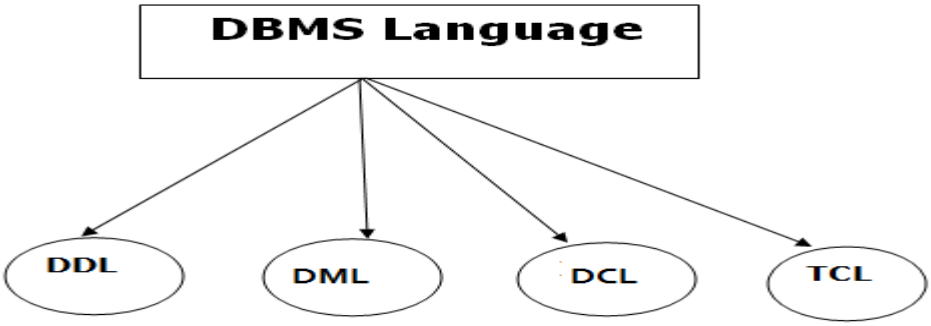
UNIT-1 (remaining part)

Database Language

- 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.

Types of Database Language



1. Data Definition Language

- **DDL** stands for **D**ata **D**efinition **L**anguage. 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.

DDL Commands:

- **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.

2. Data Manipulation Language(DML)

DML stands for **D**ata **M**anipulation **L**anguage. It is used for accessing and manipulating data in a database. It handles user requests.

DML Commands:

- **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.

3. Data Control Language(DCL)

- **DCL** stands for **D**ata **C**ontrol **L**anguage. It is used to retrieve the stored or saved data.
- The DCL execution is transactional. It also has rollback parameters.

DCL Commands:

- **Grant:** It is used to give user access privileges to a database.
- **Revoke:** It is used to take back permissions from the user.

4. Transaction Control Language(TCL)

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

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.

Database Administrator:

One of the main reasons for using DBMS is to have central control of both the data and the programs that access those data. A person who has such central control over the system is called a **database administrator (DBA).**

The Role of a DBA are:

Schema definition: The DBA creates the original database schema by executing a set of data definition statements in the DDL.

Storage structure and access-method definition: The DBA may specify some parameters pertaining to the physical organization of the data and the indices to be created.

Schema and physical-organization modification:
The DBA carries out changes to the schema and physical organization to reflect the changing needs of the organization, or to alter the physical organization to improve performance.

Granting of authorization for data access:
By granting different types of authorization, the database administrator can regulate which parts of the database various users can access. The authorization information is kept in a special system structure that the database system consults whenever a user tries to access the data in
the system.

Routine maintainance:
Examples of the database administrator's routine maintenance activities are:
° Periodically backing up the database onto remote servers, to prevent loss of data in case of disasters such as flooding.

° Ensuring that enough free disk space is available for normal operations, and upgrading disk space as required.
° Monitoring jobs running on the database and ensuring that performance is not degraded by very expensive tasks submitted by some users.

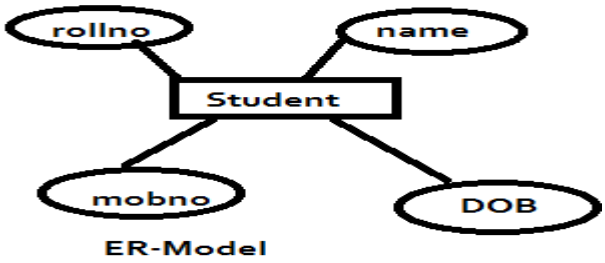
database model

A **database model** is a type of data **model** that determines the logical structure of a **database** and fundamentally determines in which manner data can be stored, organized and manipulated. The most popular example of a **database model** is the relational **model**, which uses a table-based format.

Types of database model:
1.ER Model(Entity relationship model)
2.Relational Model
3.Semi-Structured model
4.Object based model
5.Hierarchical Model
6.Network Model

1.ER Model(Entity relationship model):

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. The entity-relationship model is widely used in database design. For example: student is an entity has rollno, name, mobno, DOB, address etc
are attributes of Student Entity.



ER-Model

2.Relational Model:

The relational model uses a collection of tables to represent both data and the relationships among those data. Each table has multiple columns, and each column has a unique name. Tables are also known as relations. The relational model is an example of a record-based model. Record-based models are so named because the database is structured in fixed-format records of several types. Each table contains records of a particular type. Each record type defines a fixed number of fields, or attributes. The columns of the table are also called attributes or record type. The rows of table are called Tuples The relational data model is the most widely used data model, and a vast majority of current database systems are based on the relational model.

Roll No	Name	age
10	Arun Singh	20
11	Sumit Mishra	19
12	Rahul Kumar	19

Student table

Note: Write All advantages and disadvantages of RDBMS

3.Semi-Structured model:

The semi-structured data model permits the specification of data where individual data items of the same type may have different sets of attributes. This is in contrast to the data models , where every data item of a particular type must have the same set of attributes. *JSON* and *Extensible Markup Language* (*XML*) are widely used semi -structured data representations.

Ex:

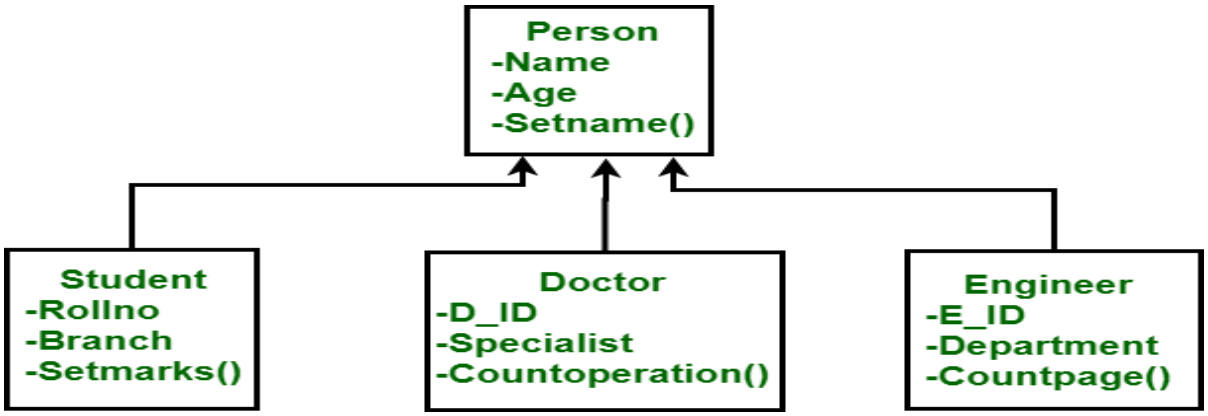
<Student>
<name>Amit</name>
<rollno>10</rollno>
<Student>

4.Object Oriented Data Model :

In Object Oriented Data Model, data and their relationships are contained in a single structure which is referred as object in this data model. In this, real world problems are represented as objects with different attributes. All objects have multiple relationships between them. Basically, it is combination of Object Oriented programming and Relational Database Model as it is clear from the following figure :

Object Oriented Data Model
= Combination of Object Oriented Programming + Relational database model

Components of Object Oriented Data Model :

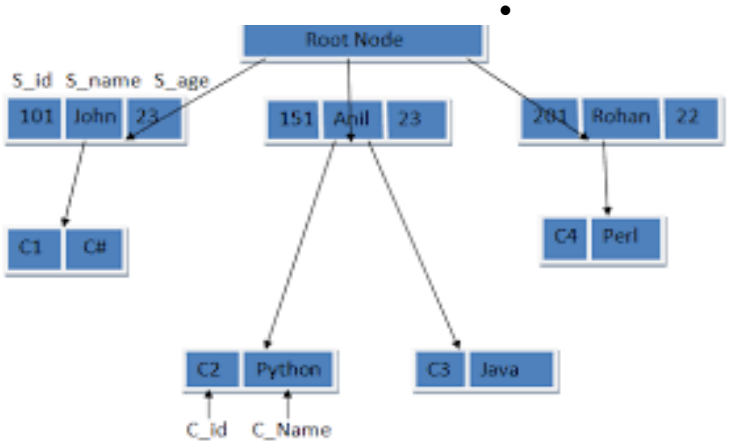


- **Objects –**
  An object is an abstraction of a real world entity or we can say it is an instance of class. Objects encapsulates data and code into a single unit which provide data abstraction by hiding the implementation details from the user. For example: Instances of student, doctor, engineer in above figure.
- **Attribute –**
  An attribute describes the properties of object. For example: Object is STUDENT and its attribute are Roll no, Branch, Semester in the Student class.
- **Methods –**
  Method represents the behavior of an object. Basically, it represents the real-world action. For example: Finding a STUDENT marks in above figure as Setmarks().
- **Class –**
  A class is a collection of similar objects with shared structure i.e. attributes and behavior i.e. methods. An object is an instance of class. For example: Person, Student, Doctor, Engineer in above figure.

- **Inheritance –**
  By using inheritance, new class can inherit the attributes and methods of the old class i.e. base class. For example: as classes Student, Doctor and Engineer are inherited from the base class Person.

5.Hierarchical Model:

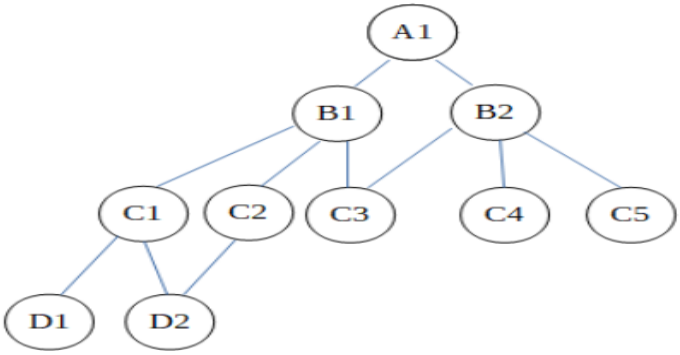
- A hierarchical database model is a data model in which the data are organized into a tree-like structure. Data is represented in form of records. Relationship among the data is represented
- by records of links. A tree may be defined as a set of nodes. At the root of the tree is the
- single parent, the parent can have none, one or more children.
- As it is arranged based on the hierarchy, every record of data tree should have at least one parent, except for the child records. The data can be accessed by following through the classified structure, always initiated from the root or the first parent.



6.Network data Model:

The network database model was created to solve the shortcomings of the hierarchical database model. In this type of model, a child can be linked to multiple parents, a feature that was not supported by the hierarchical data model. The parent nodes are known as owners and the child nodes are called members.

The network data model can be represented as –



Data Modeling Using the Entity Relationship Model (ER- Model)

ER Model stands for Entity Relationship Model is a high-level conceptual data model diagram. ER model helps to systematically analyze data requirements to produce a well-designed database. The ER Model represents real-world entities and the relationships between them. Creating an ER Model in DBMS is considered as a best practice before implementing your database.

ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships.

Entity

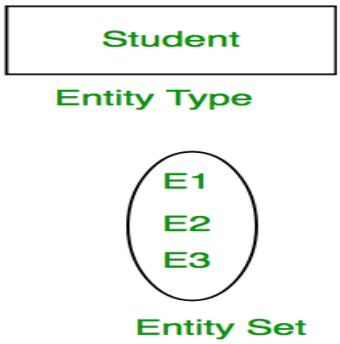
An entity is a real-world thing which can be distinctly identified like a person, place or a concept. It is an object which is distinguishable from others. If we cannot distinguish it from others then it is an object but not an entity. An entity can be of two types:

Tangible Entity: Tangible Entities are those entities which exist in the real world physically. ***Example:*** Person, car, etc.

Intangible Entity: Intangible Entities are those entities which exist only logically and

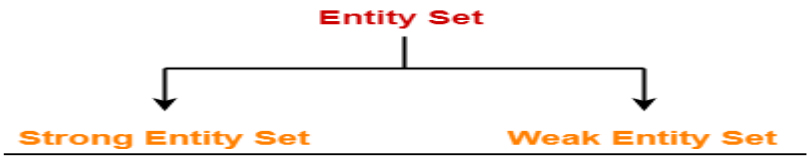
have no physical existence. ***Example:*** Bank Account, etc

Entity Set: An entity set is a set of same type of entities. 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:



Types of Entity Sets-

An entity set may be of the following two types-



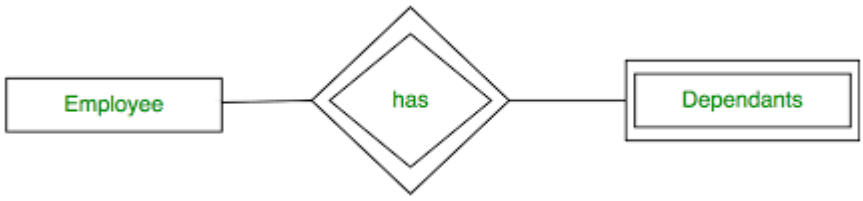
1. Strong Entity Set-

- A strong entity set is an entity set that contains sufficient attributes to uniquely identify all its entities.
- The strong entity has a primary key. Weak entities are dependent on strong entity. Its existence is not dependent on any other entity.
- Strong Entity is represented by a single rectangle –

2.Weak Entity Set:

A weak entity is a type of entity which doesn't have its key attribute. It can be identified uniquely by considering the primary key of another entity. For that, weak entity sets need to have participation.

Example: Dependants is weak Entity which is dependent on Strong entity Employee it is denoted by double rectangle and relationship between strong and weak entity is shown by double diamond.



Difference between Strong Entity and Weak Entity:

Sr. No.	Key	Strong Entity	Weak Entity
1	Key	Strong entity always have one primary key.	Weak entity have a foreign key referencing primary key of strong entity.
2	Dependency	Strong entity is independent of other entities.	Weak entity is dependent on strong entity.
3	Represented by	A strong entity is represented by single rectangle.	A weak entity is represented by double rectangle.
4	Relationship Representation	Relationship between two strong entities is represented by single diamond.	Relationship between a strong and weak entity is represented by double diamond.
5	Participation	Strong entity may or may not participate in entity relationships.	Weak entity always participates in entity relationships.

Attribute(s):
Attributes are the **properties which define the entity type**. For example, Roll_No, Name, DOB, Age, Address, Mobile_No are the attributes which defines entity type Student. In ER diagram, attribute is represented by an oval.

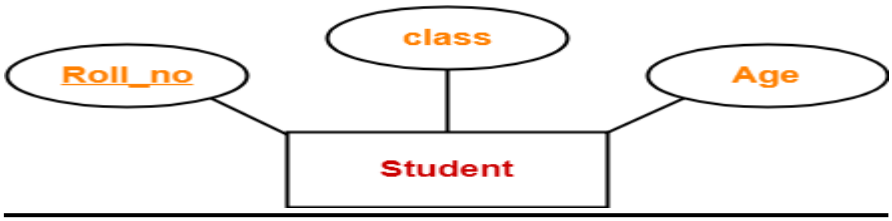
Types of Attributes:

1. Simple attributes
2. Composite attributes
3. Single valued attributes
4. Multi valued attributes
5. Derived attributes
6. Key attributes

1. Simple Attributes-

Simple attributes are those attributes which can not be divided further.

Example-



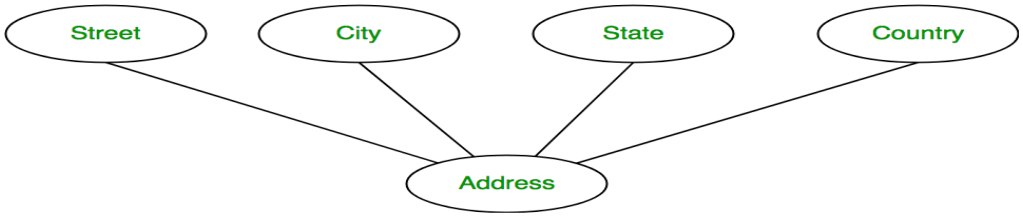
Here, all the attributes are simple attributes as they can not be divided further.

2.Composite Attribute –

Composite attributes are those attributes which are composed of many other simple attributes.

Example

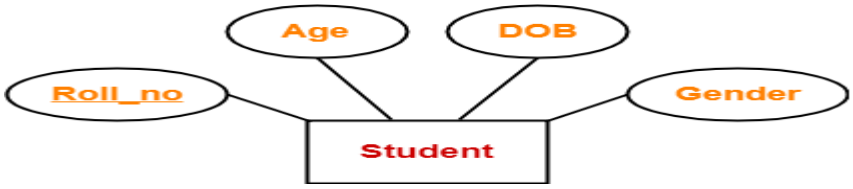
For example, Address attribute of student Entity type consists of Street, City, State, and Country. In ER diagram, composite attribute is represented by an oval comprising of ovals.



3. Single Valued Attributes-

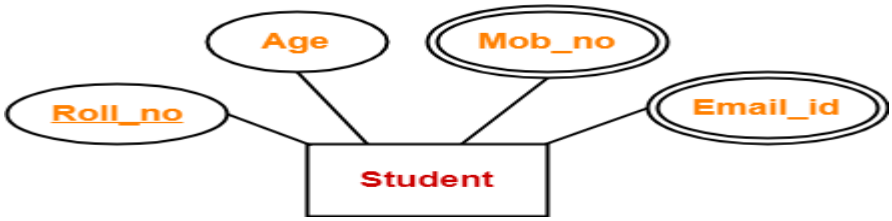
Single valued attributes are those attributes which can take only one value for a given entity from an entity set.

Example-



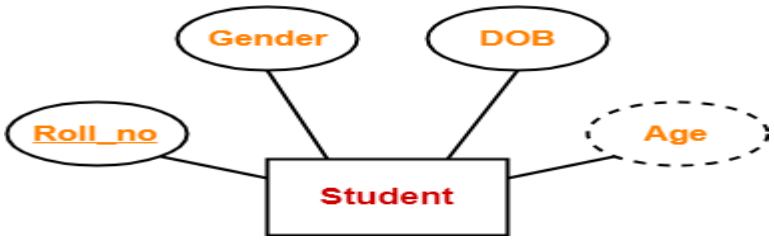
Here, all the attributes are single valued attributes as they can take only one specific value for each entity.

4. Multivalued Attribute –Multi valued attributes are those attributes which can take more than one value for a given entity from an entity set.



5. Derived Attributes-

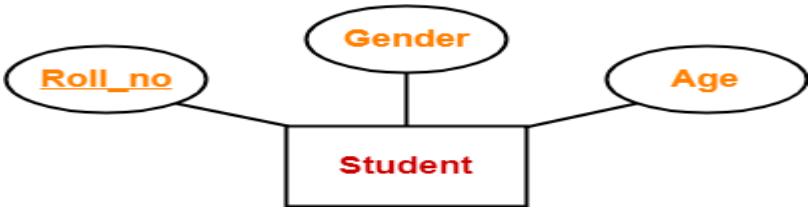
An attribute which can be **derived from other attributes** of the entity type is known as derived attribute. e.g.; Age (can be derived from DOB). In ER diagram, derived attribute is represented by dashed oval.



6. Key Attributes-

Key attributes are those attributes which can identify an entity uniquely in an entity set.

In ER diagram, key attribute is represented by an oval with underlying lines.



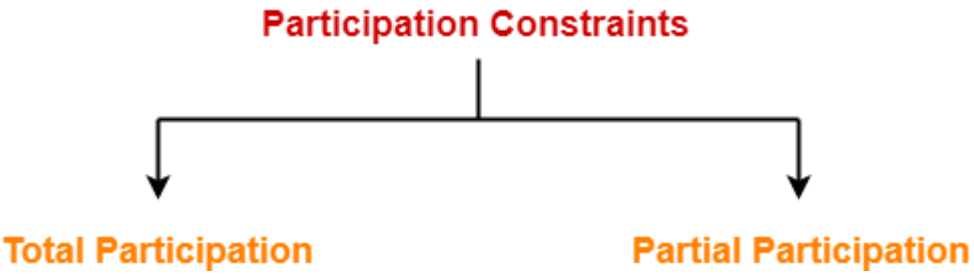
Here, the attribute “Roll_no” is a key attribute as it can identify any student uniquely.

Participation Constraints-

Participation constraints define the least number of relationship instances in which an entity must compulsorily participate.

Types of Participation Constraints-

There are two types of participation constraints-



1. Total participation
2. Partial participation

1. Total Participation-

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

Example-

Here,

- Double line between the entity set “Student” and relationship set “Enrolled in” signifies total participation.
- It specifies that each student must be enrolled in at least one course.

2. Partial Participation-

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

Example-

Here,

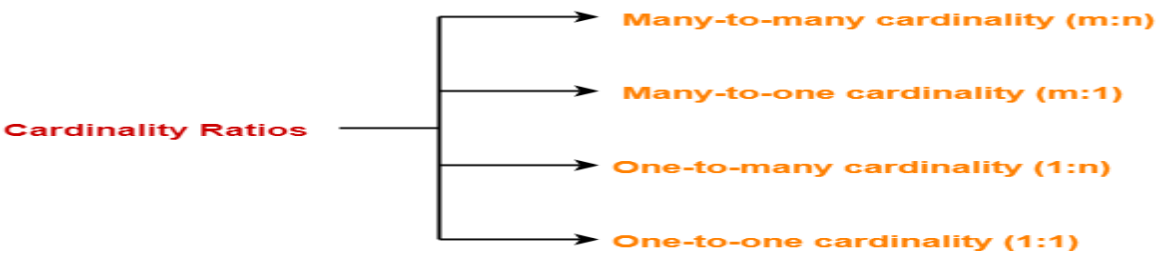
- Single line between the entity set “Course” and relationship set “Enrolled in” signifies partial participation.
- It specifies that there might exist some courses for which no enrollments are made.

Cardinality Constraint or Mapping Cardinalities:

Cardinality constraint defines the maximum number of relationship instances in which an entity can participate.

Types of Cardinality Ratios-

There are 4 types of cardinality ratios-



1. Many-to-Many cardinality (m:n)

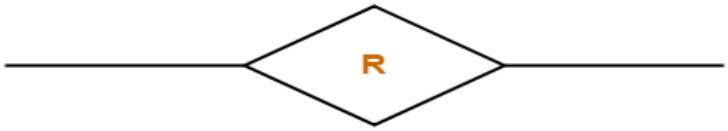
2. Many-to-One cardinality (m:1)
3. One-to-Many cardinality (1:n)
4. One-to-One cardinality (1:1)
1. to-One cardinality (1:1)

1. Many-to-Many Cardinality-

By this cardinality constraint,

- An entity in set A can be associated with any number (zero or more) of entities in set B.
- An entity in set B can be associated with any number (zero or more) of entities in set A.

Symbol Used-



Cardinality Ratio = m : n

Example-

Consider the following ER diagram-

Many to Many Relationship

Here,

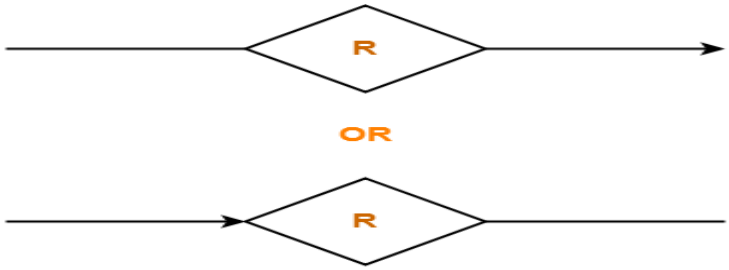
- One student can enroll in any number (zero or more) of courses.
- One course can be enrolled by any number (zero or more) of students.

2. Many-to-One Cardinality-

By this cardinality constraint,

- An entity in set A can be associated with at most one entity in set B.
- An entity in set B can be associated with any number (zero or more) of entities in set A.

Symbol Used-



Cardinality Ratio = m : 1

Example-

Consider the following ER diagram-

Many to One Relationship

Here,

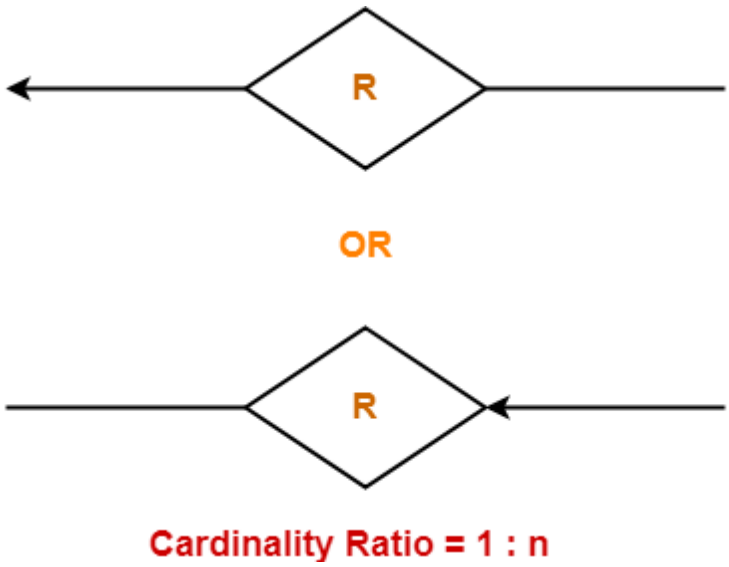
- One student can enroll in at most one course.
- One course can be enrolled by any number (zero or more) of students.

3. One-to-Many Cardinality-

By this cardinality constraint,

- An entity in set A can be associated with any number (zero or more) of entities in set B.
- An entity in set B can be associated with at most one entity in set A.

Symbol Used-



Cardinality Ratio = 1 : n

Example-

Consider the following ER diagram-

One to Many Relationship

Here,

- One student can enroll in any number (zero or more) of courses.
- One course can be enrolled by at most one student.

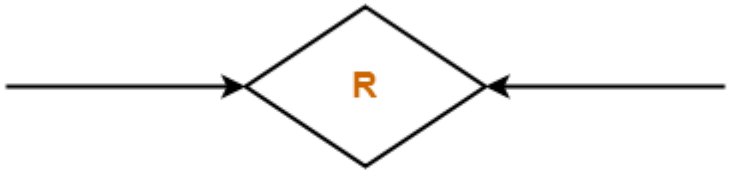
4. One-to-One Cardinality-

By this cardinality constraint,

- An entity in set A can be associated with at most one entity in set B.
- An entity in set B can be associated with at most one entity in set A.

Symbol Used-

OR



Cardinality Ratio = 1 : 1

Example-

Consider the following ER diagram-

One to One Relationship

Here,

- One student can enroll in at most one course.
- One course can be enrolled by at most one student.

The Enhanced ER Model

As the complexity of data increased in the late 1980s, it became more and more difficult to use the traditional ER Model for database modelling. Hence some improvements or enhancements were made to the existing ER Model to make it able to handle the complex applications better.

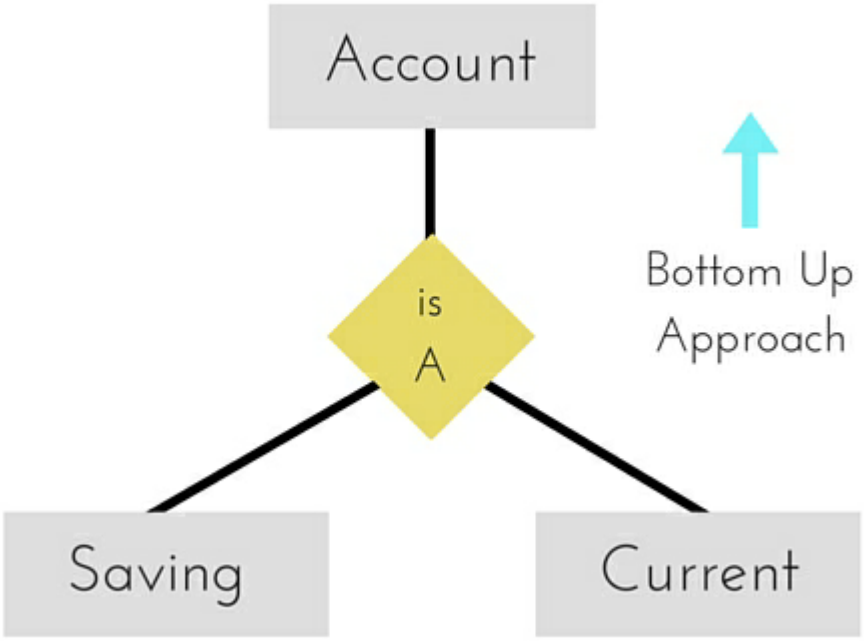
Hence, as part of the **Enhanced ER Model**, along with other improvements, three new concepts were added to the existing ER Model, they were:

1. Generalization
2. Specialization
3. Aggregration

Generalization

Generalization is a bottom-up approach in which two lower level entities combine to form a higher level entity. In generalization, the higher level entity can also combine with other lower level entities to make further higher level entity.

It's more like Superclass and Subclass system, but the only difference is the approach, which is bottom-up. Hence, entities are combined to form a more generalised entity, in other words, sub-classes are combined to form a super-class.



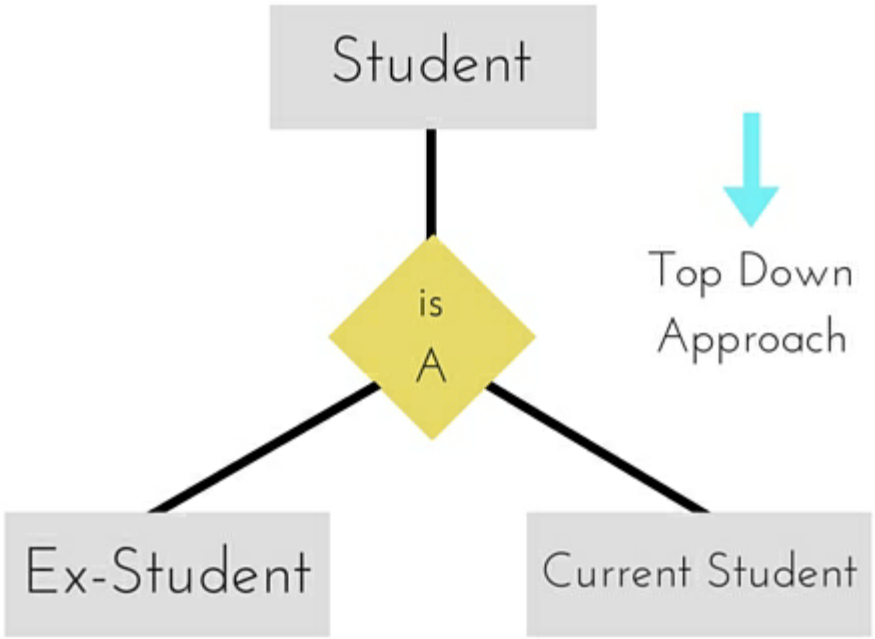
For example, **Saving** and **Current** account types entities can be generalised and an entity with name **Account** can be created, which covers both.

Specialization

Specialization is a top-down approach in which a higher-level entity is divided into multiple *specialized* lower-level entities. In addition to sharing the attributes of the higher-level entity, these lower-level entities have *specific* attributes of their own. Specialization is usually used to find subsets of an entity that has a few different or additional attributes.

Specialization is opposite to Generalization. It is a top-down approach in which one higher level entity can be broken down into two lower level entity. In specialization, a higher level entity may not have any lower-level entity sets, it's possible.

The following enhanced entity relationship diagram expresses the entities in a hierarchical database to demonstrate specialization:



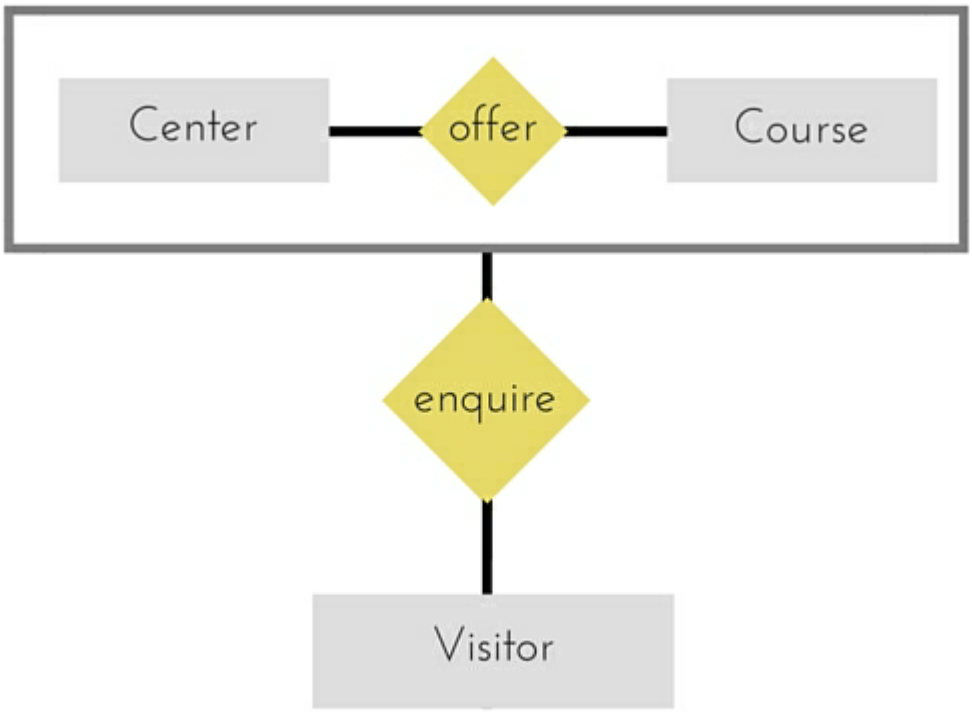
For example: Student can be ex-student and current student. So student entity is further broken in two parts. It is also IS A relationship and works like inheritance.

Aggregration

Aggregation **refers to the process by which entities are combined to form a single meaningful entity. The specific entities are combined because they do not make sense on their own. To establish a single entity,** aggregation **creates a relationship**

that combines these entities.

For example: Center entity offers the Course entity act as a single entity in the relationship which is in a relationship with another entity visitor. In the real world, if a visitor visits a coaching center then he will never enquiry about the Course only or just about the Center instead he will ask the enquiry about both.



ER-DIAGRAM (Insurance Company)

Question: Design a database for insurance company.Asume that following are the requirements that were collected:

An insurance company has different policies.Policies have pno,term_price and coverage.

Policies are categorized based on their types.There are two types:Auto_policy and Home_policy.

Policies for vehicles come under Auto policy.Auto_policy has pno,vehicle type and issue date.

Policies for house come under home policy.Home_policy has pno,issue date and term_price.

Customers take policies policy through policy agent.A customer can take only one policy .

ANSWER:

ENTITIES:

1)Policy

2)Auto_policy

3)Home_policy

4)Vehicle

5)House

6)Customer

KEY ATTRIBUTES:

1)Policy: pno

2)Auto_policy: polno

3)Home_policy:policyno

4)House:hno

5)Vehicle: vehicle_no

6)Customer: custid

OTHER ATTRIBUTES:

1)Policy:price,coverage.

2)Auto_policy:type,issue_date

3)Home_policy:date,term_price

4)House:name

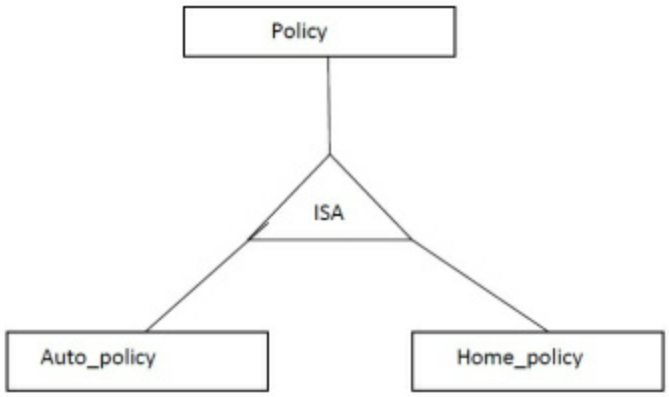
5)Vehicle:model

6)Customer:address,phno

RELATIONSHIPS:

RELATIONSHIPS

1)Policies are categorized into two types:Auto_policy and Home_policy.

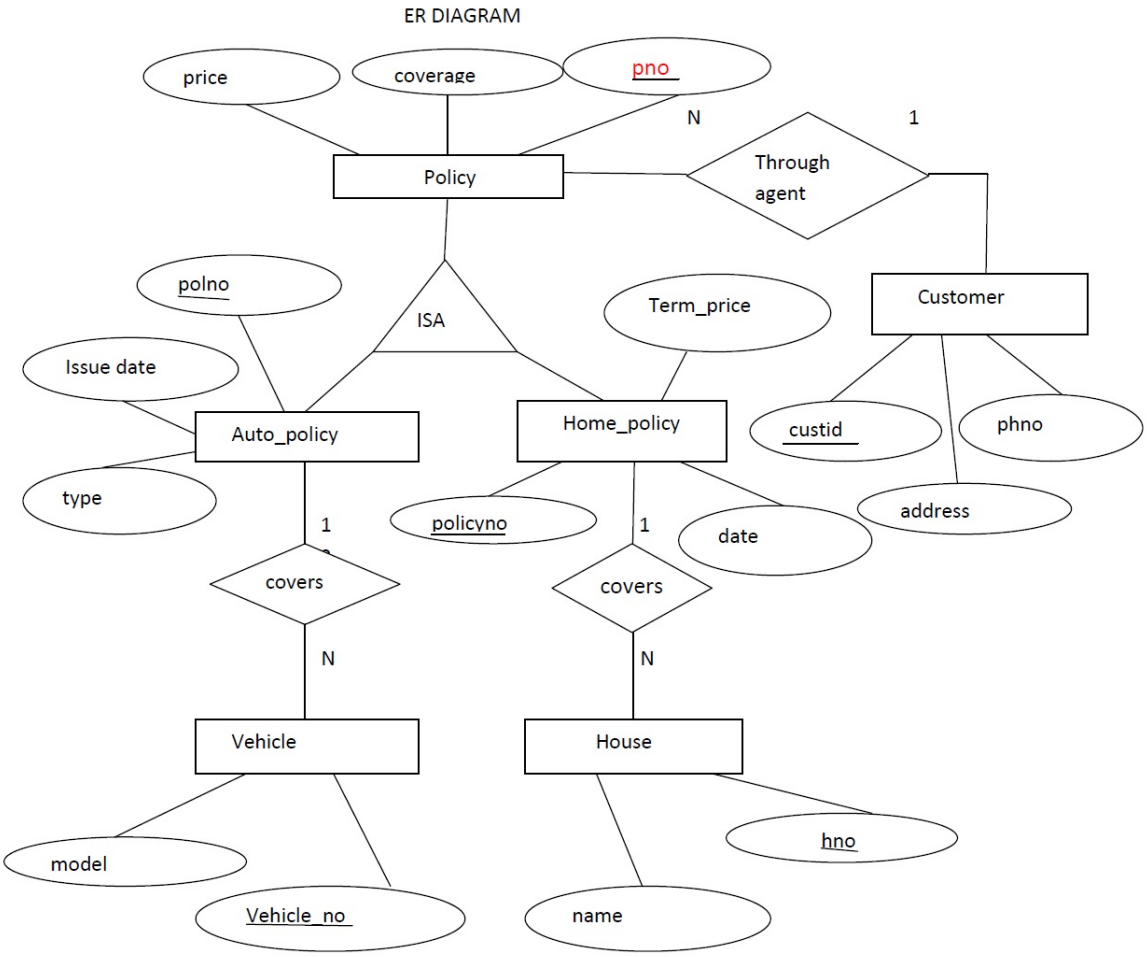


2)Auto_policy covers polices for vehicles.There will be only one policy for a vehicle.

3)Home_policy covers polices for houses.A house can take only one policy.

4)Customer takes policy through policy agent.A customer can take only one policy.

ER-Diagram:



Keys in DBMS:

Key: KEY is an attribute or set of attributes which helps you to identify a row(tuple) in a relation(table). They allow you to find the relation between two tables. Keys help you uniquely identify a row in a table by a combination of one or more columns in that table. Key is also helpful for finding unique record or row from the table. Database key is also helpful for finding unique record or row from the table.

Types of Key

1. **Super Key**
2. **Candidate key**
3. **Primary key**
4. **Alternate key**
5. **Foreign Key**
6. **Composite or Compound key**
7. **Unique Key**
8. **Surrogate key**

1. Super key:

A super key is a combination of all possible attributes that can uniquely identify the row or tuple in the given relation or table.

=>super key is a superset of a candidate key.

=>A table can have many super keys

=>A Super key may have additional attribute that are not needed for unique identity.

Employee table

Emp_id	Name	Aadhar No	Email_id	Dept_id
01	Arun	793039030	a@gmai.	1
02	Sumit	09300303	s@gmail.c	2
03	Sumit	82992092	su@gmai.	2
04	Vimal	90202020	v@gmail.c	3

Super keys:
1.{Emp_id}
2.{Aadhar No}
3.{Email_ID}
4.{Emp_id,Aadhar No}
5.{Aadhar No,Email_id}
6.{Email_id,Emp_id}
7.{Emp_id,name}
8.{Emp_id,name,Dept_id}
9.{Emp_id,name,Aadhar no,Email_id,Dept_set etc.....}

2. Candidate Key:

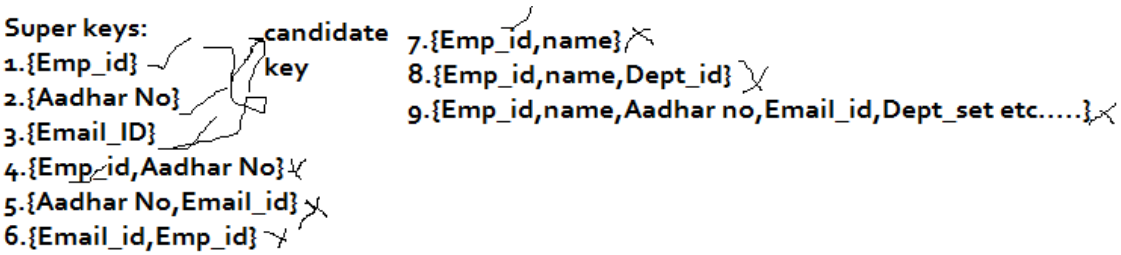
A Candidate key is an attribute or set of an attribute which can uniquely identify a tuple or row.

=>A Candidate key is a minimal super key or a super key with no redundant attributes.

=>Candidate keys are defined as distinct set of attributes from which primary key can be selected.

=> Candidate keys are not allowed to have a Null values.

Example : In the above table emp_id, Aadhar No, and Email_id are candidate **keys.**



3.Primary Key:

A Primary key is one of the candidate key chosen by the database designer to uniquely identify the tuple or row in the relation or table.

=>The value of primary key can never be NULL.

=>The value of primary key must always be unique.

=>The values of primary key can never be changed i.e updation is not possible.

=>A table is allowed to have only one primary key.

Example: In the above Employee table Emp_id is best suitable for primary key.

4.Alternate key:

Out of all candidate keys only one gets selected as primary key, remaining keys are known as alternate keys.

Example: In the above Employee table :

Emp_id is best suited for the primary key.

Rest of the candidate keys like Aadhar no, Email_id are alternate keys.

5.Foreign Key:

An attribute in one table that refers to the Primary key in another table is known as Foreign key.

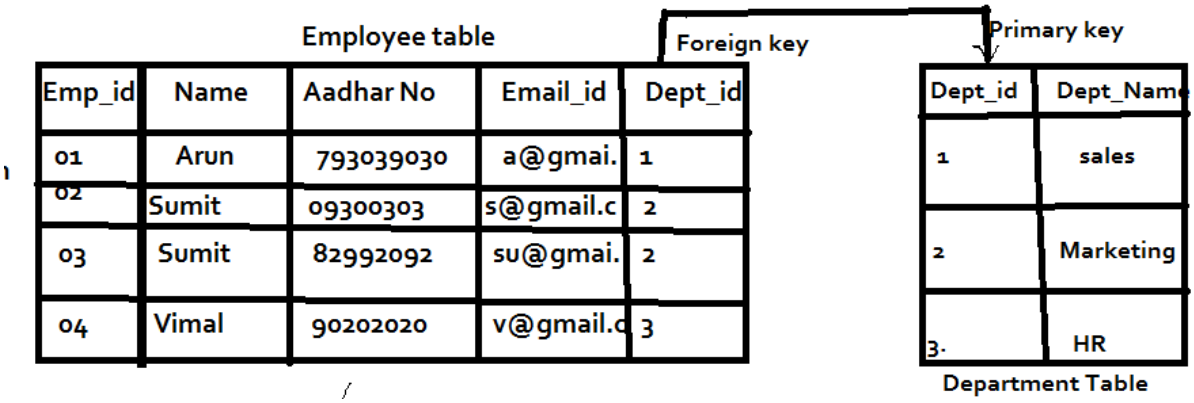
Foreign is used to join two tables together.

Foreign key can take the NULL value.

There is no restriction on a foreign key to be unique.

Foreign key may have a name other than that of a primary key.

The purpose of foreign key is to ensure referential integrity of the data.



Here, Dept_id is the Foreign key in Employee table.

6.Composite Key;

A key that has more than one attributes is known as composite key. It is also known as compound key.

cust_id	Order_id	Product_Code	Product_count
c01	001	P111	5
c02	012	P111	5
c02	012	P222	8
c01	001	P333	9

7. Unique Key-

Unique key is a key with the following properties-

- It is unique for all the records of the table.
- Once assigned, its value can not be changed i.e. it is non-updatable.
- It may have a NULL value.

Example-

The best example of unique key is Adhaar Card Numbers.

- The Adhaar Card Number is unique for all the citizens (tuples) of India (table).
- If it gets lost and another duplicate copy is issued, then the duplicate copy always has the same number as before.
- Thus, it is non-updatable.
- Few citizens may not have got their Adhaar cards, so for them its value is NULL.

8. Surrogate Key-

Surrogate key is a key with the following properties-

- It is unique for all the records of the table.
- It is updatable.
- It can not be NULL i.e. it must have some value.

Example-

Mobile Number of students in a class where every student owns a mobile phone.