


DATABASE MANAGEMENT SYSTEMS- Entity Relationship Diagram

Presented by,


Dr. S. P. Siddique Ibrahim

School of Computer Science & Engineering
VIT-AP University
Amaravati

Topics to be covered on Database Design

- Different approaches in Logical design
 - ER Modeling
 - ER notations
 - Steps in ER modeling
- 

Design process

- For small applications- Database designer who understand the application can directly decide on:-
 1. Relation to be created
 2. What are all the attributes
 3. Constraints on the relations
- 

Design process in real time applications

- However, such a **direct design process** is difficult for real-world applications
- Highly complex
- **Example:**
 - Aadhar Database, Census, Banking, Insurance, Revenue, E-commerce, etc.,

Design process in real time applications

contd.,

- Often **no one person understands** the complete data needs of an application.



Steps to be followed!!!!!!!

- Understanding the application requirements-Database Designer
- Database Designer must interact with the users of the application.



How to represent???????

1. Represent in a **high level fashion** that can be understood by the user.
2. Then, translate the requirements into lower levels of the design.

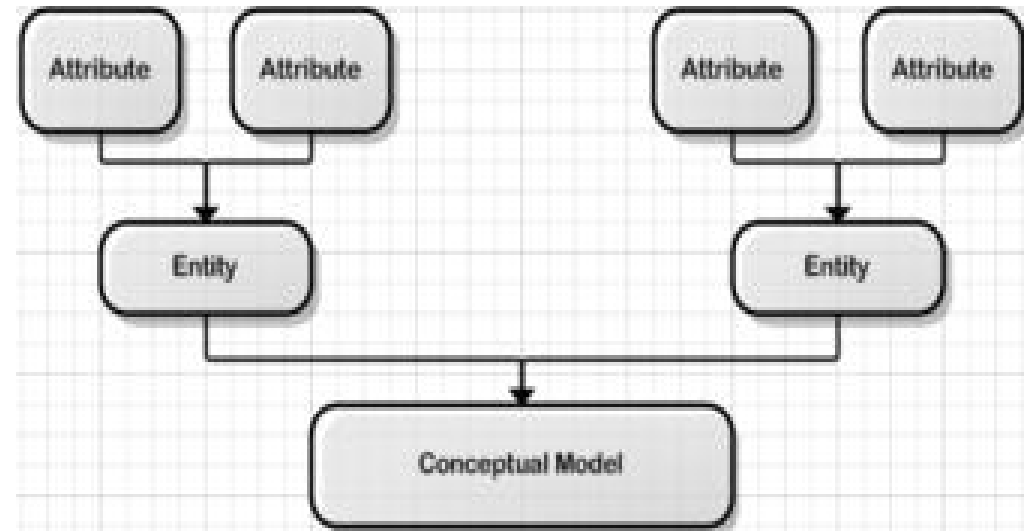
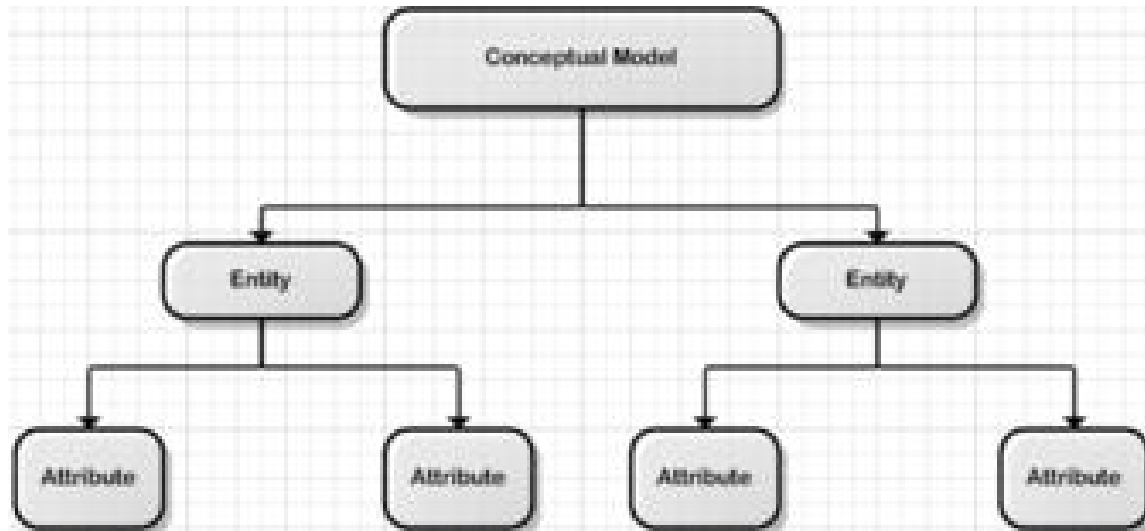


How? : Database Design Strategies in Computer Science

- Entity –Relationship Model
 - Diagrammatic representation to describe the structure of database.
 - visual representation of data that describes how data is related to each other
- Normalization Theory (Better half)
 - Formalize what designs are bad, and test for them


Database Design Strategies ..ctd

- ER Model –Top Down approach
- Normalization –Bottom up Approach

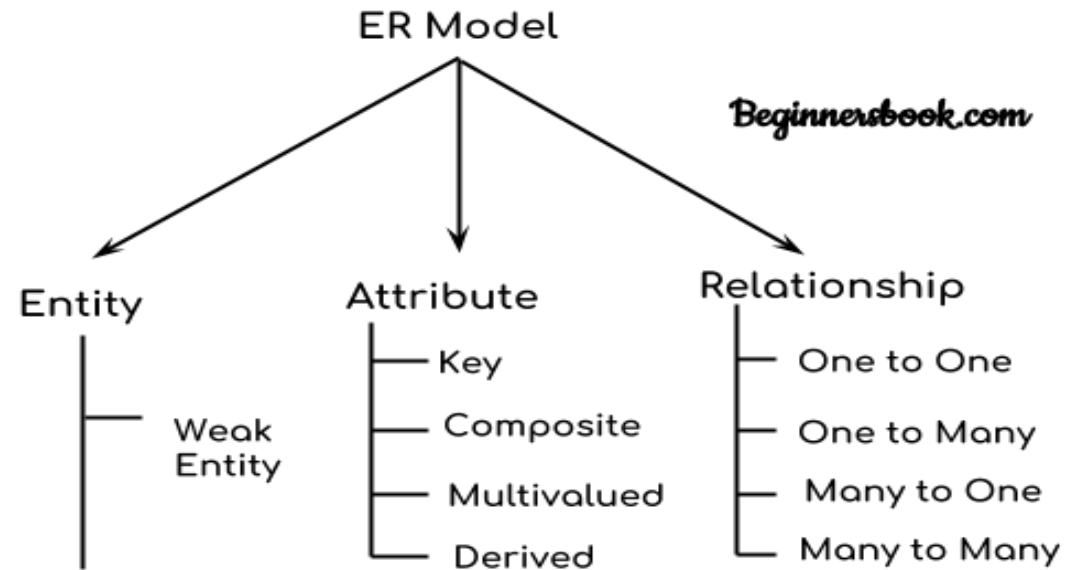


img

ER Modelling

- Defines the **conceptual view** of a database
 - E-R diagram/modelling can express the overall logical structure of a **database graphically**.
 - E-R diagrams are **simple and clear**.
 - Works around **real-world entities** and the associations among them
 - Good option for designing databases at **view level**
- 

Components of ER Model



Components of ER Diagram

1. Entity

- An entity is a *'thing'*, 'person' or *'object'* in the real world that is **unique** from all other objects.
- Can be easily identifiable.
- Named with a **noun**- **person, place, object**
- For example, In a school database
 1. Students, Teachers, classes, course offered can be considered as entities.
 2. In a bank database – customers, staff members, accounts are some examples for entity.

Employee

Department

Store

Office

- Object- car, van, table
- Person-employee, faculty, student
- Place- building, class room, park, mall




Entity and Entity sets

- **Entity set**

- Set of entities of the same type that share the same properties
- **Example:** Set of all persons, companies
- Students set may contain all the students of a school.
- Teachers set may contain all the teachers of a school.

- **Entity instance**

- Single identifiable real-world thing.
 - Example: Teja, Simth, Roger etc.,
- 

Entity Sets -- *instructor* and *student*

instructor_ID instructor_name

76766	Crick
45565	Katz
10101	Srinivasan
98345	Kim
76543	Singh
22222	Einstein

instructor

student-ID student_name

98988	Tanaka
12345	Shankar
00128	Zhang
76543	Brown
76653	Aoi
23121	Chavez
44553	Peltier

student

Let's see some examples:

- Consider an Example :

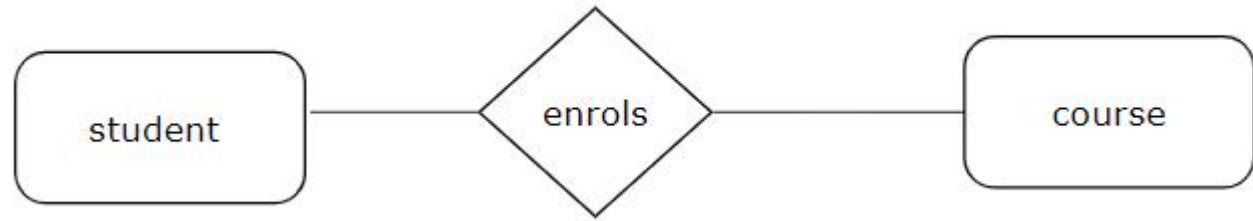
Statement: Student enrolls a course

Entity : Student, course

What is the relationship between student and course?

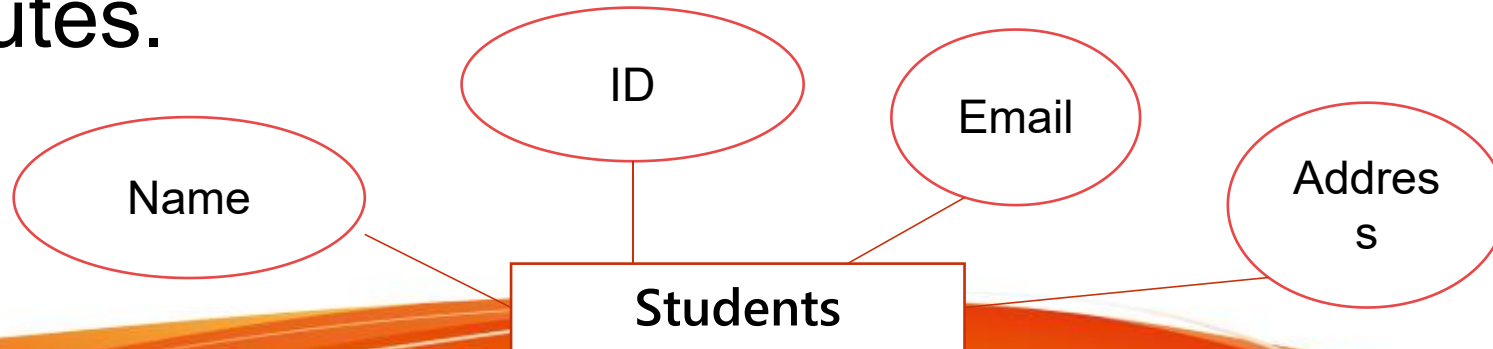
→ Enrols

ER Diagram -Entities are represented by means of rectangles.




2. Attributes

- *Entities* are represented by means of **their descriptive properties**, called attributes.
- All attributes have **values**.
- Property or characteristic of an entity type
- For Example:
- A student entity may have **name, year, class, email, age...** as attributes.



Domain/Range of Attribute

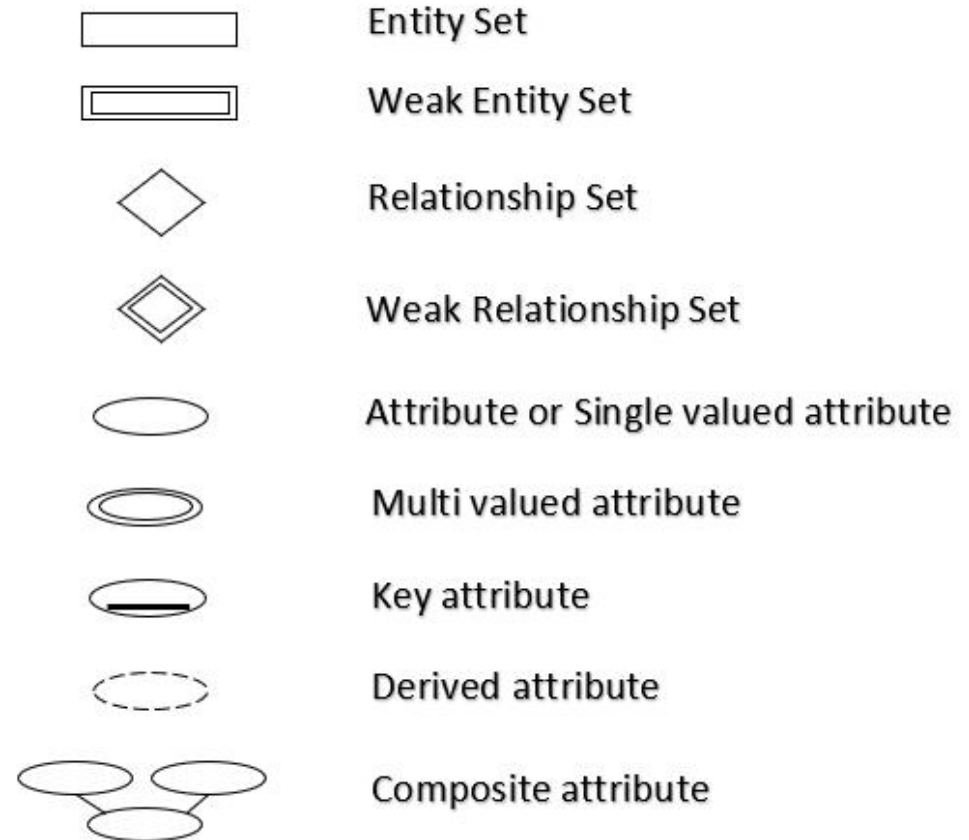
- The set of permitted values that can be assigned to each attribute is called as **domain or range** of the attribute.
 - **Example:**
 - A student's name cannot be a numeric value.
 - It has to be alphabetic.
 - A student's age cannot be negative, etc.
- 

ER Diagrams Notation


- Entity set – **Rectangle**
- Weak Entity Sets - **Double Rectangle**
- Relationship Set – **Diamond**
- Attribute – **Ellipses**
- Multivalued Attribute - **Double Ellipse**
- Derived Attribute - **Dashed Ellipse**

Lines: They link attributes to Entity Sets and Entity sets to Relationship Set

Double Lines: Total participation of an entity in a relationship set



Types of Attributes

1. Simple Attribute
 2. Composite Attribute
 3. Derived Attribute (Stored Attribute)
 4. Single value Attribute
 5. Multivalued Attribute
 6. Stored Attributes-
- 

1. Simple or Atomic attribute

- Simple attributes are **atomic** values, which cannot be divided further.
- **Example:** A student's roll number is an atomic value.
- A bank's **Account number** is an atomic value of some digits.




2. Composite or compound attribute

- Composite attributes are made of more than one simple attribute.
- **Example:** A student's full name can be divided into first_name, middle_name and last_name.



3. Stored Attribute


- An attribute, which cannot be derived from other attribute, is known as stored attribute.
 - **Example:** Date of Birth of an employee.
- 

4. Derived attribute

- Derived attributes are the attributes that do not exist in the physical database, but their values are derived from other attributes present in the database.
- **Example:**
 - **Average_salary** in a department can be derived from the individual salary of the employees.
 - **Age** can be derived from data_of_birth
 - **Average_marks** scored by a student or a class can be derived from the individual marks.



5. Single-value attribute

- Single-value attributes contain single value for a particular entity.
 - **Example:** Social_Security_Number, Age, roll number, Employee code, etc
- 

6. Multi-value attribute


- Multi-value attributes may contain more than one values for the same entity.
- **Example:** Phone number, Email_ID, etc., as a person can have more than one phone number, email_id, etc.
- Subject registered, cabin, own cars, own houses, address, bank account,




7. Complex Attribute

- If an attribute of an entity, is built using composite and multivalued attributes, then these attributes are called complex attributes.
- **Example:** A person can have more than one **residence** and each residence can have **multiple phones**, an addressphone for a person entity can be specified as –
- {Addressphone (phone {(Area Code, Phone Number)}, Address(Sector Address (Sector Number, House Number), City, State, Pin)))}
- Here {} are used to enclose multivalued attributes and () are used to enclose composite attributes with comma separating individual


8. Key Attribute

- It represents primary key.
 - This attribute has distinct value for each entity in an entity set.
 - Key attributes uniquely identifies an entity in an entity set.
 - **Example:** Roll number in a Student Entity Type.
- 


9. Non Key Attribute

- These are attributes other than candidate key attributes in a table.
 - **Example:** Firstname is a non key attribute.
- 

10. Required Attribute

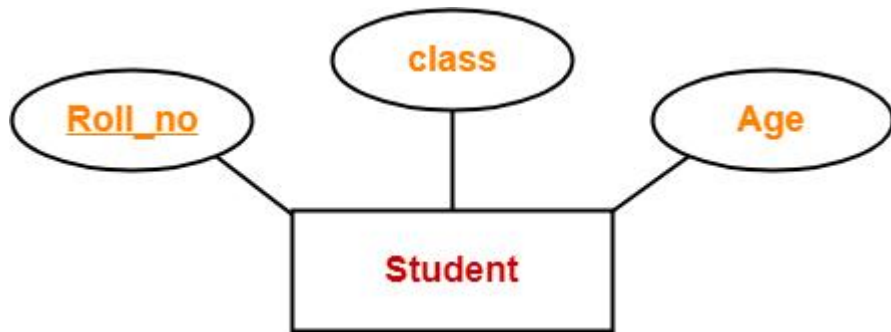
- A required attribute is an attribute that must have a data value.
 - These attributes are required because they describe what is important in the entity.
 - **Example:** firstname and lastname in a student entity are required attributes.
- 

11. Optional Attribute/Null Value Attribute

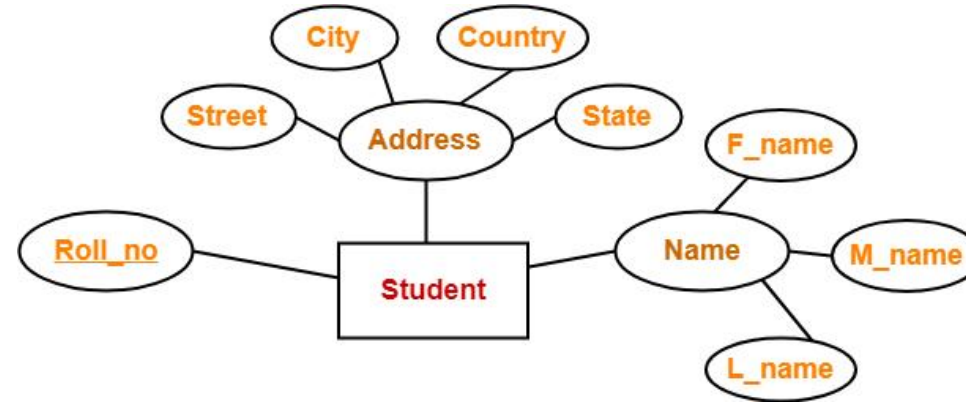
- An optional attribute may not have a value in it and can be left blank.
 - **Example:** Middlename or email address in a student entity is an optional attribute. as some students may not have middlename or email address.
- 

Types of Attributes

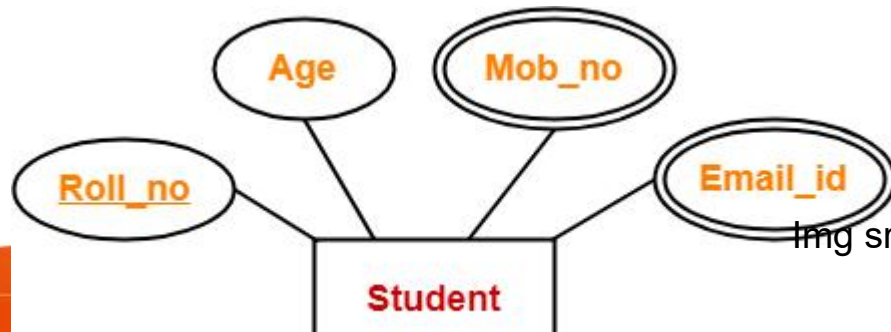
- Simple Attribute(atomic)



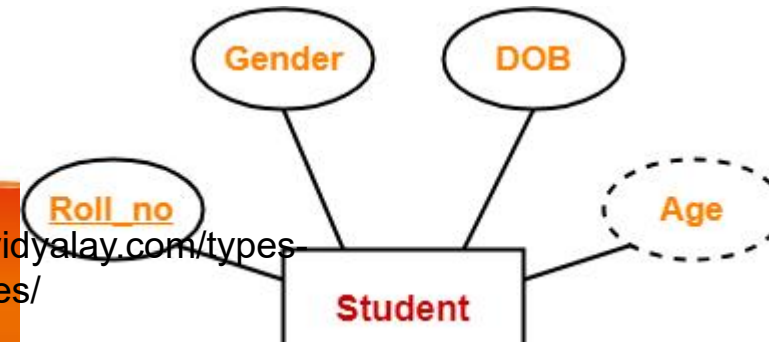
Composite Attribute



- Multivalued Attribute




Derived Attribute




Relationship

- The association among entities is called a relationship.
- Relationships are usually named using **verbs**.

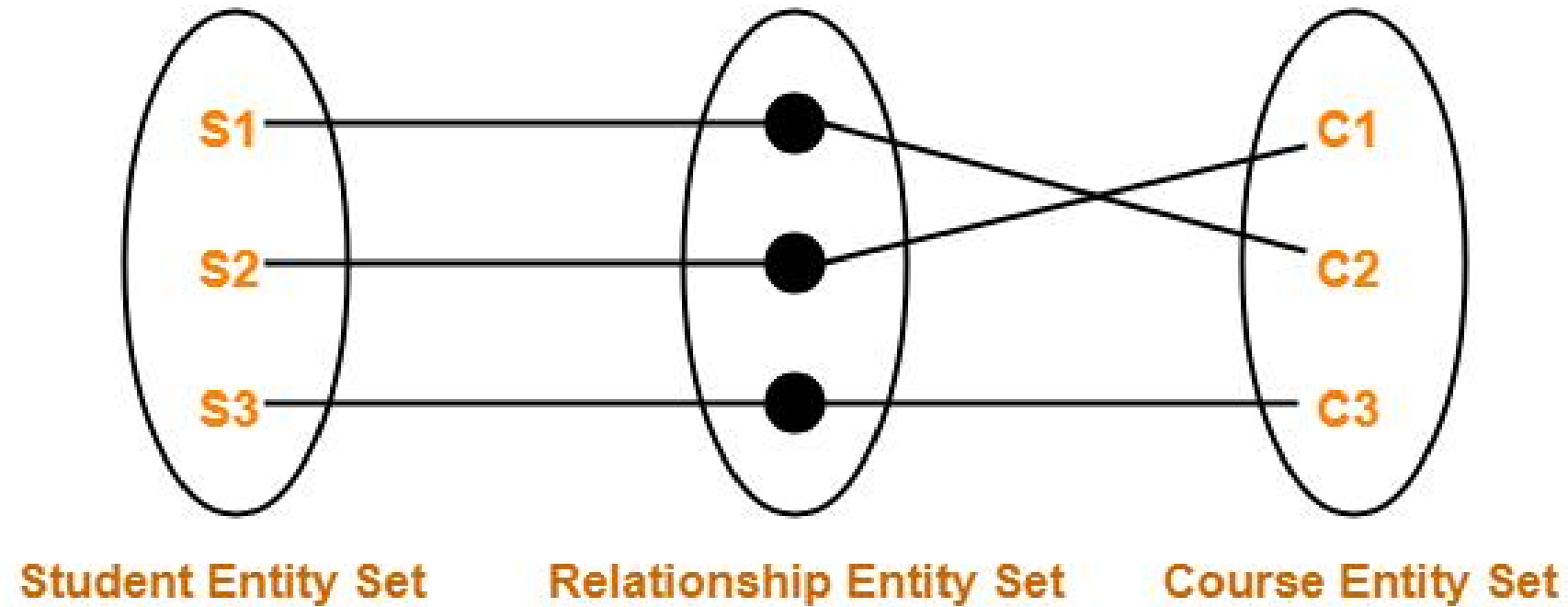
Example:

- An employee **works_at** a department
 - A student **enrolls** in a course
 - Here, Works_at and Enrolls are called relationships.
- 

Relationship Set

- A set of relationships of similar type is called a relationship set.
 - Like entities, a relationship also can have attributes, and they are called as **descriptive attributes**.
- 

Set representation of above ER diagram is-



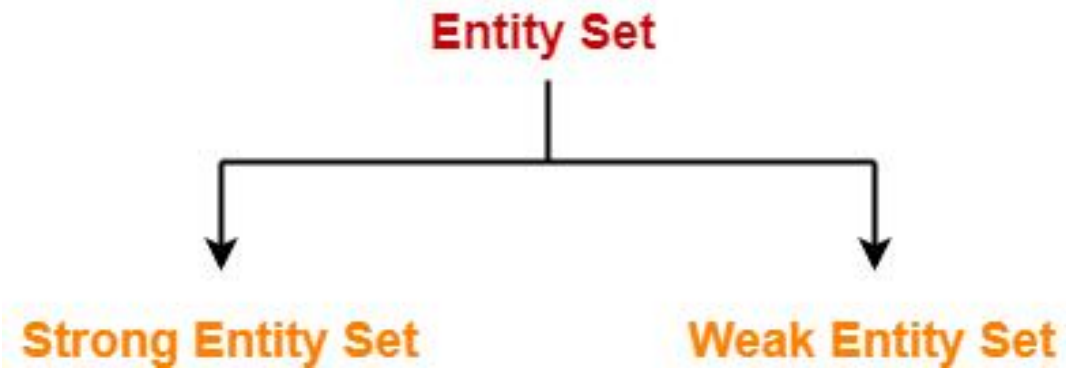
Set Representation of ER Diagram

Weak Relationship

- A relationship with a weak entity is termed as weak relationship.

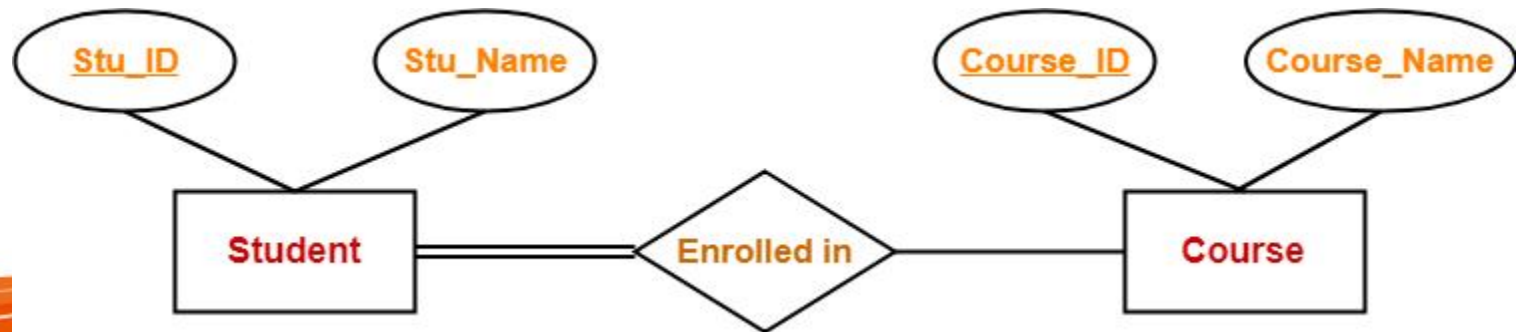


Strong Entity set Vs Weak Entity Set



Strong Entity Set

- A strong entity set is an entity set that contains sufficient attributes to uniquely identify all its entities.
- In other words, a primary key exists for a strong entity set.
- Primary key of a strong entity set is represented by underlining it.



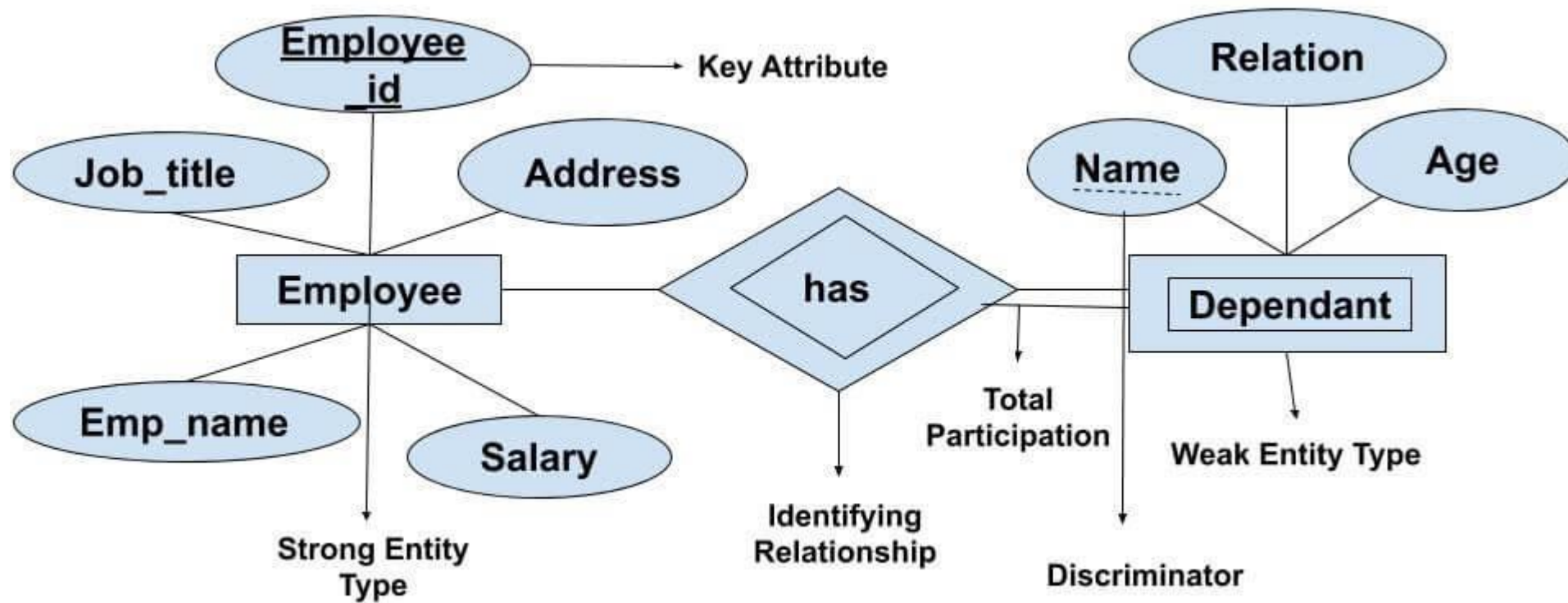
Weak Entity Set

- A weak entity set is an entity set that does not contain sufficient attributes to uniquely identify its entities.
- A **primary key does not exist** for a weak entity set.
- It contains a **partial key** called as a **discriminator**.
- Discriminator can identify a group of entities from the entity set.
- Discriminator is represented by underlining with a **dashed line**.

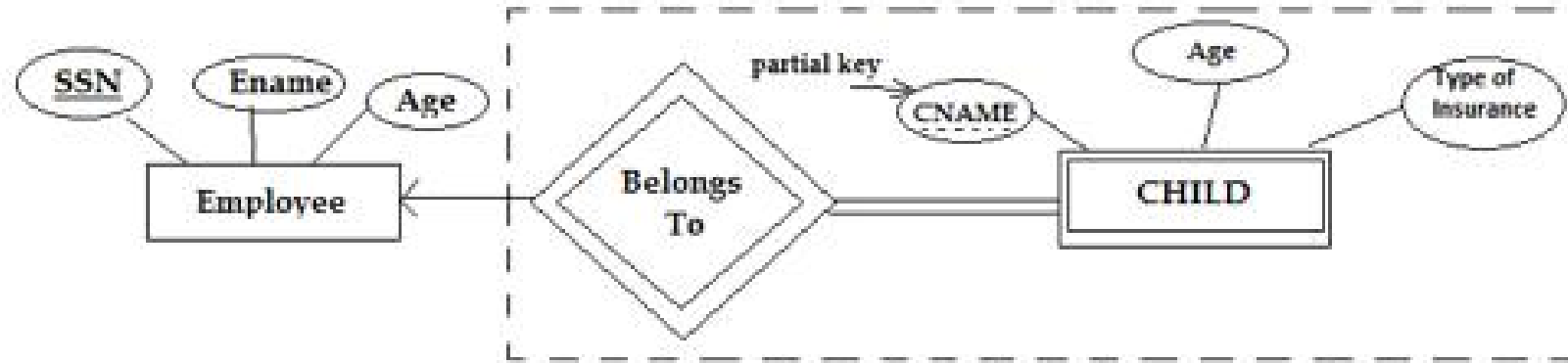
Primary key of weak entity set

= Its own discriminator + Primary key of strong entity set

Weak Entity Set-Example

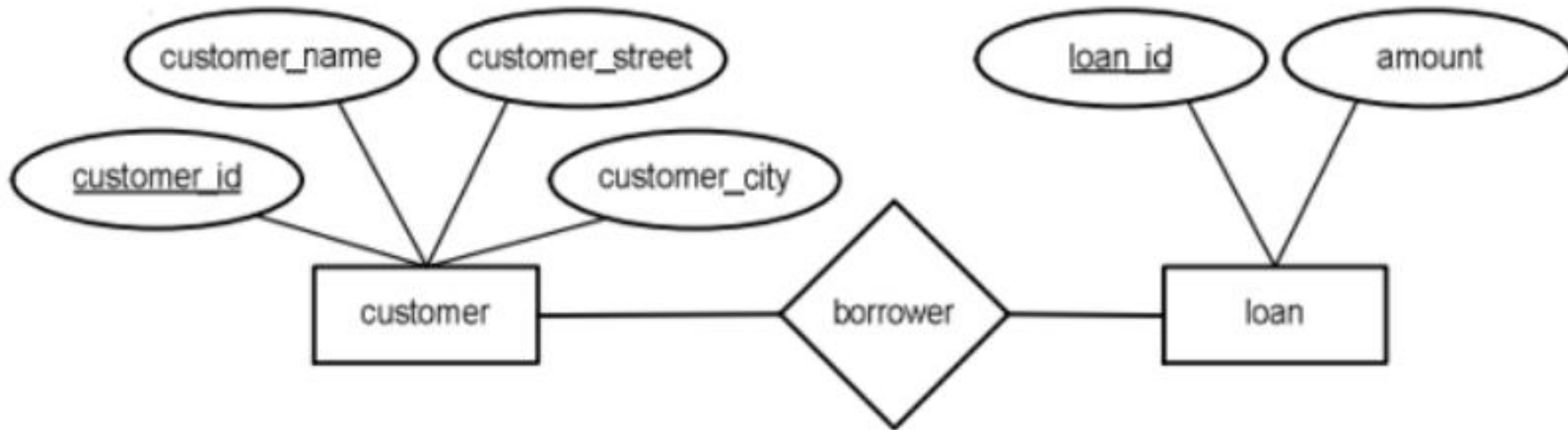


Another Example

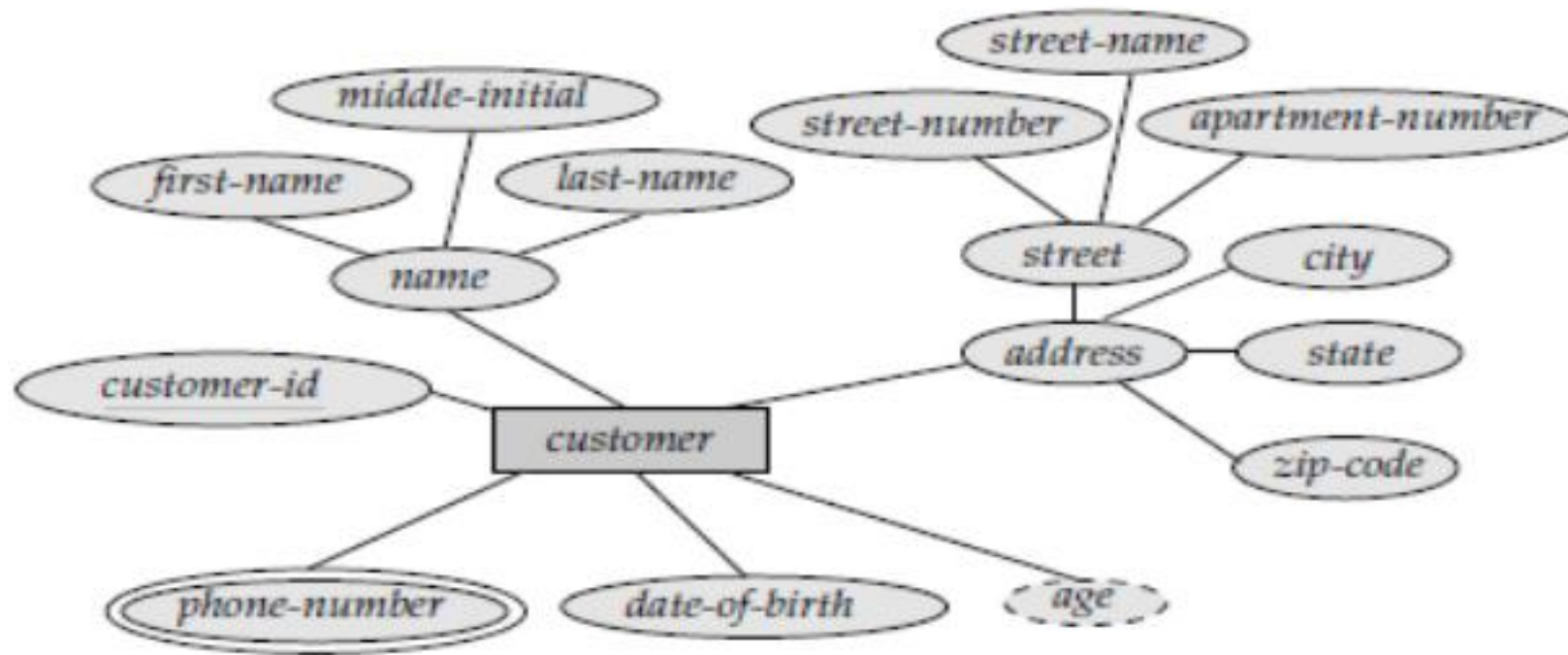


ER Diagram Example:

Bank scenario: Customer borrows loan




ER Diagram Example – Single valued, multi values and derived attributes



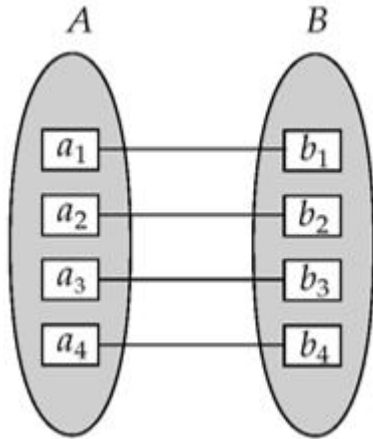
- **Cardinality** defines the number of entities in one entity set, which can be associated with the number of entities of other set via relationship set.



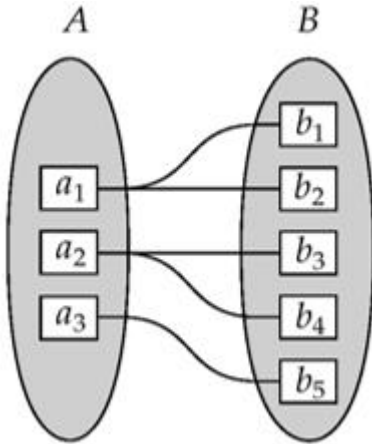
Mapping cardinality of a relationship

- Expresses the number of entities to which another entity can be associated via a relationship set
 - Useful in describing binary relationship set
 - Types of Relationships
 - One to one
 - One to many
 - Many to one
 - Many to many
- 

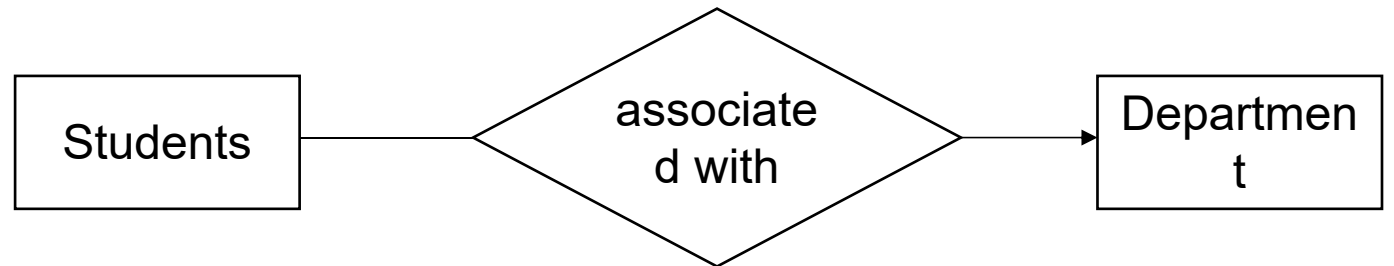
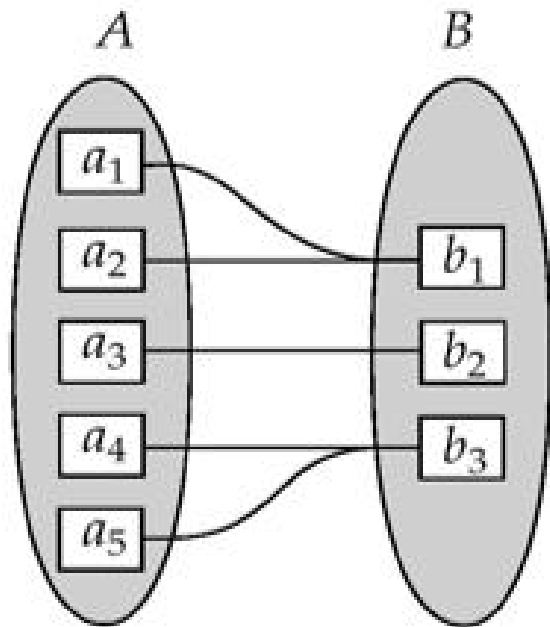
One to one



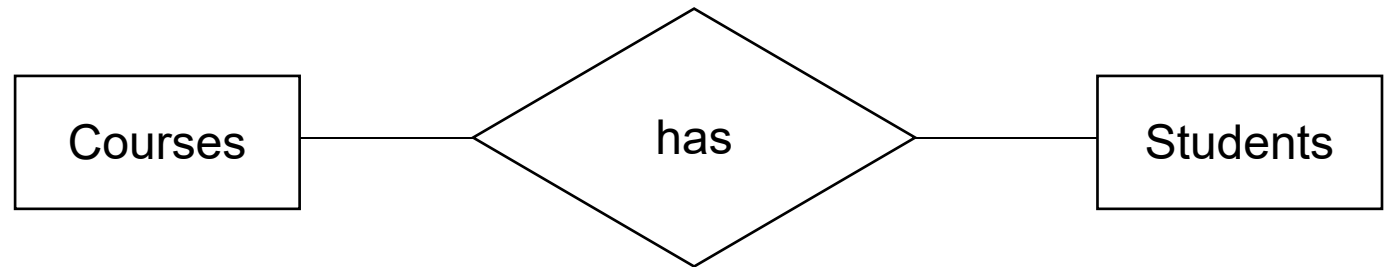
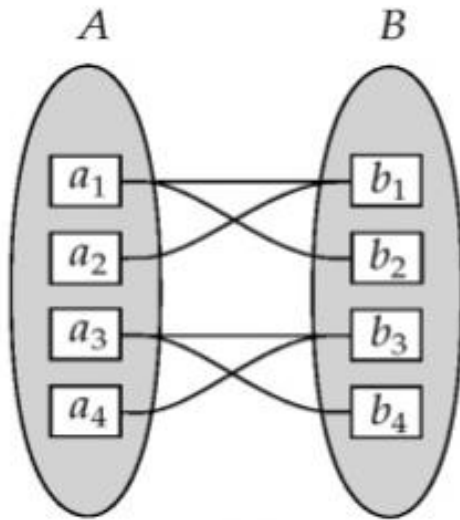
One to many



Many to one

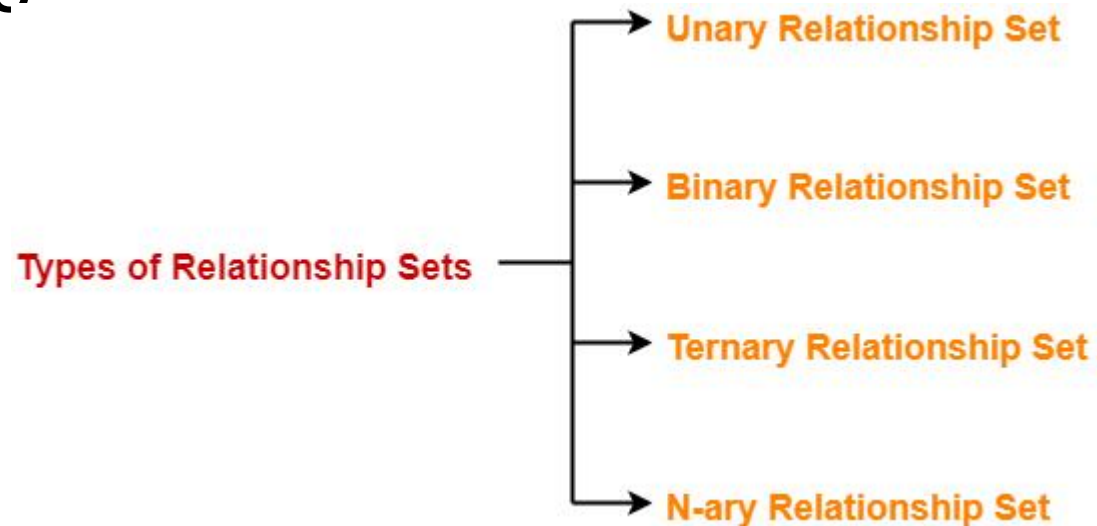


Many to many



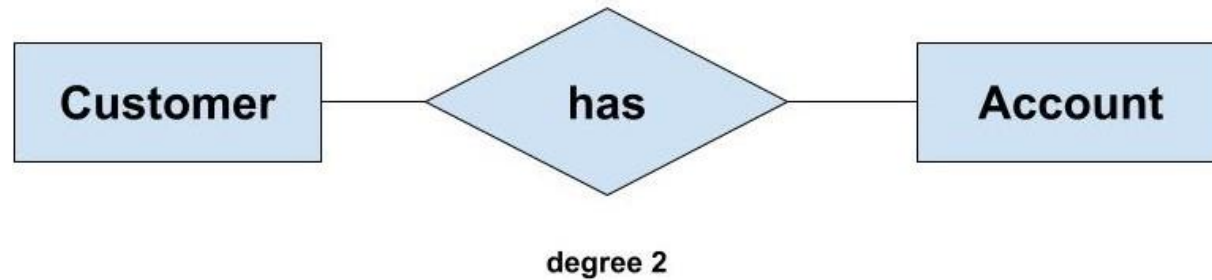
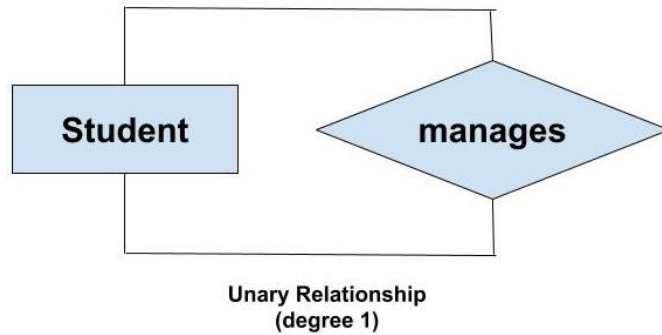
Types of Relationship set

- On the basis of degree of a relationship set, a relationship set can be classified into the following types



Examples for Relationship set

- Unary Relationship Binary Relationship




Description about Relationship set

- **Unary Relationship**

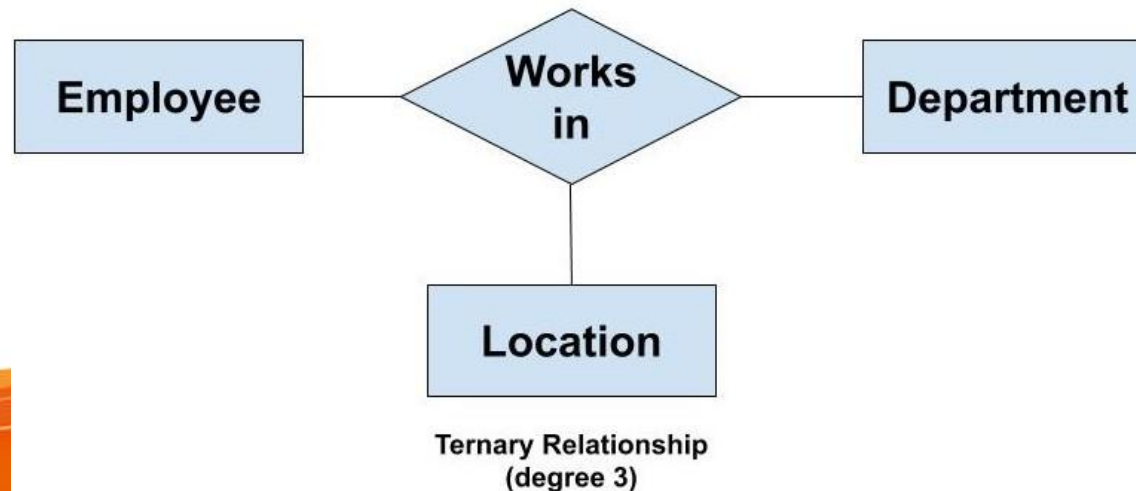
- A unary relationship exists when both the participating entity type are the same.
- When such a relationship is present we say that the degree of relationship is 1.

- **Binary Relationship**

- A binary relationship exists when exactly two entity type participates.
 - When such a relationship is present we say that the degree is 2.
 - This is the most common degree of relationship.
- 

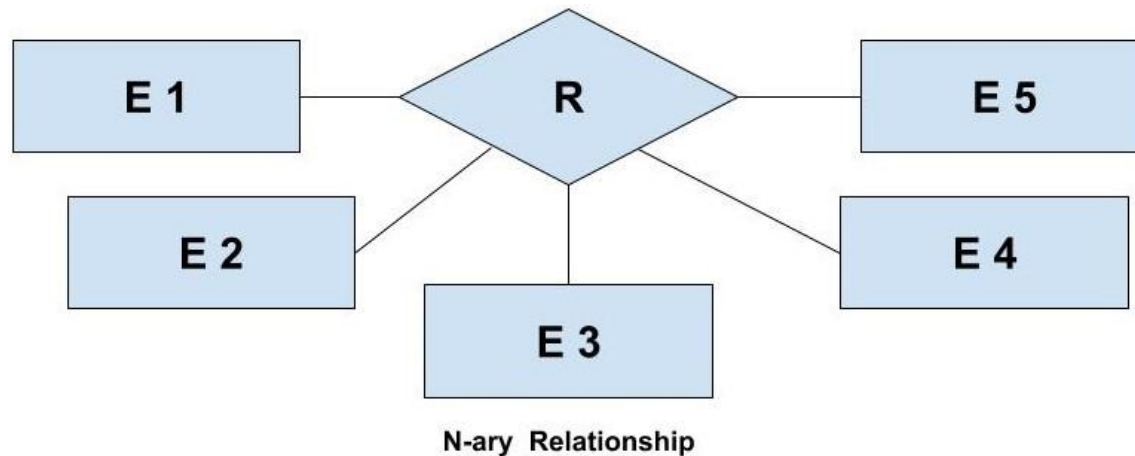
Ternary Relationship

- A ternary relationship exists when exactly **three entity type** participates.
- When such a relationship is present we say that the degree is 3.
- As the number of entity increases in the relationship, it becomes complex to convert them into relational tables



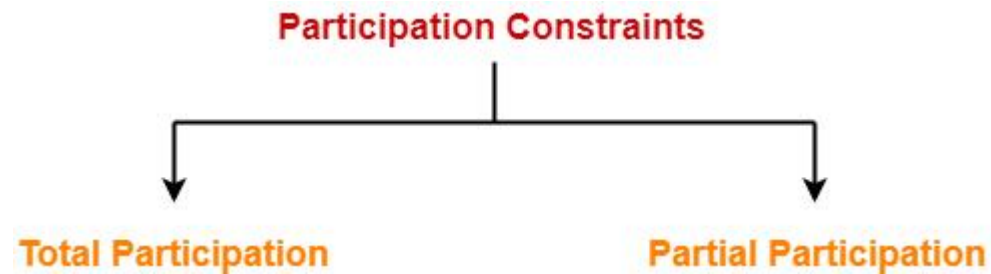
N-ary Relationship

- An N-ary relationship exists when 'n' number of entities are participating.
- Any number of entities can participate in a relationship.



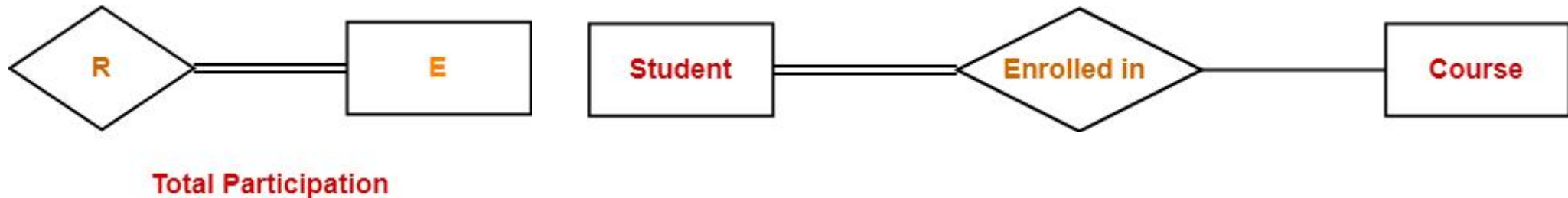
Types of Participation Constraints

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



Total Participation

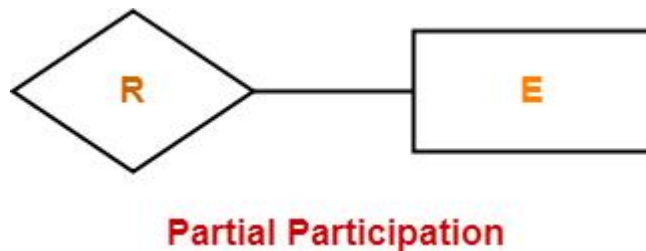
- It specifies that **each entity in the entity set must compulsorily participate in at least one relationship** instance in that relationship set.
- Total participation is represented using a **double line** between the entity set and relationship set.



- *Double line between the entity set “Student” and relationship set “Enrolled in” signifies total participation*

Partial Participation

- It specifies that each entity in the entity set **may or may not participate** in the relationship instance in that relationship set.
- Partial participation is represented using a single line between the entity set and relationship set.

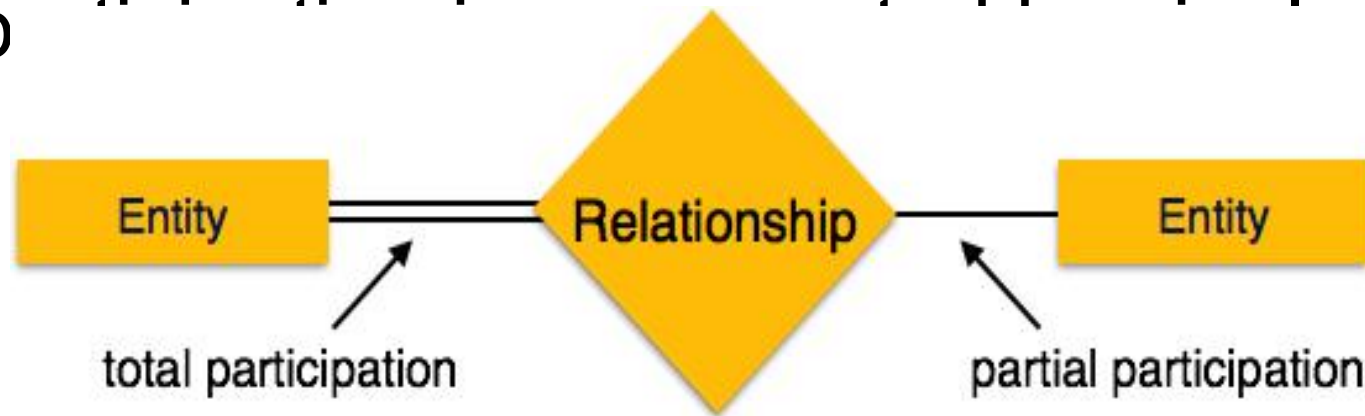


Example-



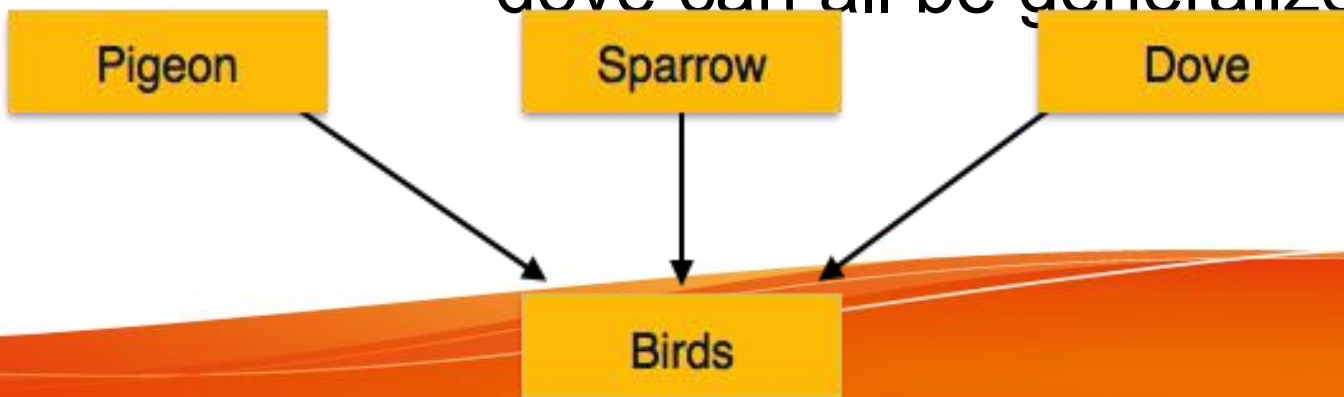
Participation Constraints

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



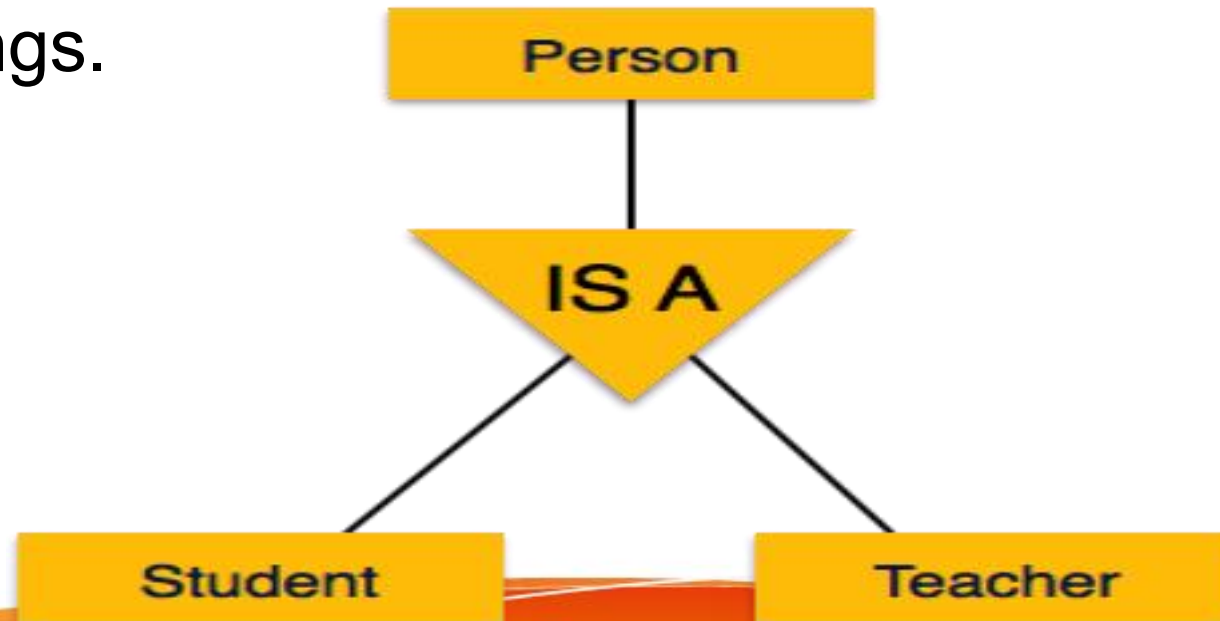
Generalization

- The process of generalizing entities,
 - where the generalized entities **contain the properties of all the generalized entities**, is called generalization.
- In generalization, a number of entities are brought together into one generalized entity based on their similar characteristics.
 - For example, pigeon, house sparrow, crow and dove can all be generalized as Birds.



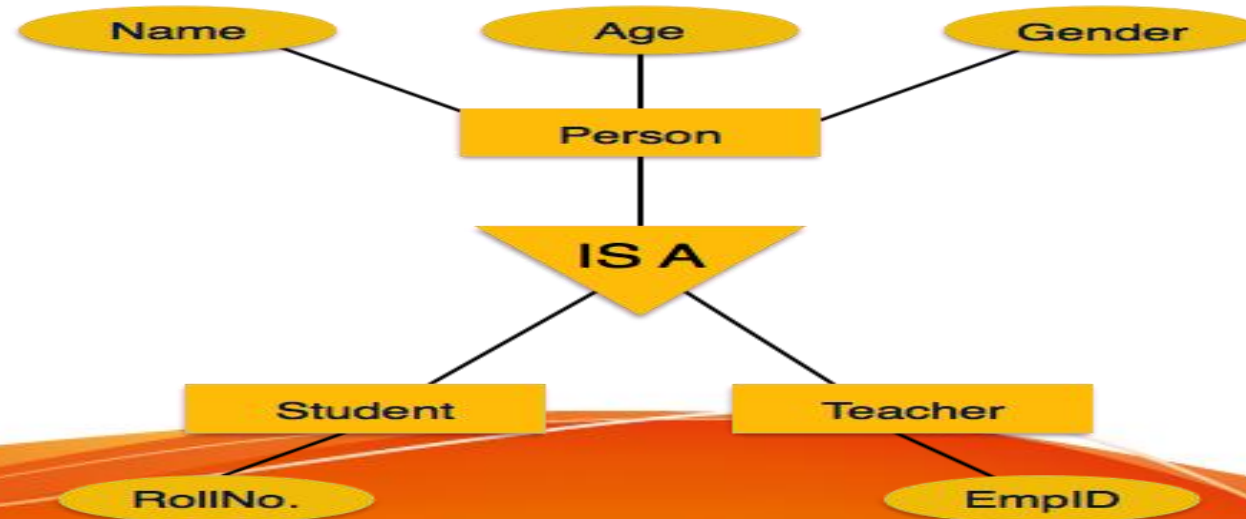
Specialization

- Specialization is the opposite of generalization.
- In specialization, a group of entities is divided into sub-groups based on their characteristics.
 - A person has name, date of birth, gender, etc.
These properties are common in all persons, human beings.



Inheritance

- **Inheritance** is an important feature of Generalization and Specialization.
- It allows lower-level entities to inherit the attributes of higher-level entities.
 - For example, the attributes of a Person class such as name, age, and gender can be inherited by lower-level entities such as Student or Teacher.



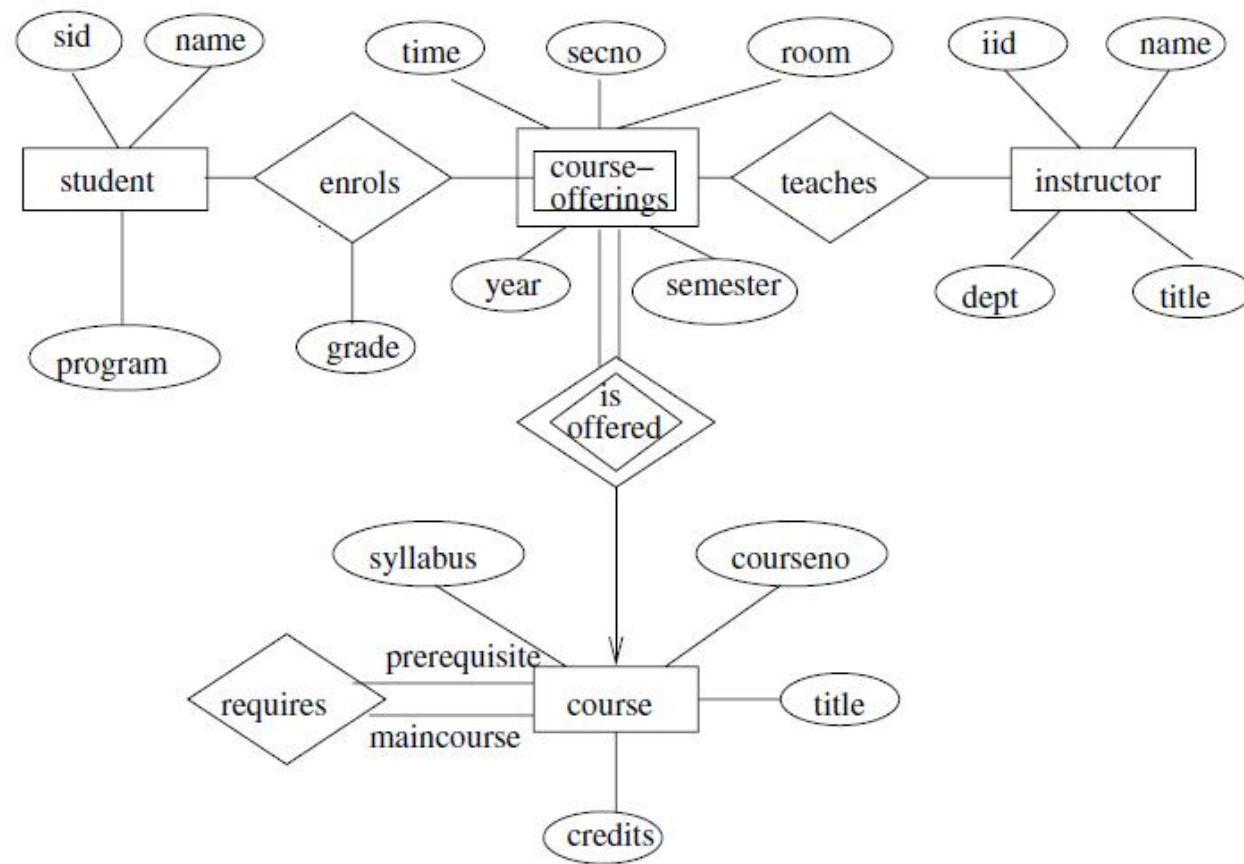
Problem statement

- A university office maintains data about the following entities:
 1. courses, including number, title, credits, syllabus, and prerequisites;
 2. course offerings, including course number, year, semester, section number, instructor(s), timings, and classroom;
 3. students, including student-id, name, and program;
 4. instructors, including identification number, name, department, and title.

Further, the enrollment of students in courses and grades awarded to students in each course they are enrolled for must be appropriately modeled.


Construct an E-R diagram for the registrar office. Document all assumptions that you make about the mapping constraints.






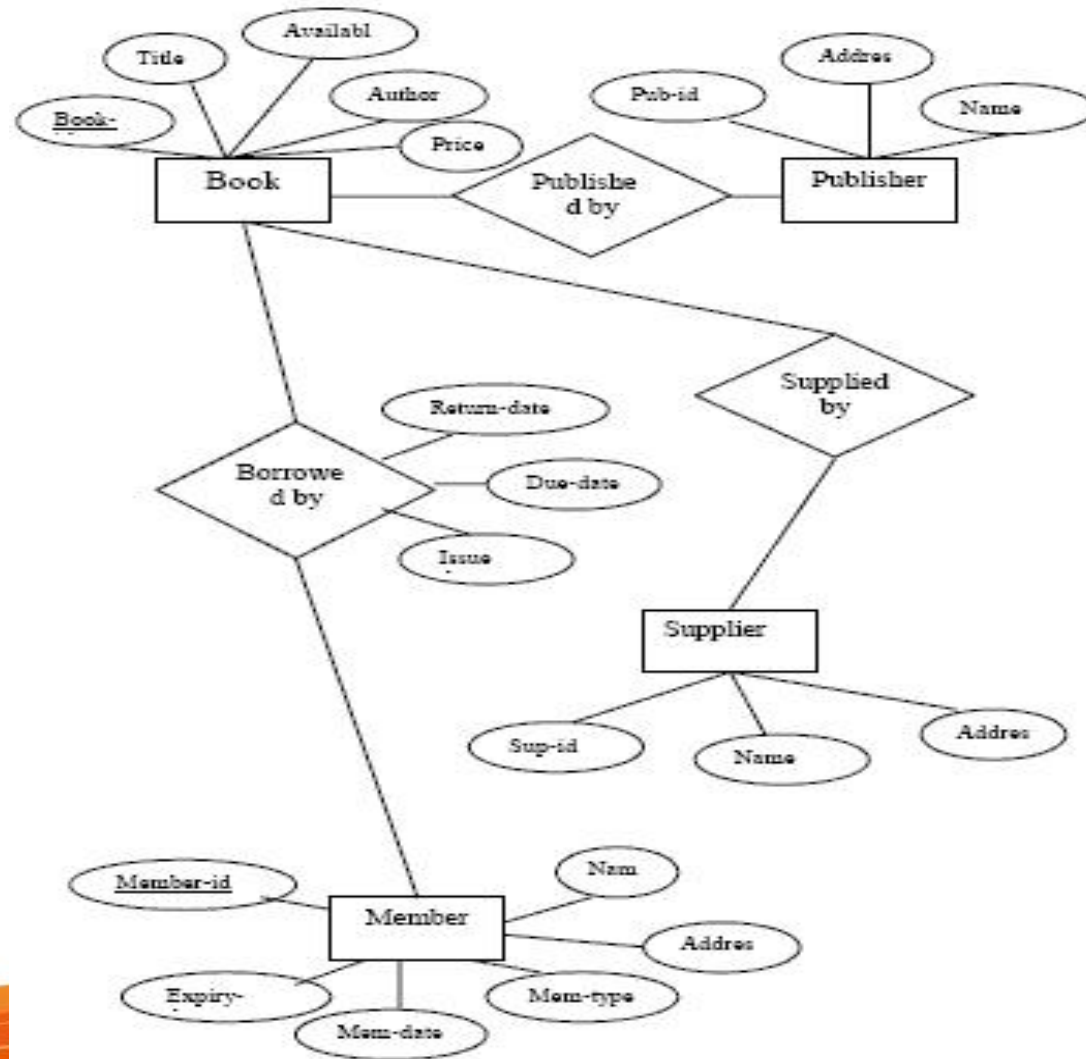
E-R diagram for a university.

Steps in designing an Entity-Relationship schema

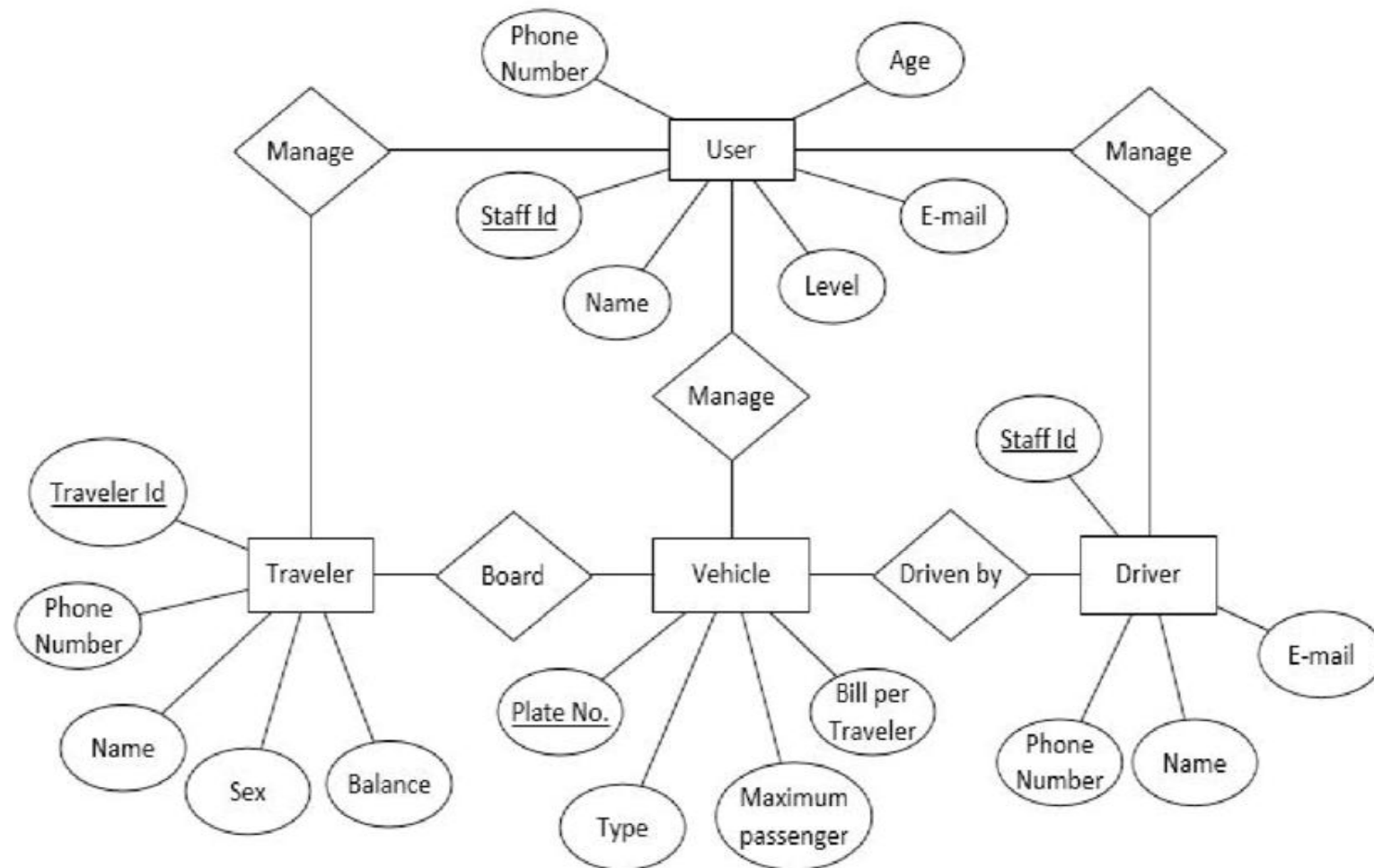
- Identify entity types (entity type vs. attribute)
 - Identify relationship types
 - Identify and associate attributes with entity and relationship types
 - Determine attribute domains
 - Determine primary key attributes for entity types
 - Associate (refined) cardinality ratio(s) with relationship types
 - Design generalization /specialization hierarchies including constraints
- 

- To create an Entity Relationship Diagram (ERD), you can follow these steps:
 - Understand the project: Understand the project's goals and objectives.
 - Identify entities: Identify the main entities involved in the system. Entities can be people, objects, events, or concepts.
 - Identify relationships: Determine the relationships between the entities.
 - List attributes: List the attributes of each entity set.
 - Determine primary keys: Determine the primary key for each entity set.
 - Identify strong and weak entities: Identify which entities are strong and which are weak.
 - List relationship attributes: List the attributes of each relationship set.
 - Specify cardinality constraints: Specify the number of instances of one entity that relate to one instance of another entity.
 - Specify participation constraints: Specify how each entity set participates in each relationship set.
 - Remove redund
- 

Library management system – ER Example



Vehicle management system – ER Example



Try it out – Bank Management System

