Database Management Systems (DBMS)

(ER Model in DBMS)

For

BCA Part-III (Session 2017-20) Students

BY

ANANT KUMAR
MCA, M. Phil, M. Tech.
Faculty Member
Department of Computer Science
J. D. Women's College, Patna

ER Model in DBMS

Entity-relationship model is a model used for design and representation of relationships between data.

The main data objects are termed as Entities, with their details defined as attributes, some of these attributes are important and are used to identity the entity, and different entities are related using relationships.

To understand about the ER Model, we must understand about:

- Entity and Entity Set
- What are Attributes? And Types of Attributes.
- Keys
- Relationships

Entity and Entity Set

Considering the above example, **Student** is an entity, **Teacher** is an entity, similarly, **Class**, **Subject** etc are also entities.

An Entity is generally a real-world object which has characteristics and holds relationships in a DBMS.

If a Student is an Entity, then the complete dataset of all the students will be the **Entity Set**

Attributes

If a Student is an Entity, then student's **roll no.**, student's **name**, student's **age**, student's **gender** etc will be its attributes.

An attribute can be of many types, here are different types of attributes defined in ER database model:

1. **Simple attribute:** The attributes with values that are atomic and cannot be broken down further are simple attributes. For example, student's **age**.

2. **Composite attribute:** A composite attribute is made up of more than one simple attribute. For example, student's **address** will contain, **house no.**, **street name**, **pincode** etc.
3. **Derived attribute:** These are the attributes which are not present in the whole database management system, but are derived using other attributes. For example, *average age of students in a class*.
4. **Single-valued attribute:** As the name suggests, they have a single value.
5. **Multi-valued attribute:** And, they can have multiple values.

Keys

If the attribute **roll no.** can uniquely identify a student entity, amongst all the students, then the attribute **roll no.** will be said to be a key.

Following are the types of Keys:

1. Super Key
2. Candidate Key
3. Primary Key

Relationships

When an Entity is related to another Entity, they are said to have a relationship. For example, A **Class** Entity is related to **Student** entity, because students study in classes, hence this is a relationship.

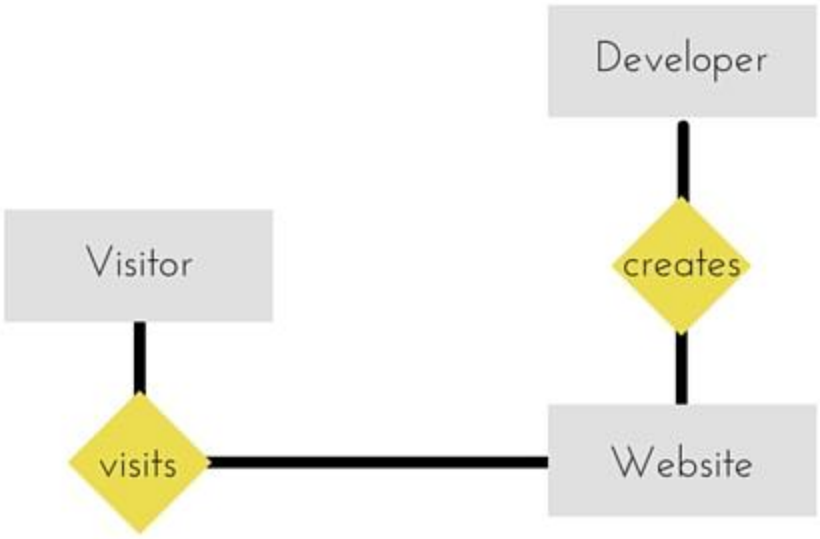
Depending upon the number of entities involved, a **degree** is assigned to relationships.

For example, if 2 entities are involved, it is said to be **Binary relationship**, if 3 entities are involved, it is said to be **Ternary** relationship, and so on.

Working with ER Diagrams

ER Diagram is a visual representation of data that describes how data is related to each other. In ER Model, we disintegrate data into entities, attributes and setup relationships between entities, all this can be represented visually using the ER diagram.

For example, in the below diagram, anyone can see and understand what the diagram wants to convey: *Developer develops a website, whereas a Visitor visits a website*.



Components of ER Diagram

Entitiy, Attributes, Relationships etc form the components of ER Diagram and there are defined symbols and shapes to represent each one of them.

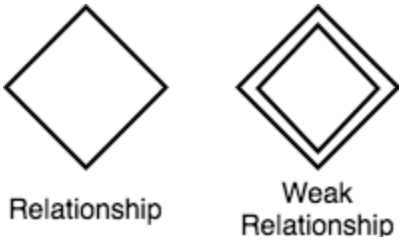
Let's see how we can represent these in our ER Diagram.

Entity

Simple rectangular box represents an Entity.

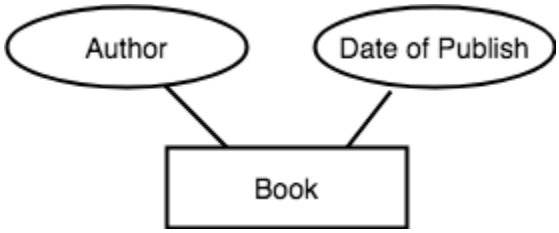
Relationships between Entities - Weak and Strong

Rhombus is used to setup relationships between two or more entities.



Attributes for any Entity

Ellipse is used to represent attributes of any entity. It is connected to the entity.



Weak Entity

A weak Entity is represented using double rectangular boxes. It is generally connected to another entity.

Key Attribute for any Entity

To represent a Key attribute, the attribute name inside the Ellipse is underlined.

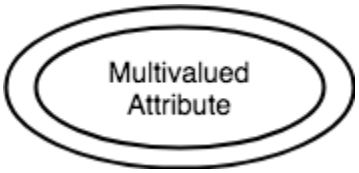
Derived Attribute for any Entity

Derived attributes are those which are derived based on other attributes, for example, age can be derived from date of birth.

To represent a derived attribute, another dotted ellipse is created inside the main ellipse.

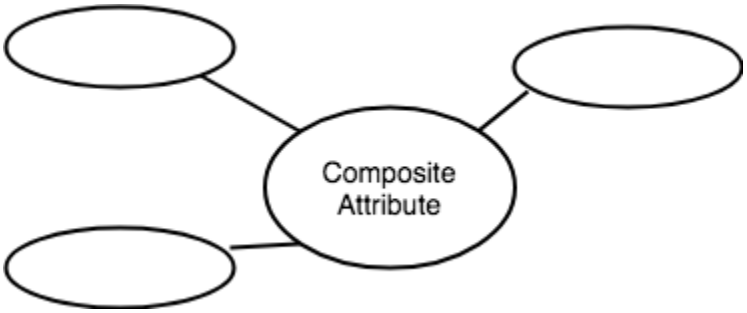
Multivalued Attribute for any Entity

Double Ellipse, one inside another, represents the attribute which can have multiple values.



Composite Attribute for any Entity

A composite attribute is the attribute, which also has attributes.



Entity

An **Entity** can be any object, place, person or class. In ER Diagram, an **entity** is represented using rectangles. Consider an example of an Organisation- Employee, Manager, Department, Product and many more can be taken as entities in an Organisation.

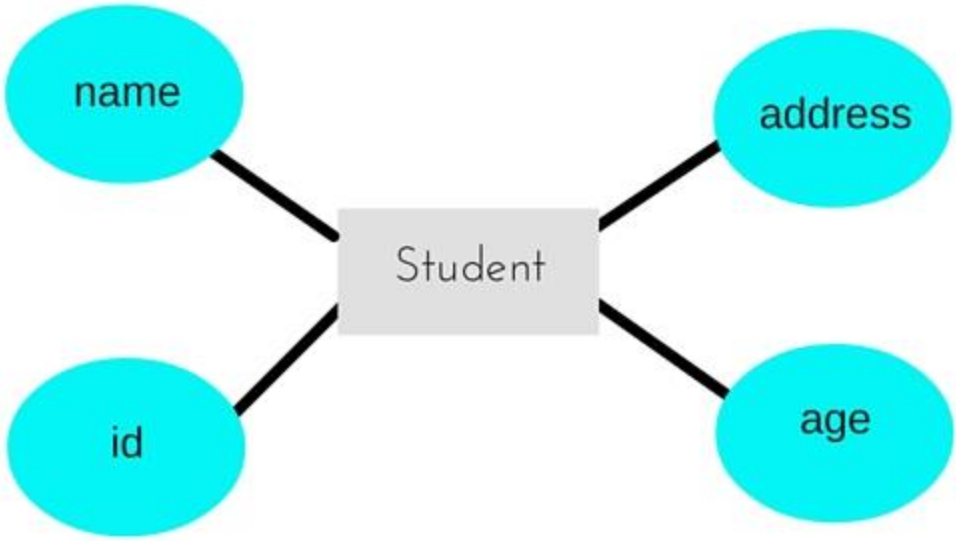
The yellow rhombus in between represents a relationship.

Weak Entity

Weak entity is an entity that depends on another entity. Weak entity doesn't have anay key attribute of its own. Double rectangle is used to represent a weak entity.

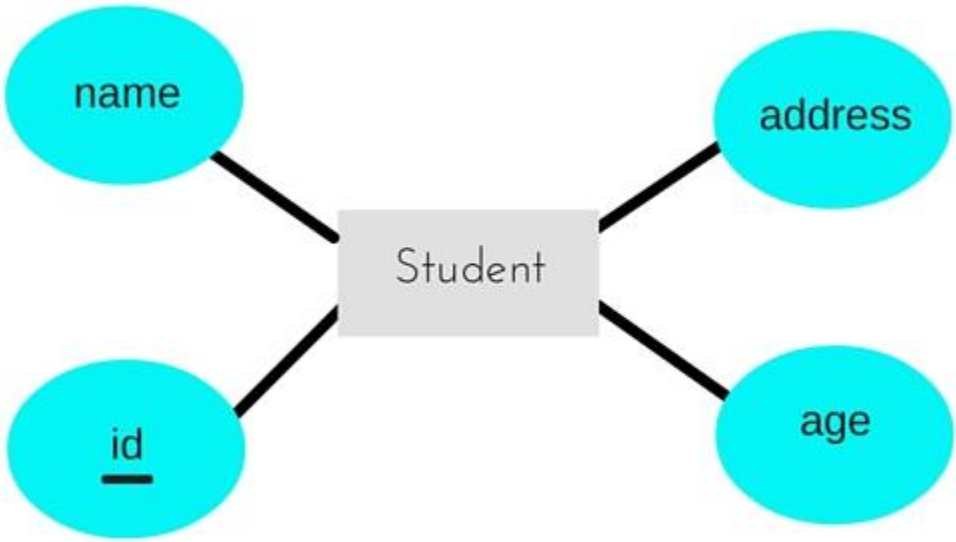
Attribute

An **Attribute** describes a property or characterstic of an entity. For example, **Name**, **Age**, **Address** etc can be attributes of a **Student**. An attribute is represented using eclipse.



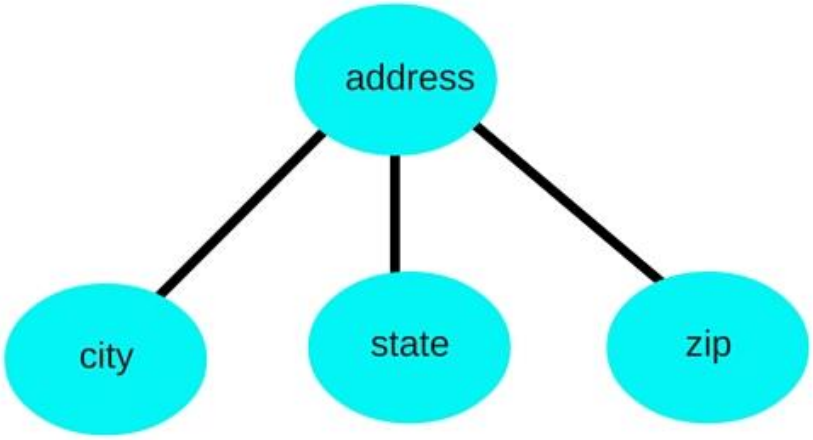
Key Attribute

Key attribute represents the main characterstic of an Entity. It is used to represent a Primary key. Ellipse with the text underlined, represents Key Attribute.



Composite Attribute

An attribute can also have their own attributes. These attributes are known as **Composite** attributes.



Relationship

A Relationship describes relation between **entities**. Relationship is represented using diamonds or rhombus.

There are three types of relationship that exist between Entities.

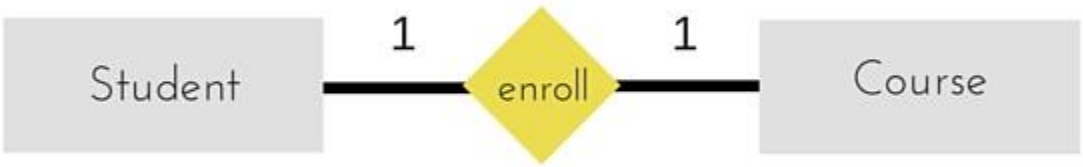
1. Binary Relationship
2. Recursive Relationship
3. Ternary Relationship

Binary Relationship

Binary Relationship means relation between two Entities. This is further divided into three types.

One to One Relationship

This type of relationship is rarely seen in real world.



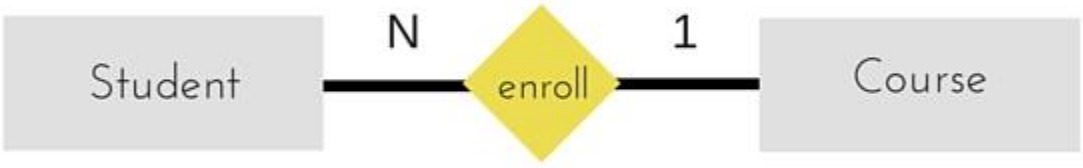
The above example describes that one student can enroll only for one course and a course will also have only one Student. This is not what you will usually see in real-world relationships.

One to Many Relationship

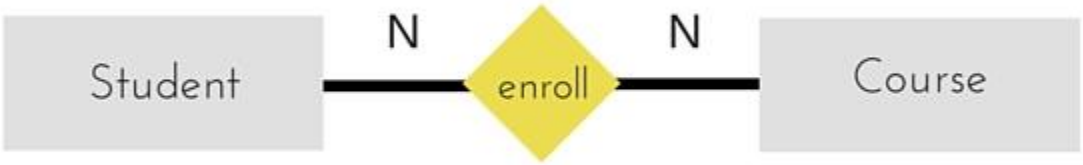
The below example showcases this relationship, which means that 1 student can opt for many courses, but a course can only have 1 student. Sounds weird! This is how it is.

Many to One Relationship

It reflects business rule that many entities can be associated with just one entity. For example, Student enrolls for only one Course but a Course can have many Students.



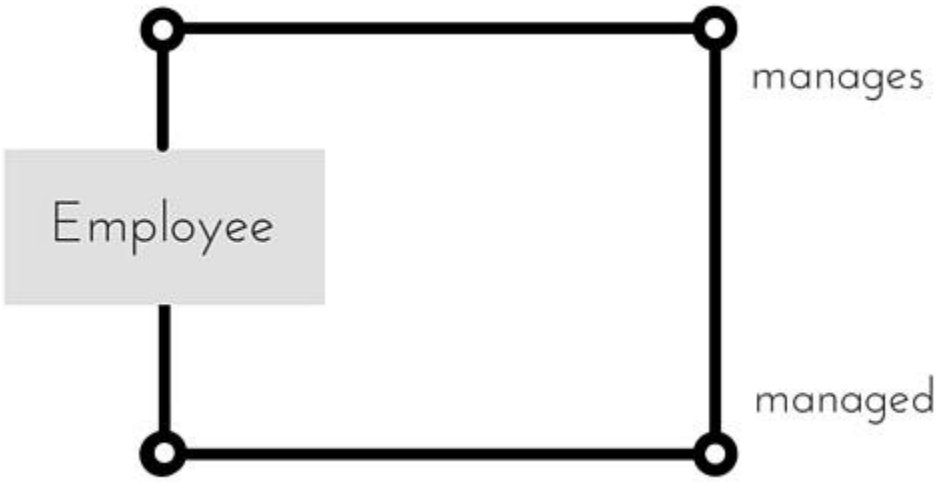
Many to Many Relationship



The above diagram represents that one student can enroll for more than one courses. And a course can have more than 1 student enrolled in it.

Recursive Relationship

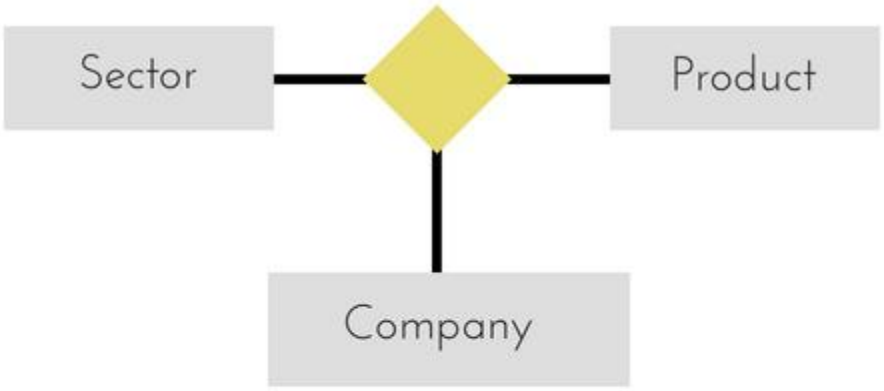
When an Entity is related with itself it is known as **Recursive** Relationship.



Ternary Relationship

Relationship of degree three is called Ternary relationship.

A Ternary relationship involves three entities. In such relationships we always consider two entites together and then look upon the third.



- The above relationship involves 3 entities.
- Company operates in Sector, producing some Products.

For example, in the diagram above, we have three related entities, **Company**, **Product** and **Sector**. To understand the relationship better or to define rules around the model, we should relate two entities and then derive the third one.

A **Company** produces many **Products**/ each product is produced by exactly one company.

A **Company** operates in only one **Sector** / each sector has many companies operating in it.

Considering the above two rules or relationships, we see that although the complete relationship involves three entities, but we are looking at two entities at a time.

The Enhanced ER Model

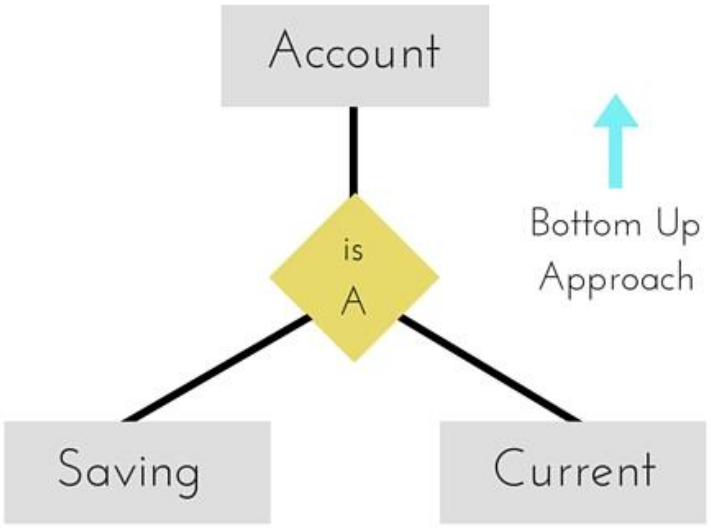
Three new concepts were added to the existing ER Model, they were:

1. Generalization
2. Specialization
3. Aggregation

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.

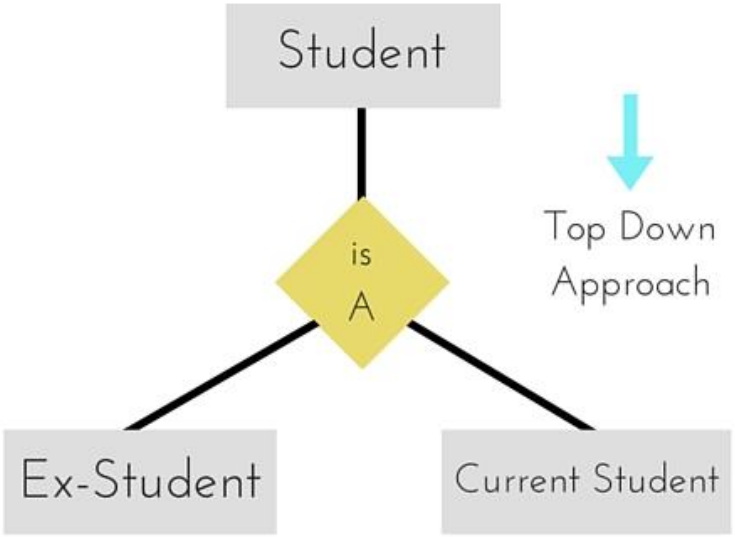
It's more like Superclass and Subclass system, but the only difference is the approach, which is bottom-up. Hence, entities are combined to form a more generalised entity, in other words, sub-classes are combined to form a super-class.



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

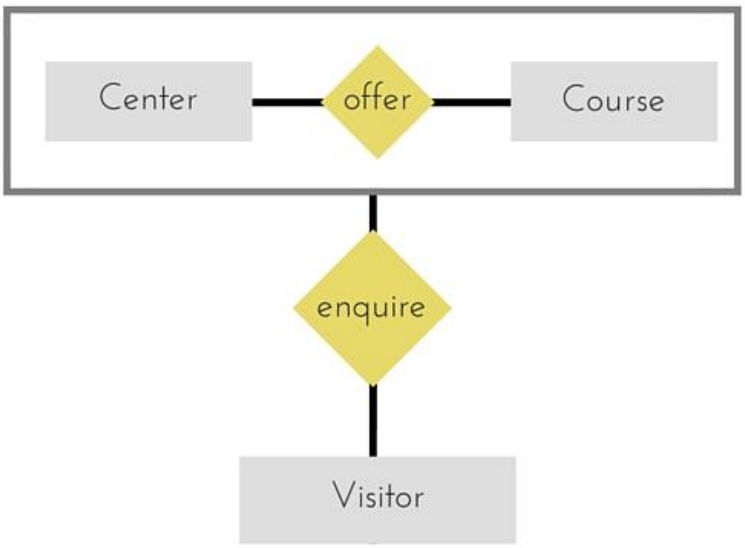
Specialization

Specialization is 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.



Aggregration

Aggregration is a process when relation between two entities is treated as a **single entity**.



In the diagram above, the relationship between **Center** and **Course** together, is acting as an Entity, which is in relationship with another entity **Visitor**. Now in real world, if a Visitor or a Student visits a Coaching Center, he/she will never enquire about the center only or just about the course, rather he/she will ask enquire about both.