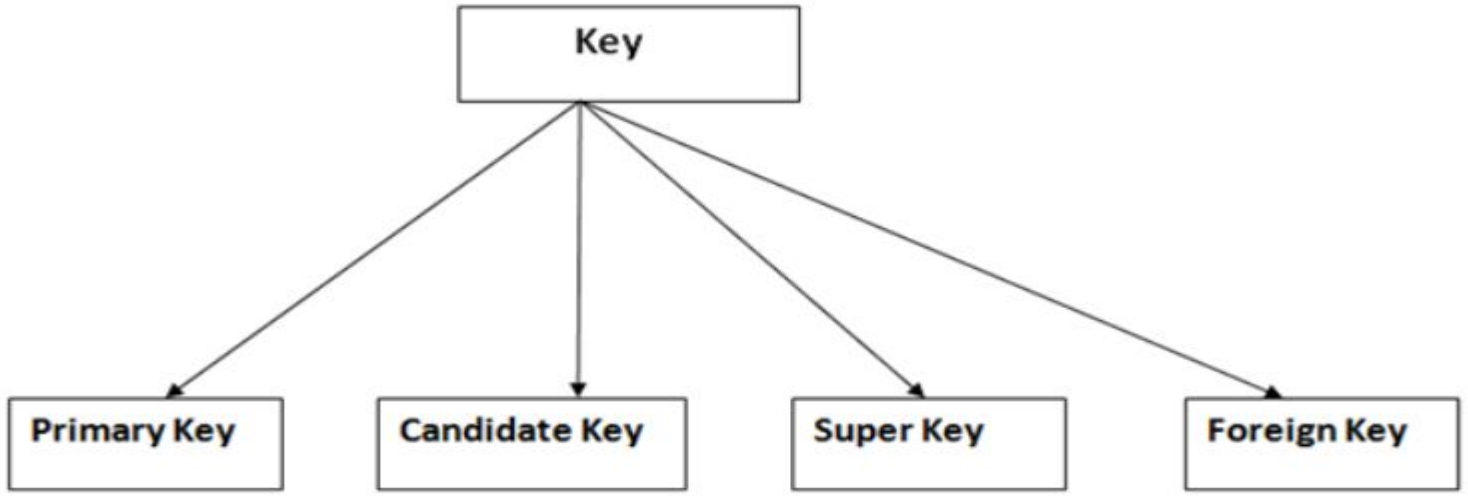
Unit 2

Relational model

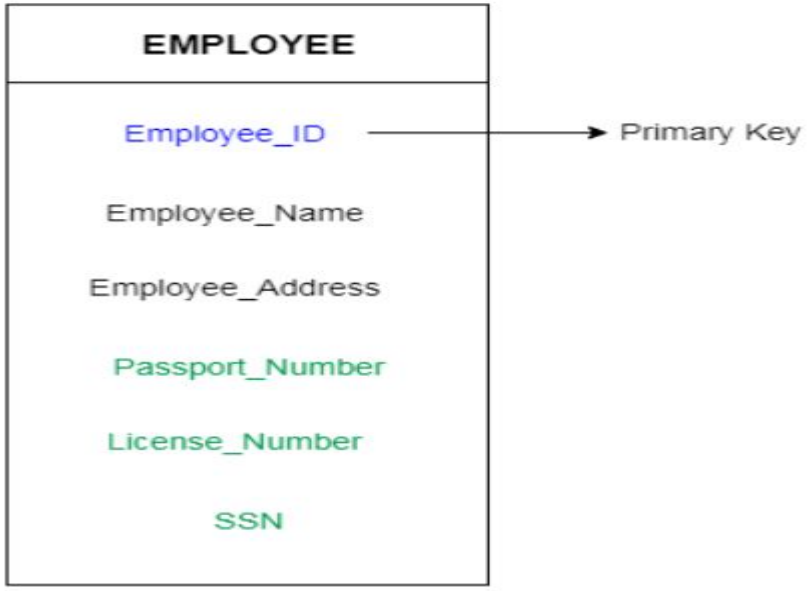
Key Constraint



- Primary key
- Foreign key
- Candidate key
- Super key

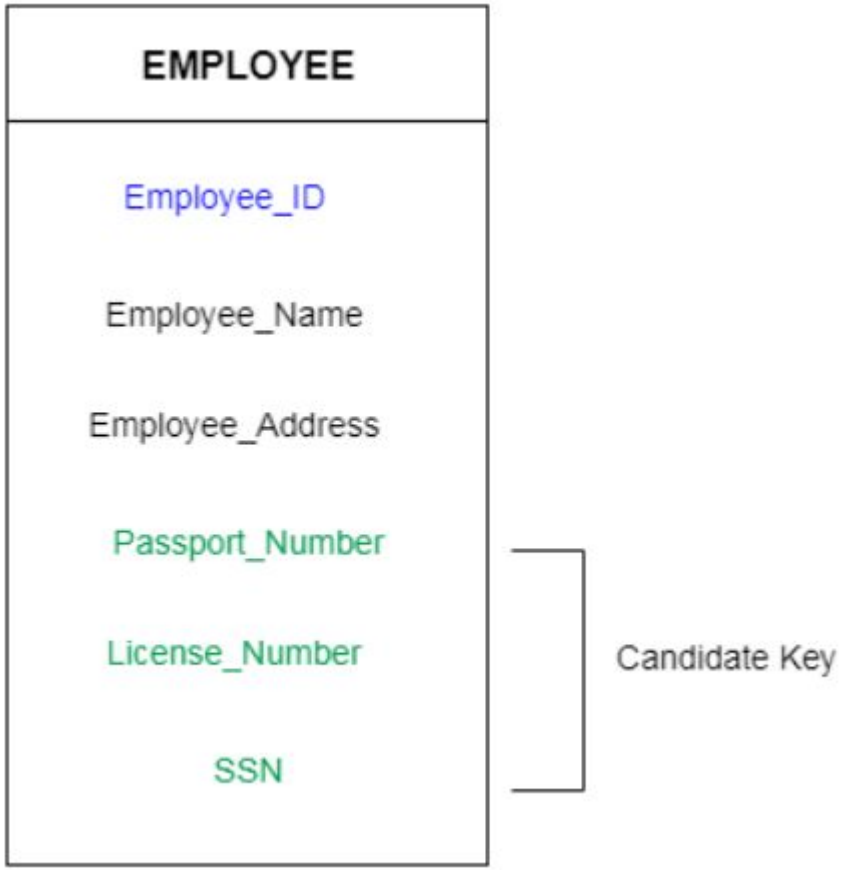
Primary key

- It is the first key which is used to identify one and only one instance of an entity uniquely. An entity can contain multiple keys as we saw in PERSON table. The key which is most suitable from those lists become a primary key.
- In the EMPLOYEE table, ID can be primary key since it is unique for each employee. In the EMPLOYEE table, we can even select License_Number and Passport_Number as primary key since they are also unique.



Candidate key

- A candidate key is an attribute or set of an attribute which can uniquely identify a tuple.
- The remaining attributes except for primary key are considered as a candidate key. The candidate keys are as strong as the primary key.

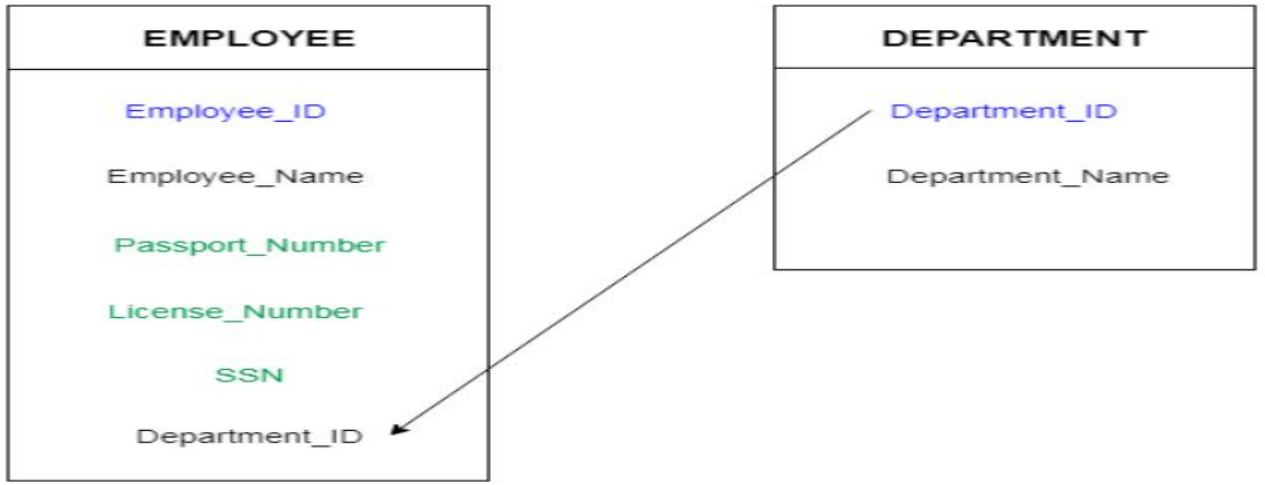


Super key

- Super key is a set of an attribute which can uniquely identify a tuple. Super key is a superset of a candidate key.
- **For example:** In the above EMPLOYEE table, for(EMPLOEE_ID, EMPLOYEE_NAME) the name of two employees can be the same, but their EMPLYEE_ID can't be the same. Hence, this combination can also be a key.

Foreign key

- Foreign keys are the column of the table which is used to point to the primary key of another table.
- In a company, every employee works in a specific department, and employee and department are two different entities. So we can't store the information of the department in the employee table. That's why we link these two tables through the primary key of one table.
- We add the primary key of the DEPARTMENT table, Department_Id as a new attribute in the EMPLOYEE table.
- Now in the EMPLOYEE table, Department Id is the foreign key, and both the tables are related.



What is Database design

- What is Database Design?
  - Database Design is a collection of processes that facilitate the **designing**, **development**, **implementation** and **maintenance** of enterprise database management systems.
- What is E-R diagram?
  - E-R diagram: (Entity-Relationship diagram)
  - It is **graphical (pictorial) representation** of database.
  - It uses different types of symbols to represent different objects of database.

Entity

Symbol

- An entity is a **person**, a **place** or an **object**.
- An entity is represented by a **rectangle** which contains the name of an entity.
- Entities of a college database are:
  - Student
  - Professor/Faculty
  - Course
  - Department
  - Result
  - Class
  - Subject

Write down the different entities of bank database.

Write down the different entities of hospital database.

Entity set

- It is a **set (group) of entities** of **same type**.
- Examples:
  - All persons having an account in a bank
  - All the students studying in a college
  - All the professors working in a college
  - Set of all accounts in a bank

Attributes

- Attribute is **properties** or details about an entity.
- An attribute is represented by an **oval** containing name of an attribute.
- Attributes of Student are:
  - Roll No
  - Student Name
  - Branch
  - Semester
  - Address
  - Mobile No
  - Age
  - SPI
  - Backlogs

Symbol

Write down the different attributes of Faculty entity.

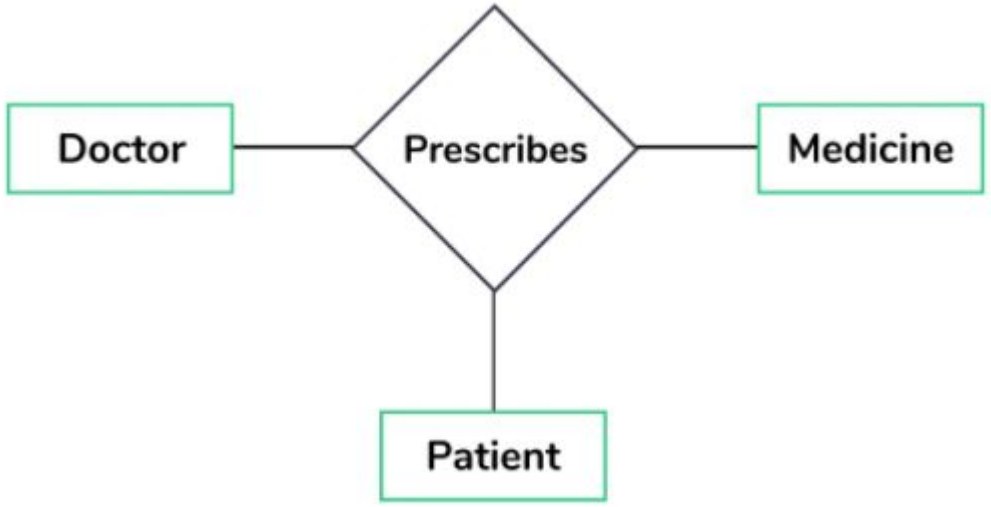
Relationship

- Relationship is an **association** (connection) between several entities.
- It should be placed between two entities and a line connecting it to an entity.
- A relationship is represented by a **diamond** containing relationship's name.

Symbol

Ternary Relationship

- Ternary relationship **three different Entities takes part in a Relationship**.



Examples

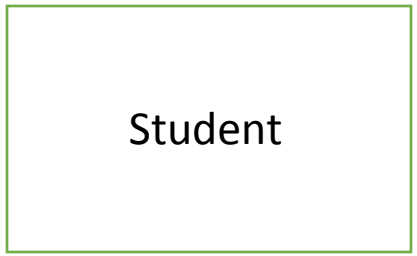
Draw an E-R diagram of following pair of entities

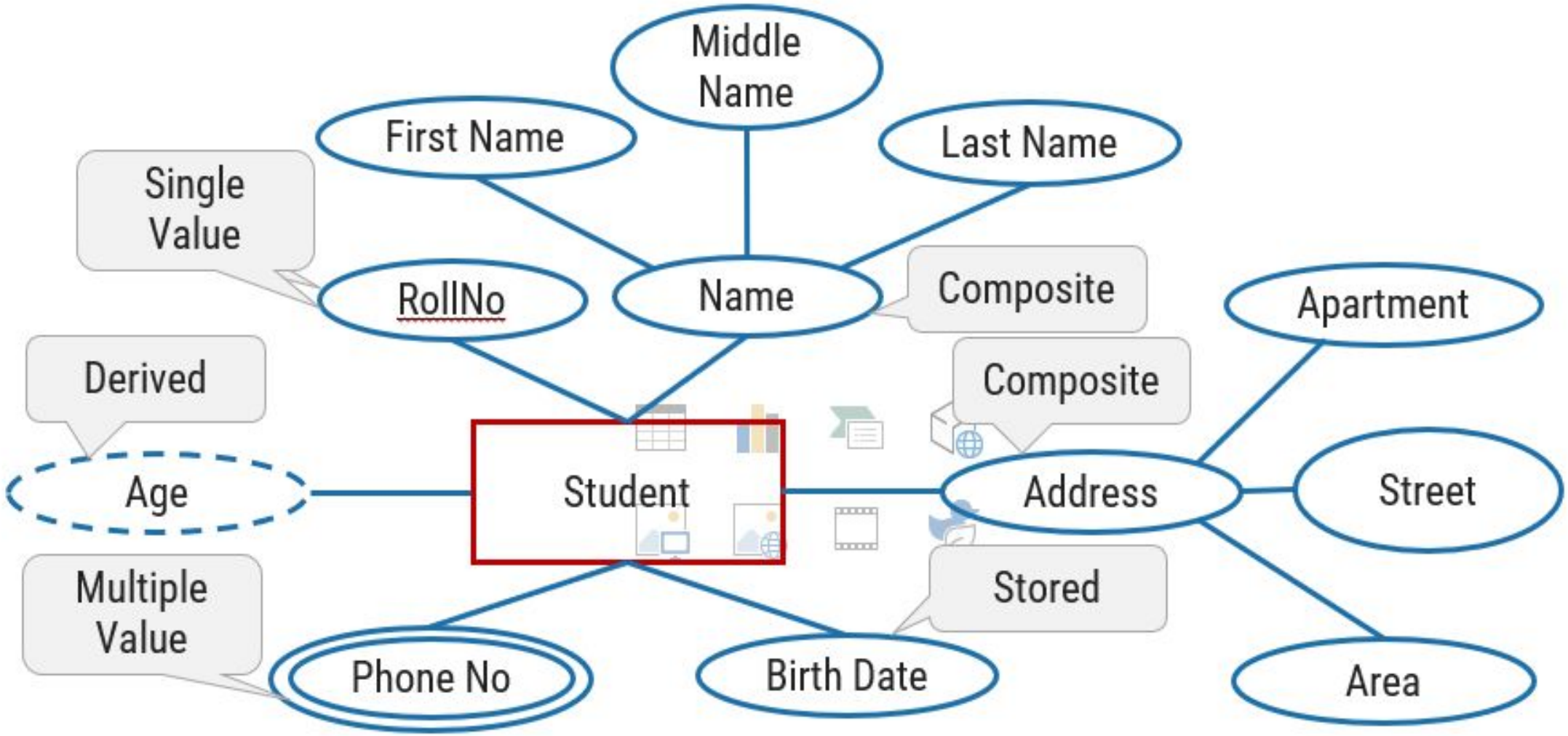
- Customer & Account
- Customer & Loan
- Doctor & Patient
- Student & Project
- Student & Teacher
  - Note: Take four attributes per entity with one primary key attribute.
    Keep proper relationship between two entities.

Type of attributes

- Single attribute
- composite attribute
- Multivalued attribute
- Derived attribute

Entity with all type of attributes



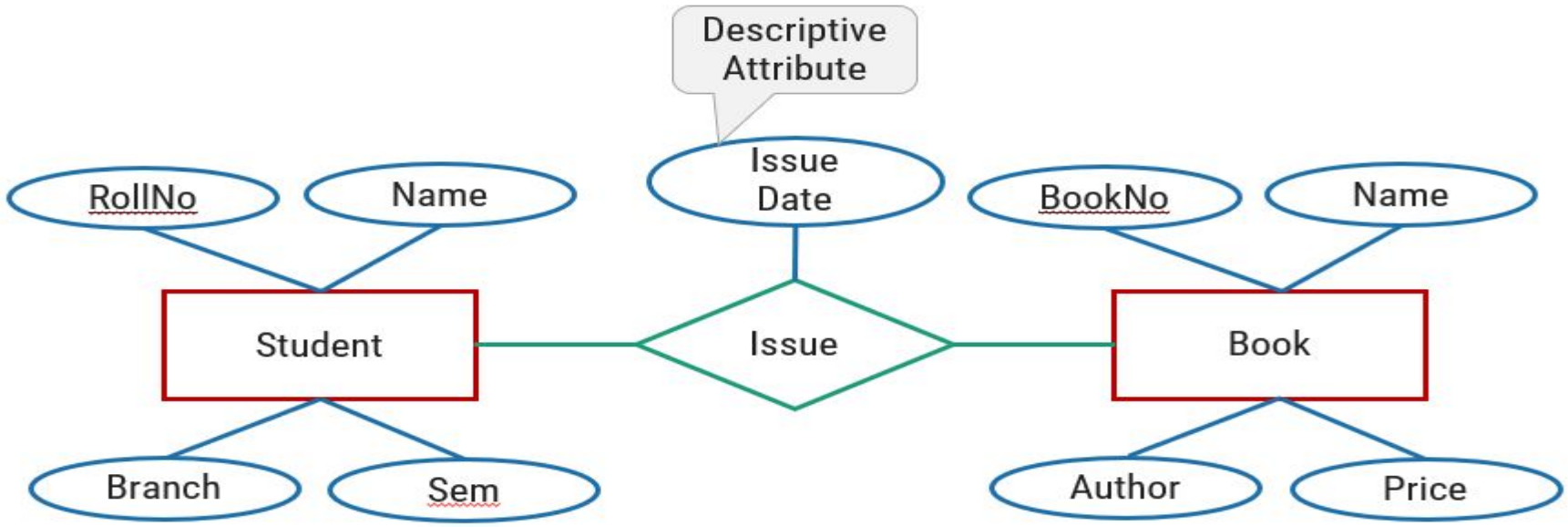

Middle Name
First Name
Last Name
Single Value
RollNo
Name
Composite
Apartment
Derived
Composite
Age
Student
Address
Street
Multiple Value
Stored
Phone No
Birth Date
Area

Exercises

- Draw an E-R diagram of Banking Management System.
- Draw an E-R diagram of Hospital Management System.
- Draw an E-R diagram of College Management System.
  - Take only 2 entities
  - Keep proper relationship between two entities
  - Use all types of attributes

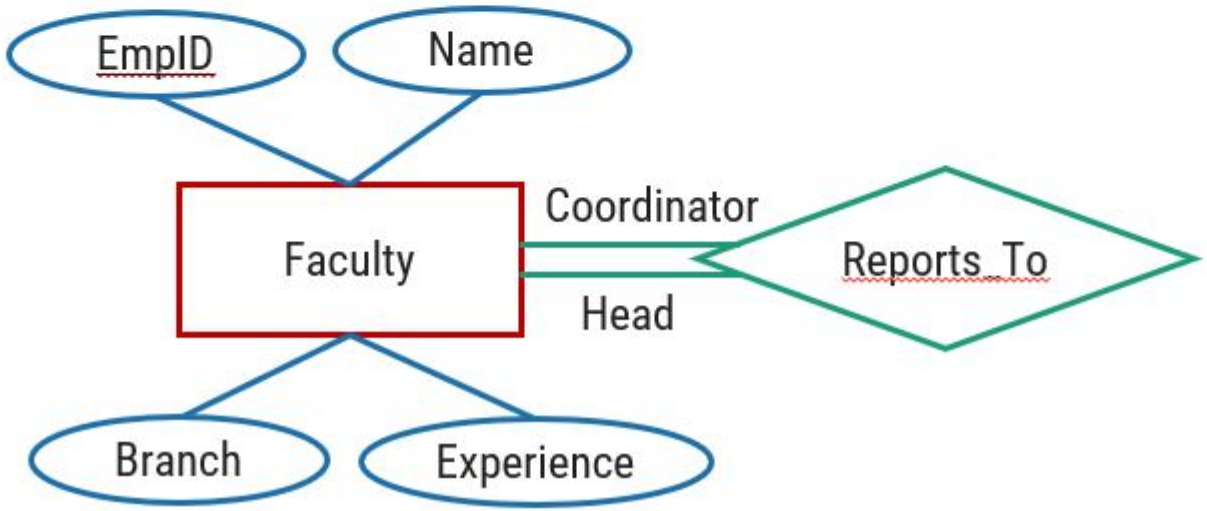
Descriptive attributes

- The attribute(s) used for describing the relationship is called descriptive attributes, also referred as relationship attributes.
- They are actually used for storing information about the relationship.
- A relationship can have zero or more attributes.



Role

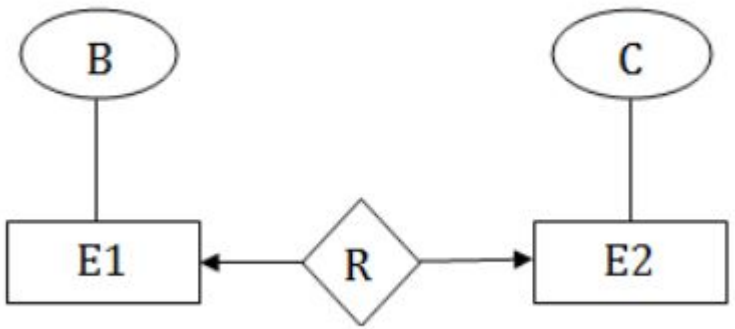
- Roles are indicated by labeling the lines that connect diamonds (relationship) to rectangles (entity).
- The labels “Coordinator” and “Head” are called roles; they specify Faculty entities interact with whom via Reports_To relationship set.
- Role labels are optional, and are used to clarify semantics (meaning) of the relationship.



Mapping cardinalities

- A mapping constraint is a data constraint that expresses the number of entities to which another entity can be related via a relationship set.
- It is most useful in describing the relationship sets that involve more than two entity sets.
- For binary relationship set R on an entity set A and B, there are four possible mapping cardinalities. These are as follows:
  - One to one (1:1)
  - One to many (1:M)
  - Many to one (M:1)
  - Many to many (M:M)

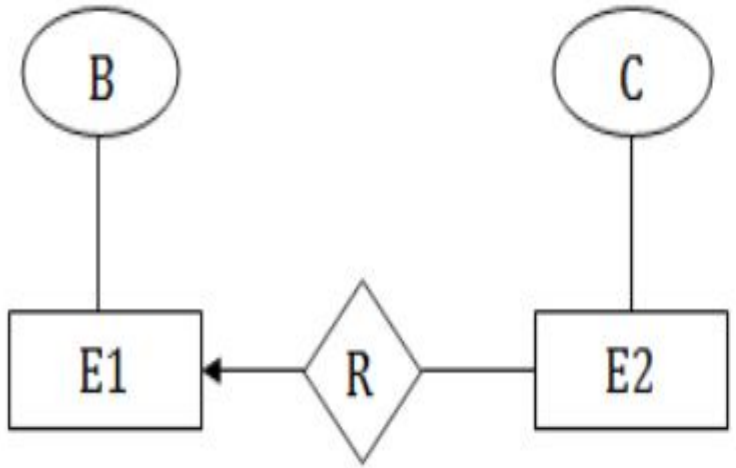
One to one



In one-to-one mapping, an entity in E1 is associated with at most one entity in E2, and an entity in E2 is associated with at most one entity in E1.

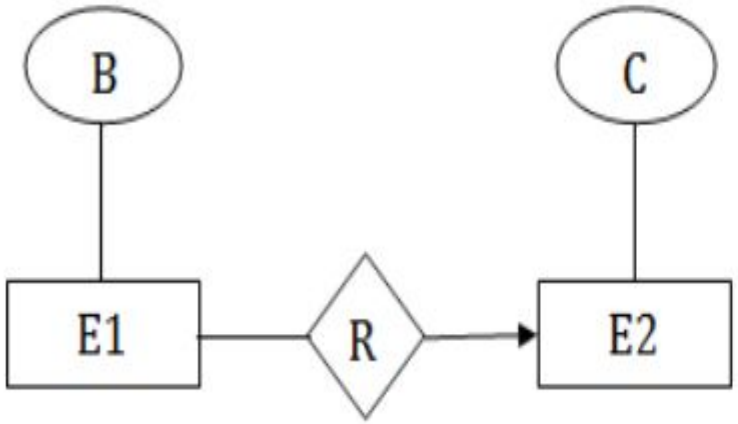
One to many

In one-to-many mapping, an entity in E1 is associated with any number of entities in E2, and an entity in E2 is associated with at most one entity in E1.

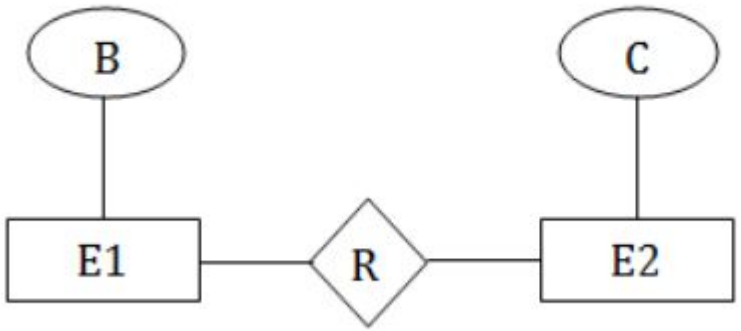


Many to one

In one-to-many mapping, an entity in E1 is associated with at most one entity in E2, and an entity in E2 is associated with any number of entities in E1.



Many to Many



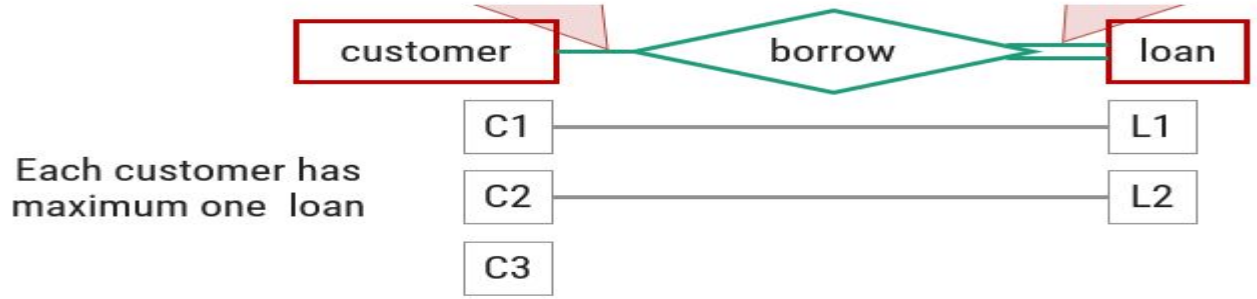
In many-to-many mapping, an entity in E1 is associated with any number of entities in E2, and an entity in E2 is associated with any number of entities in E1.

Participation constraint

- It specifies the **participation of an entity set** in a relationship set.
- There are two types participation constraints
  - Total participation
  - Partial participation

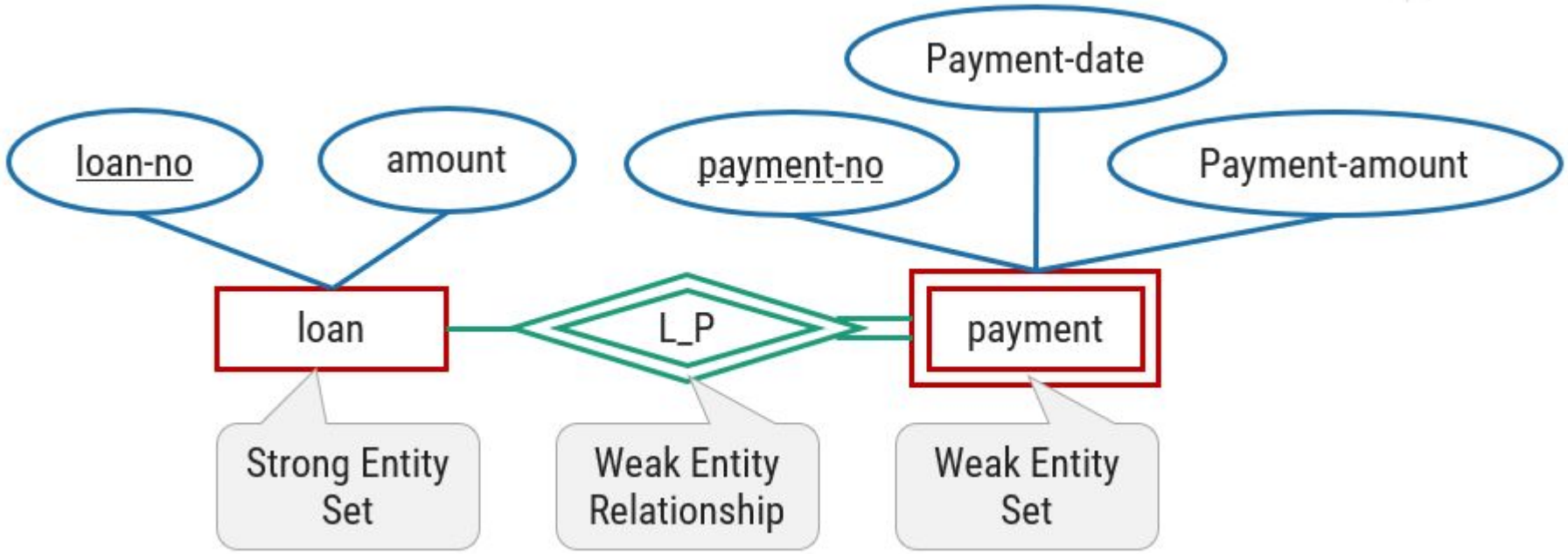
Partial participation
- some entities in the entity set may not participate in any relationship in the relationship set.
- indicated by single line

Total participation
- every entity in the entity set participates in at least one relationship in the relationship set.
- indicated by double line



Weak entity set

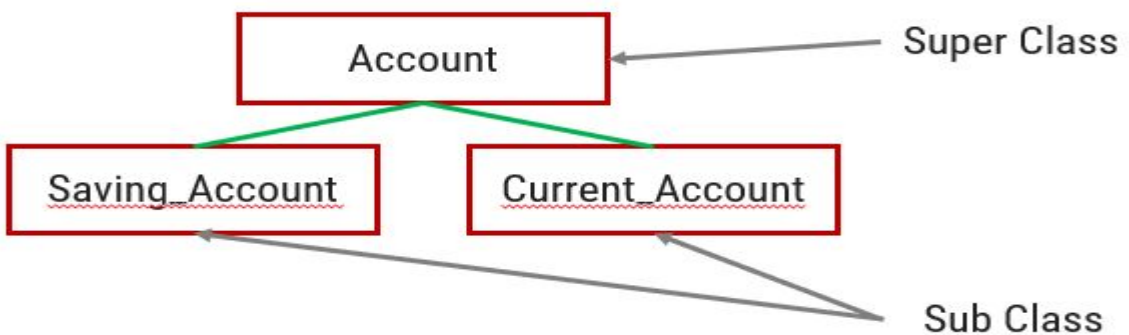
- A weak entity set is **an entity set that does not contain sufficient attributes to uniquely identify its entities**.
- A primary key does not exist for a weak entity set.



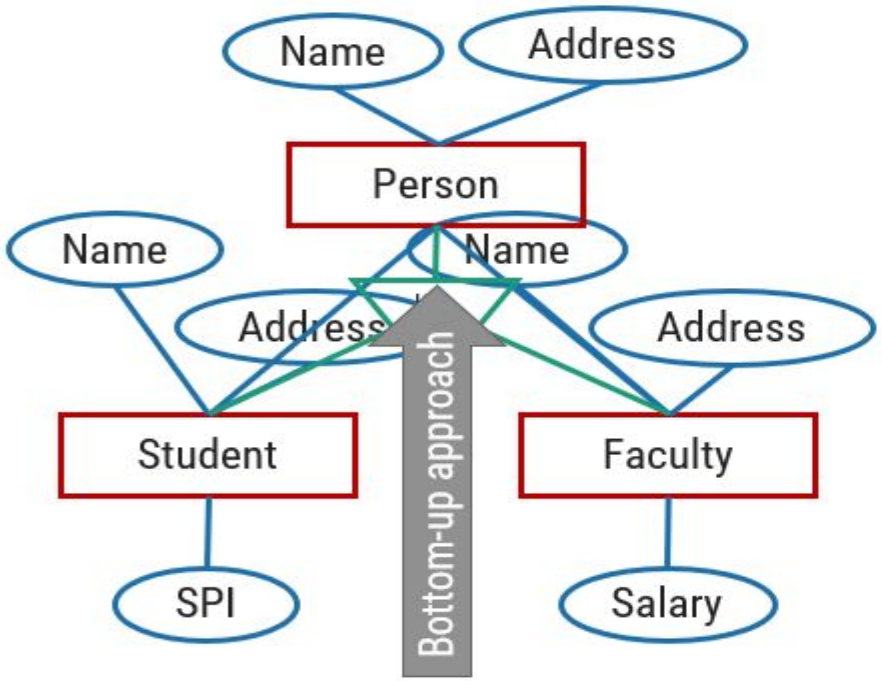
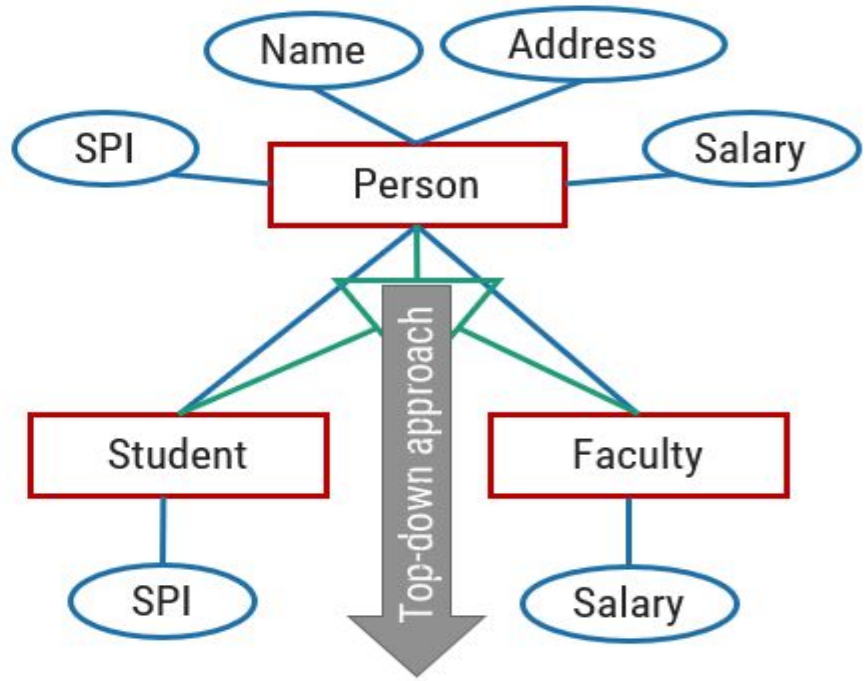
- The **existence of a weak entity set** depends on the **existence of a strong entity set**.
- The **discriminator (partial key)** of a weak entity set is the set of **attributes that distinguishes all the entities** of a weak entity set.
- The **primary key** of a weak entity set is created by **combining the primary key of the strong entity set** on which the weak entity set is existence dependent and the **weak entity set's discriminator**.
- We underline the discriminator attribute of a weak entity set with a **dashed line**.
- Payment entity has payment-no which is discriminator.
- Loan entity has loan-no as primary key.
- So primary key for payment is **(loan-no, payment-no).**

Superclass vs subclass

Super Class	Sub Class
A superclass is an entity from which **another entities can be derived**.	A subclass is an entity that is **derived from another entity**.
E.g, an entity account has two subsets saving_account and current_account So an **account is superclass**.	E.g, saving_account and current_account entities are derived from entity account. So **saving_account and current_account are subclass**.

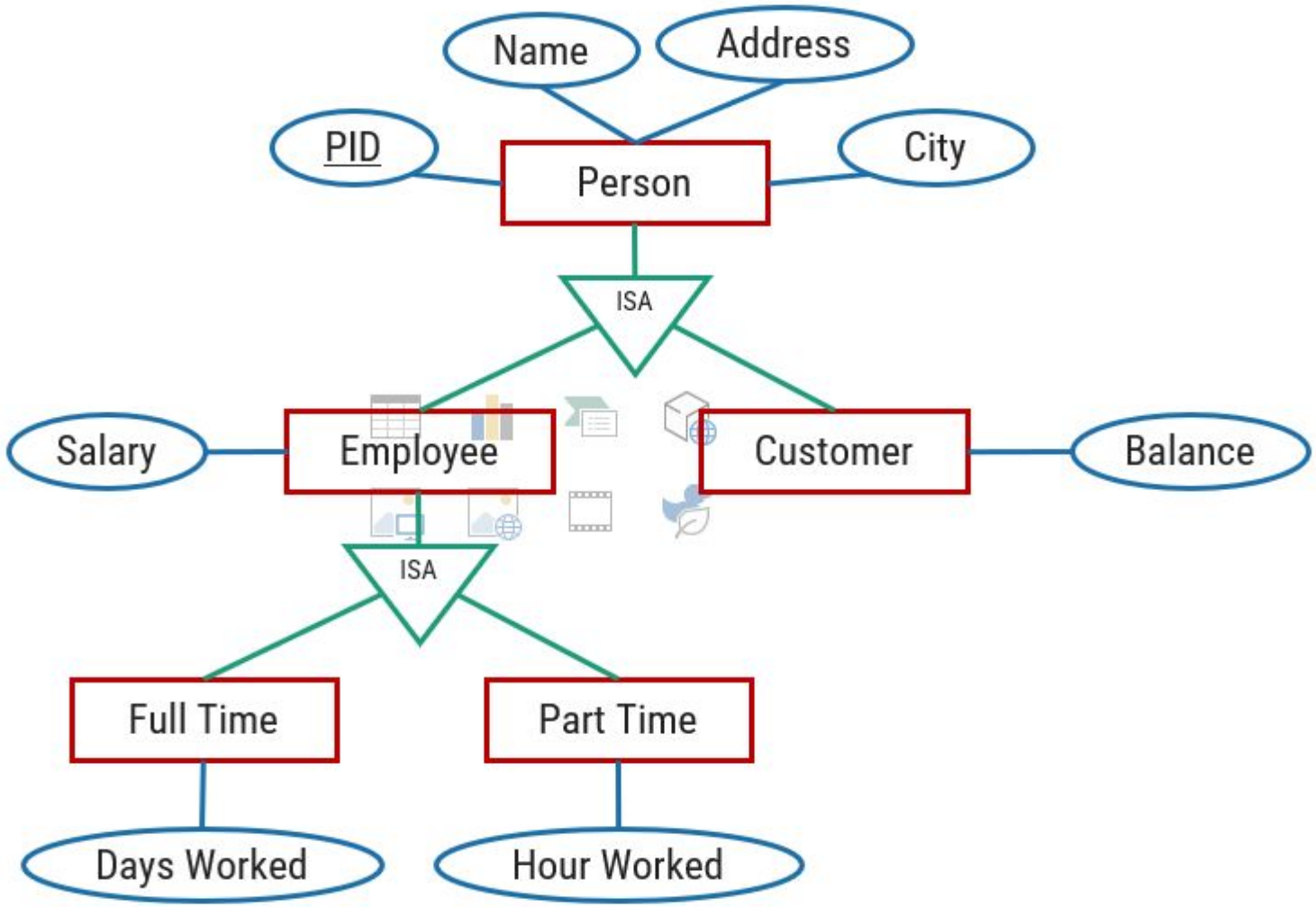


Generalization vs Specialization

Generalization	Specialization
It **extracts the common features** of **multiple entities** to **form a new entity**.	It **splits an entity to form multiple new entities** that **inherit some feature of the splitting entity**.
	

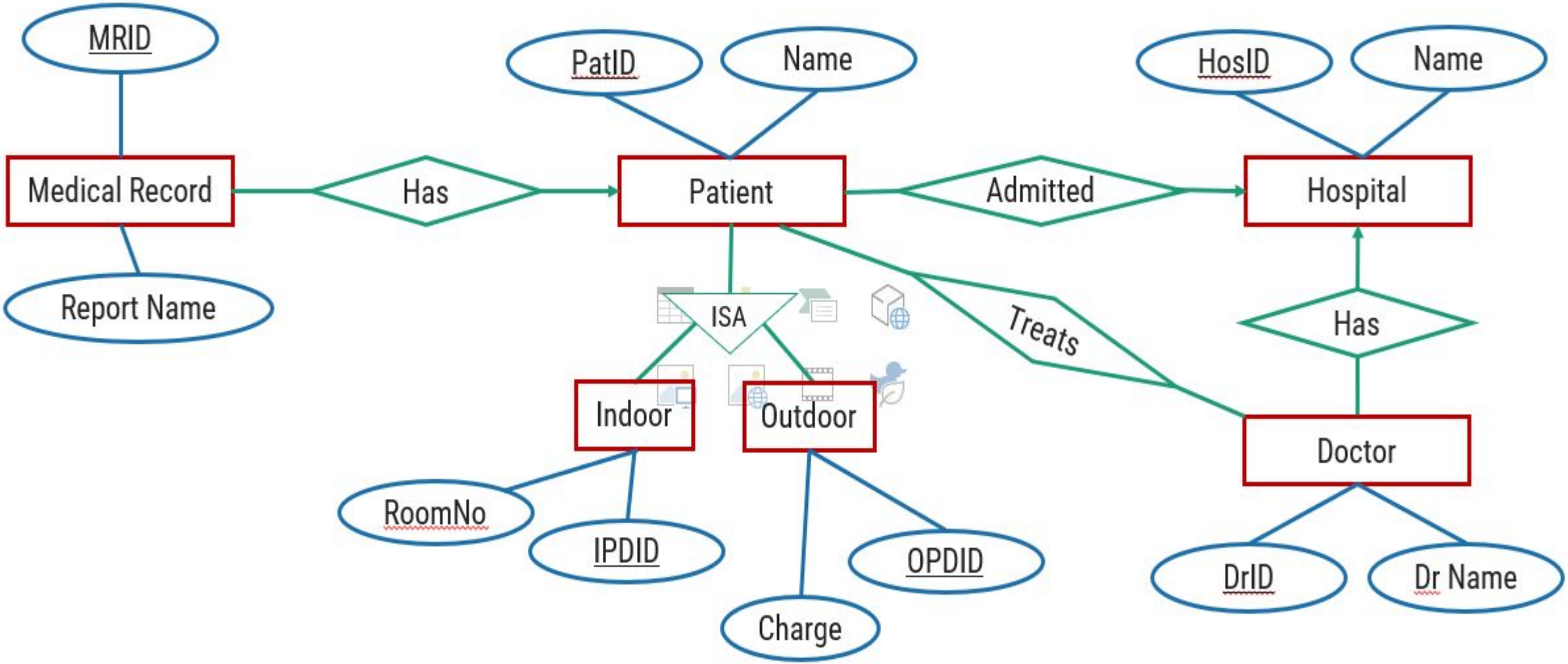
Generalization	Specialization
The process of **creation of group from various entities** is called generalization.	The process of **creation of sub-groups within an entity** is called specialization.
It is **Bottom-up** approach.	It is **Top-down** approach.
The process of taking the **union of two or more lower level entity** sets to produce a higher level entity set.	The process of taking a **sub set of higher level entity set** to form a lower level entity set.
It starts from the number of entity sets and creates high level entity set using some common features.	It starts from a single entity set and creates different low level entity sets using some different features.

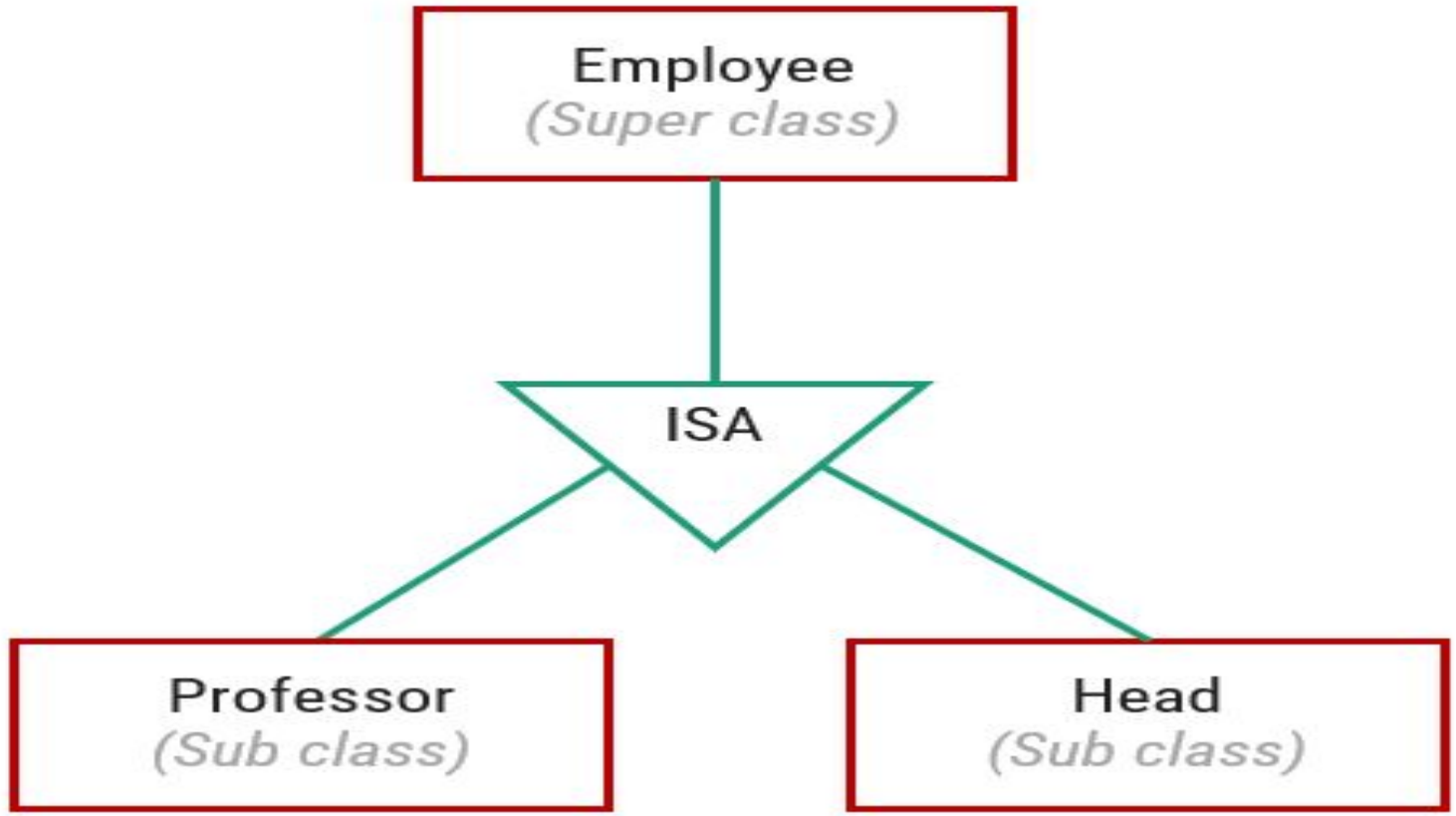
Example


Name
Address
PID
City
Person
ISA
Salary
Employee
Customer
Balance
ISA
Full Time
Part Time
Days Worked
Hour Worked

Example

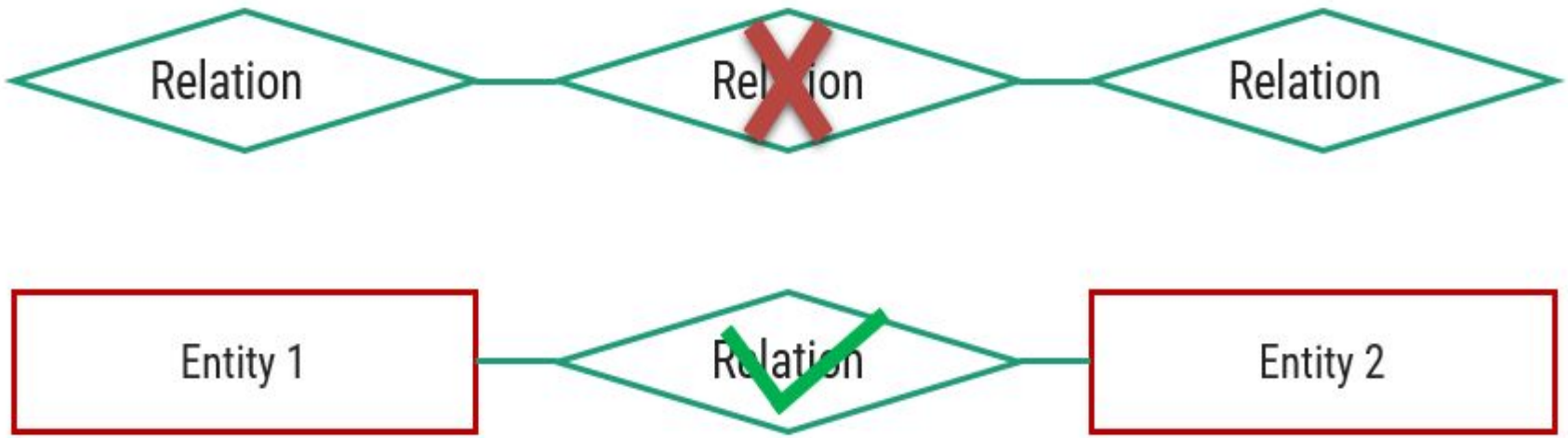
- Give the examples of Generalization/Specialization in the following E-R diagram:
  - Hospital Management System.
  - College Management System.
  - Bank Management System.
  - Insurance Company.


MRID
PatID
Name
HosID
Name
Medical Record
Has
Patient
Admitted
Hospital
Report Name
ISA
Treats
Has
Indoor
Outdoor
Doctor
RoomNo
IPDID
OPDID
Charge
DrID
Dr Name


Employee
(Super class)
ISA
Professor
(Sub class)
Head
(Sub class)

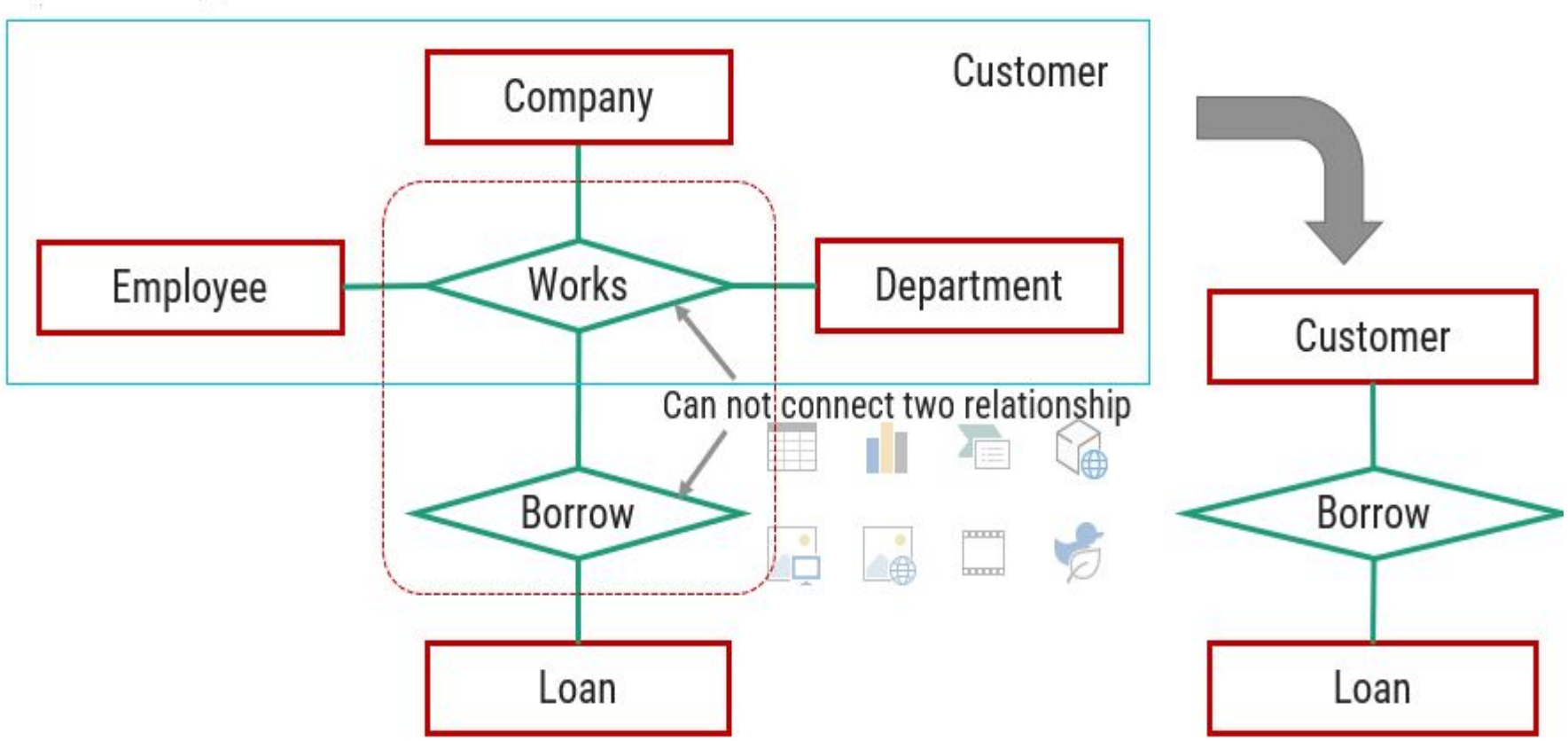
Limitation of ER diagram

- In E-R model we **cannot express relationships between two relationships**.



Aggregation

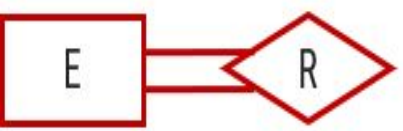
- Process of creating an entity by combining various component of E-R Diagram is called aggregation



Entity

Primary Key Attribute

Weak Entity

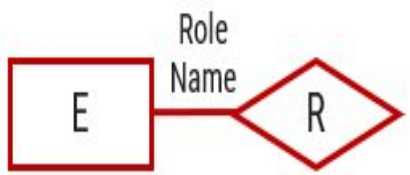


Total Participation

Attribute

Derived Attribute

Discriminating Attribute



Role Indicator

Relationship

Multi Valued Attribute

Weak Entity Relationship

Specialization/ Generalization

One to One

One to Many

Many to One

Many to Many

Type of models

- Hierarchical model (1 to M)
- Network Model (M to M)
- ER model
- Relational Model
- Object oriented database Model

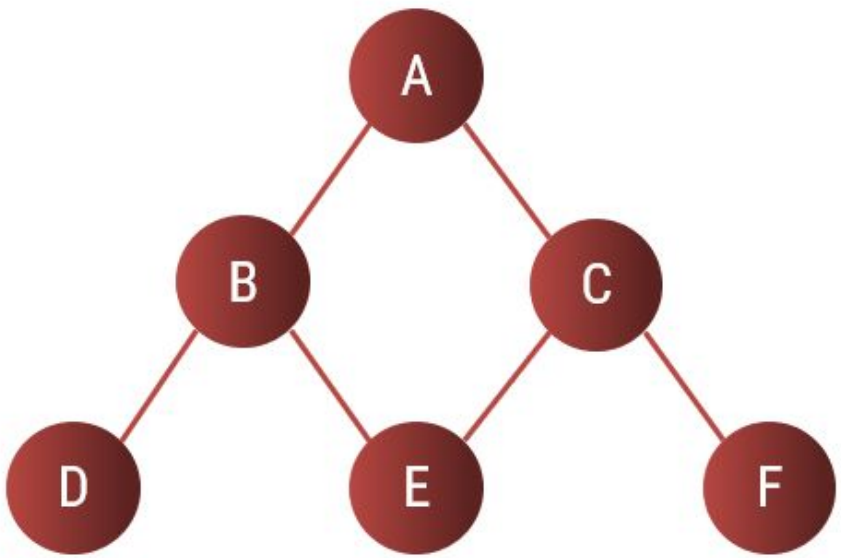
Hierarchical Model

- The hierarchical model organizes data into a **tree-like structure**, where **each record has a single parent or root**.

- The hierarchy **starts from the Root data**, and **expands like a tree**, **adding child nodes to the parent nodes**.
- In hierarchical model, data is organized into **tree-like structure** with **one-to-many relationship** between two different types of data, for example, **one department can have many professors and many students**.

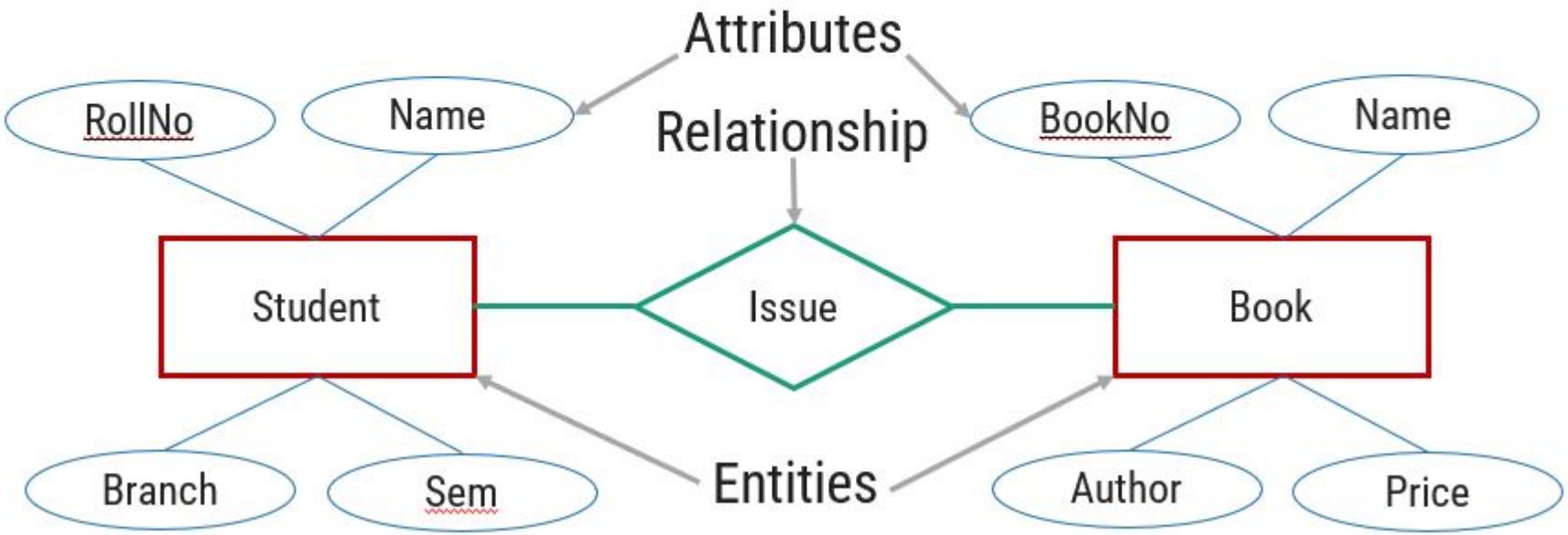
Network Model

- This is an **extension of the hierarchical model**, allowing **many-to-many relationships** in a tree-like structure that **allows multiple parents**.



Entity Relationship Model

- In this database model, **relationships are created by dividing object of interest into entity** and **its characteristics into attributes**.



Relational Model

- In this model, **data is organized in two-dimensional tables** and the **relationship is maintained by storing a common attribute**.

Rno	Student_Name	Age
101	Raj Patel	20
102	Meet Shah	21

SubID	Subject_Name	Teacher
1	DBMS	Doshi
2	DS	Vyash

Foreign Key

Foreign Key

ResID	Rno	SubID	Marks
1	101	1	80
2	101	2	85
3	102	1	75
4	102	2	80

Object Oriented database Model

- This data model is another method of representing real world objects.
- It considers **each object in the world as objects** and isolates it from each other.
- It **groups its related functionalities together** and **allows inheriting its functionality** to other related sub-groups.

Integrity Constraints

- Integrity constraints are a **set of rules**. It is used to **maintain the quality** of information.
- Integrity constraints ensure that the data insertion, updating, and other processes have to be performed in such a way that data integrity is not affected.
- Thus, integrity constraint is used to **guard against accidental damage** to the database.
- Various Integrity Constraints are:
  - Check
  - Not null
  - Unique
  - Primary key
  - Foreign key

- Check
  - This constraint defines a business rule on a column. All the rows in that column must satisfy this rule.
  - Limits the data values of variables to a **specific set, range, or list of values**.
  - The constraint can be applied for a single column or a group of columns.
  - E.g. value of SPI should be between 0 to 10.
- Not null
  - This constraint ensures all rows in the table contain a definite value for the column which is specified as not null. Which means a **null value** is not allowed.
  - E.g. name column should have some value.
- Unique
  - This constraint ensures that a column or a group of columns in each row have a **distinct (unique)** value.
  - A column(s) can have a null value but the values cannot be duplicated.
  - E.g. enrollmentno column should have unique value.