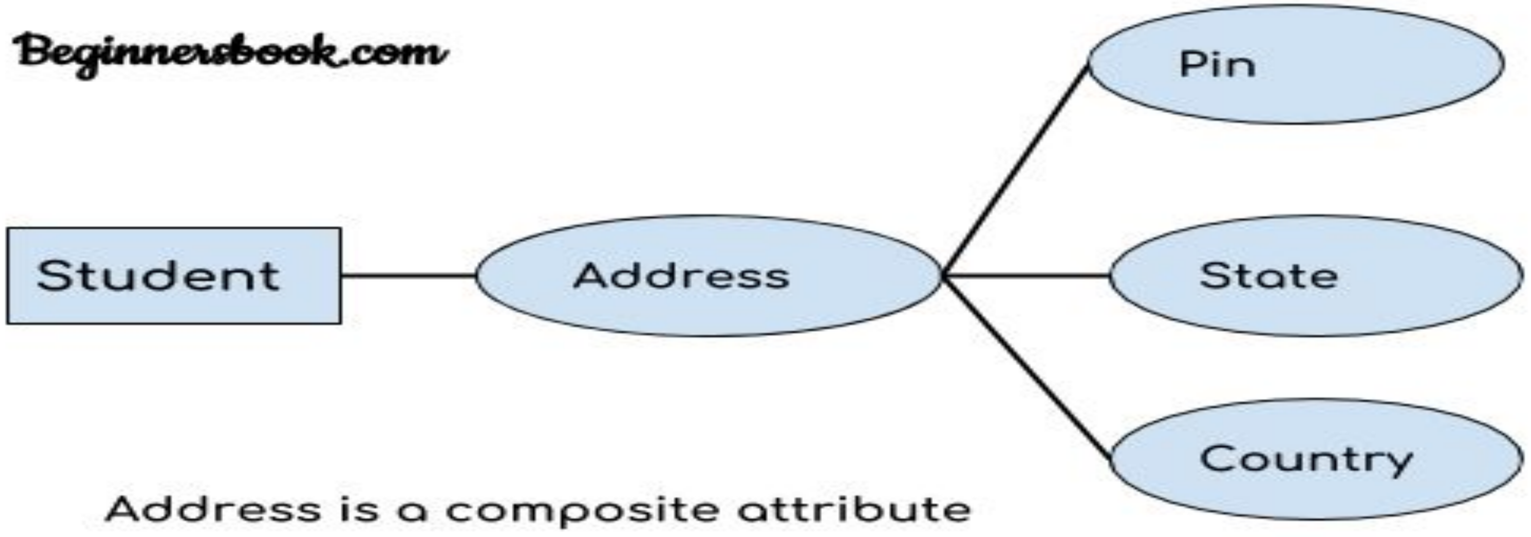
Attribute: An attribute describes the property of an entity. For example, Roll_No, Name, DOB, Age, Address, Mobile_No are the attributes which defines entity type Student.

There are four types of attributes:

1. Key attribute
2. Composite attribute
3. Multivalued attribute
4. Derived attribute

2. Composite attribute:

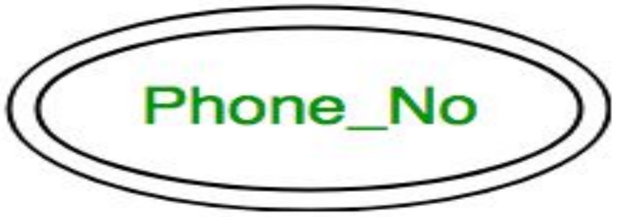
An attribute that is a combination of other attributes is known as composite attribute. For example, In student entity, the student address is a composite attribute as an address is composed of other attributes such as pin code, state, country.



Address is a composite attribute

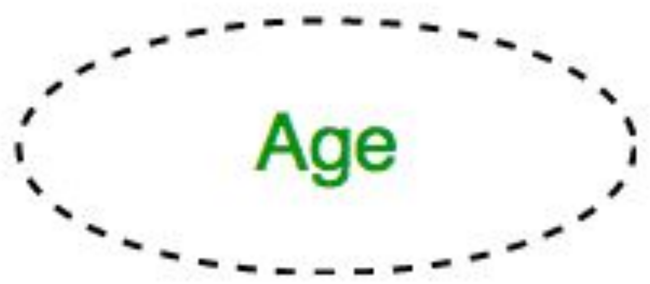
3. Multivalued Attribute:

An attribute consisting more than one value for a given entity. For example, Phone_No (can be more than one for a given student). In ER diagram, multivalued attribute is represented by double oval.



4. Derived Attribute:

An attribute which can be derived from other attributes of the entity type is known as derived attribute. e.g.; Age (can be derived from DOB). In ER diagram, derived attribute is represented by dashed oval.



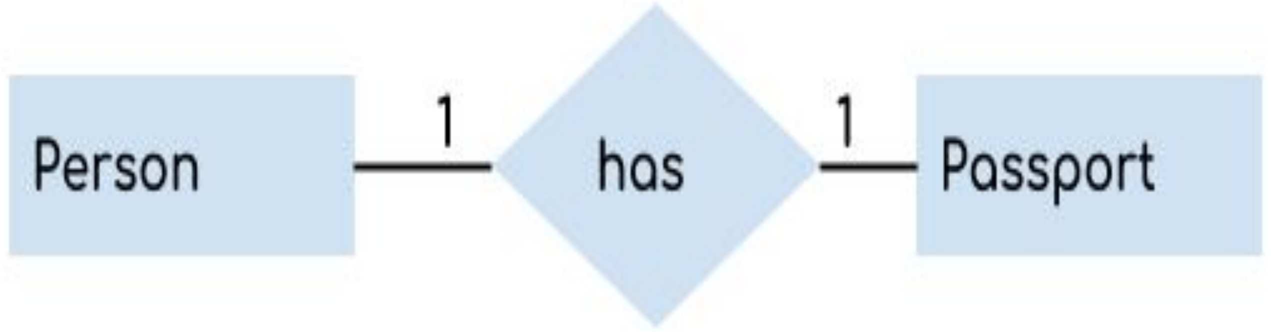
3. Relationship: A relationship is represented by diamond shape in ER diagram, it shows the relationship among entities.

There are four types of relationships:

1. One to One
2. One to Many
3. Many to One
4. Many to Many

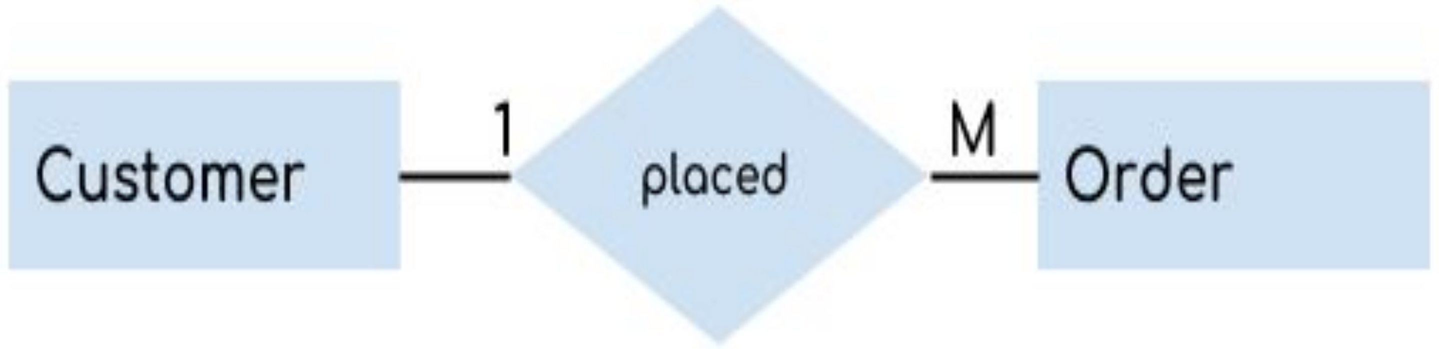
1. One to One Relationship

When a single instance of an entity is associated with a single instance of another entity then it is called one to one relationship. For example, a person has only one passport and a passport is given to one person.



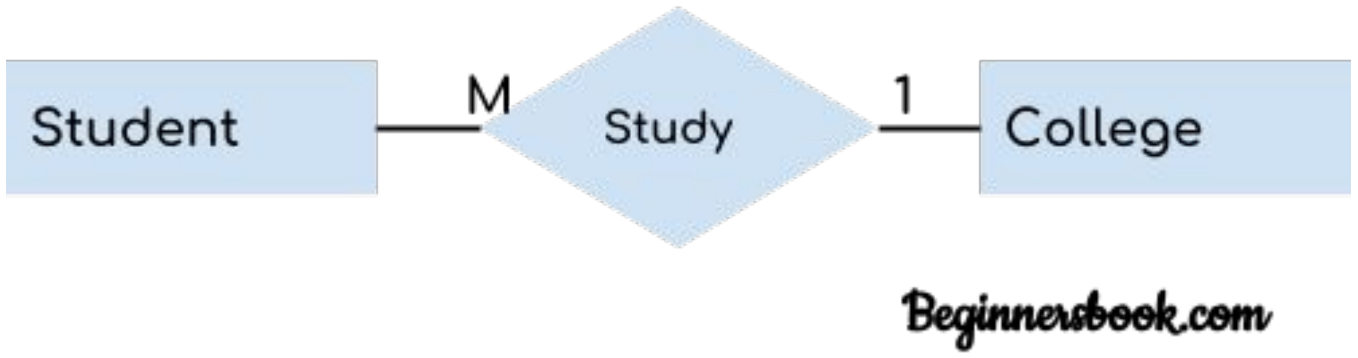
2. One to Many Relationship

When a single instance of an entity is associated with more than one instances of another entity then it is called one to many relationship. For example – a customer can place many orders but a order cannot be placed by many customers.



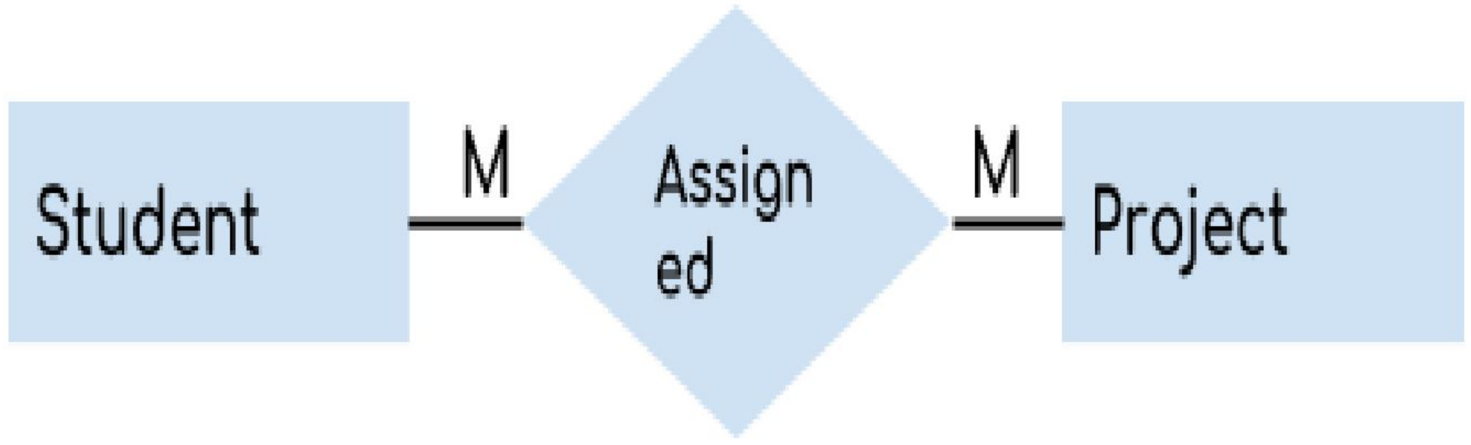
3. Many to One Relationship

- When more than one instances of an entity is associated with a single instance of another entity then it is called many to one relationship. For example – many students can study in a single college but a student cannot study in many colleges at the same time.



4. Many to Many Relationship

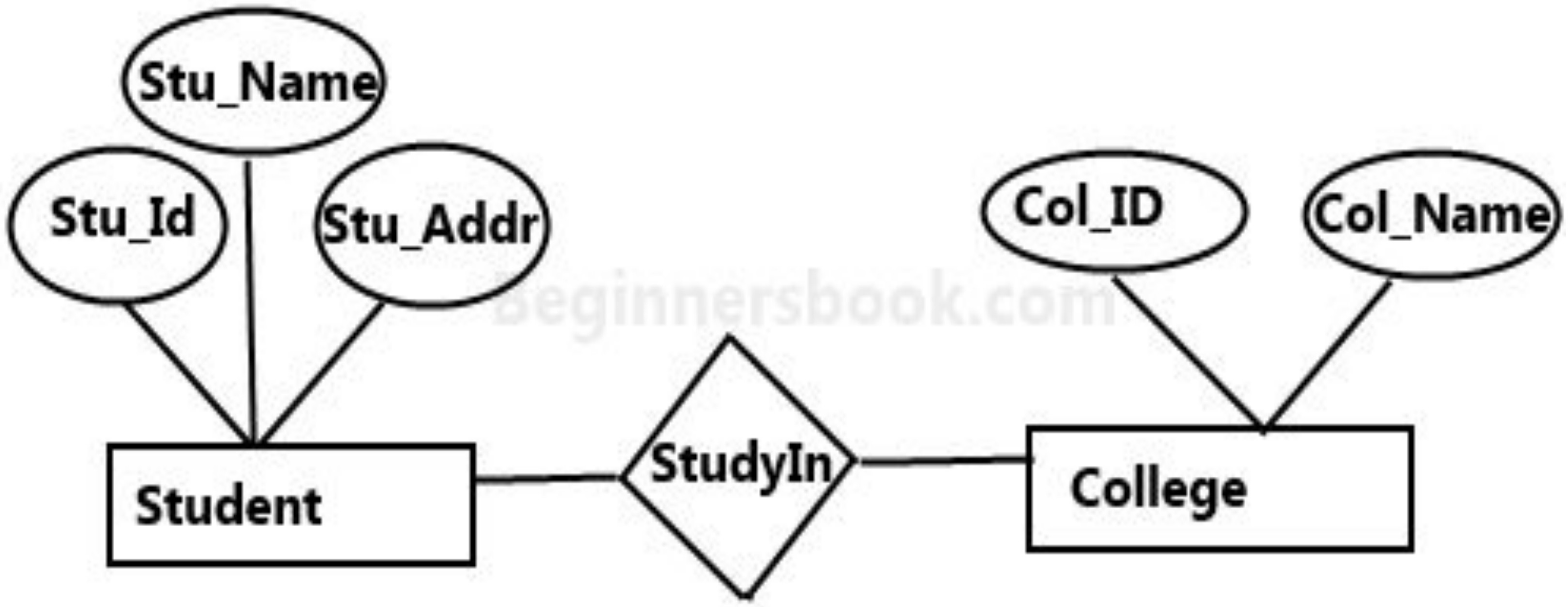
- When more than one instances of an entity is associated with more than one instances of another entity then it is called many to many relationship. For example, a can be assigned to many projects and a project can be assigned to many students.



Beginnersbook.com

A simple ER Diagram:

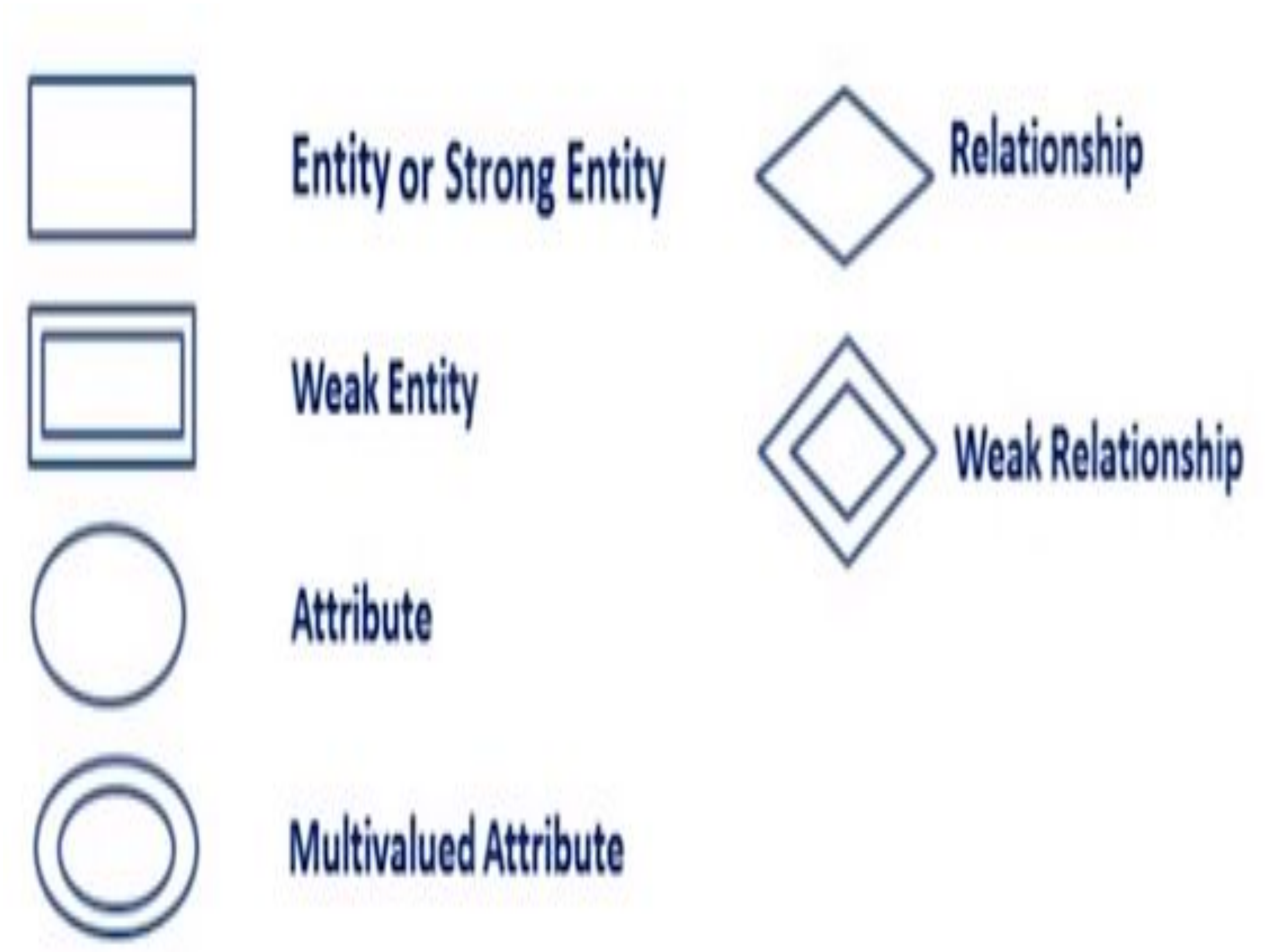
In the following diagram we have two entities Student and College and their relationship. The relationship between Student and College is many to one as a college can have many students however a student cannot study in multiple colleges at the same time. Student entity has attributes such as Stu_Id, Stu_Name & Stu_Addr and College entity has attributes such as Col_ID & Col_Name.



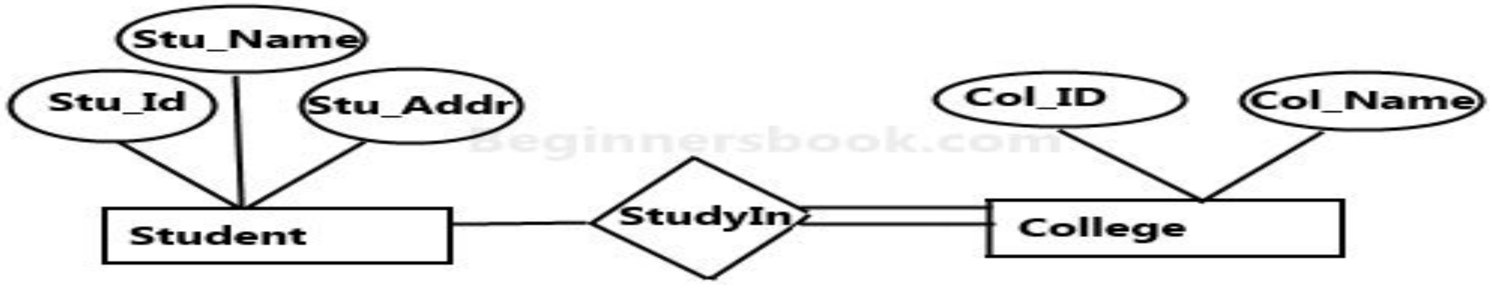
Sample E-R Diagram

The geometric shapes and their meaning in an E-R Diagram.

Rectangle: Represents Entity sets.
Ellipses: Attributes
Diamonds: Relationship Set
Lines: They link attributes to Entity Sets and Entity sets to Relationship Set
Double Ellipses: Multivalued Attributes
Dashed Ellipses: Derived Attributes
Double Rectangles: Weak Entity Sets
Double Lines: Total participation of an entity in a relationship set


Entity or Strong Entity
Relationship
Weak Entity
Weak Relationship
Attribute
Multivalued Attribute

Total Participation of an Entity set

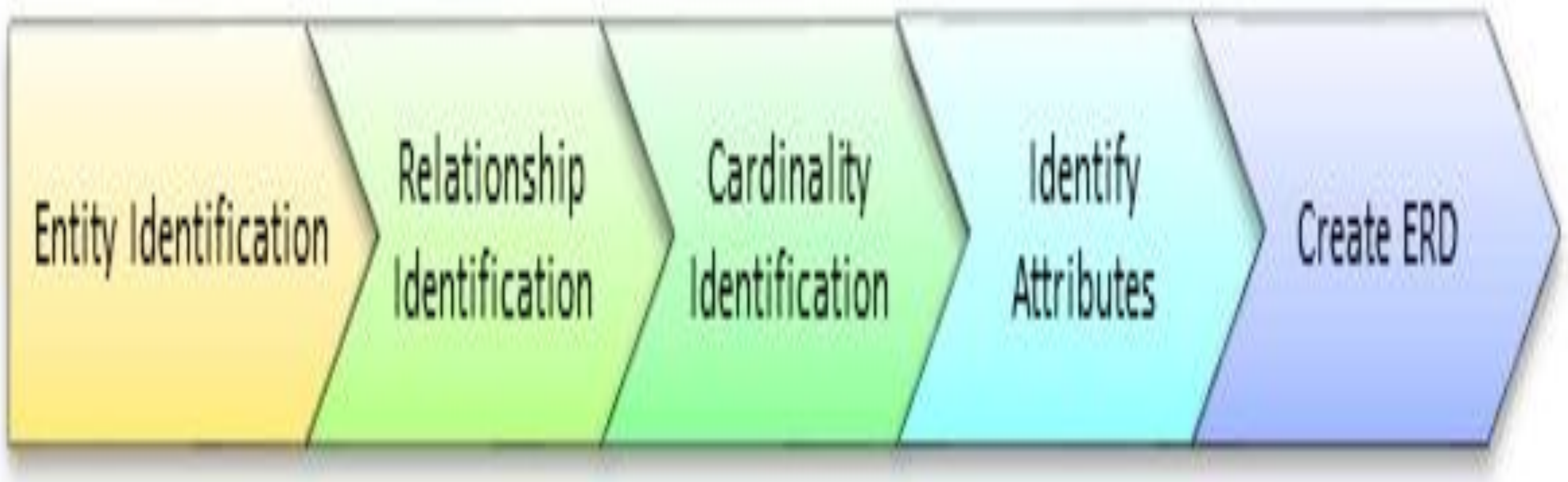


E-R Digram with total participation of College entity set in StudyIn relationship Set - This indicates that each college must have atleast one associated Student.

- A Total participation of an entity set represents that each entity in entity set must have at least one relationship in a relationship set. For example: In the below diagram each college must have at-least one associated Student.

Steps to Create an ERD

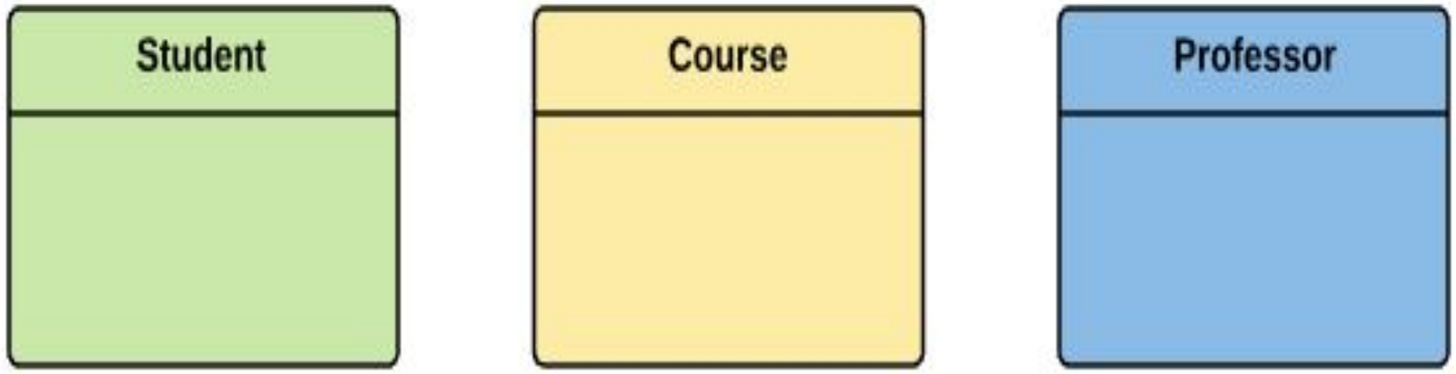
- Following are the steps to create an ERD.

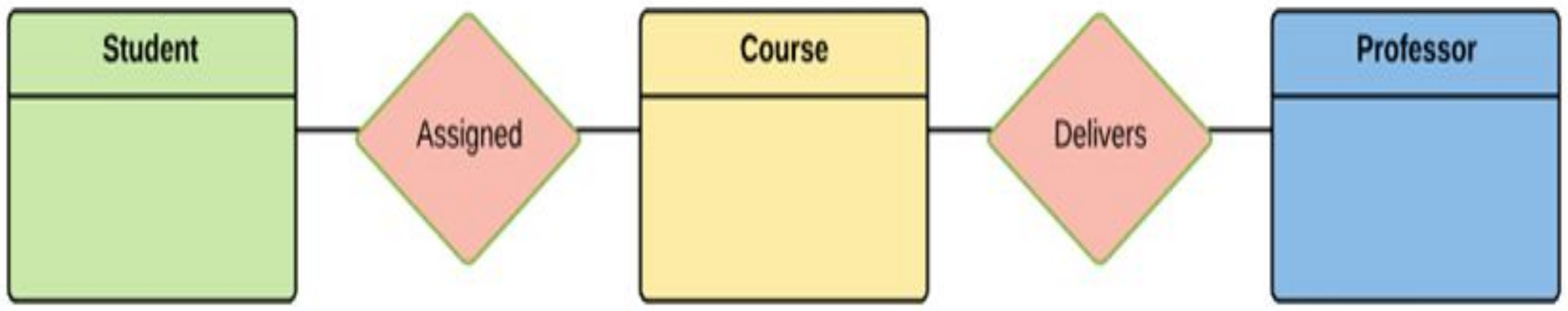


In a university, a Student enrolls in Courses. A student must be assigned to at least one or more Courses. Each course is taught by a single Professor. To maintain instruction quality, a Professor can deliver only one course.

Step 1) Entity Identification

We have three entities
Student
Course
Professor

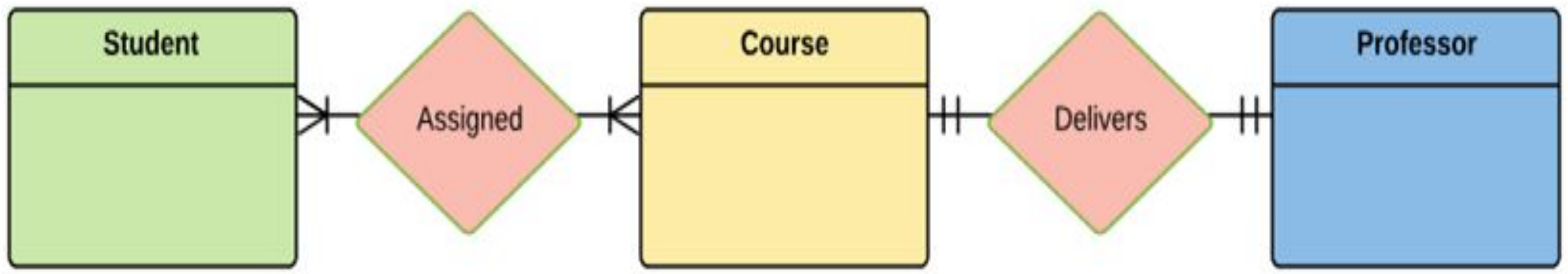




Step 2) Relationship Identification

We have the following two relationships

The student is assigned a course
Professor delivers a course



Step 3) Cardinality Identification

For them problem statement we know that,

A student can be assigned multiple courses
A Professor can deliver only one course

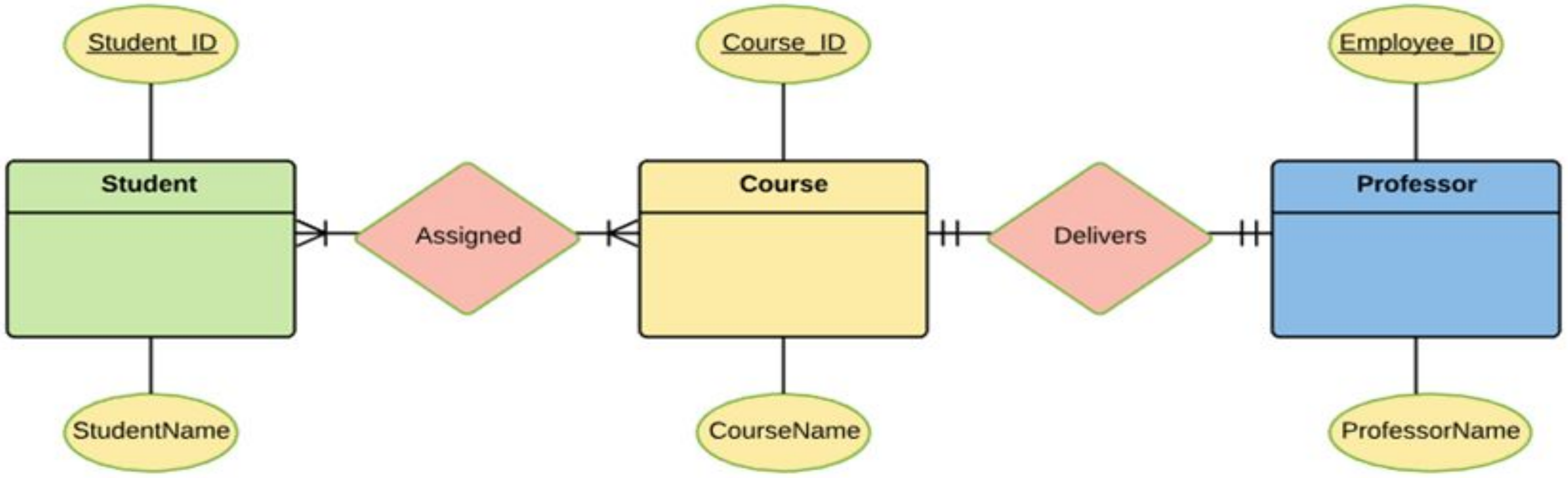
Step 4) Identify Attributes

You need to study the files, forms, reports, data currently maintained by the organization to identify attributes. You can also conduct interviews with various stakeholders to identify entities. Initially, it's important to identify the attributes without mapping them to a particular entity.

Once, you have a list of Attributes, you need to map them to the identified entities. Ensure an attribute is to be paired with exactly one entity. If you think an attribute should belong to more than one entity, use a modifier to make it unique.

Once the mapping is done, identify the primary Keys. If a unique key is not readily available, create one.

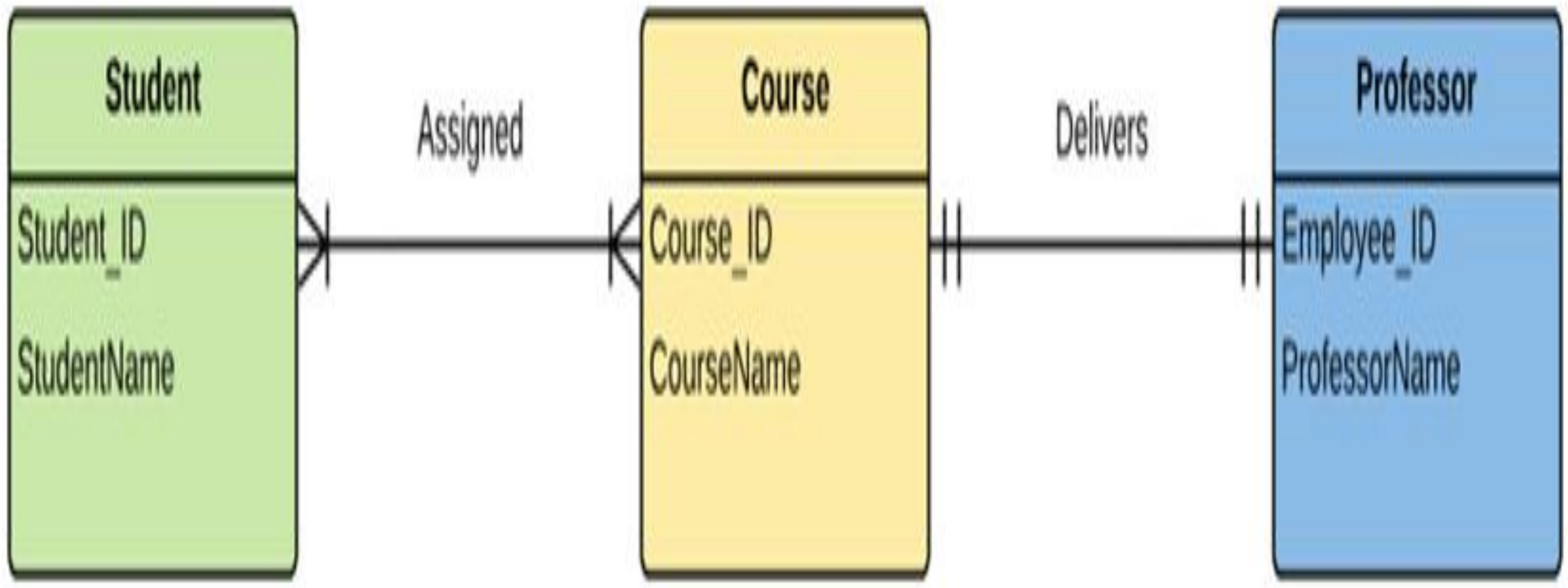
Entity	Primary Key	Attribute
Student	Student_ID	StudentName
Professor	Employee_ID	ProfessorName
Course	Course_ID	CourseName



For Course Entity, attributes could be Duration, Credits, Assignments, etc. For the sake of ease we have considered just one attribute.

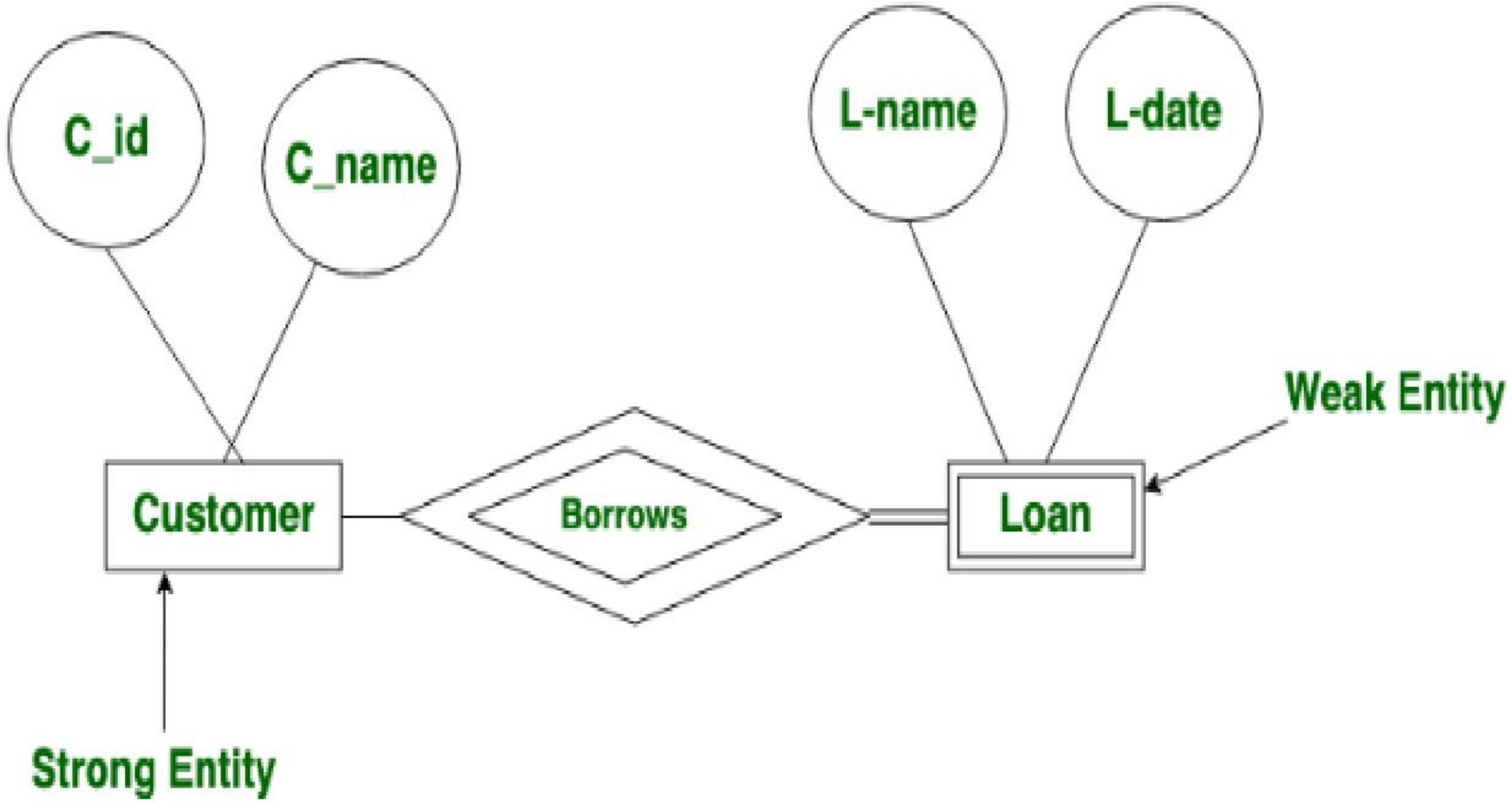
Step 5) Create the ERD

A more modern representation of ERD Diagram



Strong and Weak Entity

- **Strong Entity:**
  A strong entity is not dependent of any other entity in the schema. A strong entity will always have a primary key. Strong entities are represented by a single rectangle. The relationship of two strong entities is represented by a single diamond.
  Various strong entities, when combined together, create a strong entity set.
- **Weak Entity:**
  A weak entity is dependent on a strong entity to ensure the its existence. Unlike a strong entity, a weak entity does not have any primary key. It instead has a partial discriminator key. A weak entity is represented by a double rectangle.
  The relation between one strong and one weak entity is represented by a double diamond.


C_id
C_name
L-name
L-date
Weak Entity
Customer
Borrows
Loan
Strong Entity

Difference between Strong and Weak Entity:

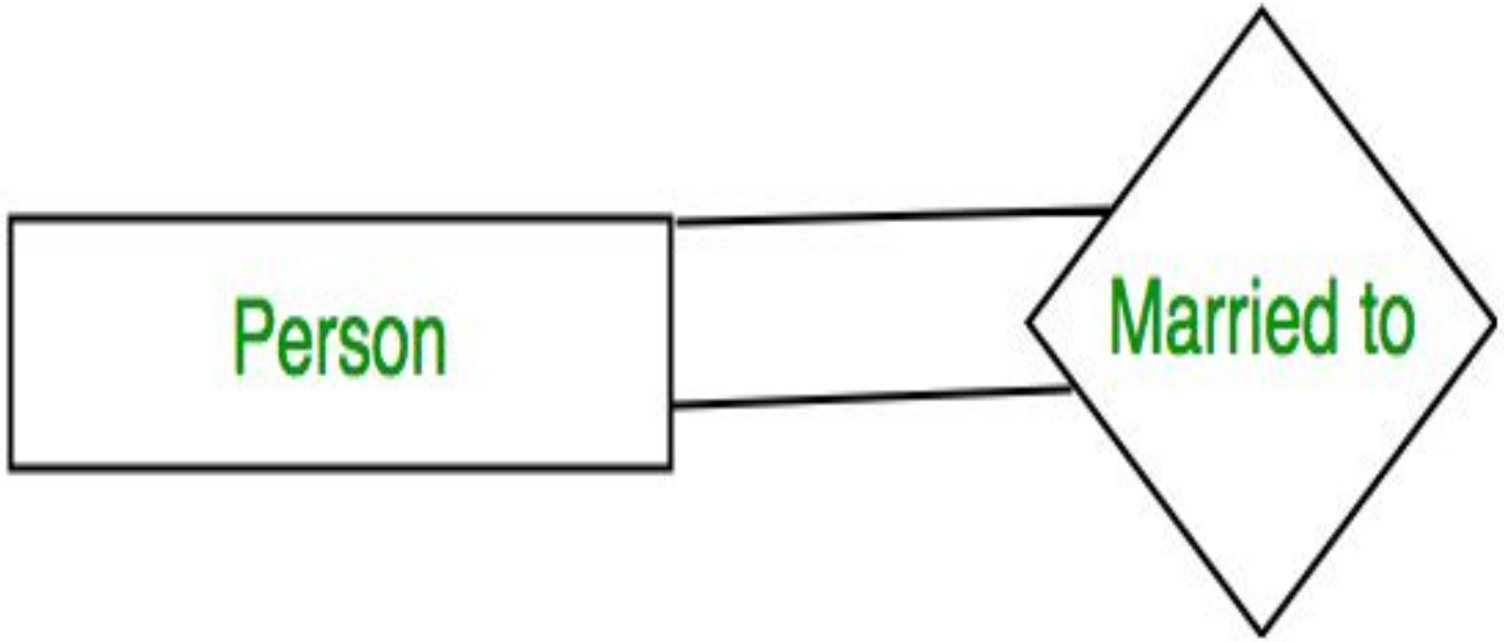
Sr, No	STRONG ENTITY	WEAK ENTITY
1.	Strong entity always has primary key.	While weak entity has partial discriminator key.
2.	Strong entity is not dependent of any other entity.	Weak entity is depend on strong entity.
3.	Strong entity is represented by single rectangle.	Weak entity is represented by double rectangle.
4.	Two strong entity's relationship is represented by single diamond.	While the relation between one strong and one weak entity is represented by double diamond.
5.	Strong entity have either total participation or not.	While weak entity always has total participation.

Best Practices for Developing Effective ER Diagrams

- Eliminate any redundant entities or relationships
- You need to make sure that all your entities and relationships are properly labeled
- There may be various valid approaches to an ER diagram. You need to make sure that the ER diagram supports all the data you need to store
- You should assure that each entity only appears a single time in the ER diagram
- Name every relationship, entity, and attribute are represented on your diagram
- Never connect relationships to each other
- You should use colors to highlight important portions of the ER diagram

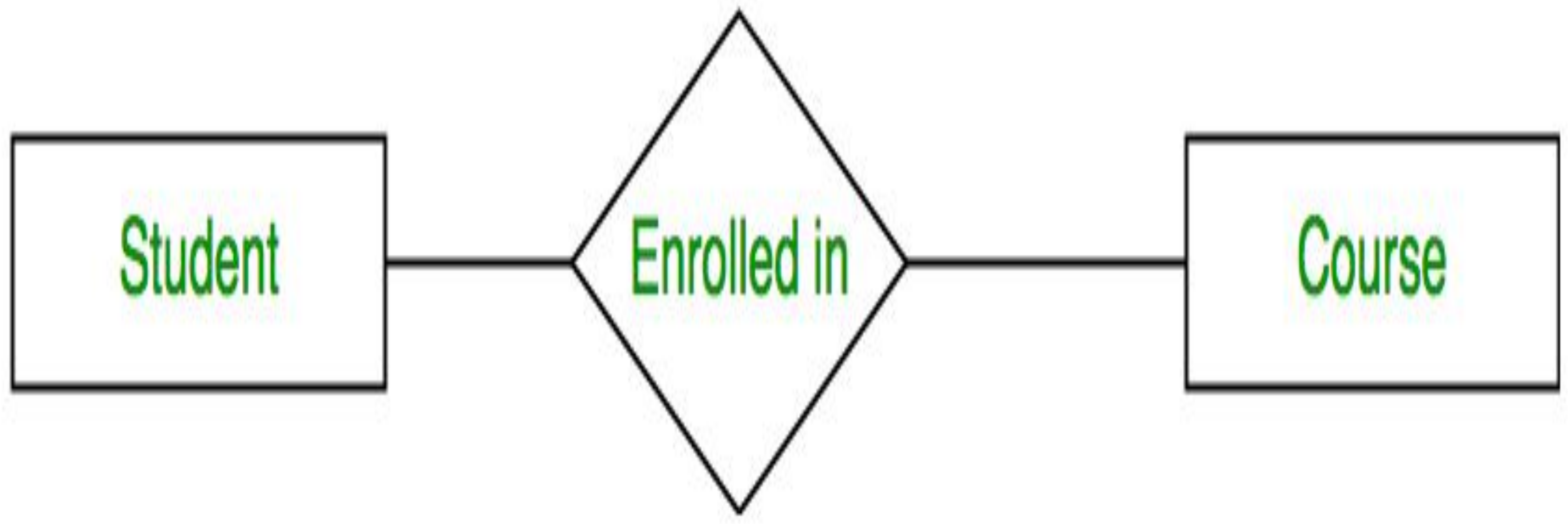
Degree of a relationship set:

- The number of different entity sets **participating in a relationship** set is called as degree of a relationship set.
- **Unary Relationship –**
  When there is **only ONE entity set participating in a relation**, the relationship is called as unary relationship. For example, one person is married to only one person.


Person
Married to

2. Binary Relationship –

When there are TWO entities set participating in a relation, the relationship is called as binary relationship.For example, Student is enrolled in Course.



3. n-ary Relationship –

When there are n entities set participating in a relation, the relationship is called as n-ary relationship.

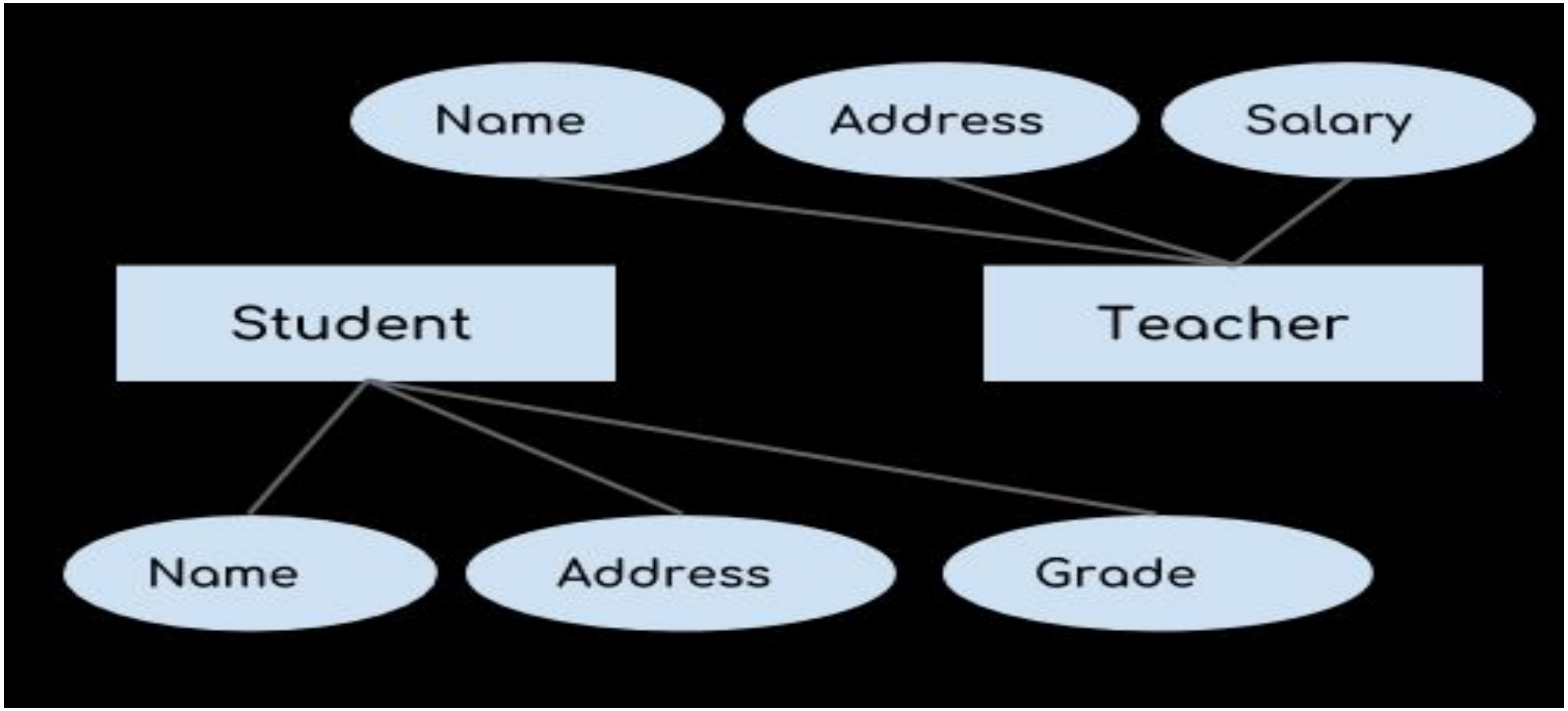
Generalization

- **Generalization** is a process in which the common attributes of more than one entities form a new entity. This newly formed entity is called generalized entity.

Generalization Example

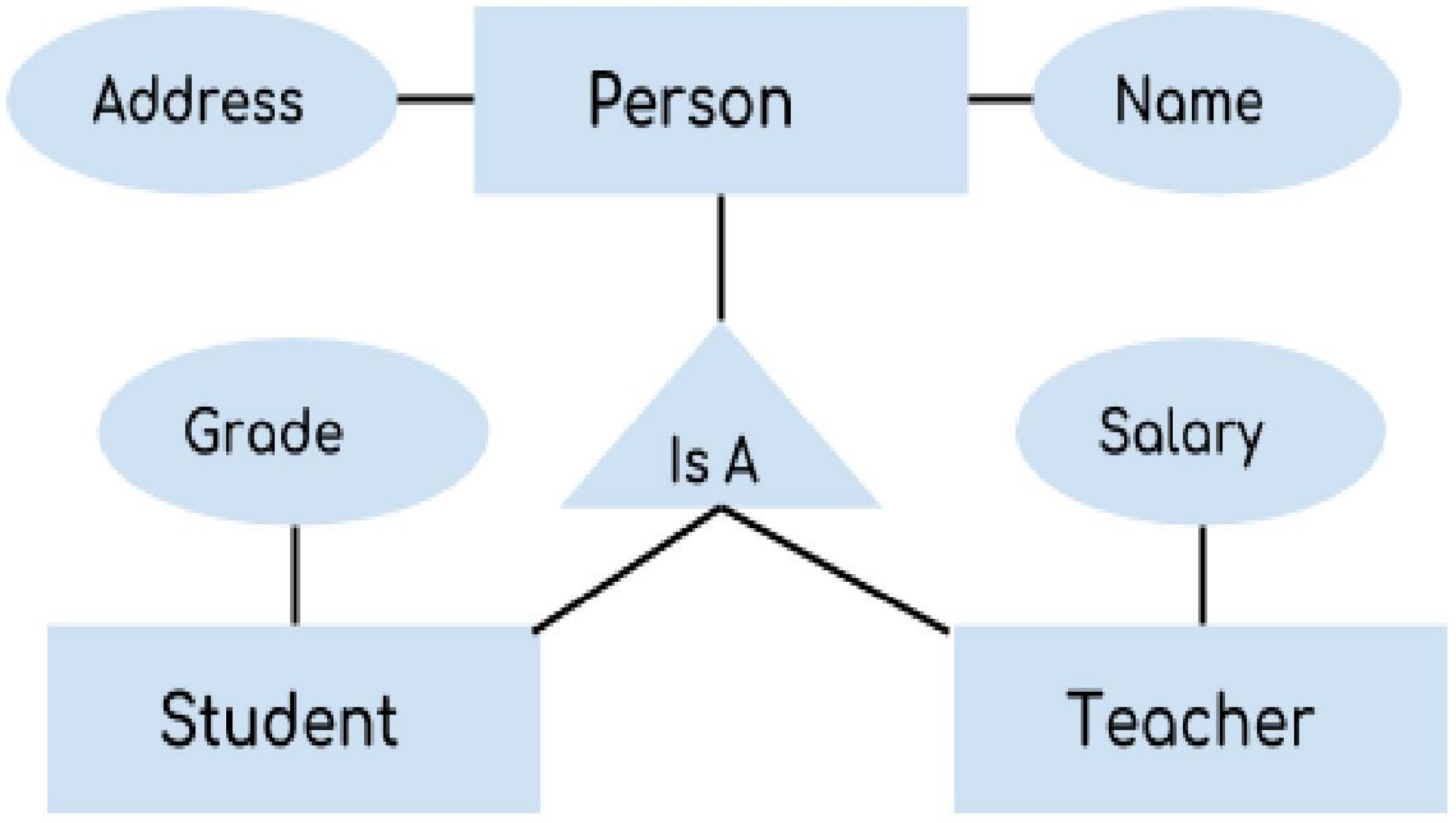
- Lets say we have two entities Student and Teacher.
  Attributes of Entity Student are: Name, Address & Grade
  Attributes of Entity Teacher are: Name, Address & Salary

The ER diagram before generalization looks like this:



The ER diagram after generalization:

We have created a new generalized entity Person and this entity has the common attributes of both the entities. As you can see in the following ER diagram that after the generalization process the entities Student and Teacher only has the specialized attributes Grade and Salary respectively and their common attributes (Name & Address) are now associated with a new entity Person which is in the relationship with both the entities (Student & Teacher).

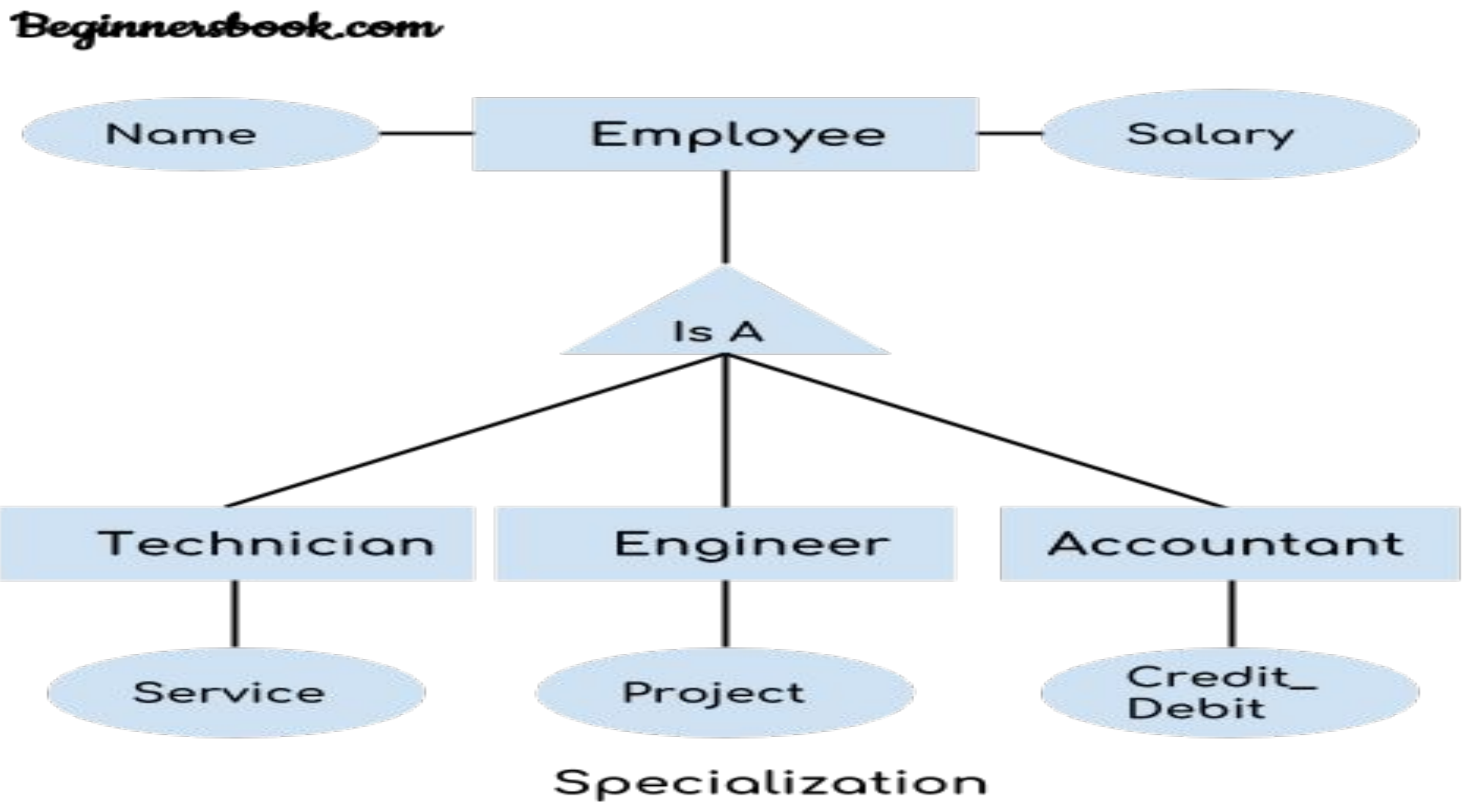


Generalization

Specialization

- **Specialization** is a process in which an entity is divided into sub-entities. You can think of it as a reverse process of **generalization**, in generalization two entities combine together to form a new higher level entity. Specialization is a top-down process.
- The idea behind Specialization is to find the subsets of entities that have few distinguish attributes. For example – Consider an entity employee which can be further classified as sub-entities Technician, Engineer & Accountant because these sub entities have some distinguish attributes.

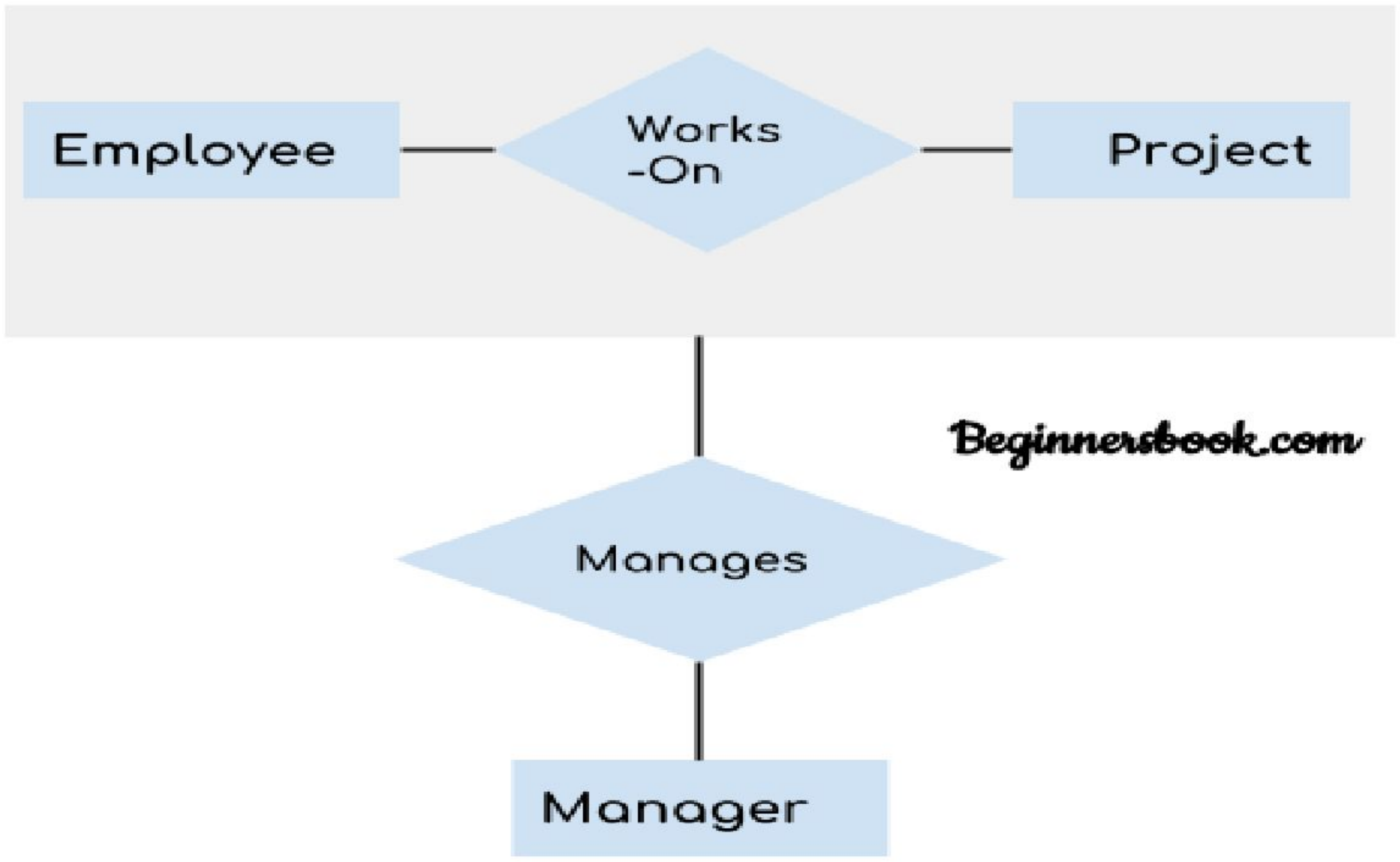
Example



Beginnersbook.com
Name
Employee
Salary
Is A
Technician
Engineer
Accountant
Service
Project
Credit_ Debit
Specialization

Aggregration

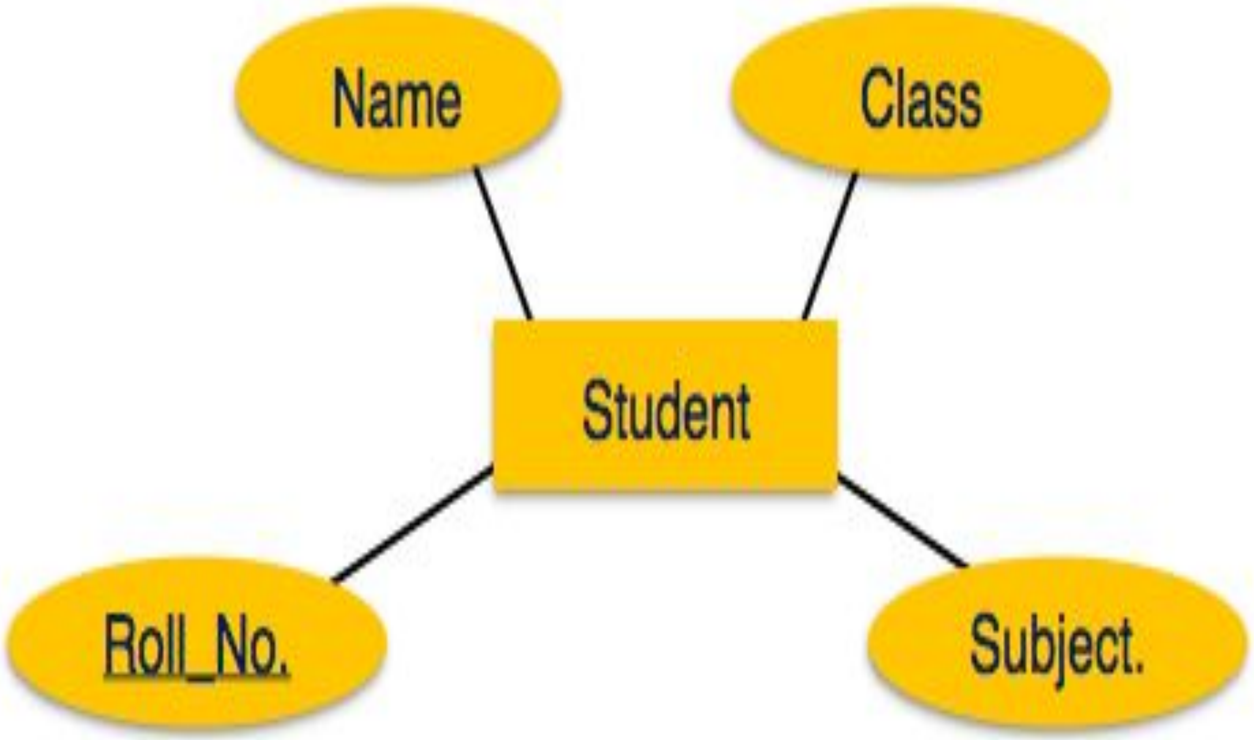
- **Aggregation** is a process in which a single entity alone is not able to make sense in a relationship so the relationship of two entities acts as one entity. I know it sounds confusing but don’t worry the example we will take, will clear all the doubts.


Employee
Works -On
Project
Beginnersbook.com
Manages
Manager

Mapping ER Model into Relational Model

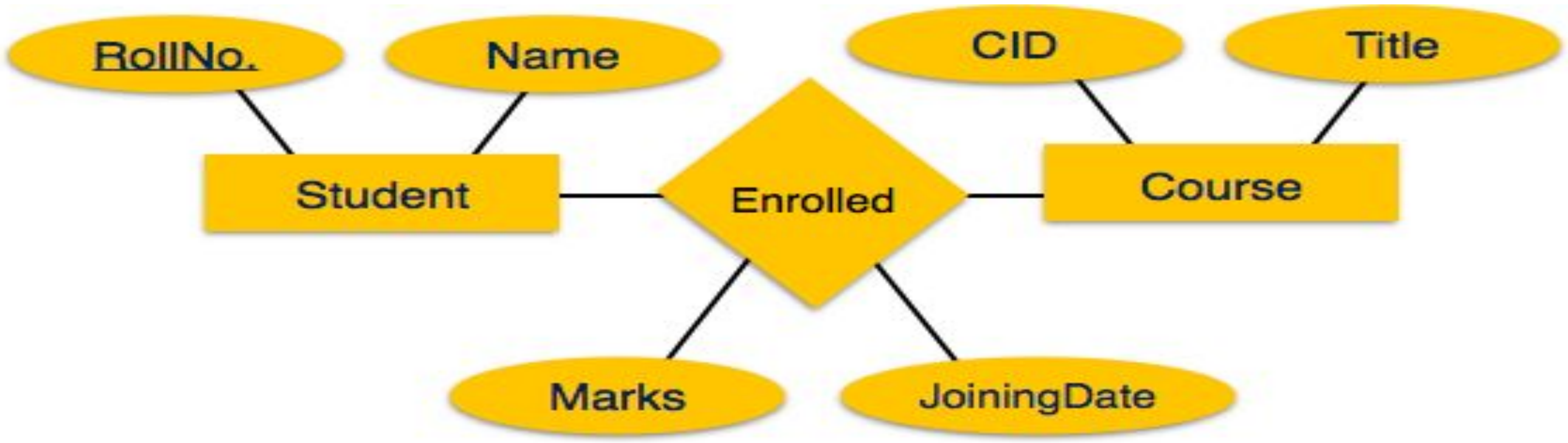
- ER diagrams can be mapped to relational schema, that is, it is possible to create relational schema using ER diagram. We cannot import all the ER constraints into relational model, but an approximate schema can be generated.
- ER diagrams can be mapped to relational schema, that is, it is possible to create relational schema using ER diagram. We cannot import all the ER constraints into relational model, but an approximate schema can be generated.

Mapping Entity
An entity is a real-world object with some attributes.



Mapping Process (Algorithm)
Create table for each entity.
Entity's attributes should become fields of tables with their respective data types.
Declare primary key.

Mapping Relationship
A relationship is an association among entities.



Mapping Process

Create table for weak entity set.

Add all its attributes to table as field.

Add the primary key of identifying entity set.

Declare all foreign key constraints.

Student Relation

Roll_No	Name	Class	Subject
1	RAM	5 Sem	DBMS
2	SHYAM	5 Sem	DBMS

QUESTIONS ON ER diagram

1) Draw ER Diagram of College.
2) Draw ER Diagram of Hospital.
3) Draw ER Diagram of Hotel.

MCQs

- **1. A collection of related data.**
  a) Information
  b) Valuable information
  c) Database
  d) Metadata
- **2. DBMS is software.**
  a) True
  b) False

3. DBMS manages the interaction between __________ and database.

a) Users

b) Clients

c) End Users

d) Stake Holders

4. Which of the following is not involved in DBMS?

a) End Users

b) Data

c) Application Request

d) HTML

5. Database is generally __________

a) System-centered

b) User-centered

c) Company-centered

d) Data-centered

6. A characteristic of an entity.

a) Relation
b) Attribute
c) Parameter
d) Constraint

7. The restrictions placed on the data.

a) Relation
b) Attribute
c) Parameter
d) Constraint

8. IMS stands for?

a) Information Mastering System
b) Instruction Management System
c) Instruction Manipulating System
d) Information Management System

9. A model developed by Hammer and Mc Leod in 1981.

a) SDM

b) OODBM

c) DDM

d) RDM

10. Object = __________+ relationships.

a) data

b) attributes

c) entity

d) constraints

11. A logical schema

a) Is the entire database.

b) describe data in terms of relational tables and columns, object-oriented classes, and XML tags.

c) Describes how data is actually stored on disk.

d) Both (A) and (C)

12. Related fields in a database are grouped to form a

a)Data file

b)Data record

c)Menu

d)Bank

13. The database environment has all of the following components except:

a)Users

b)Separate file

c)Database

d)Database Administrators

14. The way a particular application views the data from the database that the application uses is a

a) Module.

b) Relational model.

c)Schema.

d)Sub schema

15. The property / properties of a database is / are :
a) It is an integrated collection of logically related records.
b) It consolidates separate files into a common pool of data records.
c) Data stored in a database is independent of the application programs using it
d) All of the above

16. A relational database developer refers to a record as

a) criteria.
b) A relation.
c) A tuple.
d) An attribute.

17. The relational model feature is that there

a) Is no need for primary key data
b) Is much more data independence than some other database models
c)Are explicit relationships among records
d)Are tables with many dimensions

18. Conceptual design

a)Is a documentation technique.
b) Needs data volume and processing frequencies to determine the size of the database.
c)Involves modeling independent of the DBMS
d)Is designing the relational model.