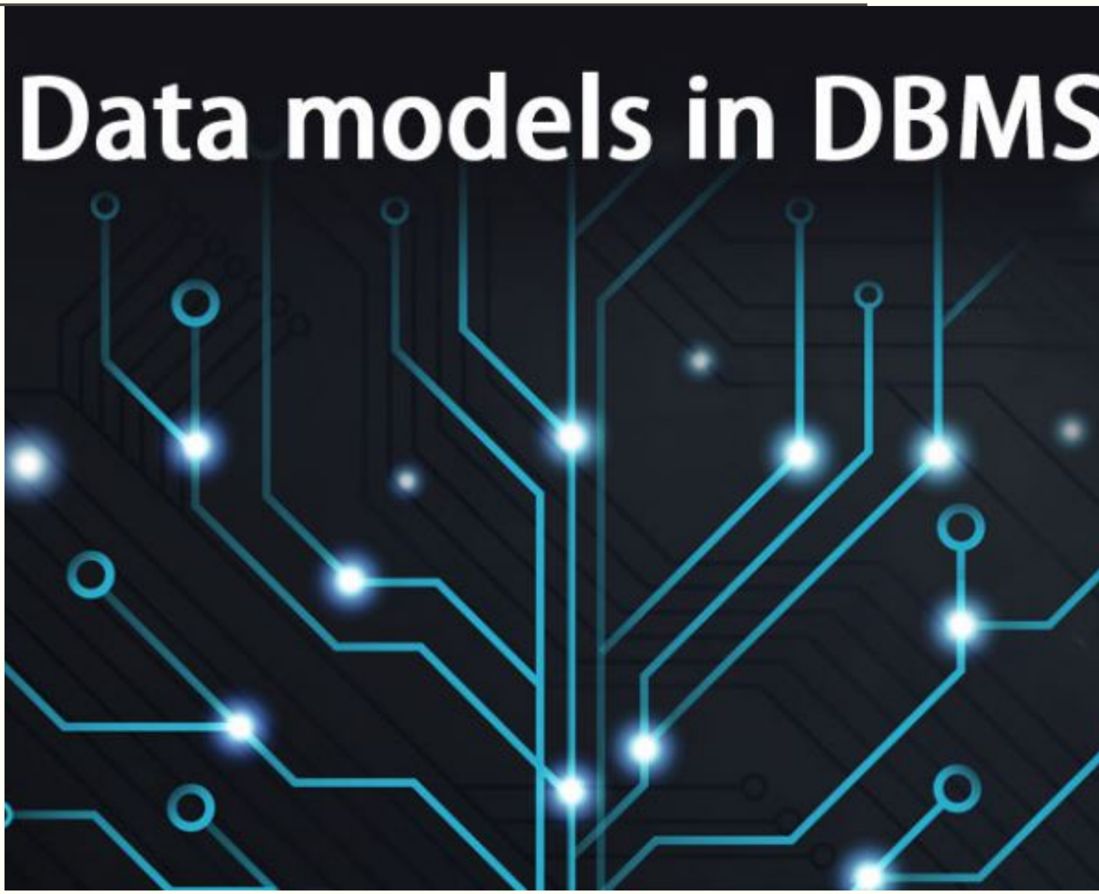
DATABASE MANAGEMENT SYSTEM

Unit - 2

Data Models



Prepared by
Prof. Rutika Patel

Basic Concept of E-R Diagram

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)

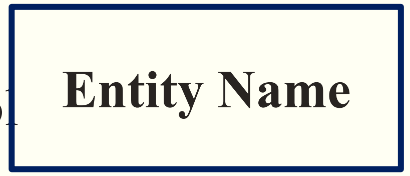
It is graphical (pictorial) representation of database.

It uses different types of symbols to represent different objects of database.

Entity

- An entity is **a person, a place** or **an object**.
- An entity is represented by a **rectangle** which contains the name of an entity.

Symbol



 Entities of a college database are:

- Student, Professor/Faculty, Course, Department, Class, Subject

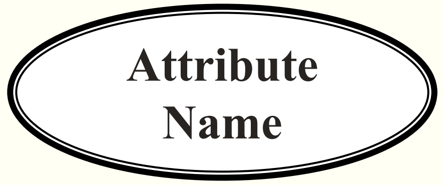
Student
Faculty
Course

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

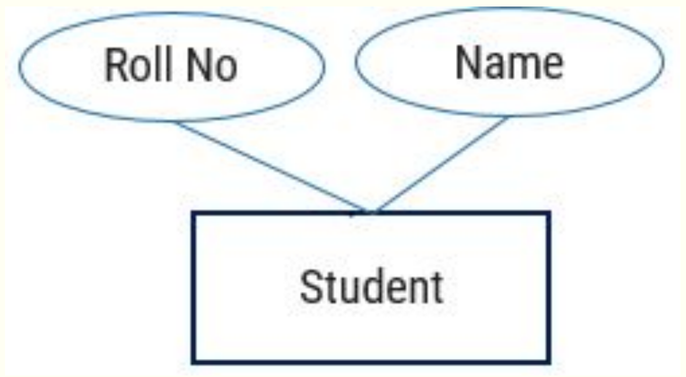
Attributes

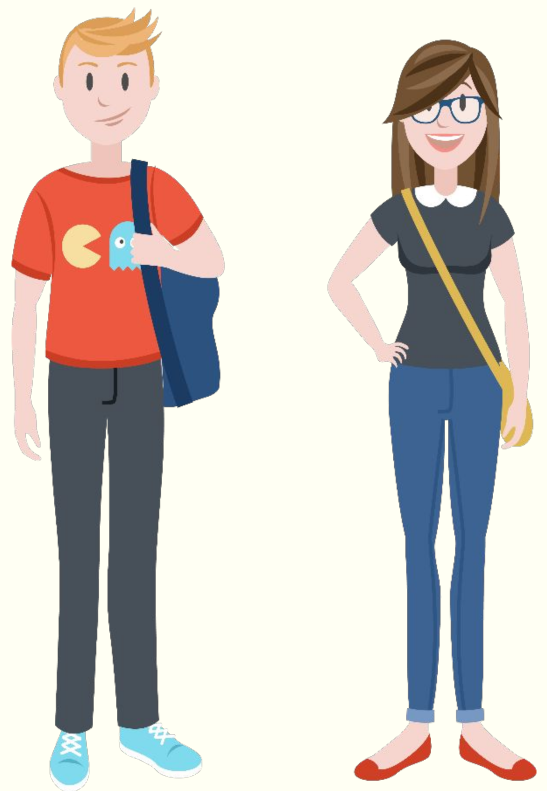
- Attribute is **properties** or details about an entity.
- An attribute is represented by an **oval** containing name of an attribute.



Symbol

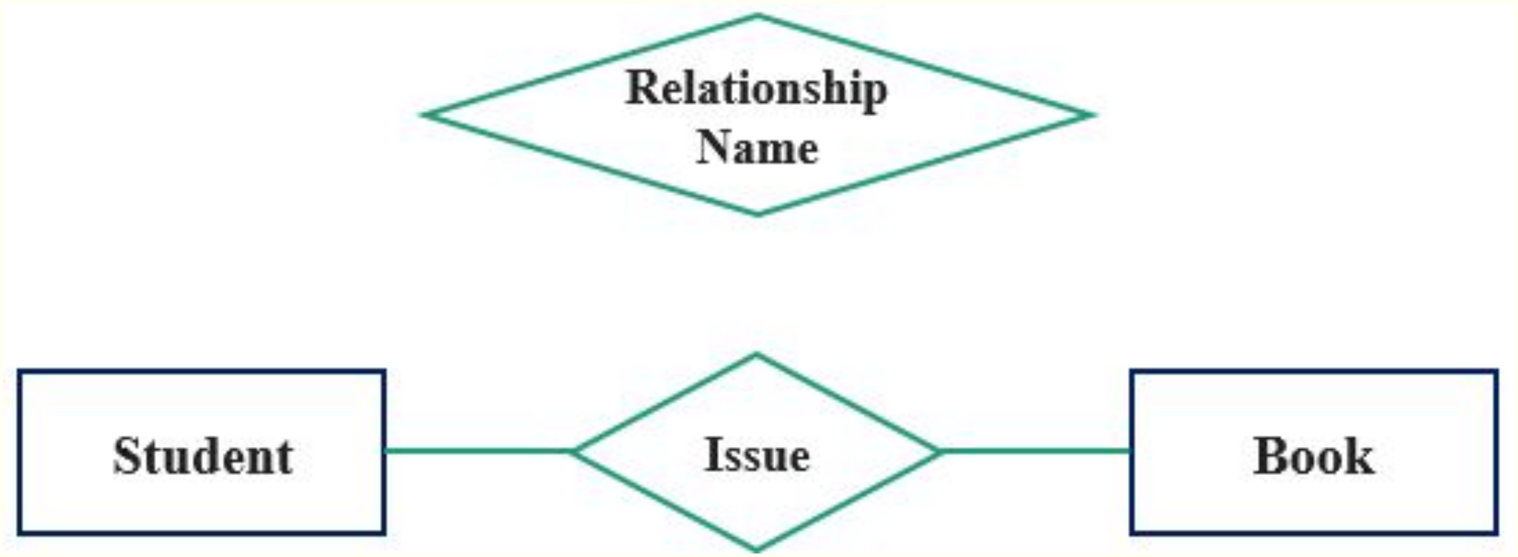
- Attributes of Student are:
  - Roll No
  - Student Name
  - Branch
  - Semester
  - Address
  - Mobile No
  - Age
  - SPI
  - Backlogs



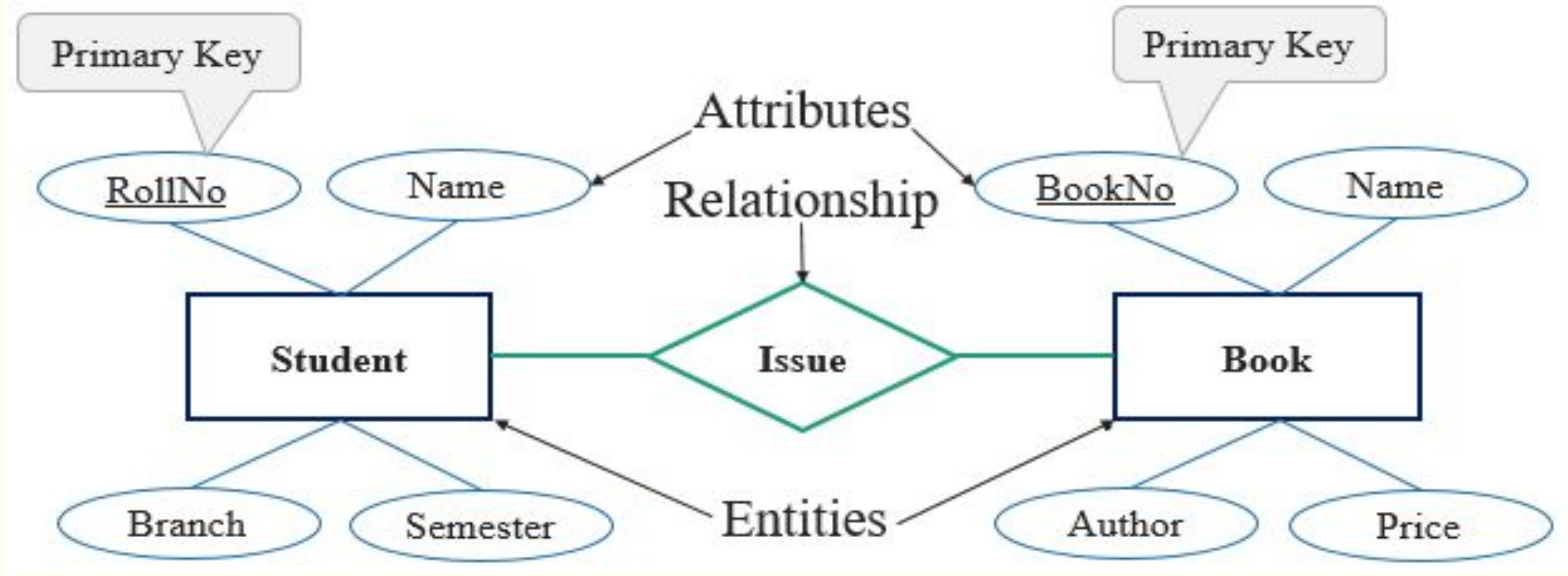


Relationship

- 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.

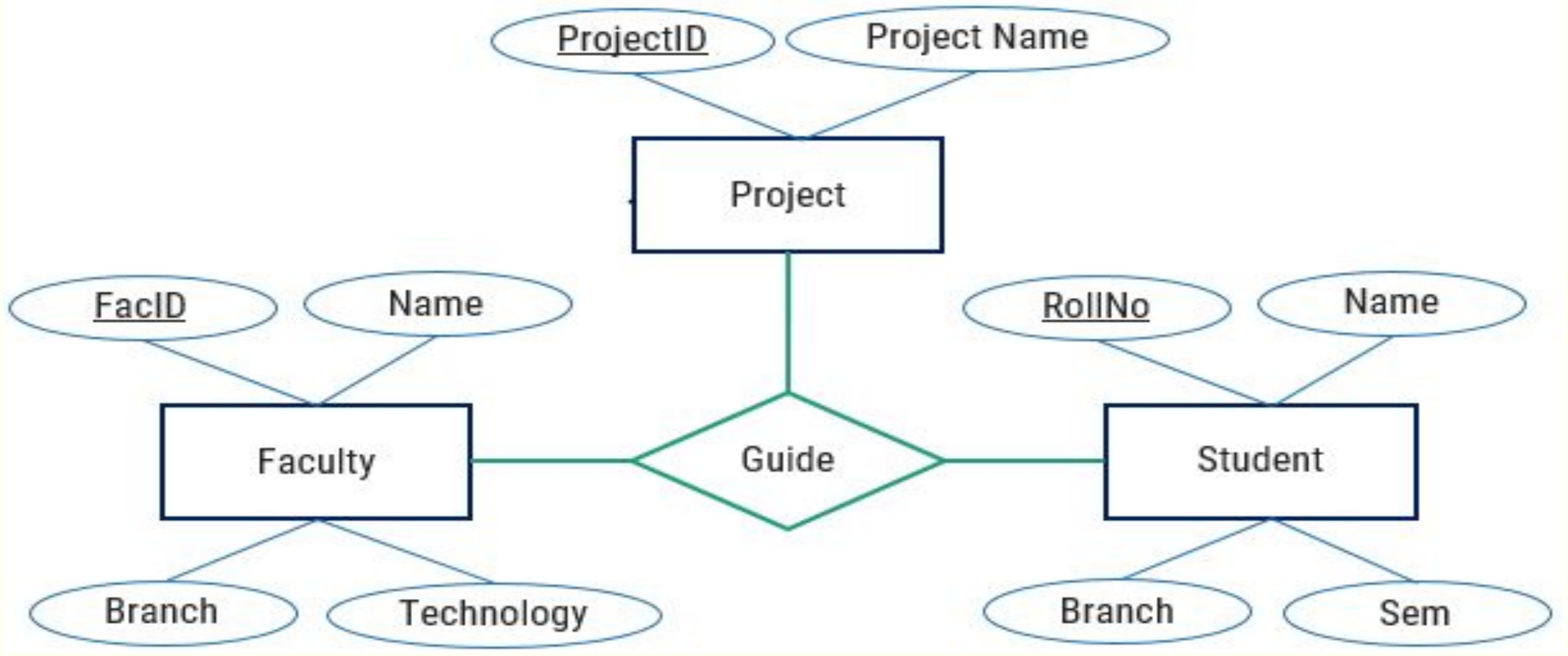


E-R Diagram of a Library System



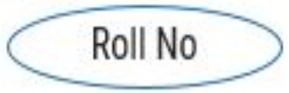
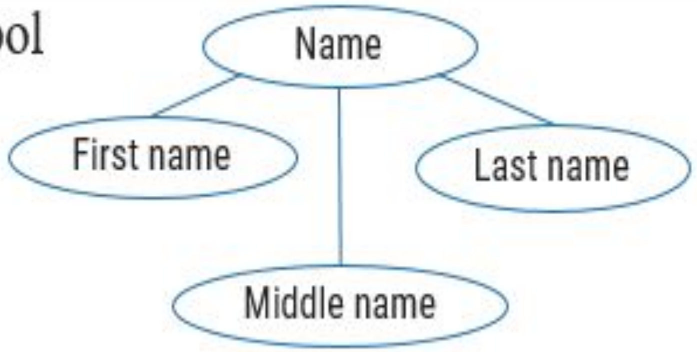
Each and every entity must have one primary key attribute.
Relationship between 2 entities is called binary relationship.

Ternary Relationship

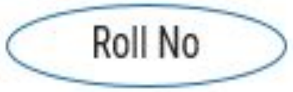
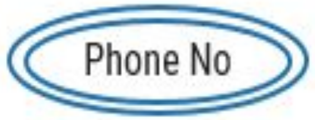


Relationship between 3 entities is called ternary relationship.

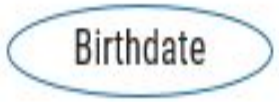
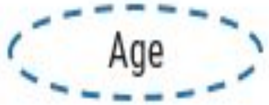
Types of Attributes

Simple Attribute	Composite Attribute
Cannot be divided into subparts	Can be divided into subparts
E.g. RollNo, CPI	E.g. Name (first name, middle name, last name) Address (street, road, city)
Symbol  	Symbol 

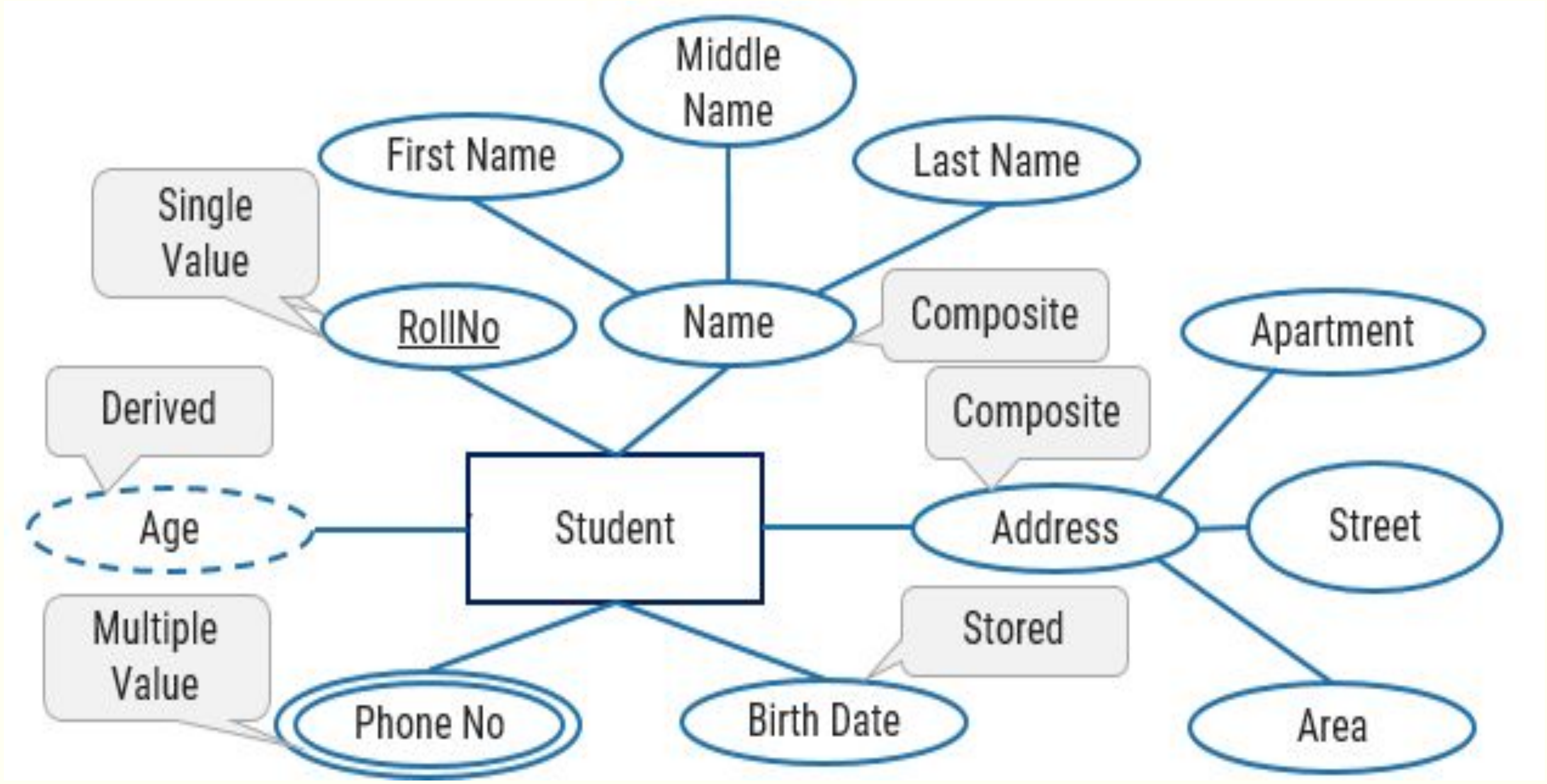
Cont..

Single-valued Attribute	Multi-valued Attribute
Has single value	Has multiple (more than one) value
E.g. RollNo, CPI	E.g. PhoneNo (person may have multiple phone nos) EmailID (person may have multiple emails)
Symbol  	Symbol 

Cont..

Stored Attribute	Derived Attribute
It's value is stored manually in database	It's value is derived or calculated from other attributes
E.g. Birthdate	E.g. Age (can be calculated using current date and birthdate)
Symbol 	Symbol 

Entity with all types of Attributes

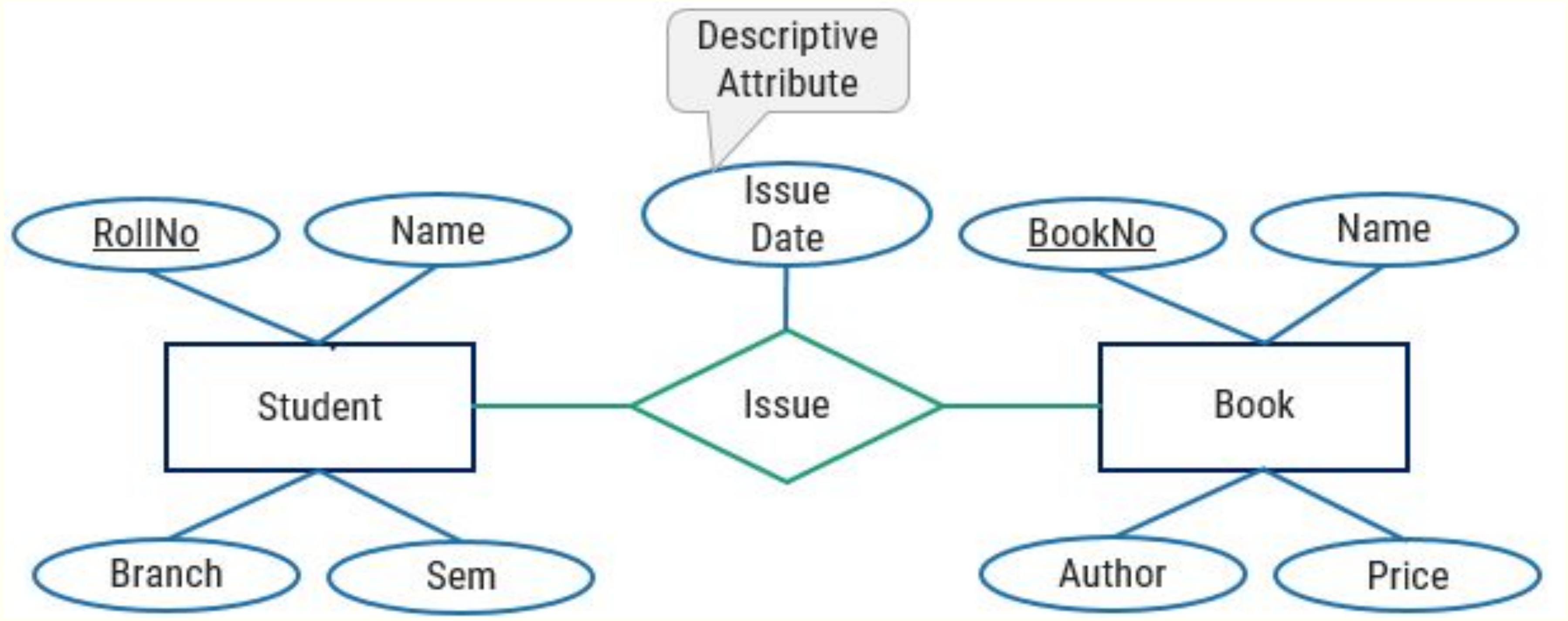


Exercise

- 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.

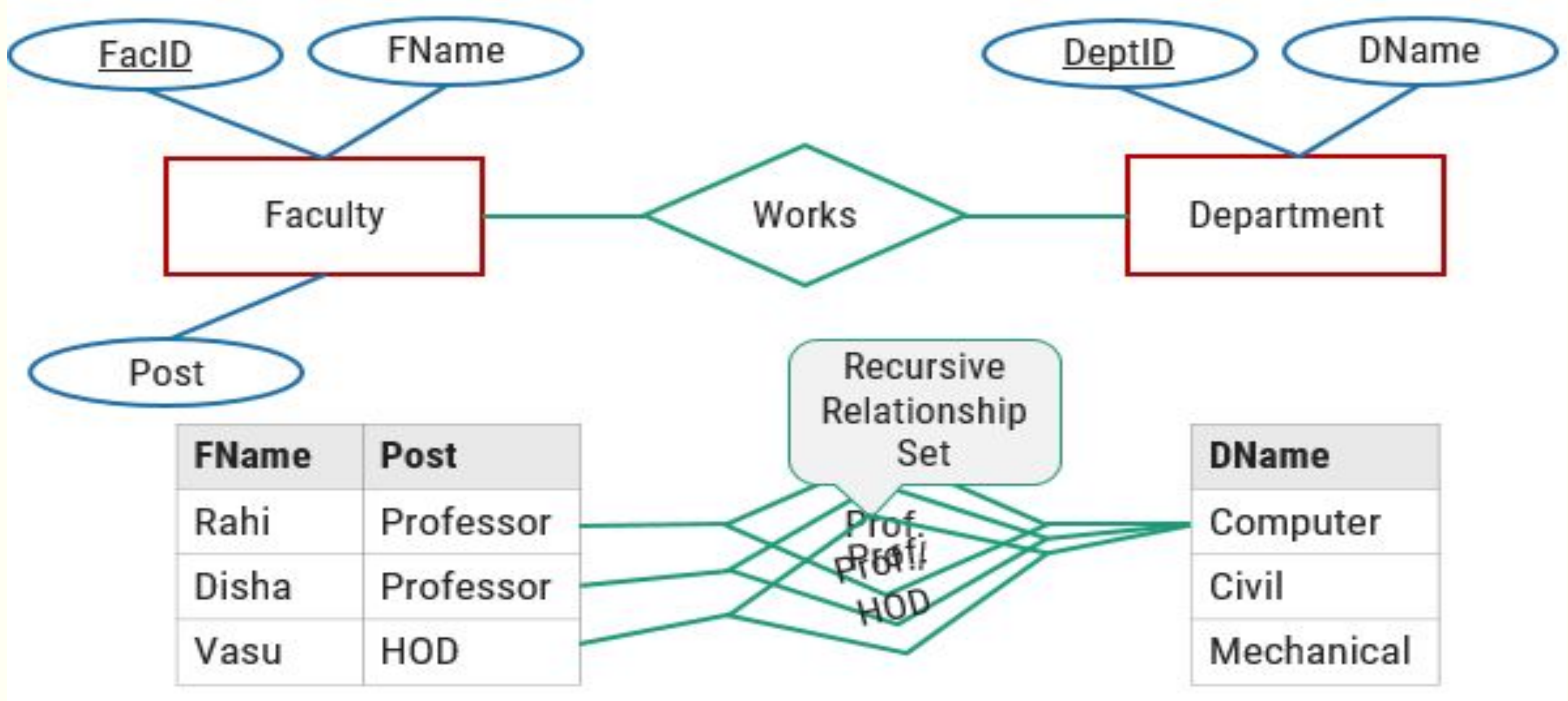
Descriptive Attribute

- **Attributes of the relationship** is called descriptive attribute.



Recursive Relationship Set

- The same **entity participates in a relationship set more than once** then it is called recursive relationship set.



FName	Post
Rahi	Professor
Disha	Professor
Vasu	HOD

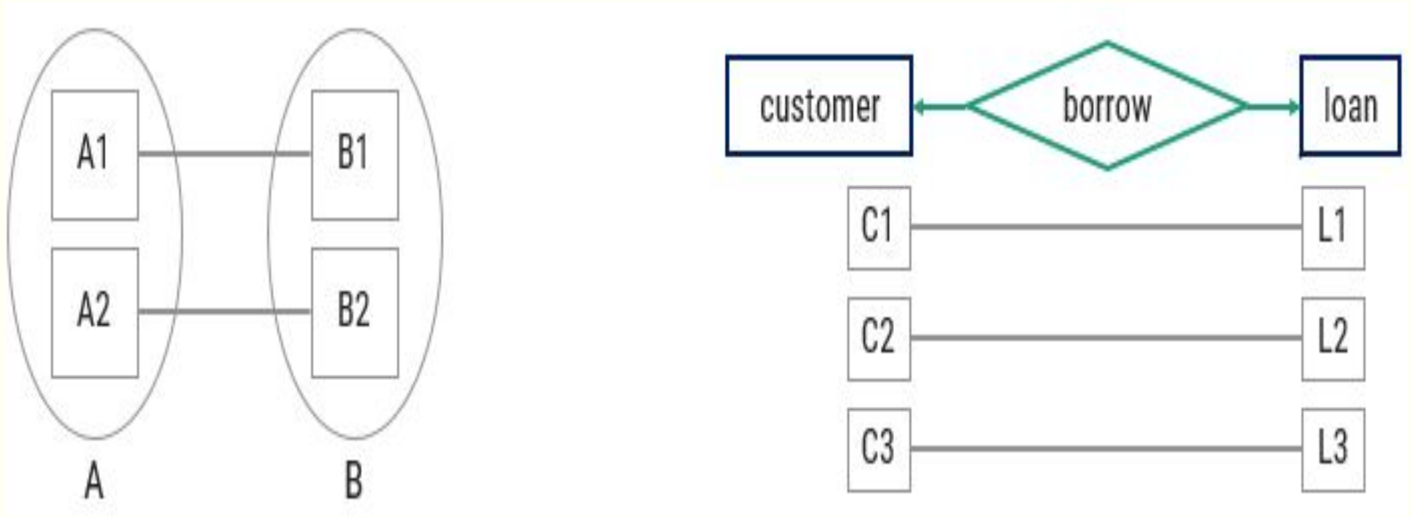
DName
Computer
Civil
Mechanical

Mapping Cardinality (Cardinality Constraints)

- It represents the **number of entities of another entity set** which are **connected to an entity** using a relationship set.
- It is most **useful in describing binary relationship sets**.
- For a binary relationship set the mapping cardinality must be one of the following types:
  - One to One
  - One to Many
  - Many to One
  - Many to Many

One-to-One relationship (1 – 1)

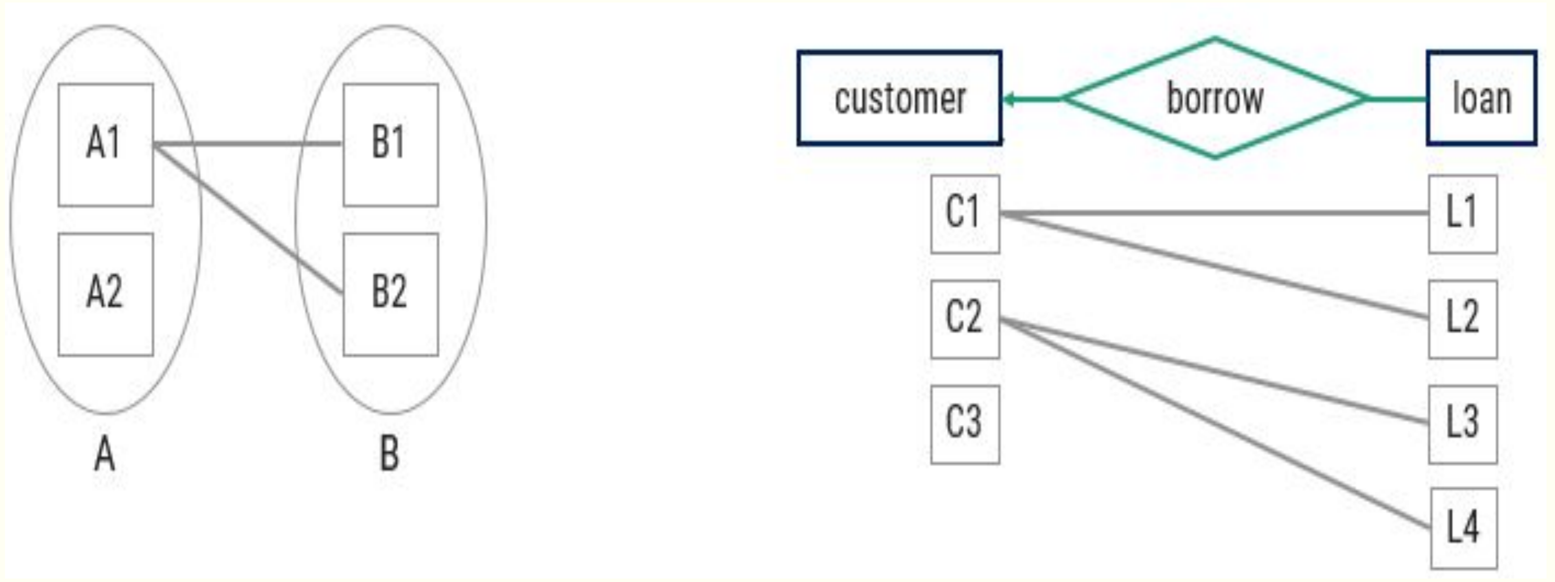
- An entity in A is associated with only one entity in B and an entity in B is associated with only one entity in A.



- Example: A customer is connected with only one loan using the relationship borrower and a loan is connected with only one customer using borrower.

One-to-Many relationship (1 – N)

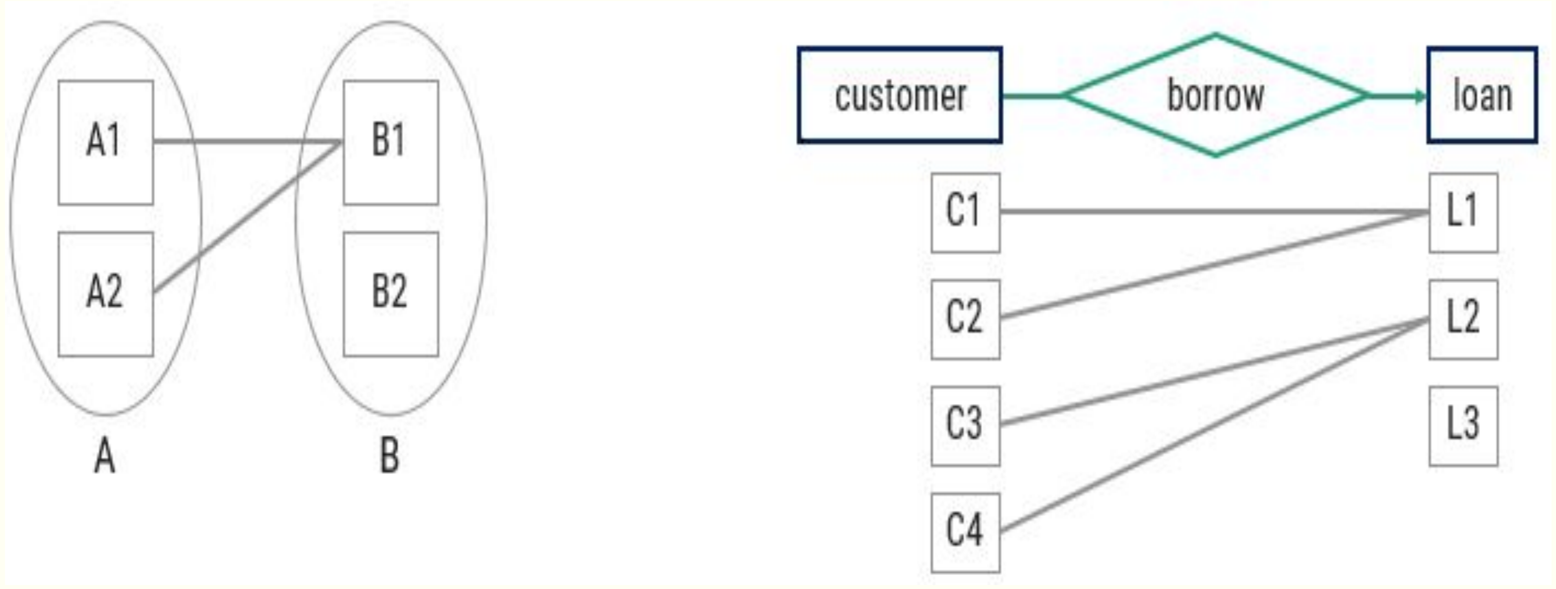
- An entity in **A is associated with more than one entities in B** and an entity in **B is associated with only one entity in A**.



- Example: A **loan is connected with only one customer** using borrower and a **customer is connected with more than one loans using borrower**.

Many-to-One relationship (N – 1)

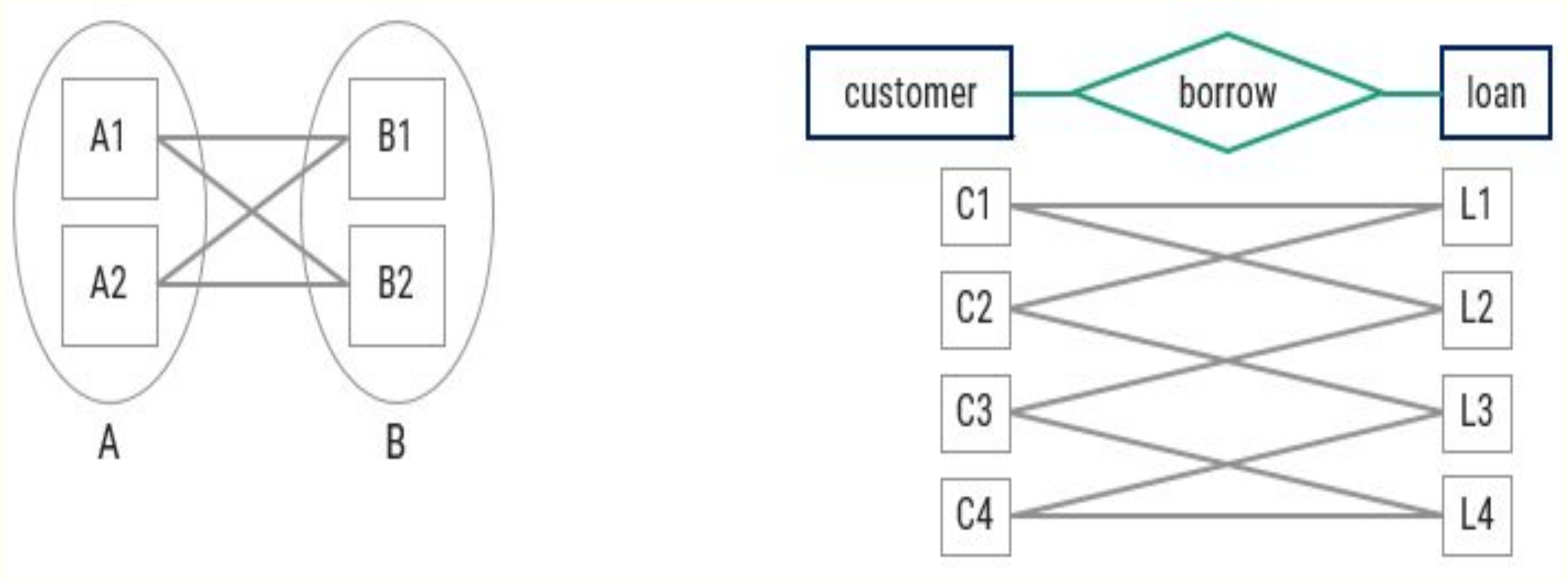
- An entity in **A is associated with only one entity in B** and an entity in **B is associated with more than one entities in A**.



- Example: A **loan is connected with more than one customer** using borrower and a **customer is connected with only one loan** using borrower.

Many-to-Many relationship (N – N)

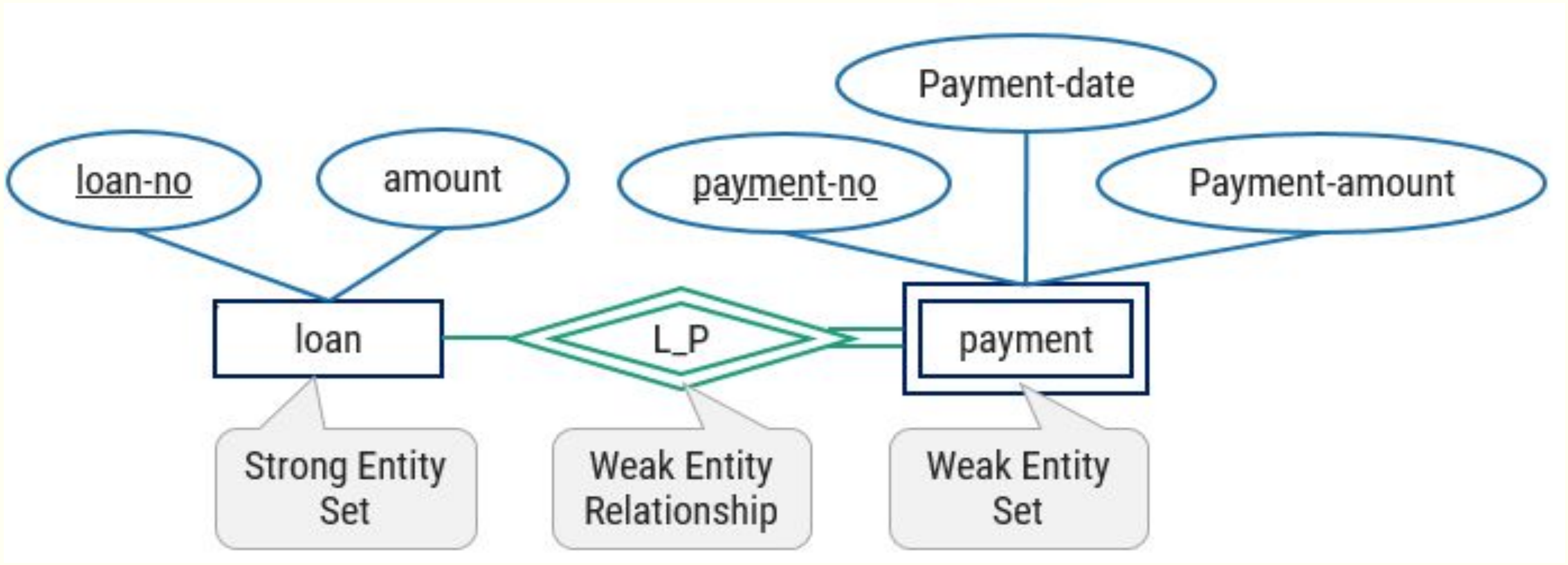
- An entity in **A is associated with more than one entities in B** and an entity in **B is associated with more than one entities in A**.



- Example: A **customer is connected with more than one loan** using borrower and a **loan is connected with more than one customer** using borrower.

Weak Entity Set

- An **entity set that does not have a primary key** is called weak entity set.



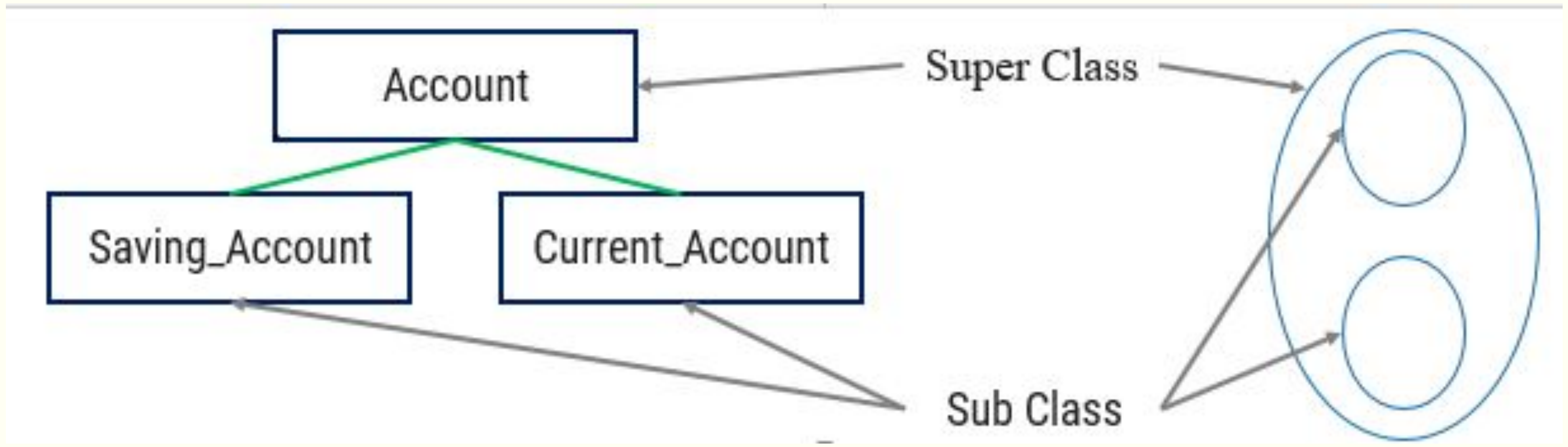
- Weak entity set is indicated by double rectangle.
- Weak entity relationship set is indicated by double diamond.

Cont..

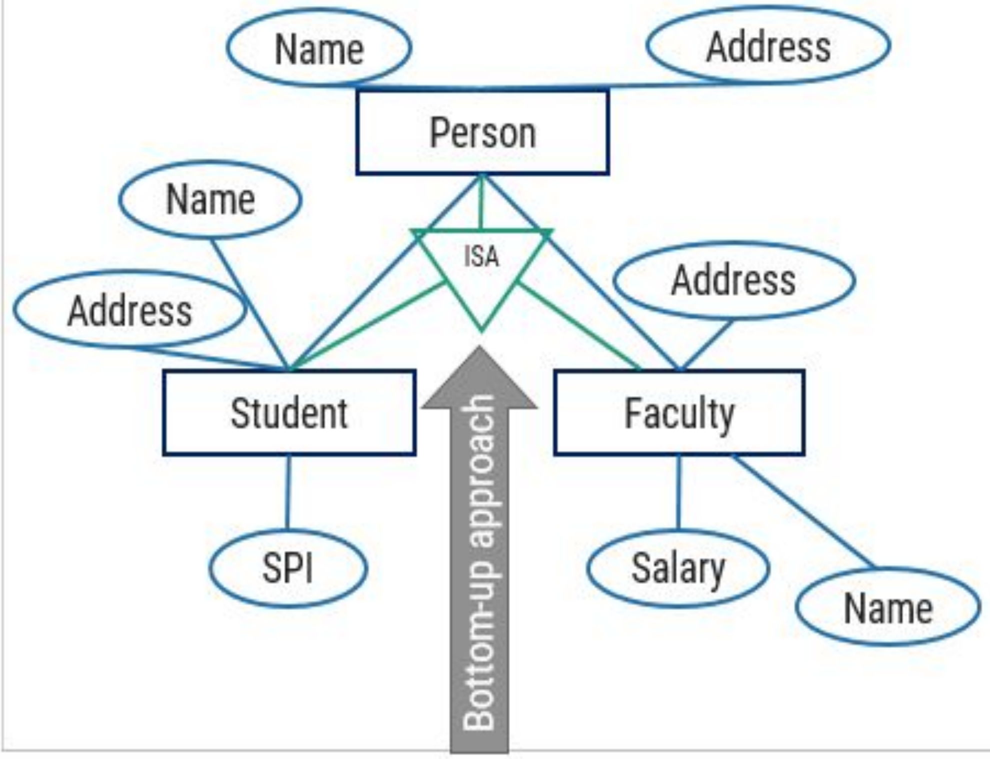
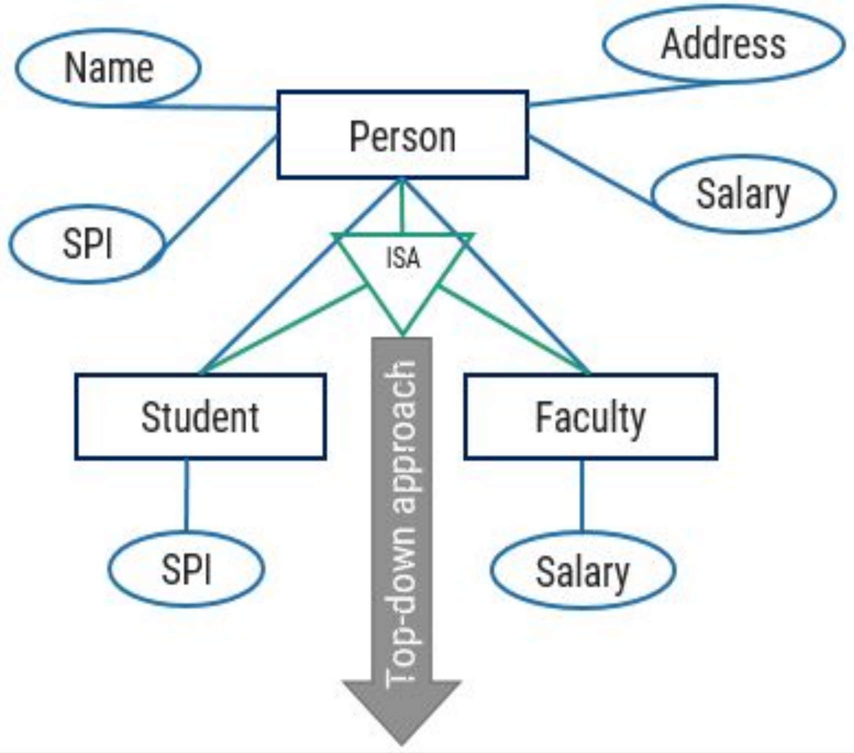
- The **existence of a weak entity set** depends on the **existence of a strong entity set**.
- The **discriminator (partial key)** of a weak entity set is the set of **attributes that distinguishes all the entities** of a weak entity set.
- The **primary key** of a weak entity set is created by **combining 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 v/s Subclass

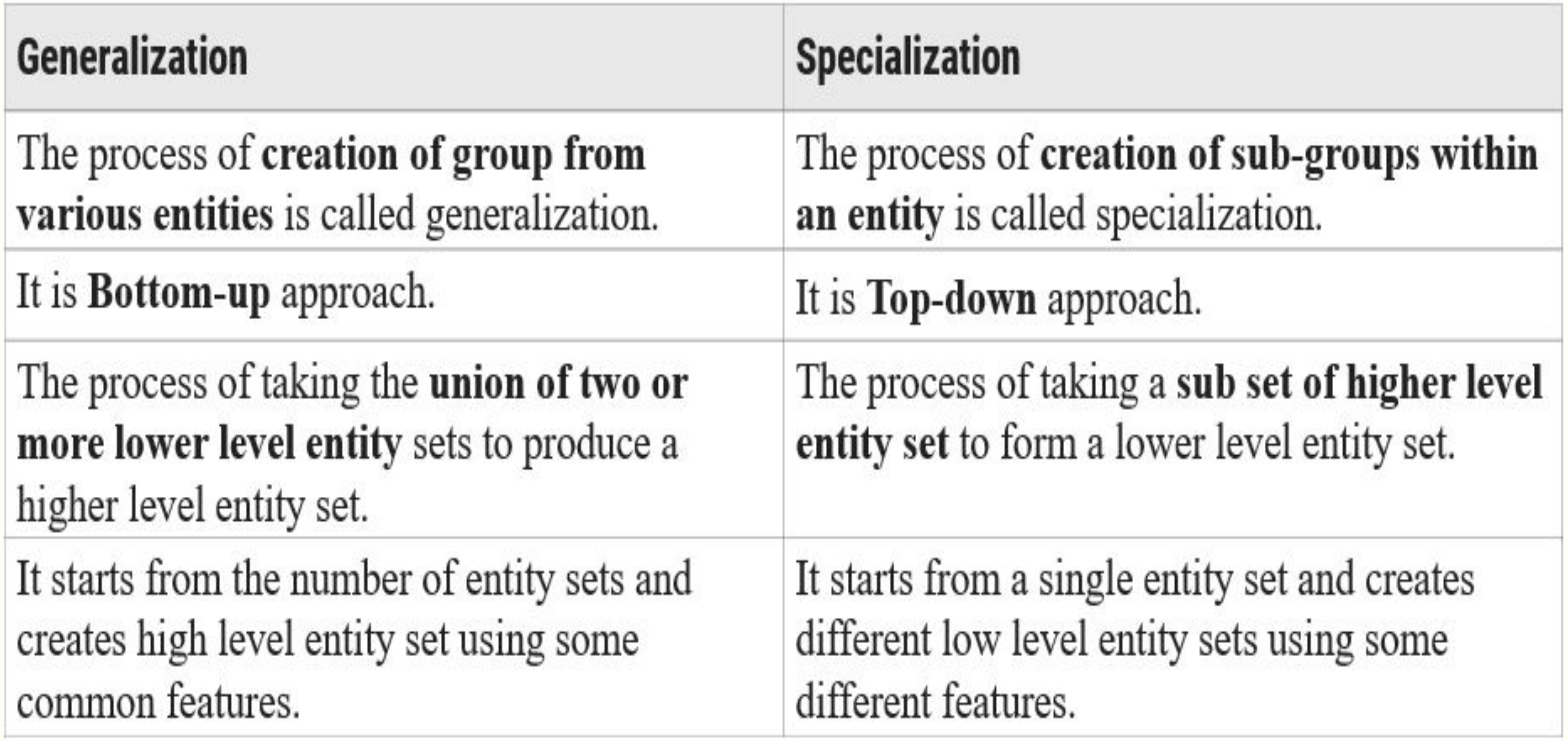
Super Class	Sub Class
A superclass is an entity from which **another entities can be derived**.	A subclass is an entity that is **derived from another entity**.
E.g, an entity account has two subsets saving_account and current_account So an **account is superclass**.	E.g, saving_account and current_account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



Generalization v/s 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**.
	

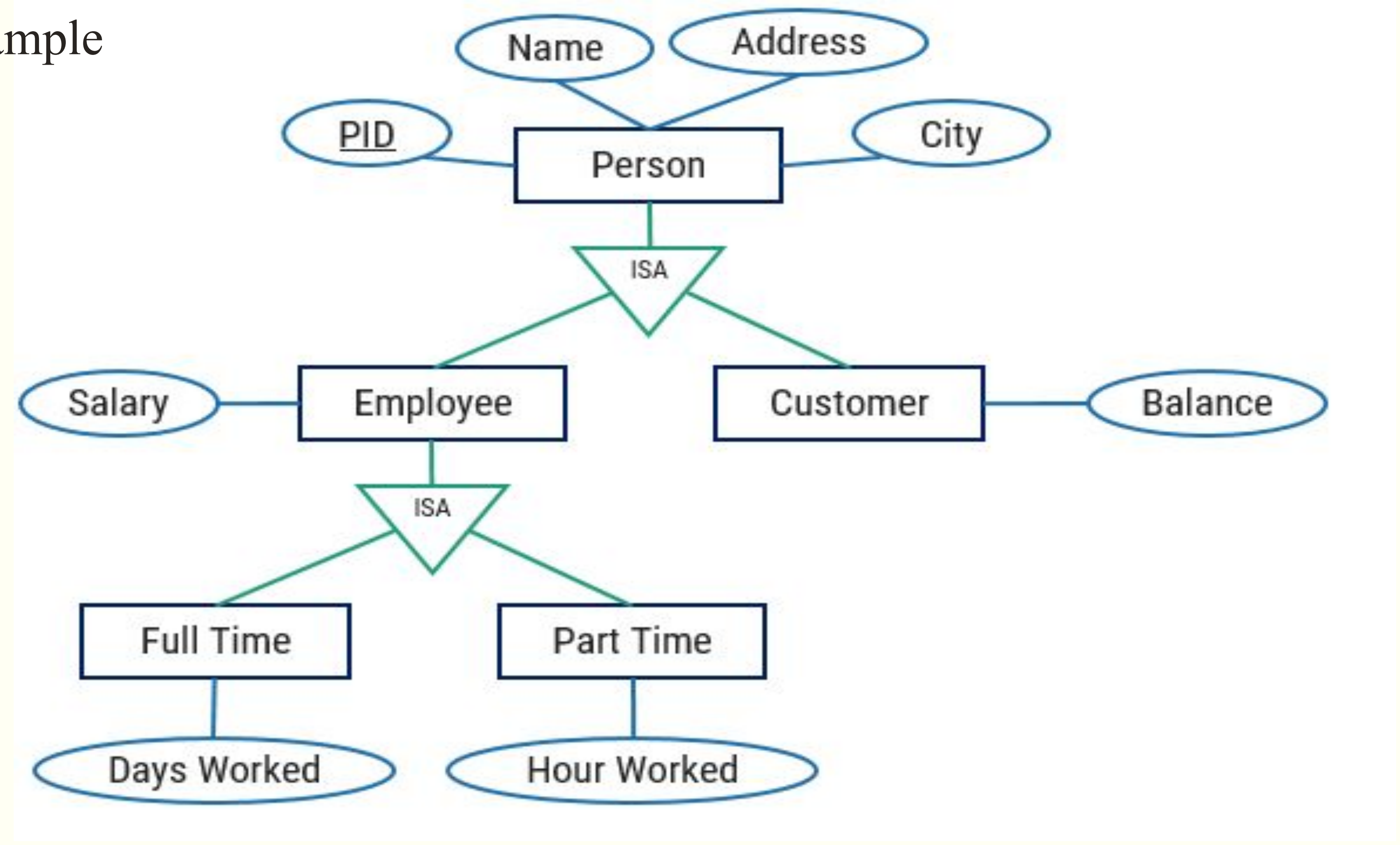
Cont..



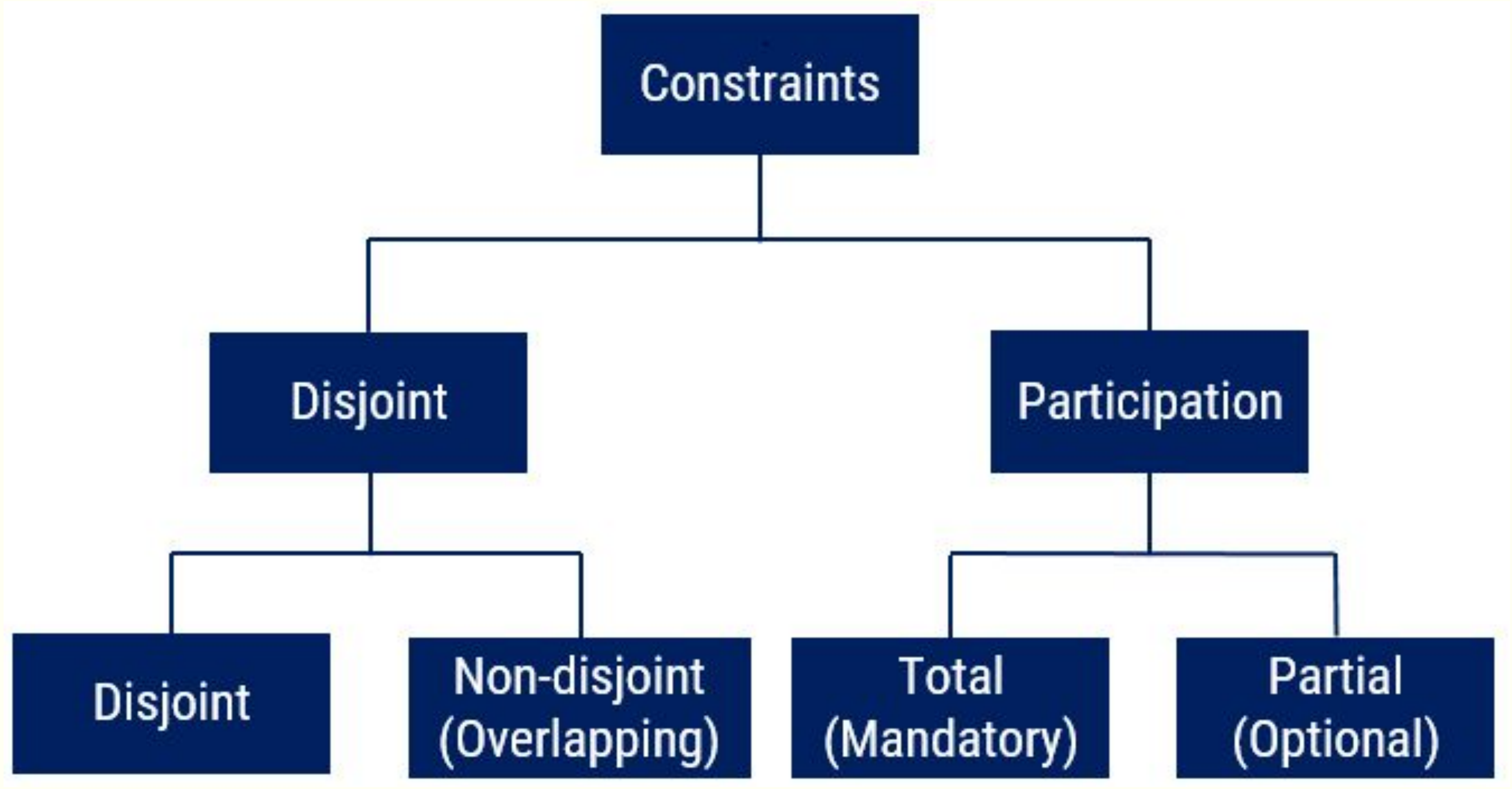
Generalization	Specialization
The process of **creation of group from various entities** is called generalization.	The process of **creation of sub-groups within an entity** is called specialization.
It is **Bottom-up** approach.	It is **Top-down** approach.
The process of taking the **union of two or 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.

Cont..

Example

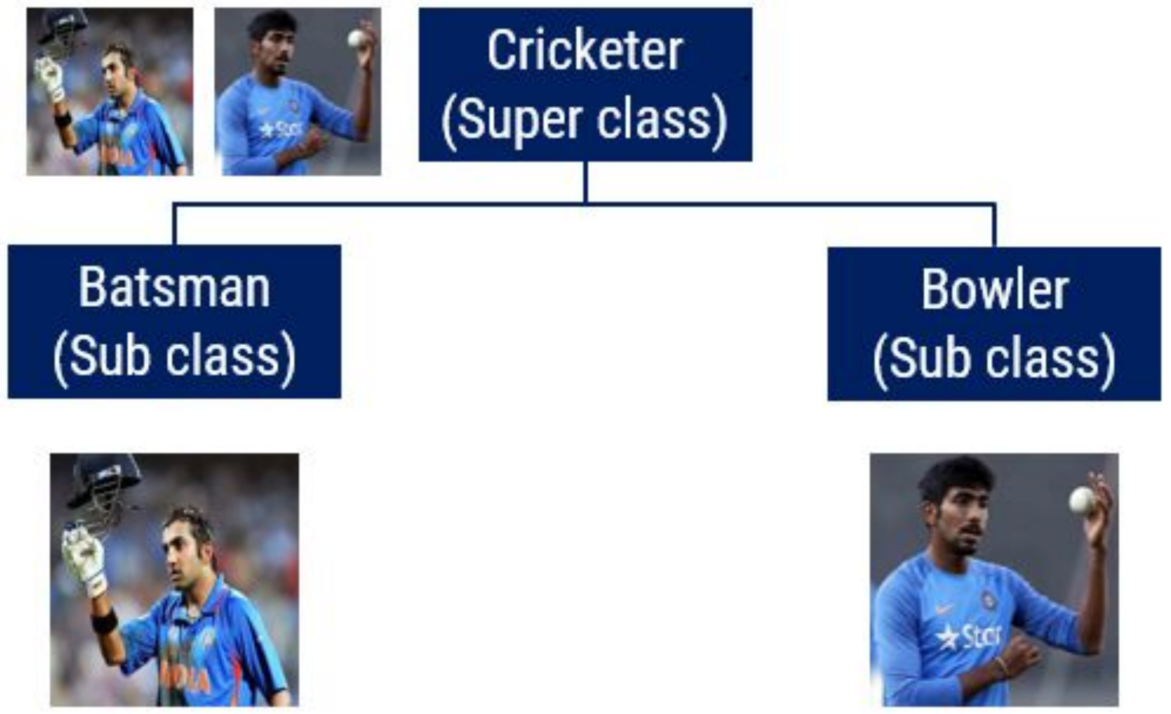


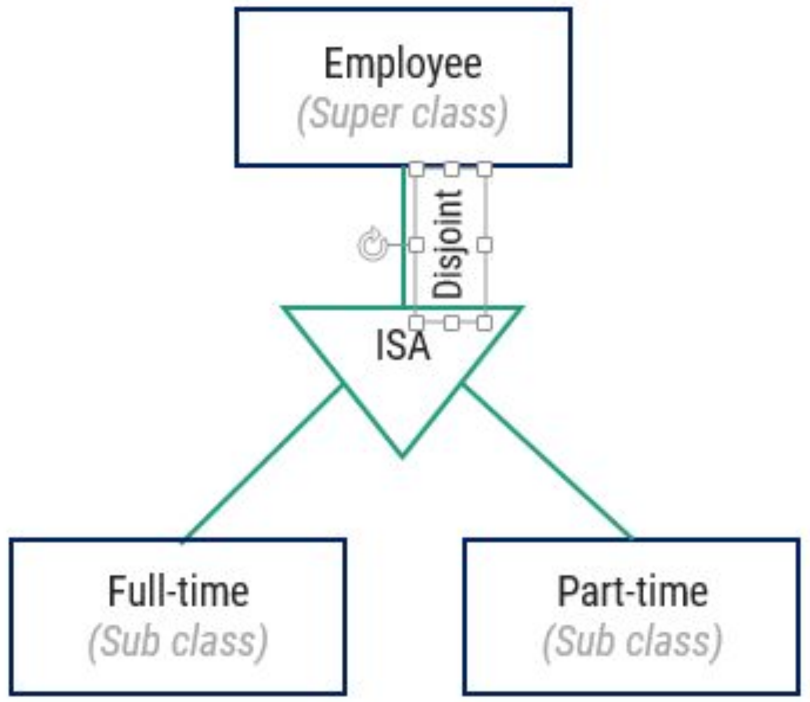
Constraints on Specialization and Generalization



Disjoint Constraint

- It describes **relationship between members of the superclass and subclass** and indicates whether member of a superclass can be a member of one, or more than one subclass.
- It specifies that the **entity of a super class can belong to only one lower-level entity set** (sub class).
- Specified by '**d**' or by writing **disjoint** near to the ISA triangle.

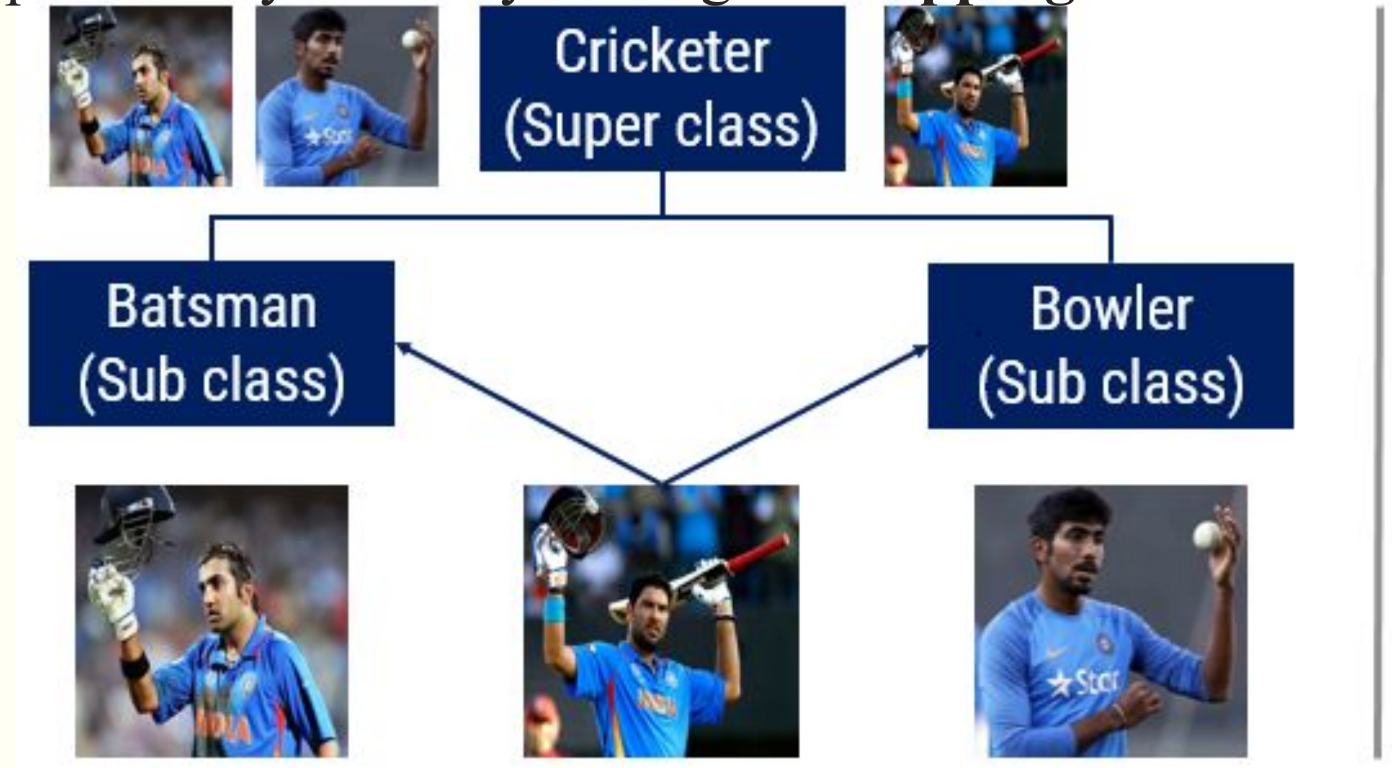




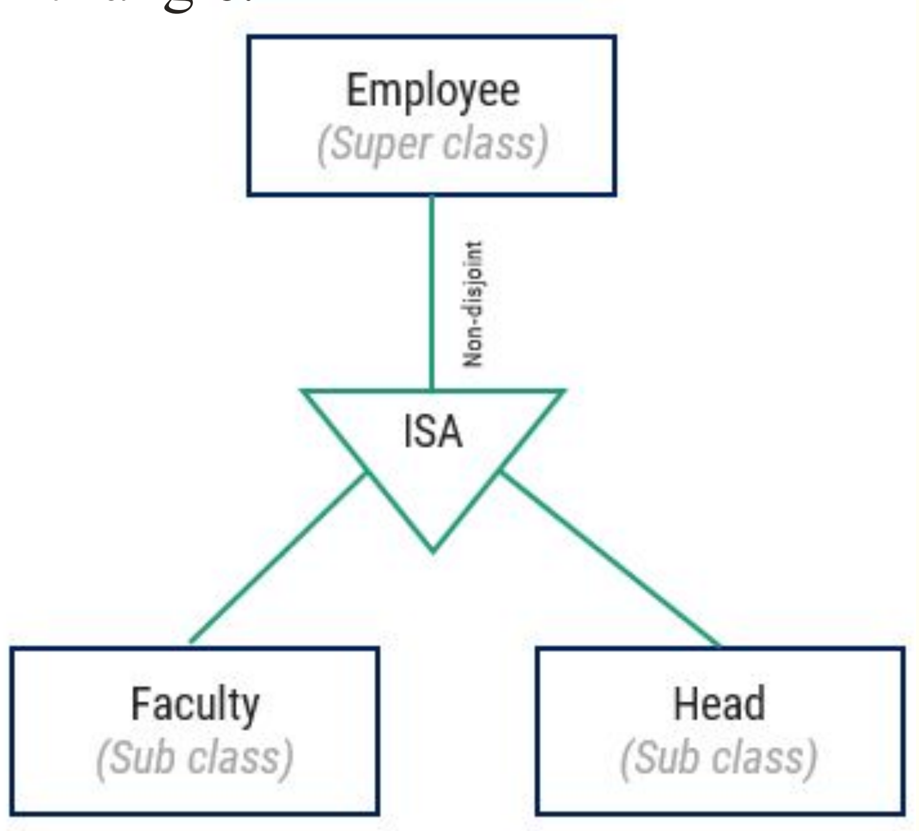
All the players are associated with only one sub class either (Batsman or Bowler).

Non-disjoint (Overlapping) Constraint

It specifies that an **entity of a super class can belong to more than one lower-level entity** set (sub class).
Specified by ‘**o**’ or by writing **overlapping** near to the ISA triangle.

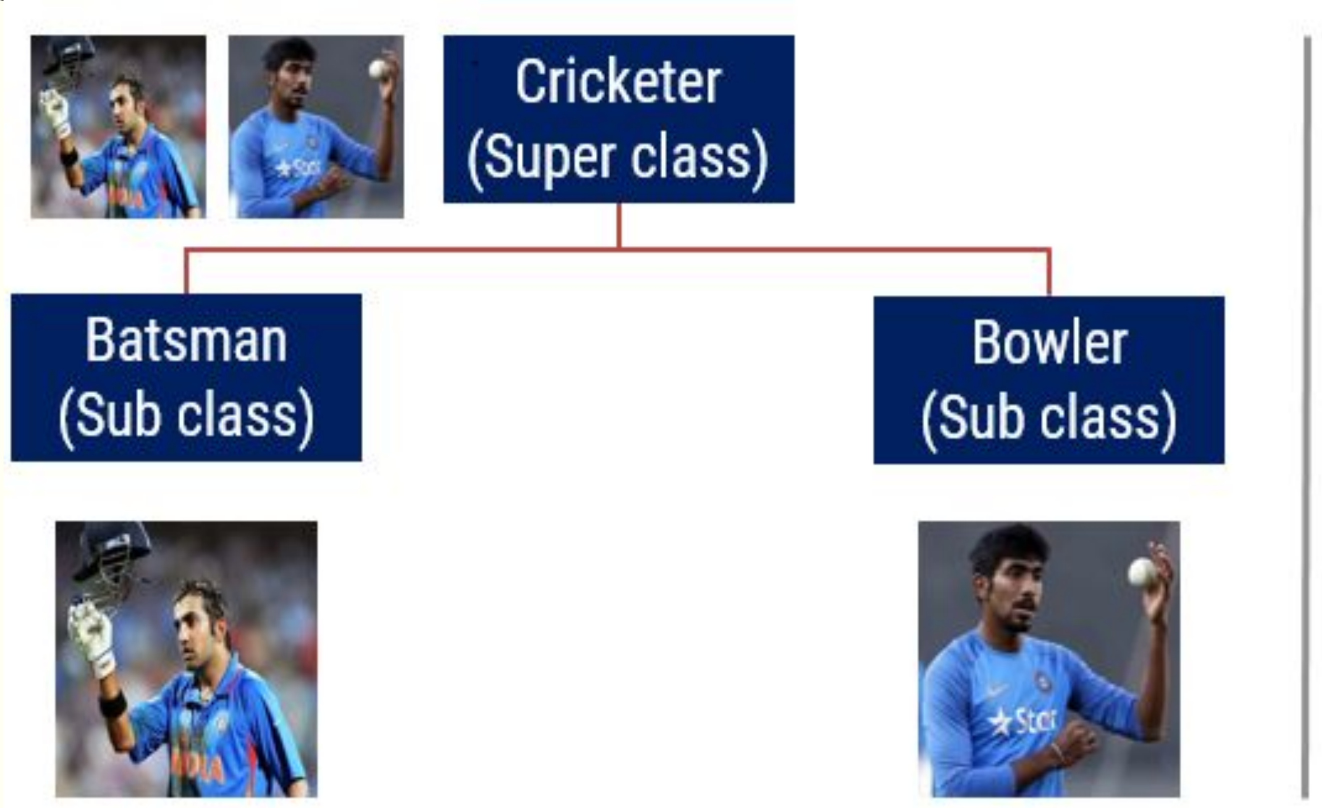


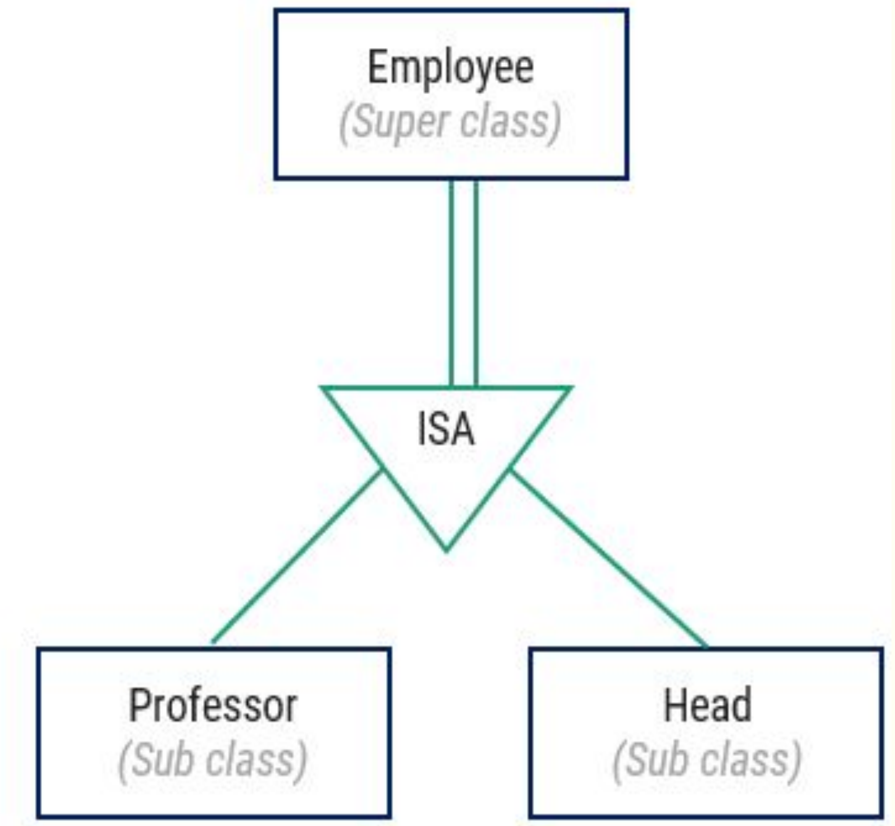
One player (Yuvraj singh) is associated with more than one sub class.



□ Total (Mandatory) Participation

- Total participation specifies that **every entity in the superclass must be a member of some subclass** in the specialization.
- Specified by a **double line** in E-R diagram.

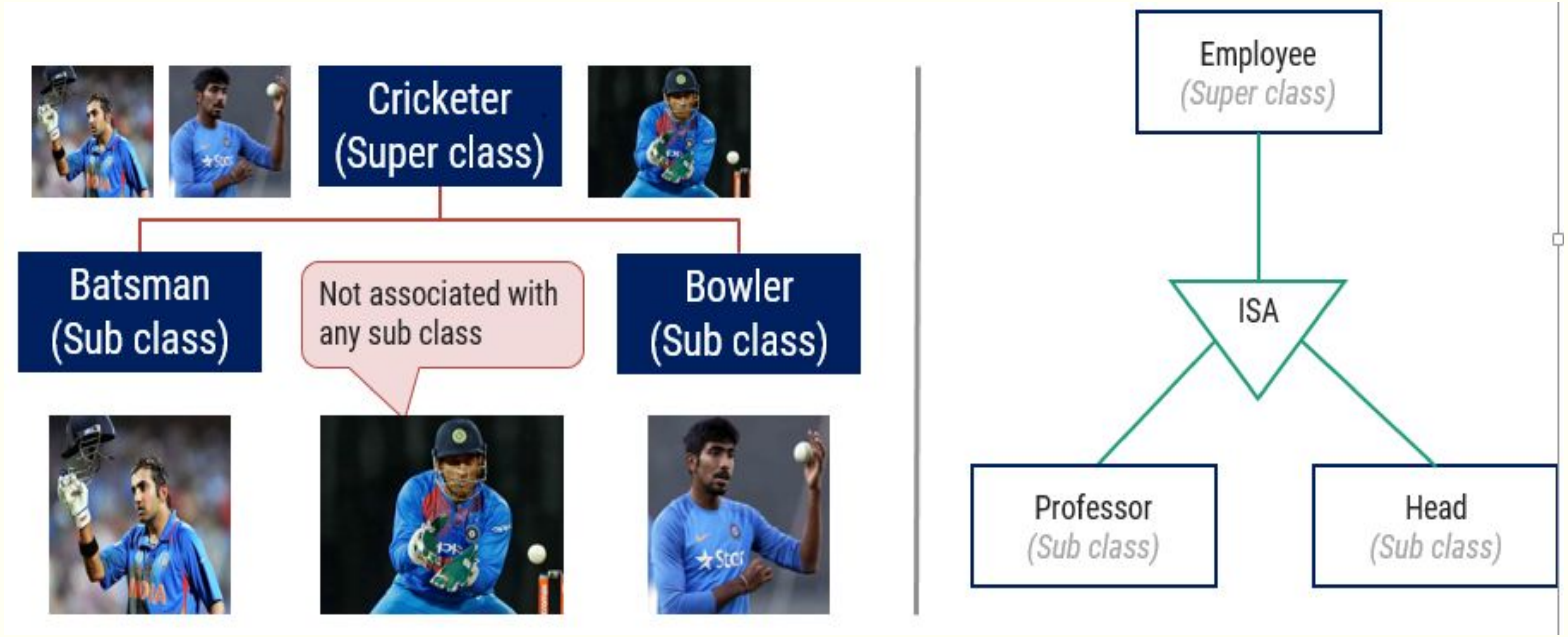




All the players are associated with minimum one sub class either (Batsman or Bowler).

Partial (Optional) Participation

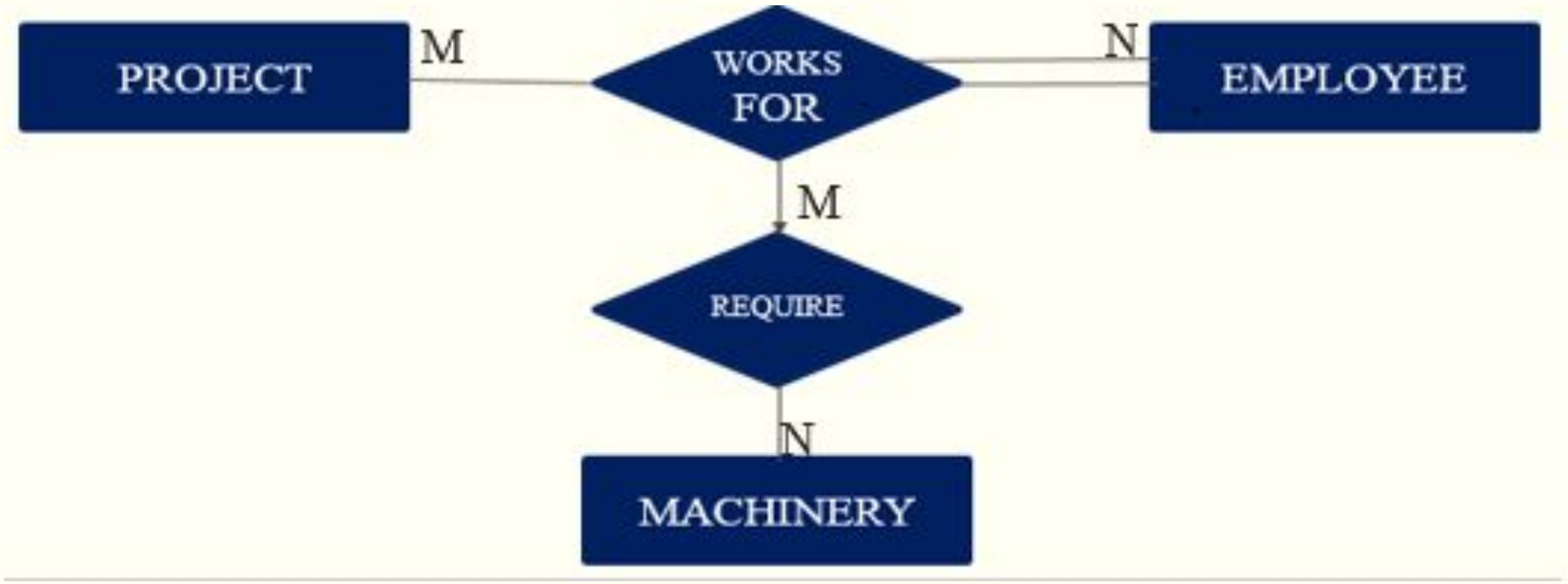
- Partial participation specifies that **every entity in the super class does not belong to any of the subclass** of specialization.
- Specified by a **single line** in E-R diagram.



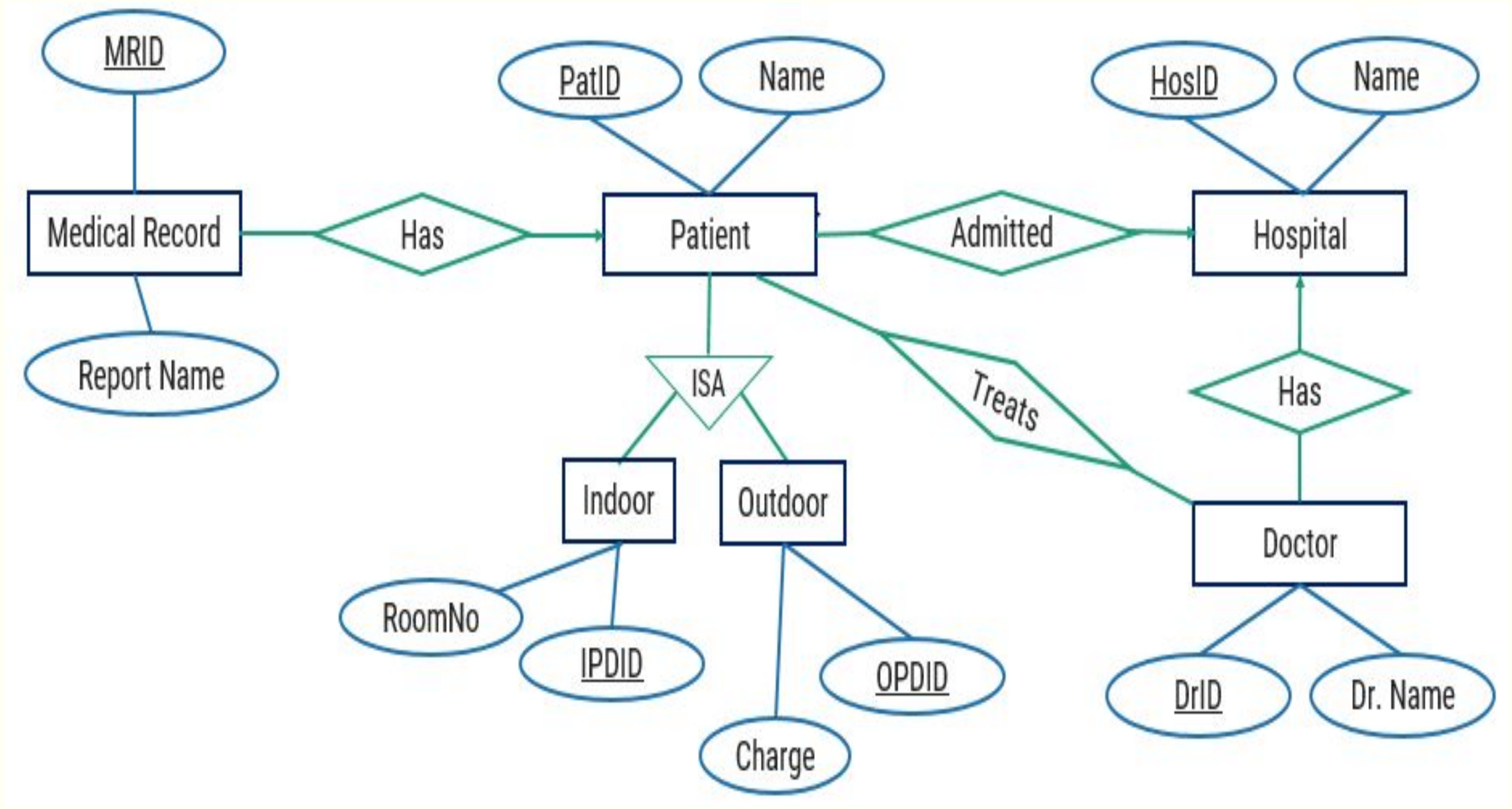
Aggregation in E-R diagram

- An ER diagram is not capable of representing relationship between an entity and a relationship which may be required in some scenarios.
- In those cases, a relationship with its corresponding entities is aggregated into a higher level entity.
- Aggregation is an abstraction through which we can represent relationships as higher level entity sets.

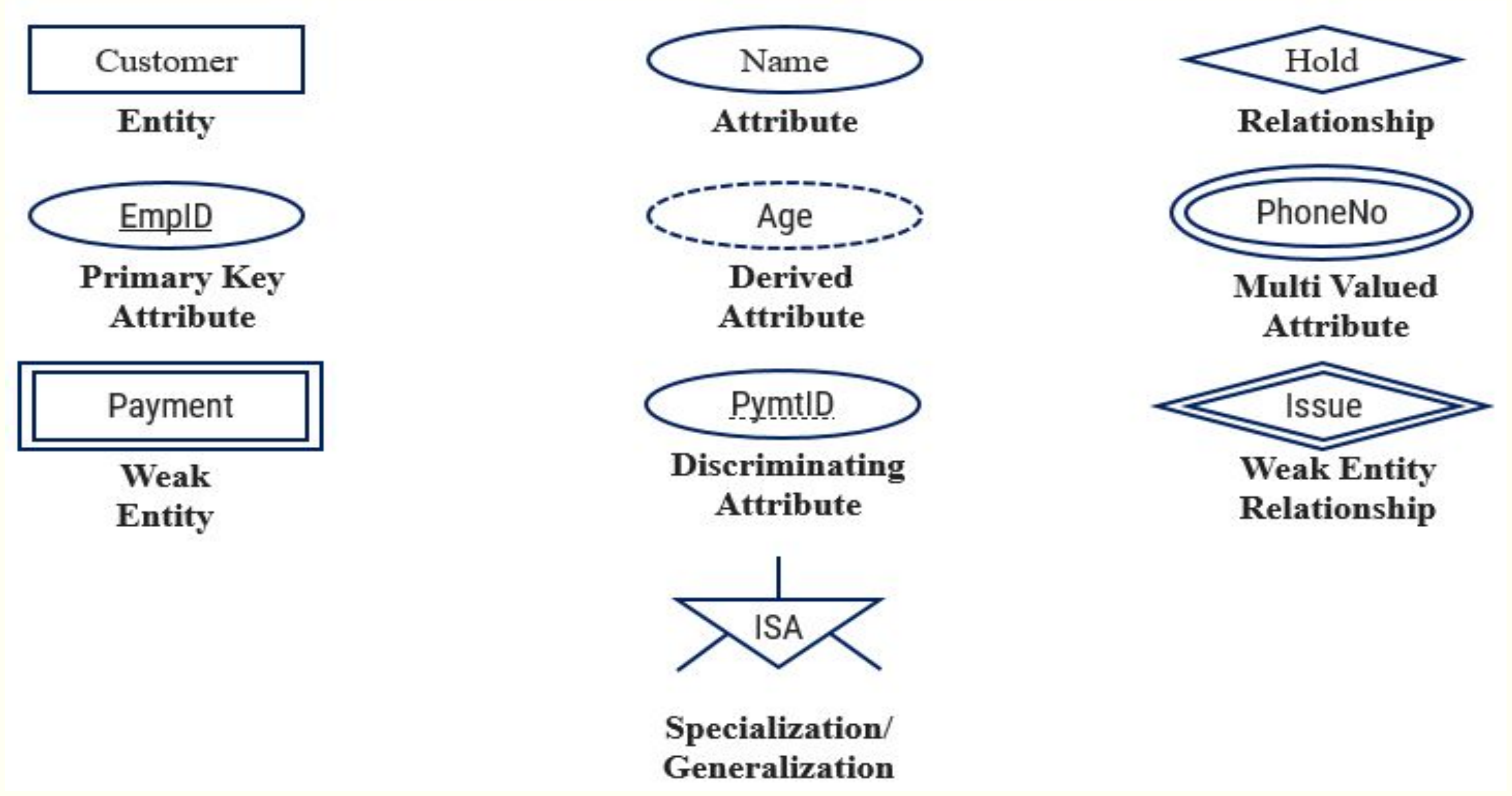
- For Example, Employee working for a project may require some machinery. So, REQUIRE relationship is needed between relationship WORKS_FOR and entity MACHINERY. Using aggregation, WORKS_FOR relationship with its entities EMPLOYEE and PROJECT is aggregated into single entity and relationship REQUIRE is created between aggregated entity and MACHINERY.



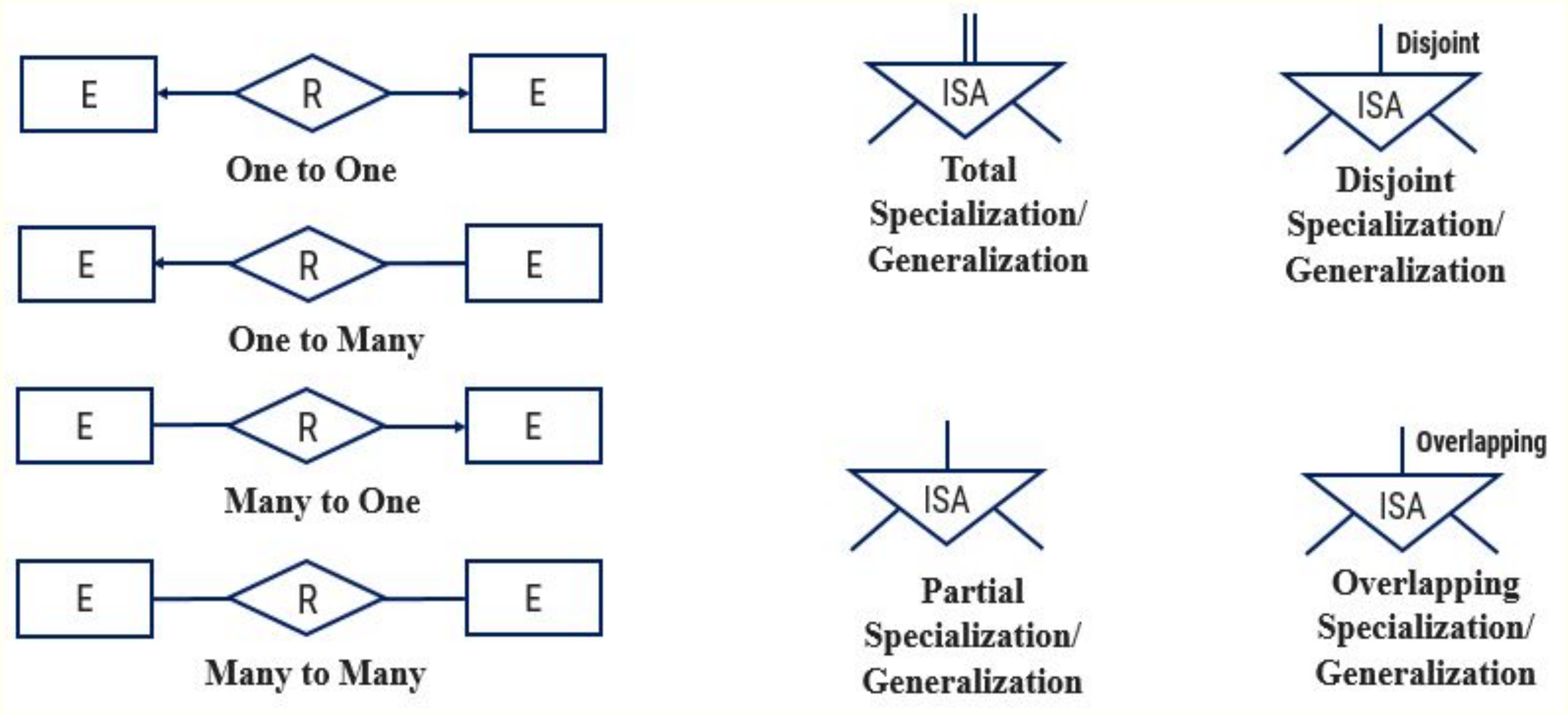
E-R diagram of Hospital Management System



Summery of Symbols used in E-R diagram



Cont…


E
R
E
One to One
E
R
E
One to Many
E
R
E
Many to One
E
R
E
Many to Many
ISA
Total Specialization/ Generalization
Disjoint
ISA
Disjoint Specialization/ Generalization
ISA
Partial Specialization/ Generalization
Overlapping
ISA
Overlapping Specialization/ Generalization

Data Models

What is a Database Models?

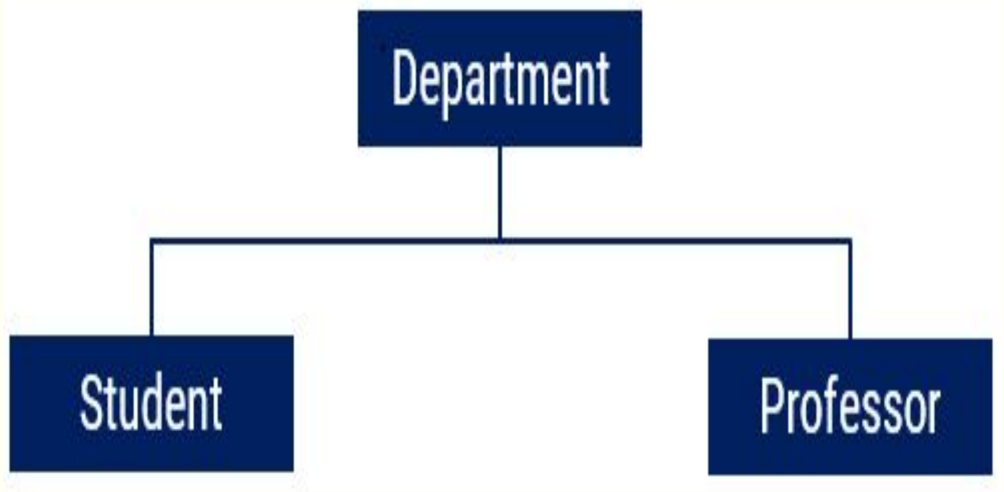
- A database model is a type of data model that **defines the logical structure of a database**.
- It determine **how data can be stored, accessed and updated** in a database management system.
- The most popular example of a database model is the relational model, which uses a table-based format.

Type of Database Models

- Hierarchical Model
- Network Model
- Entity-relationship Model
- Relational Model
- Object-oriented database Model

Hierarchical Model

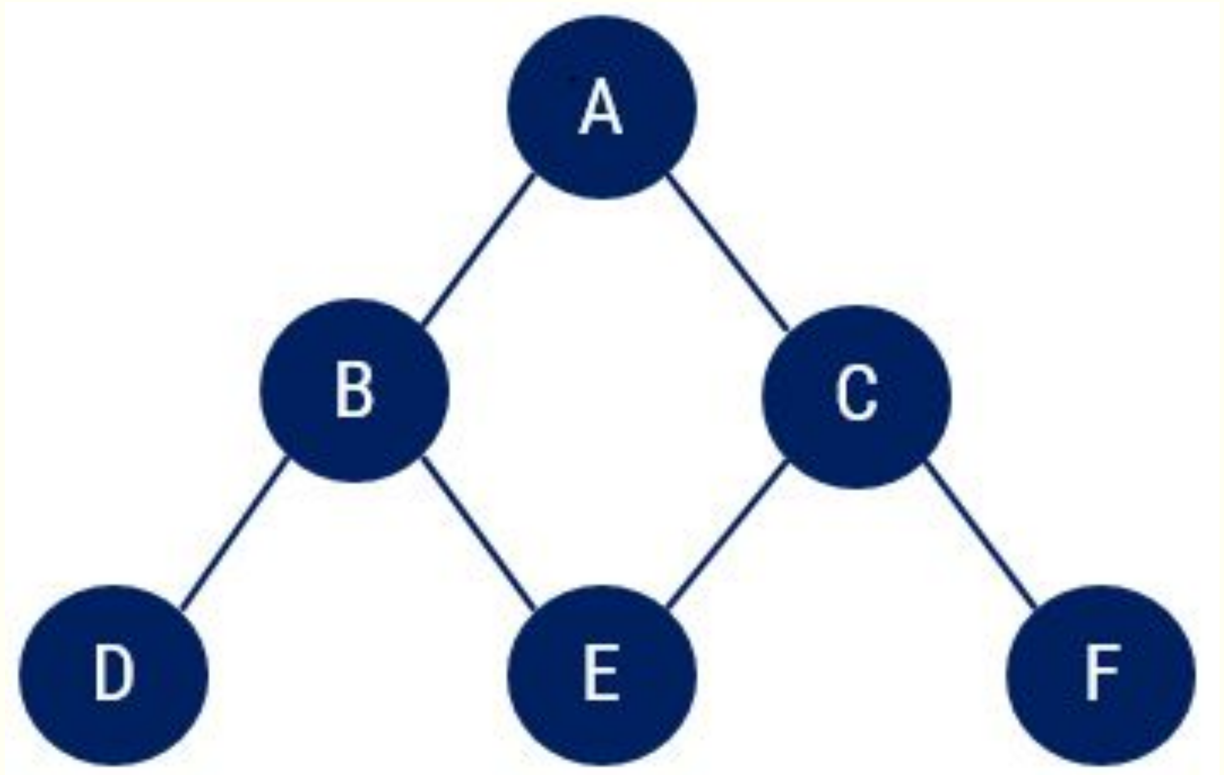
- The hierarchical model organizes data into a **tree-like structure**, where **each record has a single parent or root**.



- 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**.

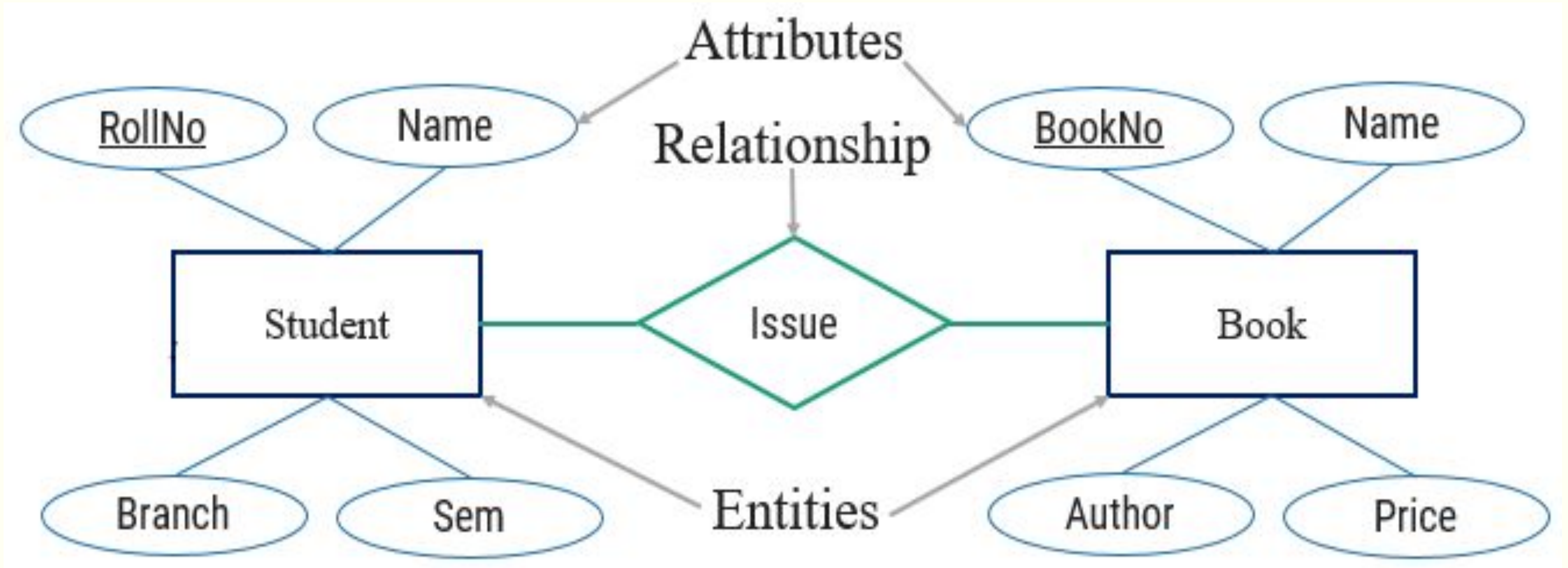
Network Model

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



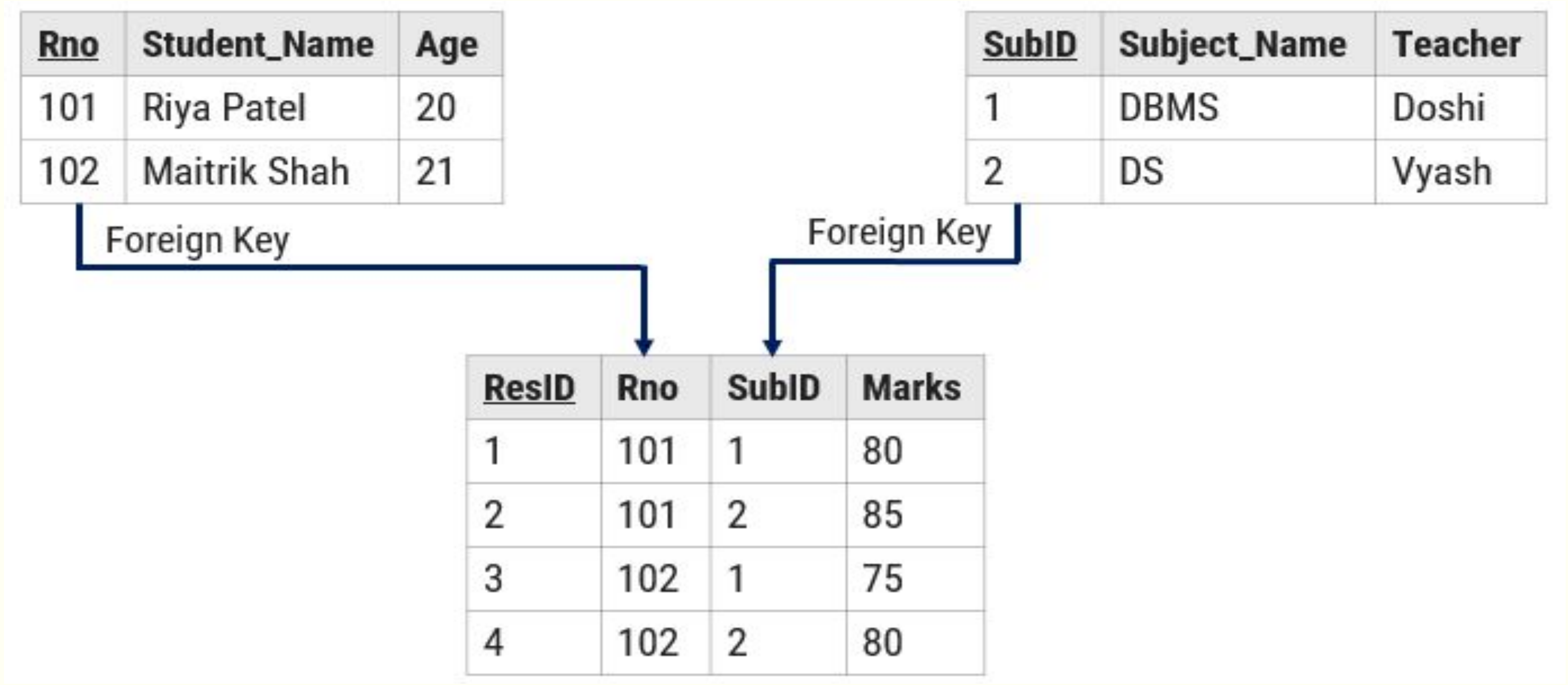
Entity-relationship Model

- 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	Riya Patel	20
102	Maitrik Shah	21

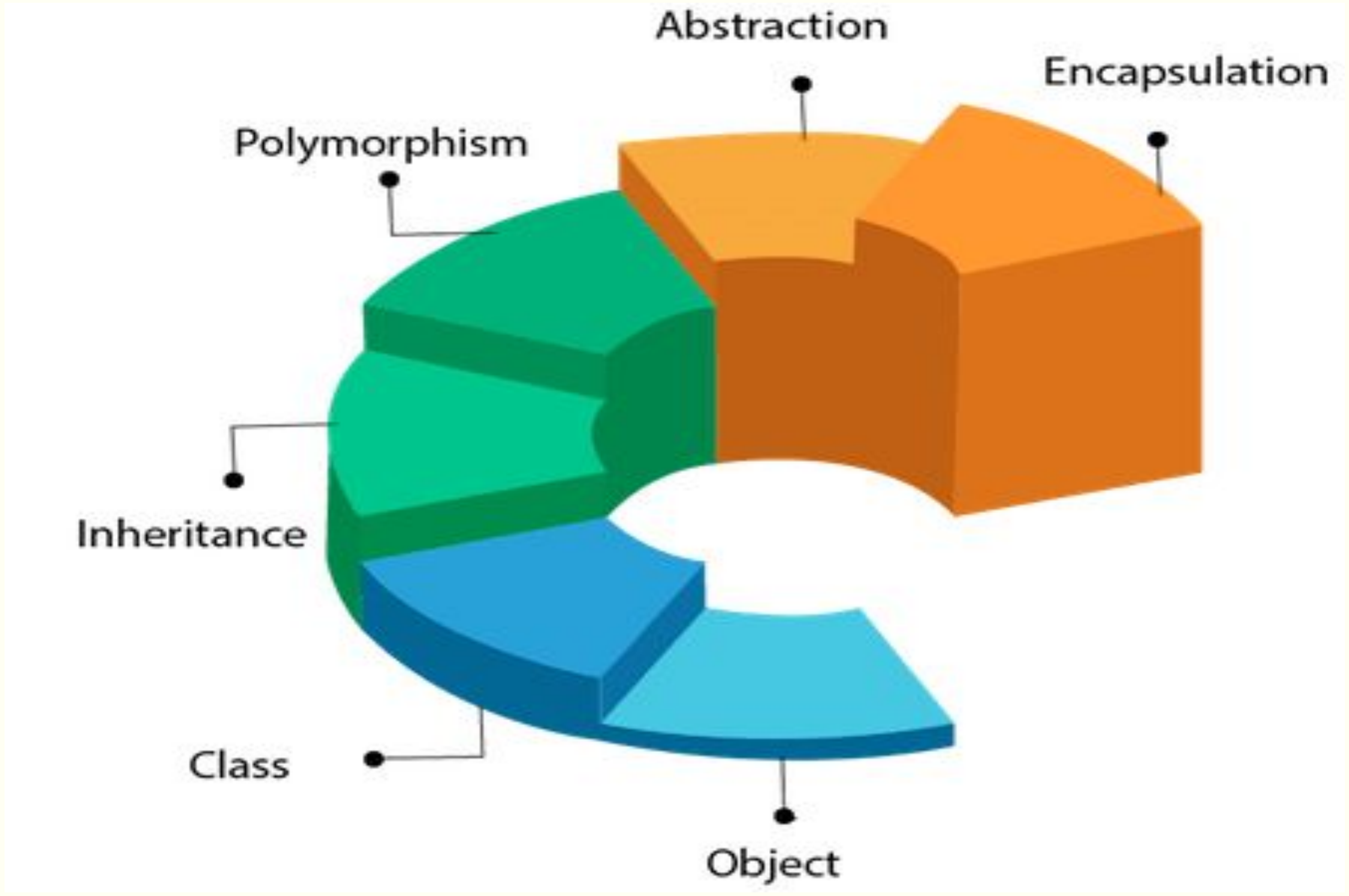
SubID	Subject_Name	Teacher
1	DBMS	Doshi
2	DS	Vyash

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.

Cont..


Abstraction
Encapsulation
Polymorphism
Inheritance
Class
Object

Cont..

Object

Any entity that has state and behavior is known as an object. For example, a chair, pen, table, keyboard, bike, etc. It can be physical or logical.
Example: A dog is an object because it has states like color, name, breed, etc. as well as behaviors like wagging the tail, barking, eating, etc.

Class

Collection of objects is called class. It is a logical entity.
A class can also be defined as a blueprint from which you can create an individual object. Class doesn't consume any space.

Cont..

- **Inheritance**
  - When one object acquires all the properties and behaviors of a parent object, it is known as inheritance. It provides code reusability. It is used to achieve runtime polymorphism.
- **Polymorphism**
  - If one task is performed in different ways, it is known as polymorphism.
  - For example can be to speak something; for example, a cat speaks meow, dog barks woof, etc.

"Now Speak!"
"Quack"
"Woof"
"Meow"

Cont..

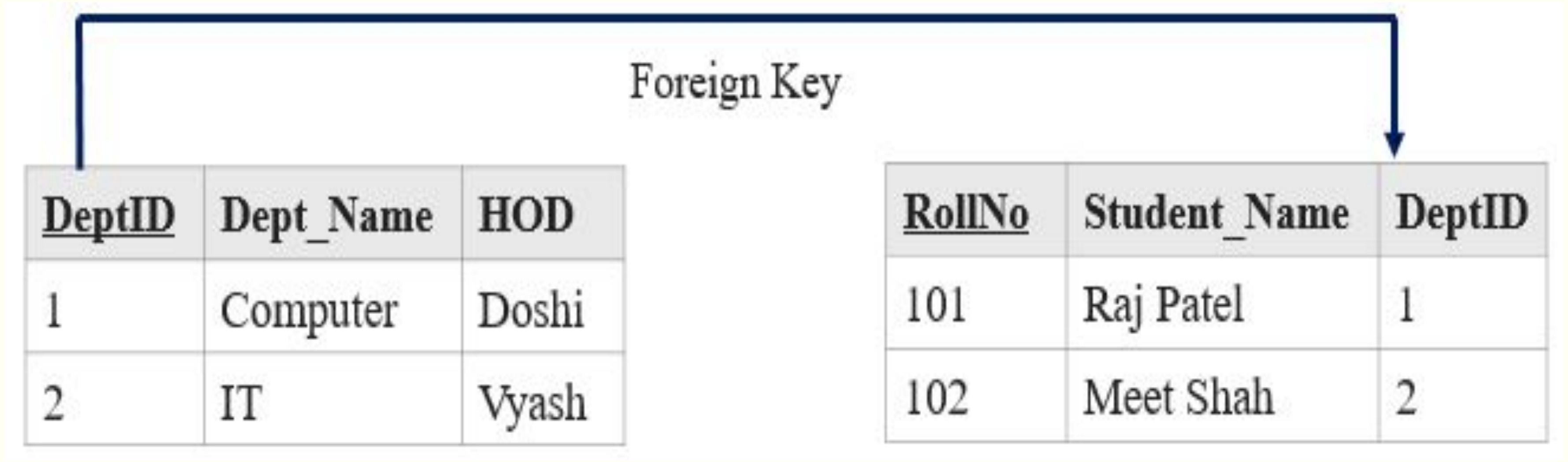
- **Abstraction**
  - Hiding internal details and showing functionality is known as abstraction. For example phone call, we don't know the internal processing.
- **Encapsulation**
  - Binding (or wrapping) code and data together into a single unit are known as encapsulation. For example, a capsule, it is wrapped with different medicines.

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.

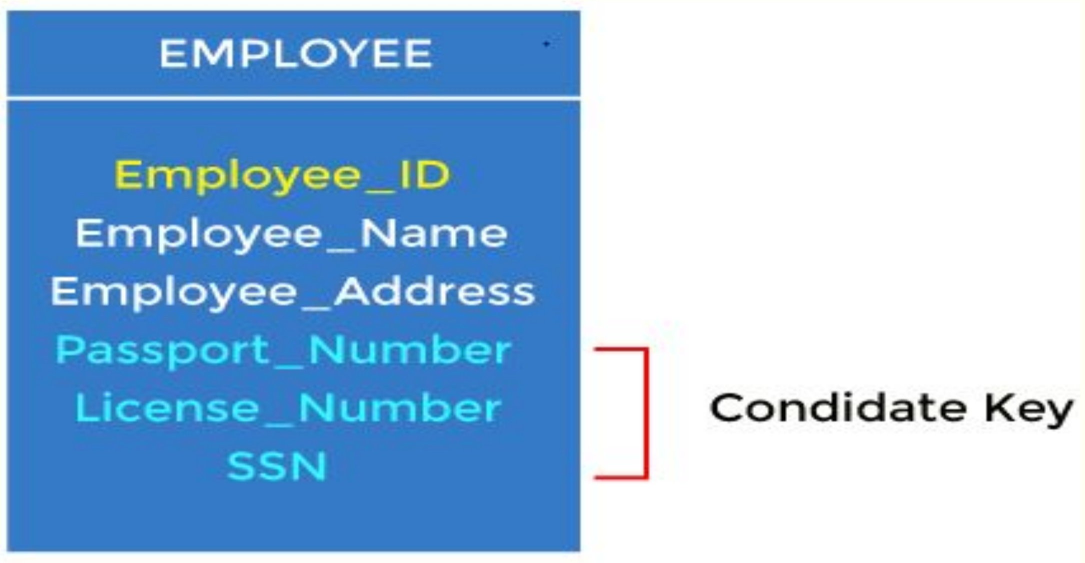
- **Primary key**
  - This constraint defines a column or combination of columns which uniquely identifies each row in the table.
  - Primary key = **Unique key + Not null**
  - E.g. enrollmentno column should have unique value as well as can’t be null.
- **Foreign key (referential integrity constraint)**
  - A referential integrity constraint (foreign key) is specified between two tables.
  - In the referential integrity constraints, if a foreign key column in table 1 refers to the primary key column of table 2, then every value of the foreign key column in table 1 must be null or be available in primary key column of table 2.



DeptID	Dept_Name	HOD
1	Computer	Doshi
2	IT	Vyash

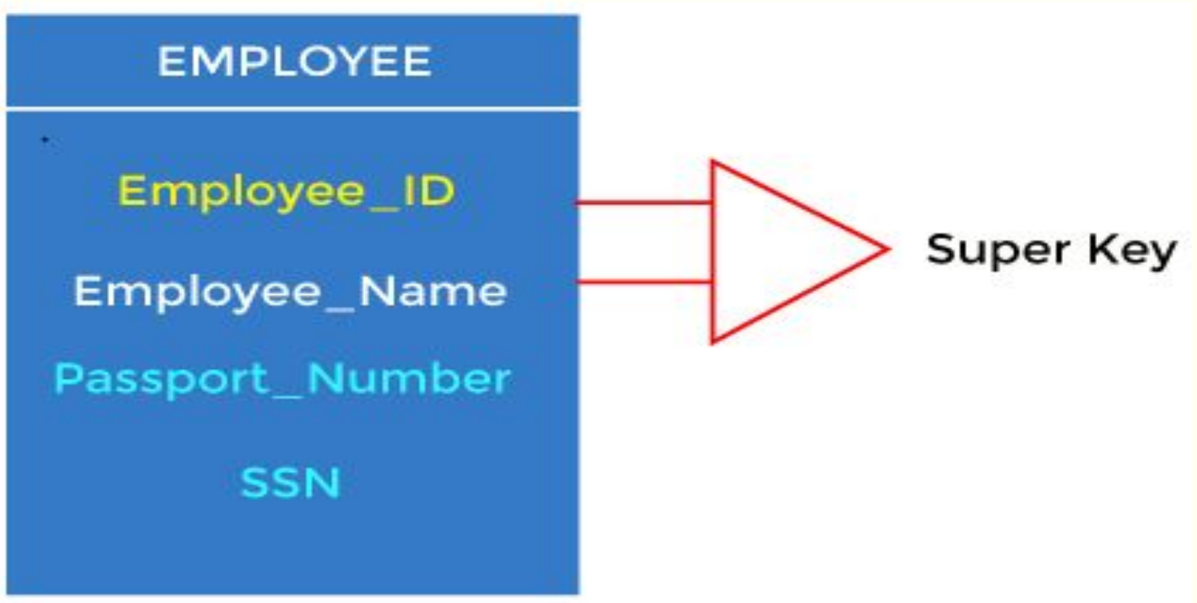
RollNo	Student_Name	DeptID
101	Raj Patel	1
102	Meet Shah	2

- **Candidate key**
  - A candidate key is an attribute or set of attributes that can uniquely identify a tuple.
  - Except for the primary key, the remaining attributes are considered a candidate key. The candidate keys are as strong as the primary key.
  - E.g. : In the EMPLOYEE table, id is best suited for the primary key. The rest of the attributes, like SSN, Passport_Number, License_Number, etc., are considered a candidate key.



Super Key

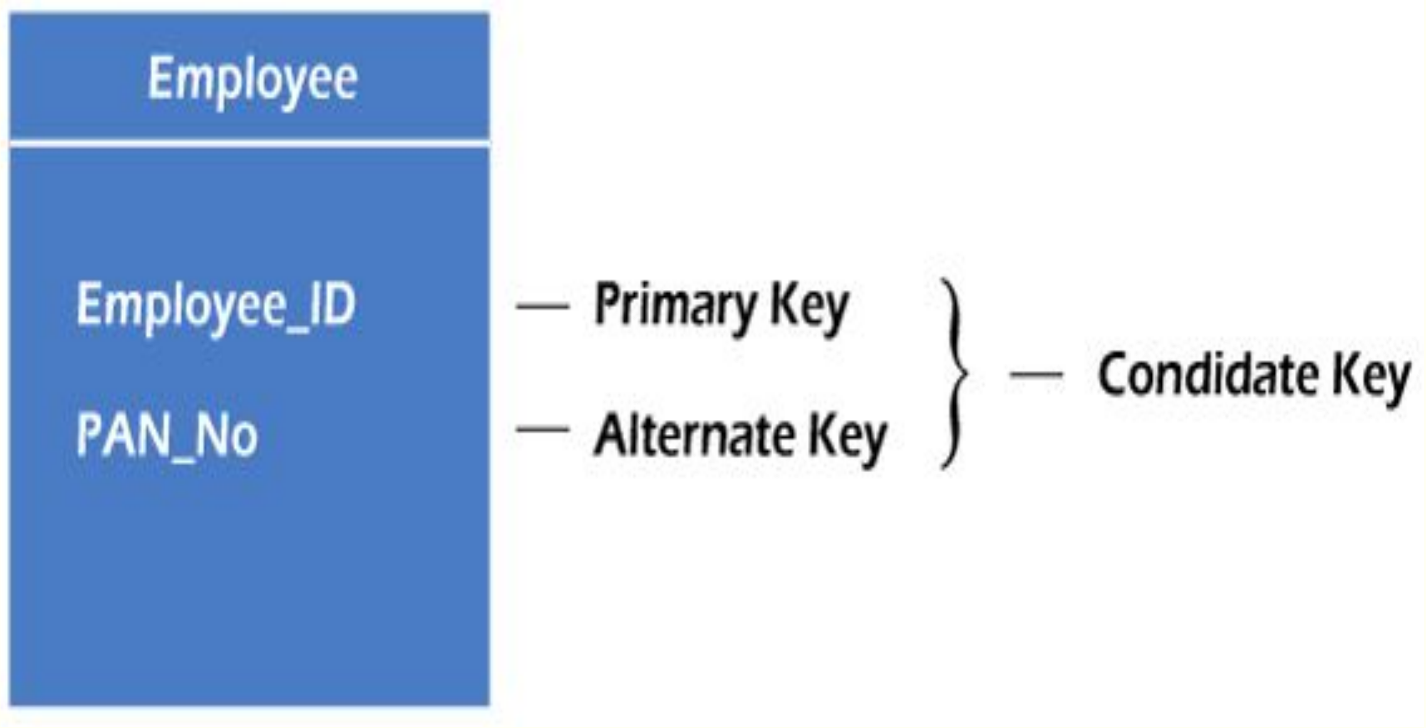
- Super key is an attribute set that can uniquely identify a tuple. A super key is a superset of a candidate key.
- **For example:** In the above EMPLOYEE table, for(EMPLOEE_ID, EMPLOYEE_NAME), the name of two employees can be the same, but their EMPLYEE_ID can't be the same. Hence, this combination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



- The super key would be EMPLOYEE-ID (EMPLOYEE_ID, EMPLOYEE-NAME), etc.

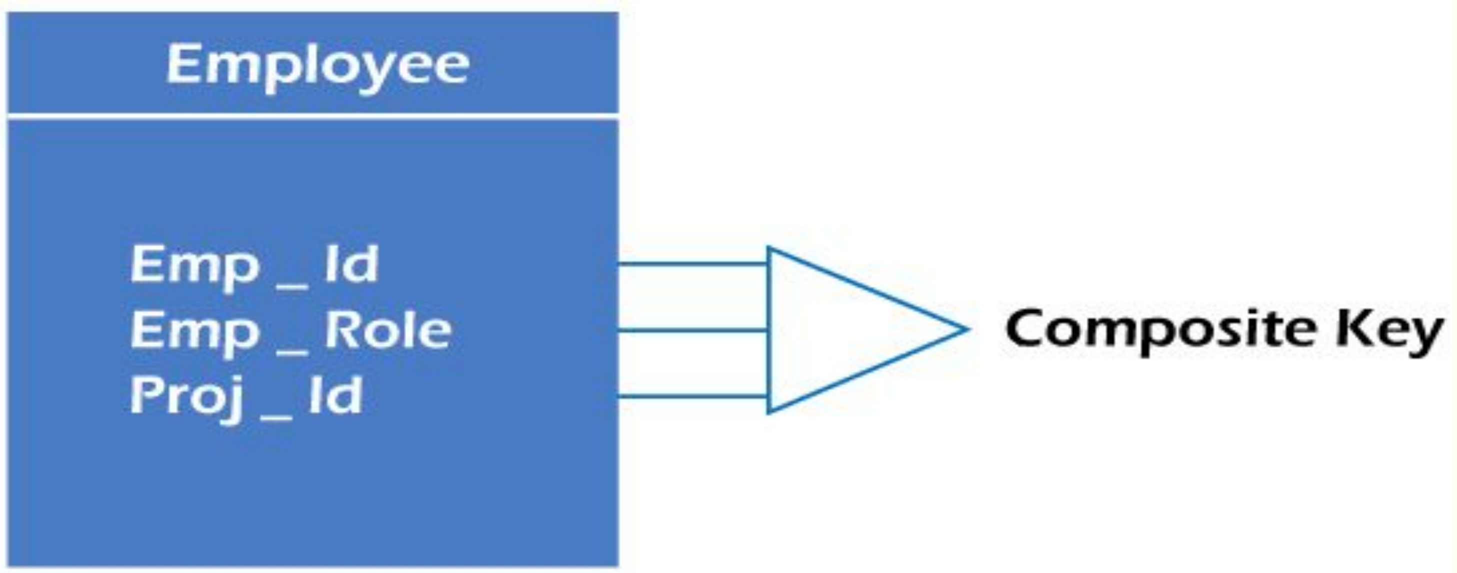
Alternate key

- There may be one or more attributes or a combination of attributes that uniquely identify each tuple in a relation. These attributes or combinations of the attributes are called the candidate keys. One key is chosen as the primary key from these candidate keys, and the remaining candidate key, if it exists, is termed the alternate key.
- **For example,** employee relation has two attributes, Employee_Id and PAN_No, that act as candidate keys. In this relation, Employee_Id is chosen as the primary key, so the other candidate key, PAN_No, acts as the Alternate key.



Composite key

- Whenever a primary key consists of more than one attribute, it is known as a composite key. This key is also known as Concatenated Key.
- **For example,** in employee relations, we assume that an employee may be assigned multiple roles, and an employee may work on multiple projects simultaneously. So the primary key will be composed of all three attributes, namely Emp_ID, Emp_role, and Proj_ID in combination. So these attributes act as a composite key since the primary key comprises more than one attribute.



Questions asked in GTU

1. Write a note on mapping cardinality in E-R diagram.
2. Explain the difference between a weak and a strong entity set.
3. Explain the difference between generalization and specialization. **OR** Explain specialization and generalization concept in E-R diagram with suitable example.
4. Write a note on constraints on specialization and generalization.
5. Explain aggregation in E-R diagram with example.
6. What do you mean by integrity constraints? Discuss various integrity constraints.

Questions asked in GTU

7. Draw E-R diagram for Bank Management System.
8. Define E-R diagram. Draw an E-R diagram for Library Management System. Assume relevant entities and attributes for the given system.
9. Construct an E-R diagram for a car-insurance company whose customers own one or more cars each. Each car has associated with it zero to any number of recorded accidents.
10. Design a generalization–specialization hierarchy for a motor-vehicle sales company. The company sells motorcycles, passenger cars, vans, and buses. Justify your placement of attributes at each level of the hierarchy. Explain why they should not be placed at a higher or lower level.

Questions asked in GTU

11. Design a database for an airline. The database must keep track of customers and their reservations, flights and their status, seat assignments on individual flights, and the schedule and routing of future flights. Your design should include an E-R diagram, a set of relational schemas, and a list of constraints, including primary-key and foreign-key constraints.

12. Design a database for a hospital with a set of patients and a set of medical doctors. Associate with each patient a log of the various tests and examinations conducted. Your design should include an E-R diagram, a set of relational schemas, and a list of constraints, including primary-key and foreign-key constraints.

Thank you...