



Unit 3- Data Models

Subject Code: 303105203

Prof. S.W.Thakare
Assistant Professor,
Computer science & Engineering





CHAPTER-3

Data Models



DATA MODELS

Data Models

- Data models are how the logical structure of a database is designed.
- Data Models are fundamental entities to introduce abstraction in a DBMS.
- Data models define how data is connected to each other and how they are processed and stored inside the system.
- It defines how data can be stored, accessed and updated in database management systems.

Data Models

- There are different types of data models in DBMS:-
 - ER(Entity-Relationship)- Model
 - Relational Model
 - Network Model
 - Object Oriented Model

ER Model

Basic Concepts:-

- **What is E-R diagram?**
 - E-R diagram means Entity-Relationship diagram
 - It is a visual tool for **graphical (pictorial) representation** of database.
 - ER Model was proposed by Peter Chen in 1970's to use it for a conceptual modelling/designing of database.



ER Model

Basic Concepts:-

- **What is E-R diagram?**
 - It is based on the view of real-world entities and relationships among them.
 - While expressing real-world scenario into the database model, the ER Model creates **entity set, relationship set, general attributes and constraints**.
 - ER Model mainly focuses on **Entities and their attributes and Relationships** among entities.
 - It uses various types of symbols to represent objects of database.

Entity

- Entity is real-world objects, place, person about which we collect data
- Example: For University database entity can be Student, Teacher, Department, Course, Result, Class, etc.
- Entity is denoted by a **Rectangle** containing name of entity.

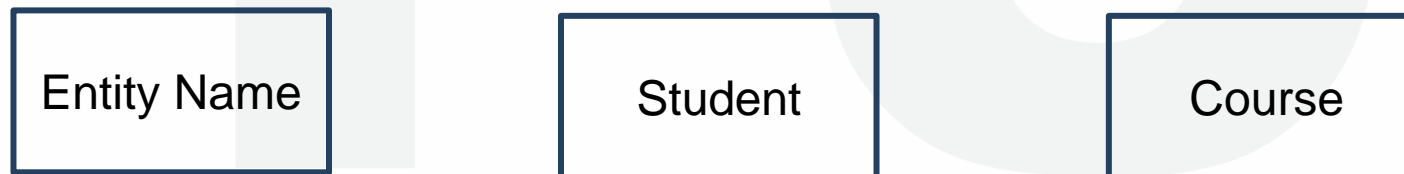


Figure: 1.25 Entity

Attributes

- Attribute represent **properties or details** about an entity.
- Example: For student entity, attributes can be name, enrollment no, address, date of birth, result, etc.
- It is denoted by an **oval** symbol having name of an attribute.



Figure: 1.26 Attributes

Relationship

- Relationship is an association/connection between several entities.
- It defines how entities are related to each other.
- It is denoted by a **diamond** containing relationship's name.
- Diamond should be placed between two entities and a line connecting to both entity.
- Example: Book from library is issued by student, here book & student are entities and issue is relationship.



Figure:1.27 Relationship



Entity Set

- Entity Set is a set of entities of the same type having same properties or attributes.
 - Example: set of all persons, companies, trees, holidays, all students studying in a university
- **Relationship Set:** When there are set of relationships of a same type is called Relationship set

E-R Diagram of Library Management System

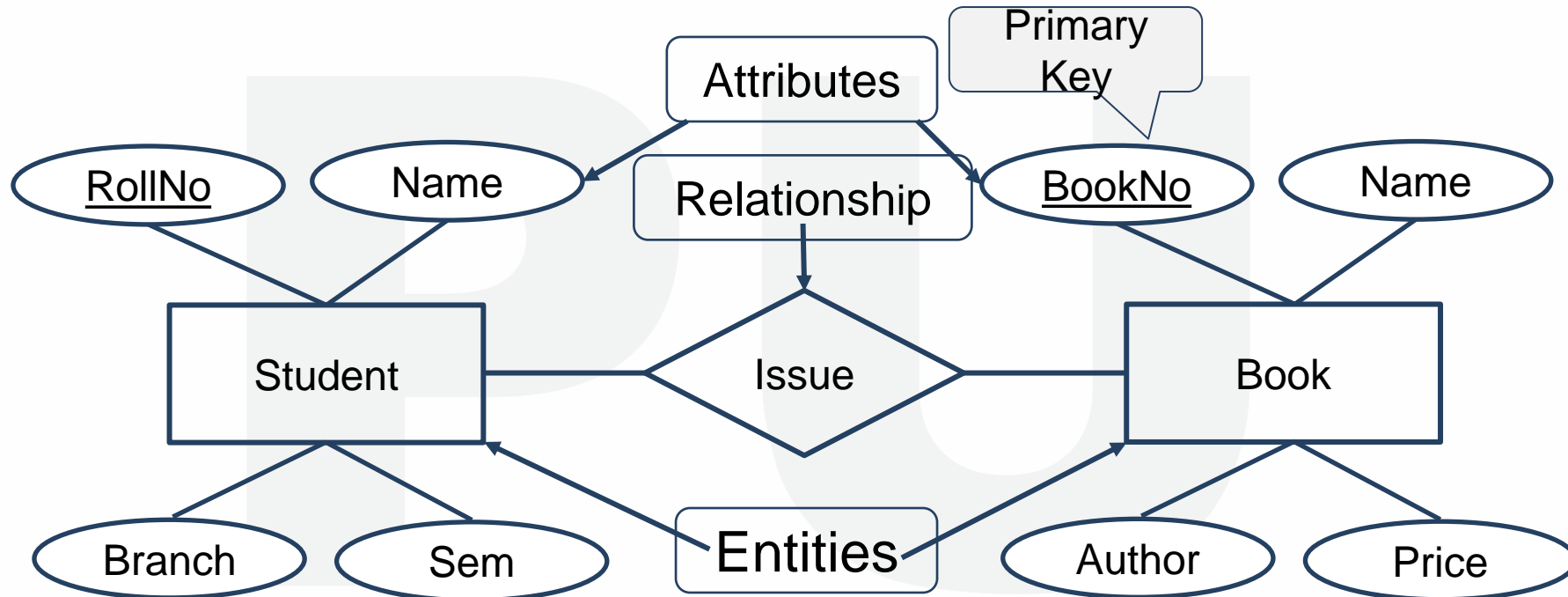


Figure: 1.28 An example of E-R Diagram

Types of Attributes

- Let's understand different types of attributes
 1. Simple and composite attributes.
 2. Single-valued and multi-valued attributes
 3. Stored attribute and Derived attributes
 4. Complex Attribute
 5. Key Attribute

1. Simple and composite attributes

➤ Simple Attribute:

- It cannot be divided further in more subparts.
- It is like undivided atomic value.
- Example: Price, Year, Enno, CPI

The text "CPI" is centered within a dark blue oval, which is superimposed on a large, light gray background watermark of the letters "PU".

CPI

Figure: 1.29 Simple Attribute

1. Simple and composite attributes

➤ Composite Attribute:

- It can be divided further in more subparts.
- It is an attribute composed of many other attributes.
- Example: Name

(first name, middle name, last name)

Address

(street, road, city)

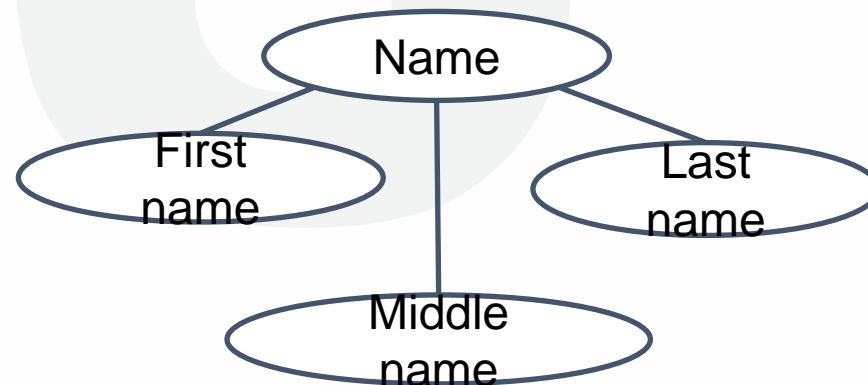


Figure: 1.30 Composite Attribute

2. Single-valued and multi-valued attributes

➤ Single-Valued Attribute:

- As name suggests it has single value only.
- Example: Enno, Birthdate

Birthdate

Figure: 1.31 Single-valued Attribute

2. Single-valued and multi-valued attributes

➤ Multi-Valued Attribute:

- It has multiple/more than one values.
- Example: PhoneNo

(person may have multiple phone nos)

EmailID

(person may have multiple emails)



Figure: 1.32 Multi-valued Attribute

3. Stored attribute and Derived attributes

➤ **Stored Attribute:**

- In this attribute value needs to be stored/defined manually.
- Example: Birthdate, Height, Weight

Height

Figure: 1.33 Stored Attribute

3. Stored attribute and Derived attributes

➤ **Derived Attribute:**

- Derived attribute value can be calculated or derived from other attributes.
- Example: Age (Can be derived from current date and birth date)



Figure: 1.34 Derived Attribute

4. Complex Attribute

- Attribute that are derived by nesting the composite and multivalued attributes are called complex attributes.

Address_phone((phone), address (H.no., city, state))

Complex Attribute

MV Attribute

Composite Attribute

5. Key Attribute

- Attribute which uniquely identifies each entity in the entity set is called key attribute.
- For example, RollNo will be unique for each student.
- It is denoted by an oval with underlying lines.



Figure: 1.35 Key Attribute

Example of All Attributes

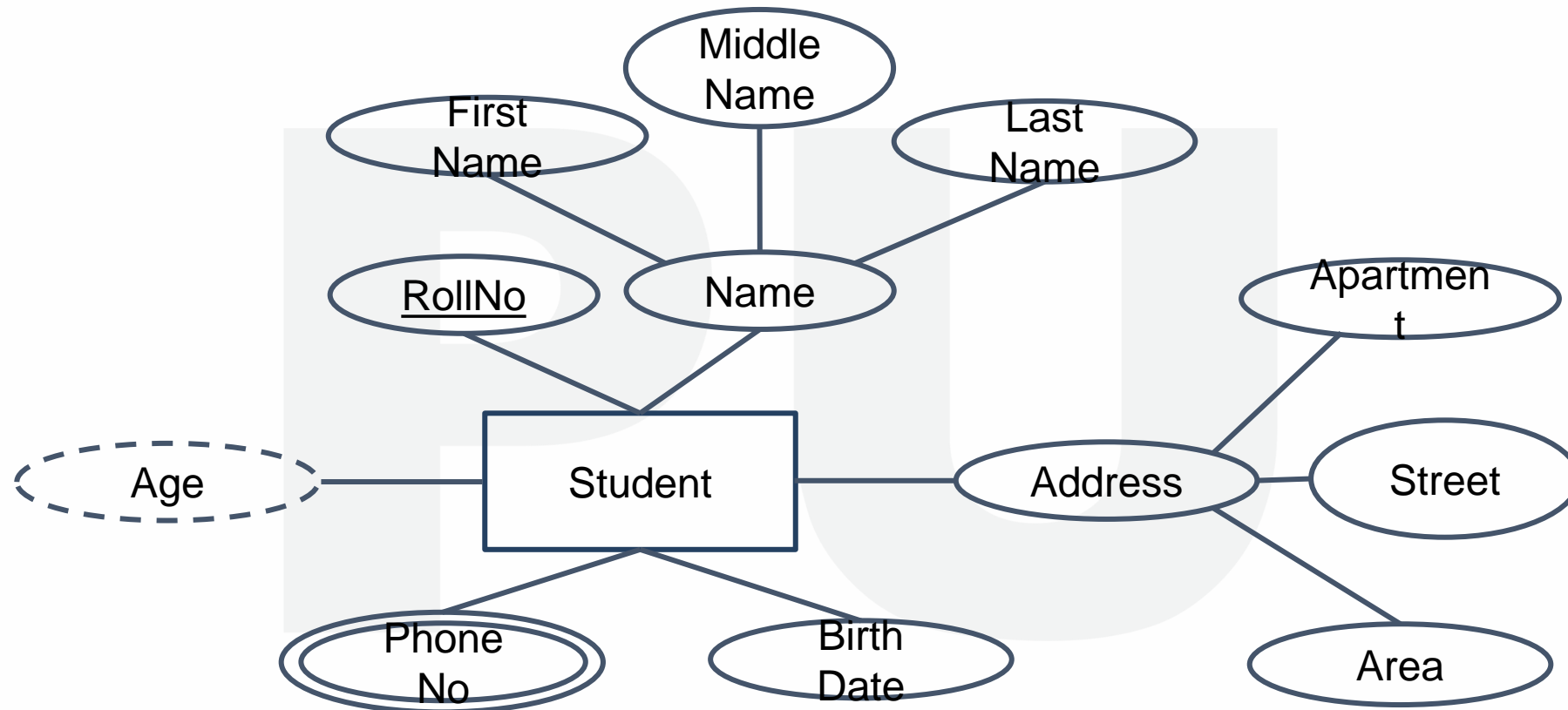


Figure: 1.36 An example of all Attributes

Descriptive Attributes

- If any relationship has a attribute like entity then its known as Descriptive Attributes.
- Example: Student had been issued degree certificate on 25/5/2020. Certificate date is an attribute of relationship.

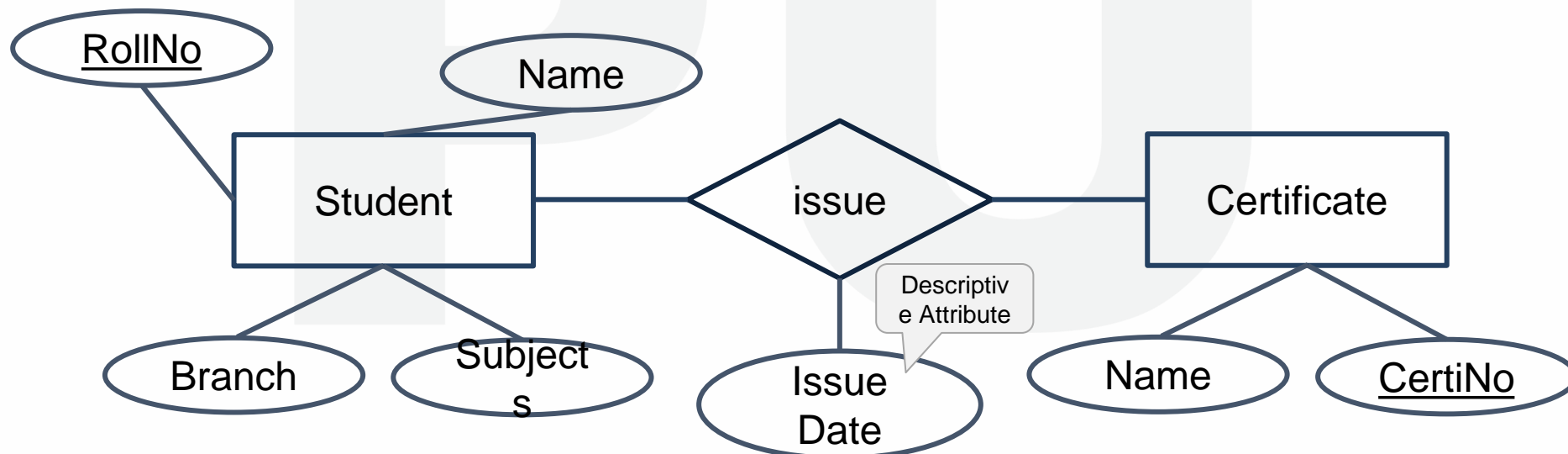


Figure: 1.37 Descriptive Attributes

Recursive Relationship Set

- When one entity set participate in a relationship for more than once then it is called recursive relationship set.

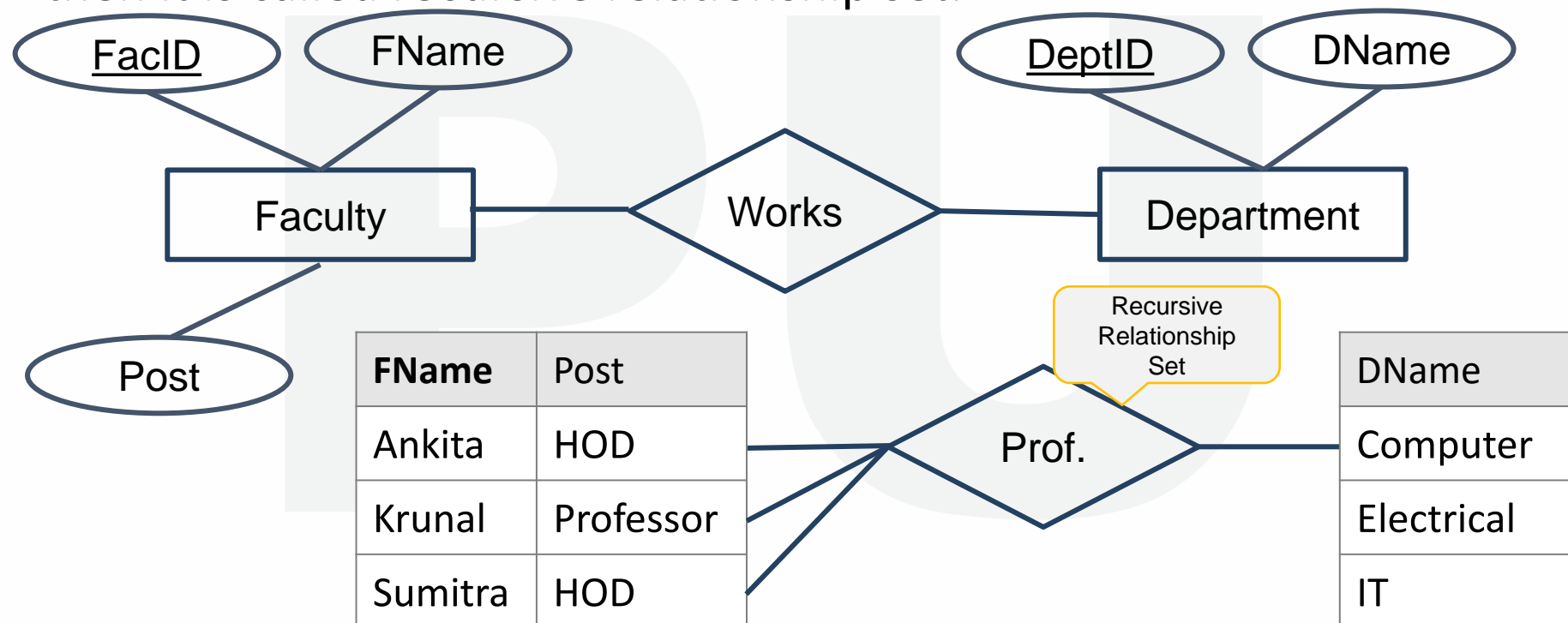


Figure: 1.38 Recursive Relationship set



Degree of a Relationship Set

- Number of entity sets joining in a relationship set is known as a degree of a relationship set.
1. Unary Relationship
 2. Binary Relationship
 3. n-ary Relationship

Degree of a Relationship Set

1. Unary Relationship:

- When only one entity set participating in a relationship is called Unary Relationship.
- Example: One person is married to only one person.

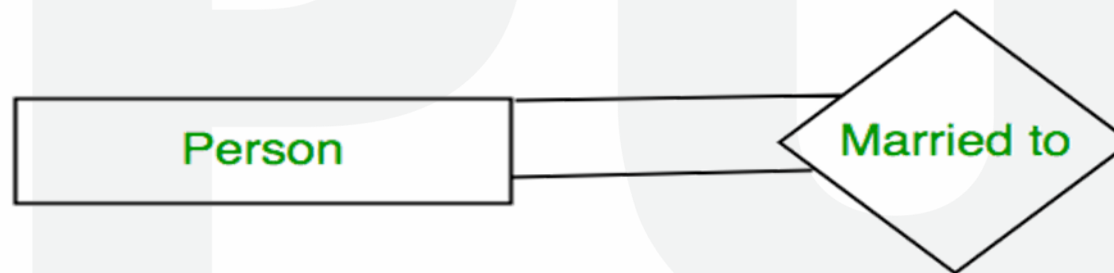


Figure: 1.39 Unary Relationship

Degree of a Relationship Set

2. Binary Relationship:

- When Two entity set participating in a one relationship is called Binary Relationship.
- Example: Student is Enrolled in Course



Figure: 1.40 Binary Relationship

Degree of a Relationship Set

3. N-ary / Ternary Relationship:

- When there are 3 or N entity set participating in a relationship is called Ternary / N-ary Relationship.

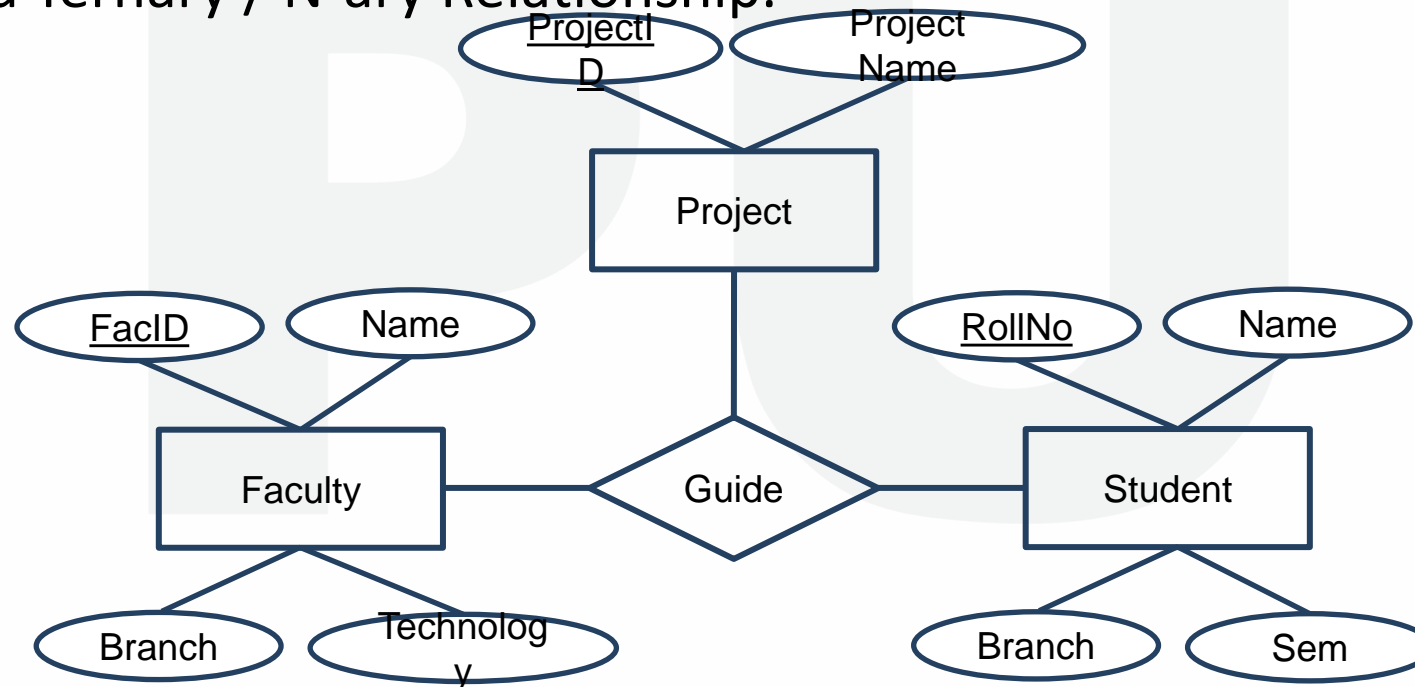


Figure: 1.41 Ternary Relationship



Cardinality Constraints (Mapping Cardinality)

- It defines numbers of times an entity of another entity set participate in a relationship set.
- It helps in defining binary relationship sets.
- Mapping Cardinality for binary relationship can be following types:
 1. One to One
 2. One to Many
 3. Many to One
 4. Many to Many

Cardinality Constraints (Mapping Cardinality)

1. 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.

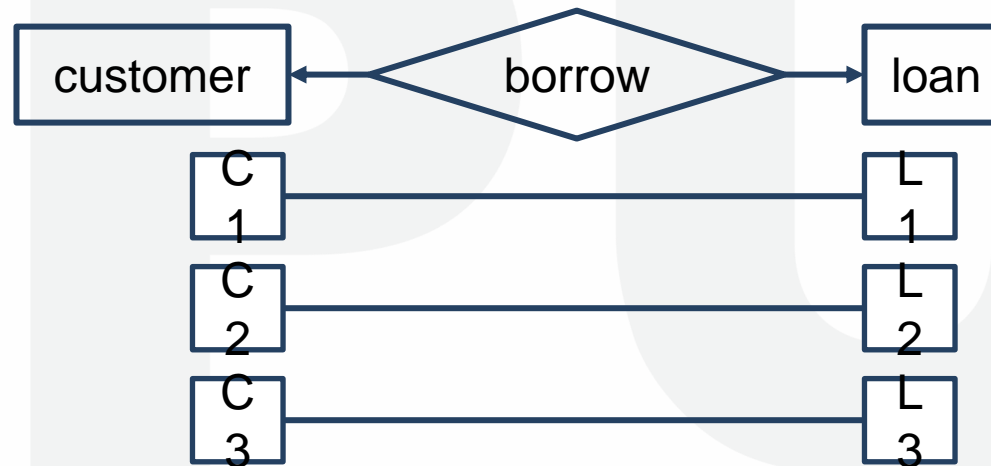


Figure: 1.42 1-1 Relationship

- **Example:** One customer is connected with only one loan through borrower relationship and loan is connected to one customer using borrower.

Cardinality Constraints (Mapping Cardinality)

2. One to Many Relationship(1-M):

- 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.

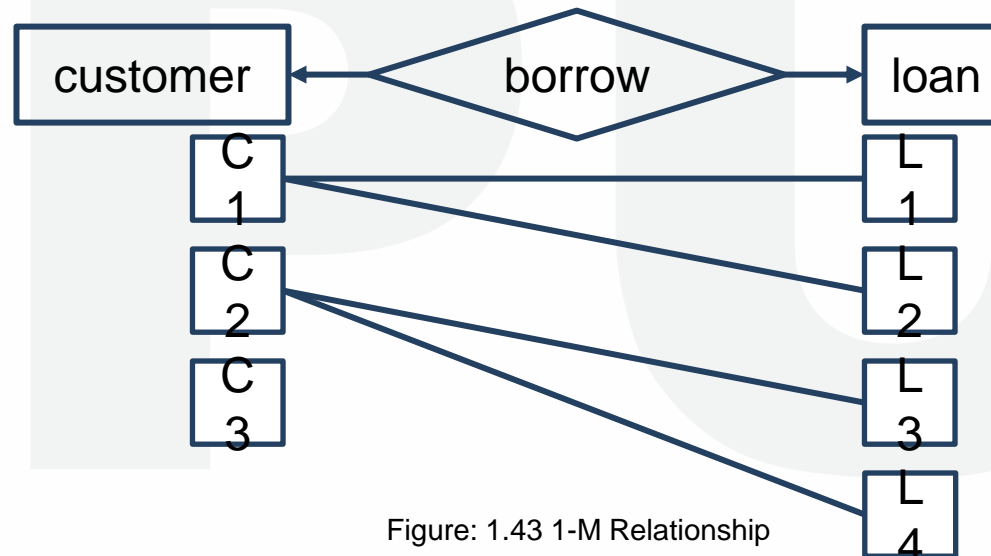


Figure: 1.43 1-M Relationship

- **Example:** Loan is connected with only one customer through borrower relationship and customer is connected with more than one loan.

Cardinality Constraints (Mapping Cardinality)

3. Many to One Relationship(M-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.



Figure: 1.44 M-1 Relationship

Cardinality Constraints (Mapping Cardinality)

4. Many to Many Relationship(M-M):

- 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.



Figure: 1.45 M-M Relationship



Weak Entity Set

- Any Entity set that does not have a primary key of own is known as Weak Entity Set. This entity is known as Dependent Entity.
- Any Entity that has a key attribute is strong entity type and known as a Independent Entity.
- Existence of a weak entity set depends on the existence of a strong entity set.
- Weak Entity set is denoted by double rectangle.
- Weak Entity Relationship set is denoted by double diamond

Weak Entity Set

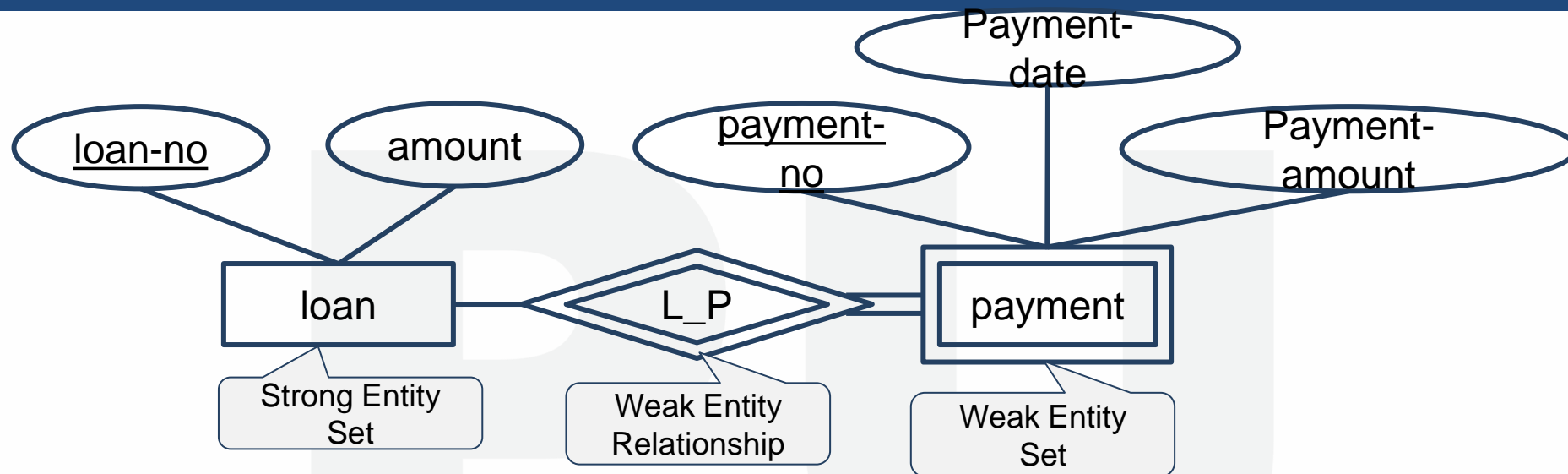


Figure: 1.46 Weak Entity Set

- Primary key of a weak entity set is known as discriminator (partial key), it's created by combining primary key of strong entity set.

Weak Entity Set

- Here Loan-No is primary key of a Loan entity.
- Payment-no is discriminator of payment entity.
- Primary key for Payment is (loan-no, payment-no)
- Discriminator attribute of weak entity set is denoted with dashed underline.

Super Class & Sub Class

- A **superclass** is an entity from which another entities can be derived.
 - Example: an entity person has two subsets employee and teacher
 - Here person is superclass.
- A **subclass** is an entity that is derived from another entity.
 - employee and teacher entities are derived from entity account.
 - Here employee and teacher are subclass.

Super Class & Sub Class

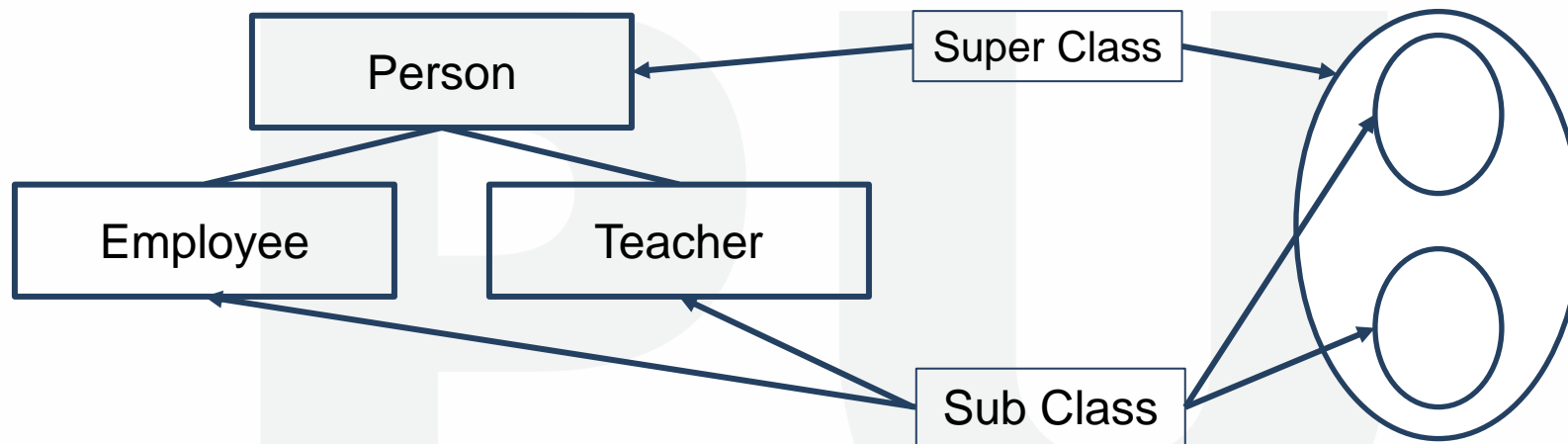


Figure: 1.47 Super & Sub class



Generalization & Specialization

➤ Generalization:

- It determines the common features of multiple entities to create a new entity.
- Generalization is a process of creating group from several entities.
- It follows a bottom-up approach.
- It's like union of two or more lower level entity sets to make higher level entity set.

Generalization & Specialization

➤ Generalization:

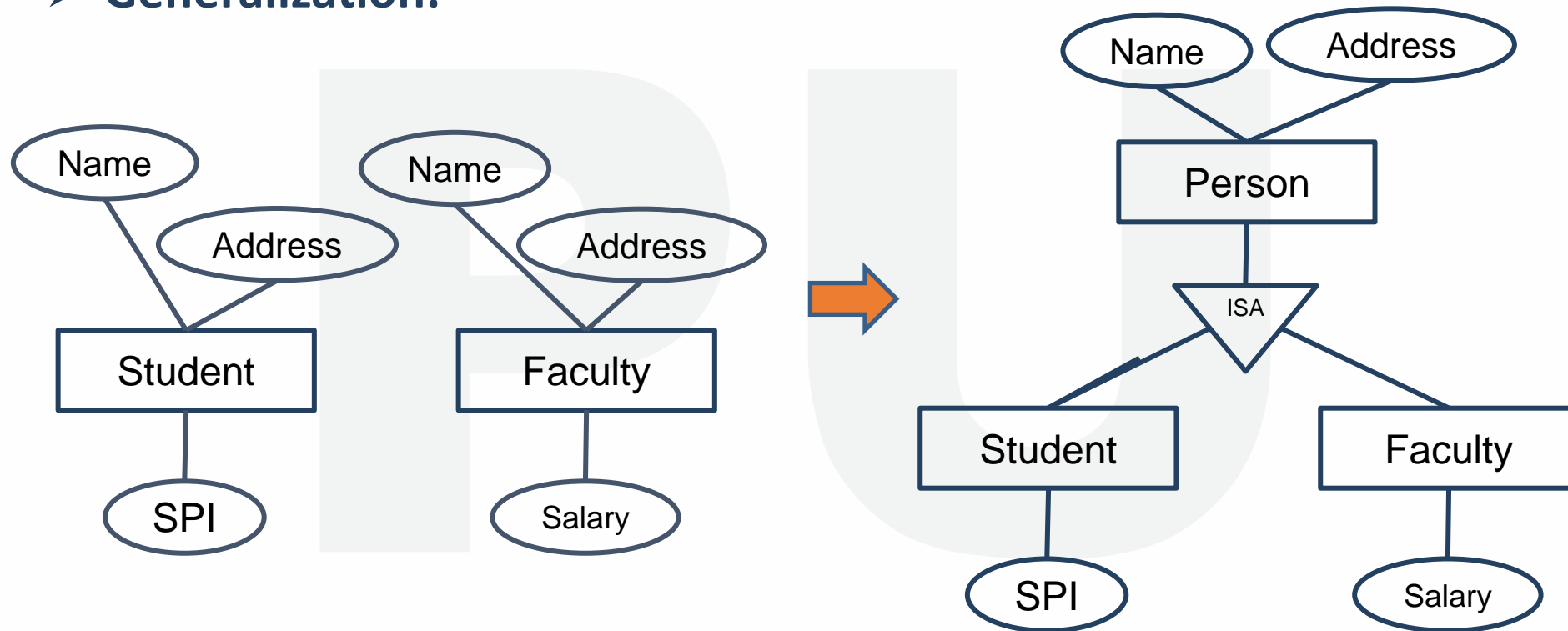


Figure: 1.48 Generalization



Generalization & Specialization

➤ Specialization:

- It divides entity to make multiple entities that inherits some features of splitting entity.
- It is a process of creating subgroups within entities.
- It follows a top-down approach.
- It's like subset of higher level entity set to form a lower level entity set.

Generalization & Specialization

➤ Specialization:

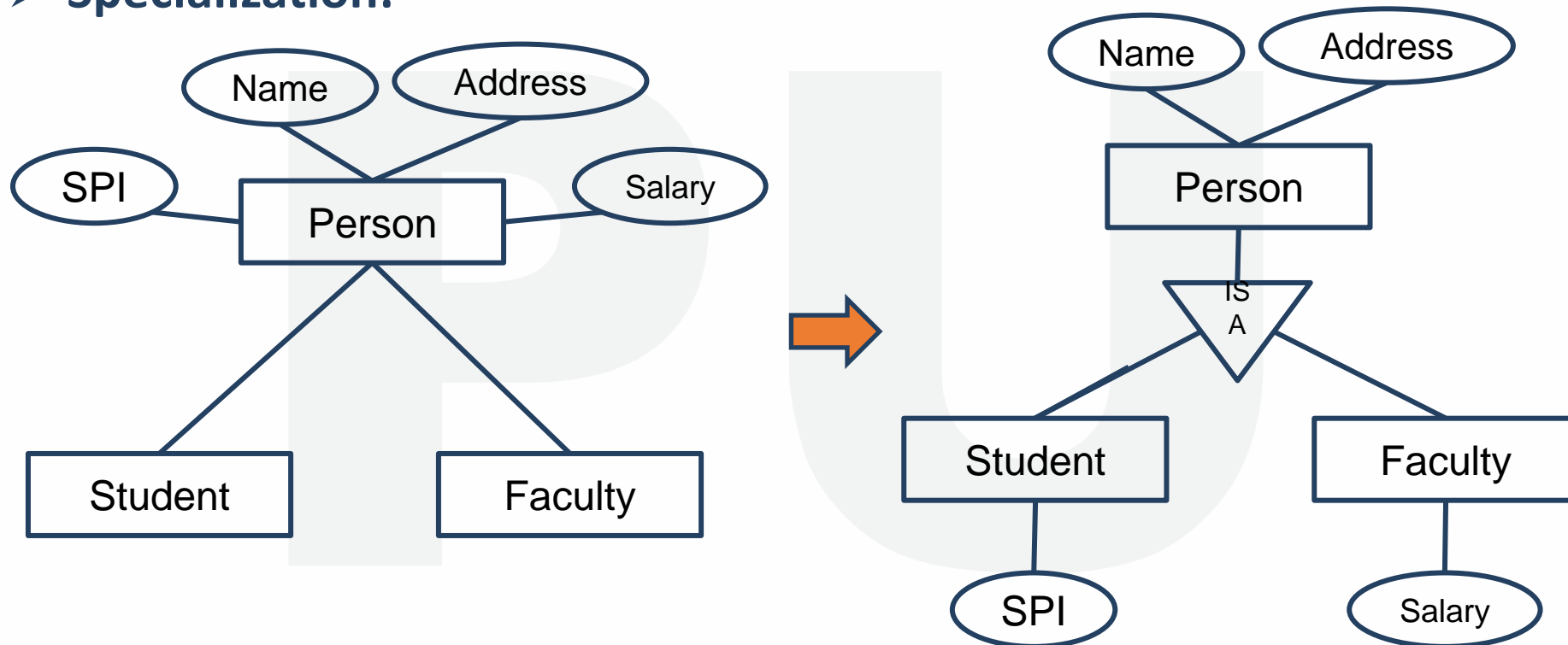


Figure: 1.49 Specialization

Example

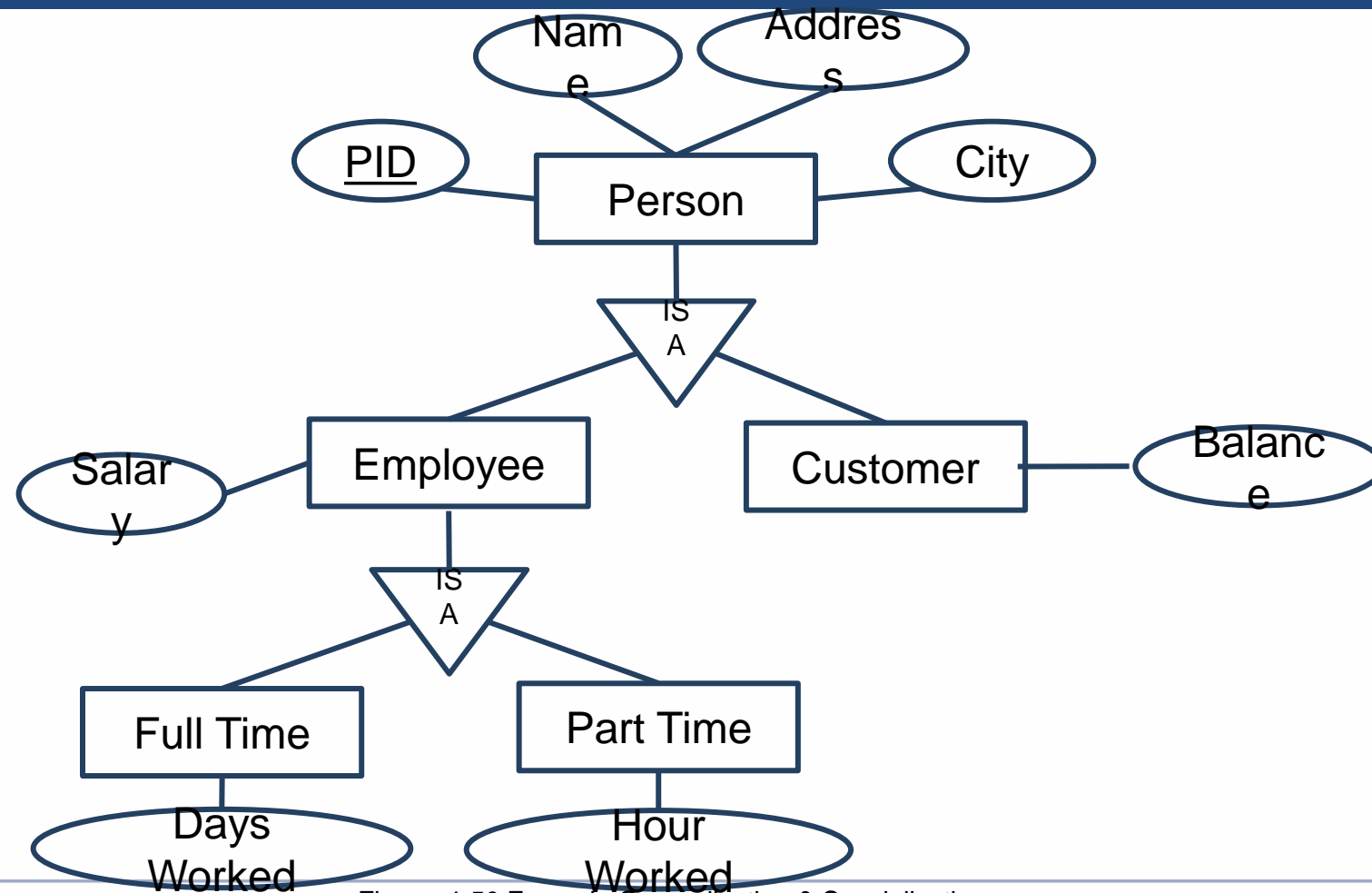


Figure: 1.50 Example Generalization & Specialization



Constraints on Specialization and Generalization

- Constraints are normally divided in two types
 1. Disjoint Constraint
 2. Participation Constraint



Disjoint Constraints

- It defines relationship of members of superclass and subclass and also indicates member of superclass can be a member of one or more subclass.
 1. Disjoint constraint
 2. Non-disjoint(Overlapping) constraint

Disjoint Constraint

1. Disjoint Constraint

- It defines entity of a super class can belong to only one sub class entity set
- It is denoted by **d** or **disjoint** near ISA triangle.

2. Non-Disjoint Constraint:

- It defines entity of a super class can belong to more than one sub class entity set.
- It is denoted by **o** or **non-disjoint** near ISA triangle.

Example

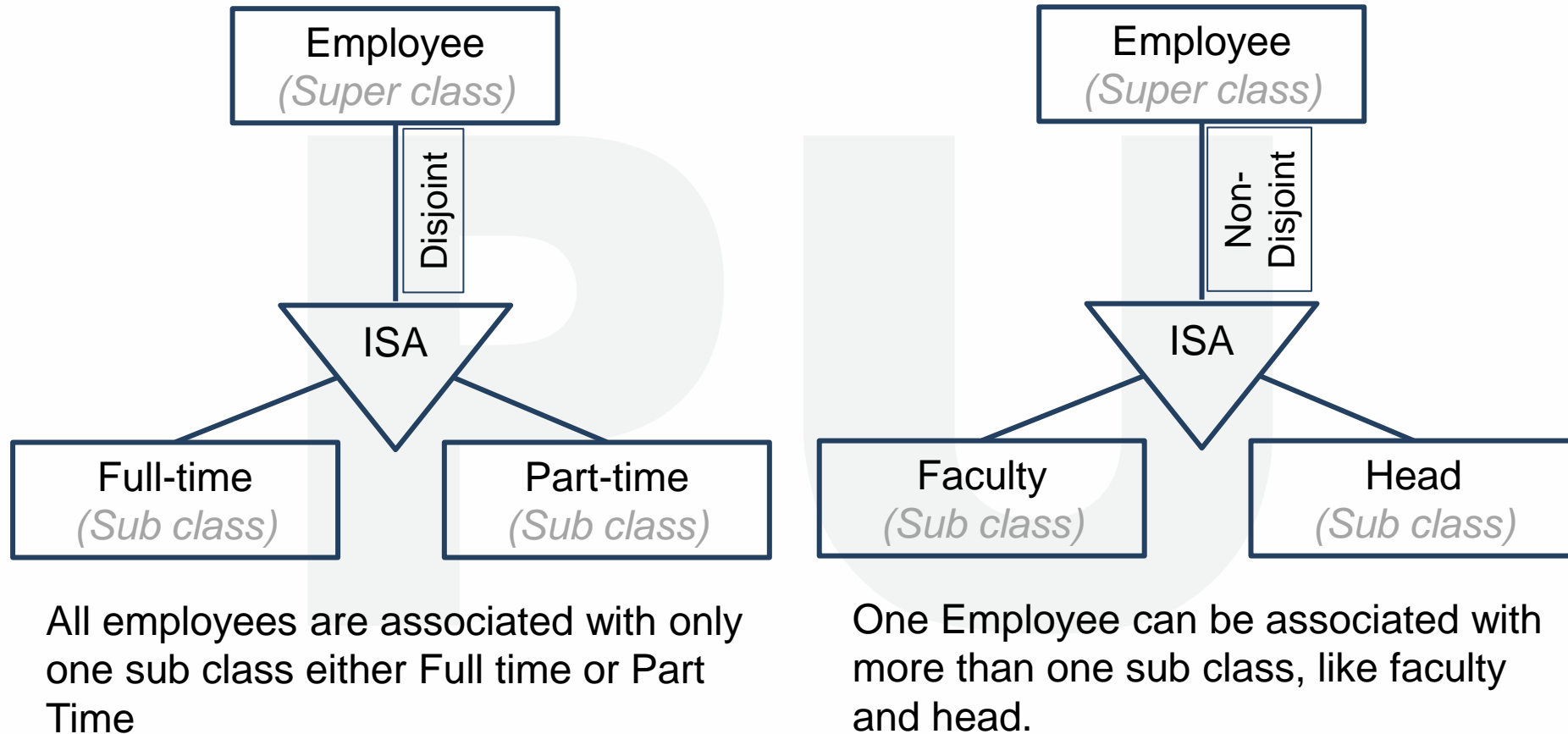


Figure: 1.51 Example of Disjoint & Non-Disjoint Constraint

Participation(Completeness) Constraints

- It defines every member of super class must participate as a member of subclass or not.
- It defines how much entity set participates in a relationship set.
 1. Total(Mandatory) Participation
 2. Partial(Optional) Participation

Participation Constraints

1. Total Participation:

- Every entity in the entity set participates in at least one relationship in the relationship set such participation is total.
- Every entity of superclass must be a member of subclass.
- It is denoted by double line connected with relationship

2. Partial Participation:

- Some entities in the entity set may not participate in any relationship in the relationship set such participation is partial.
- Every entity in super class does not belong to any of the subclass.
- It is denoted by single line connected with relationship.

Participation Constraints

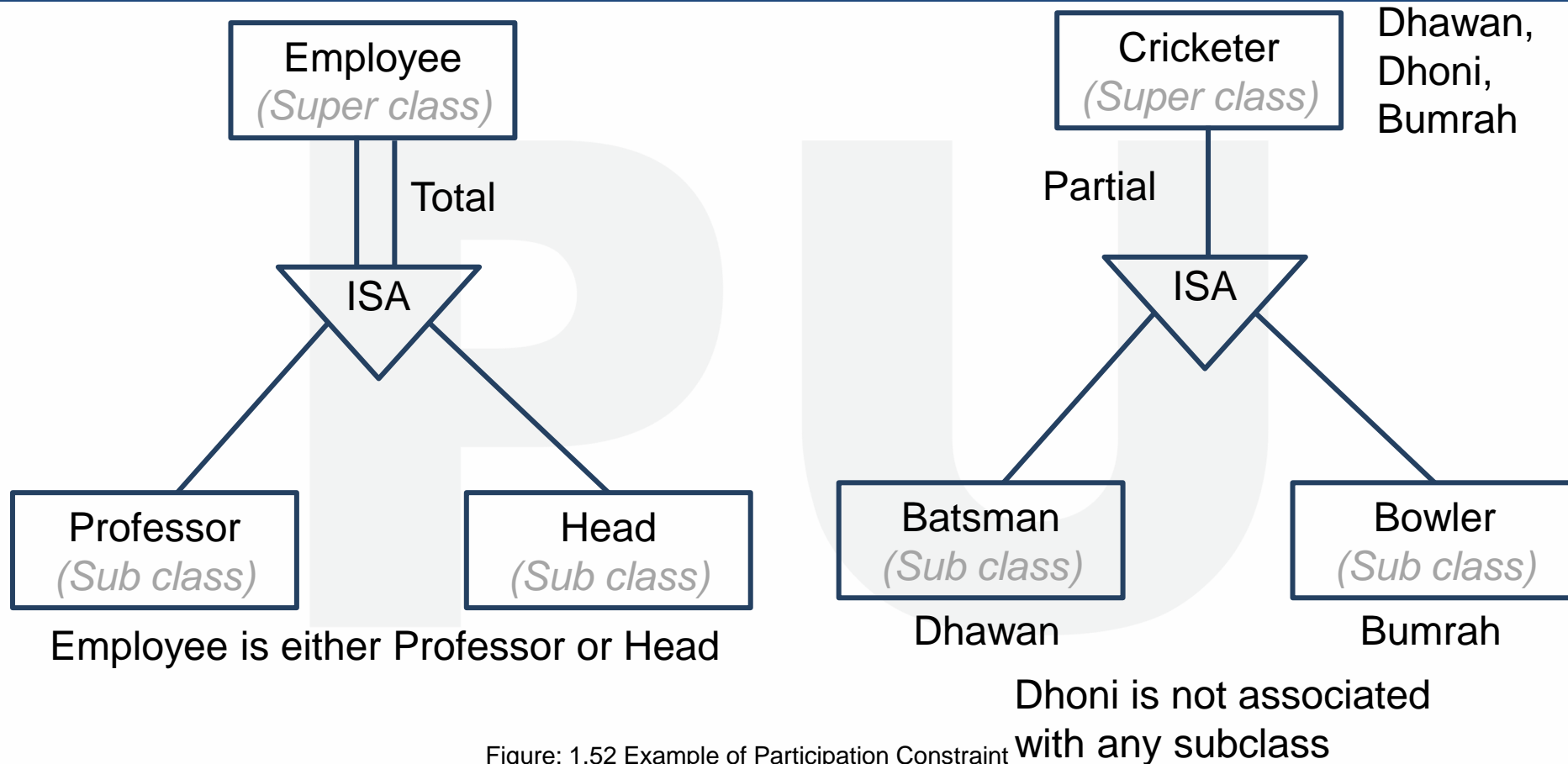


Figure: 1.52 Example of Participation Constraint

Participation Constraints

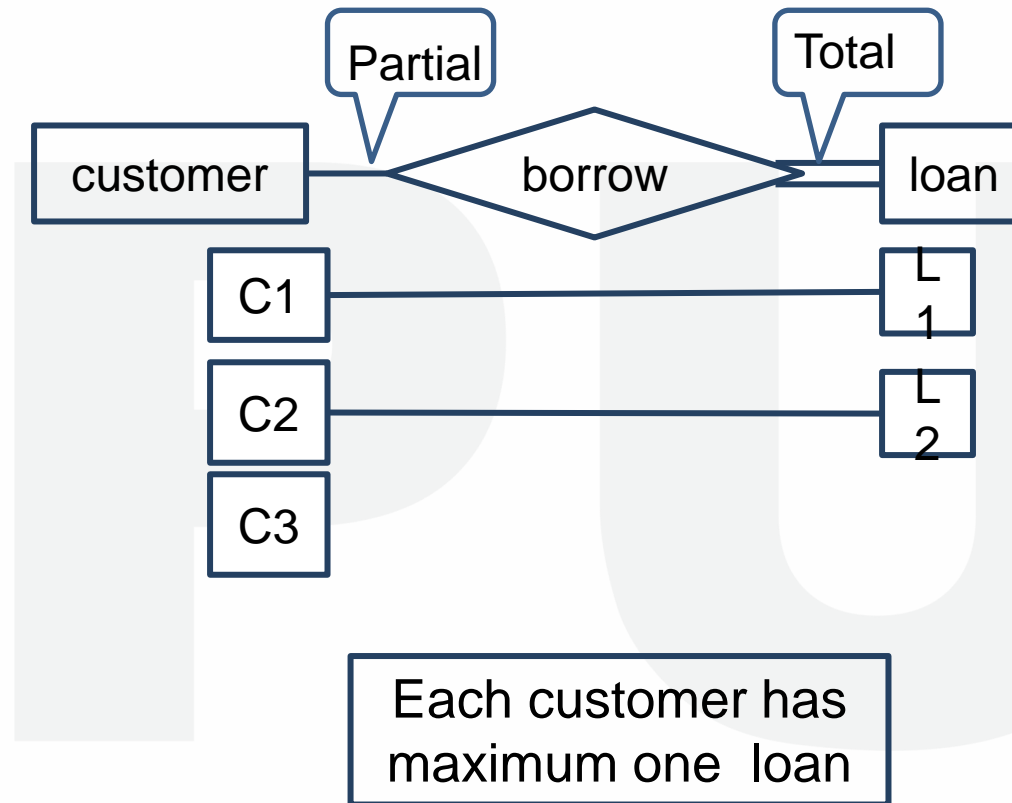


Figure: 1.53 Example of participation Constraint

Aggregation

- Normally in E-R model relationship between entities are possible to define, but relationships between two relationships defining is not possible, that's limitation here.
- Aggregation is kind of abstraction which treats relationships as entities.
- Aggregation is process of creating single entity by combining components & relationship between two entity of ER model.

Aggregation

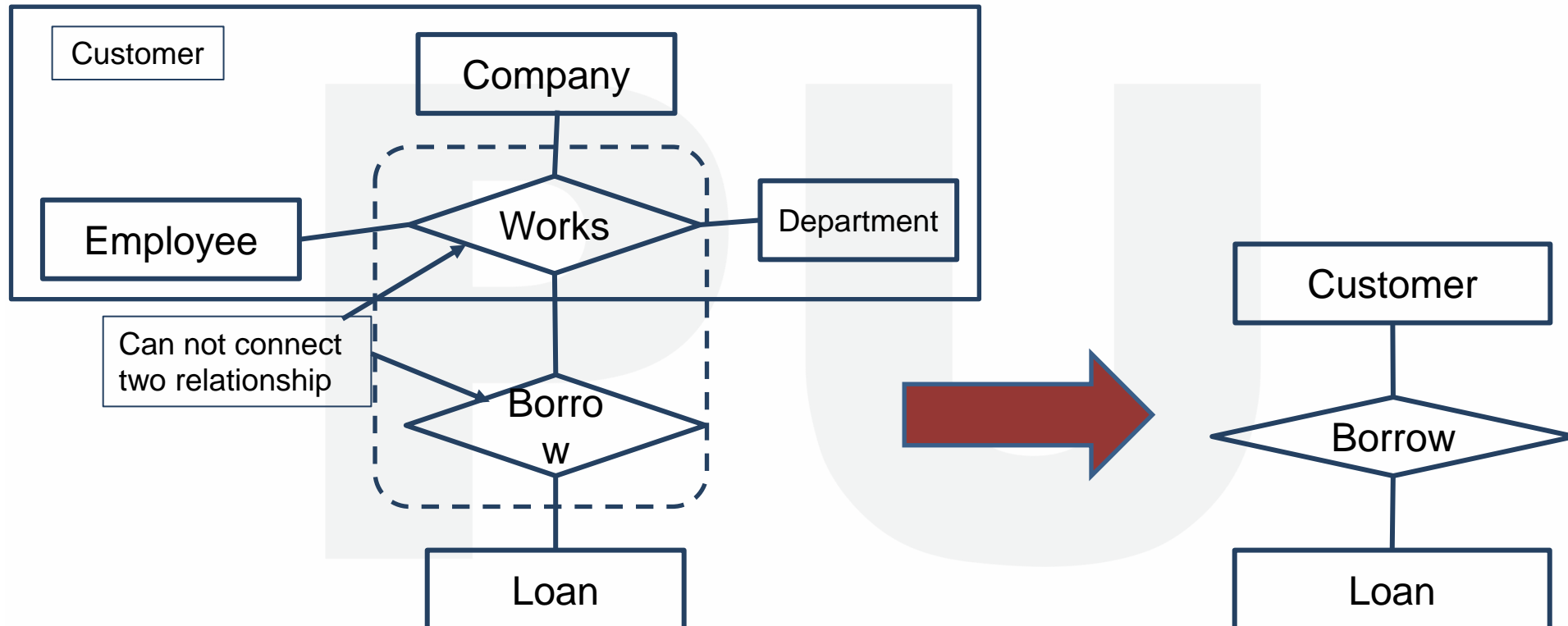


Figure: 1.54 Example of Aggregation



E-R Model Symbols

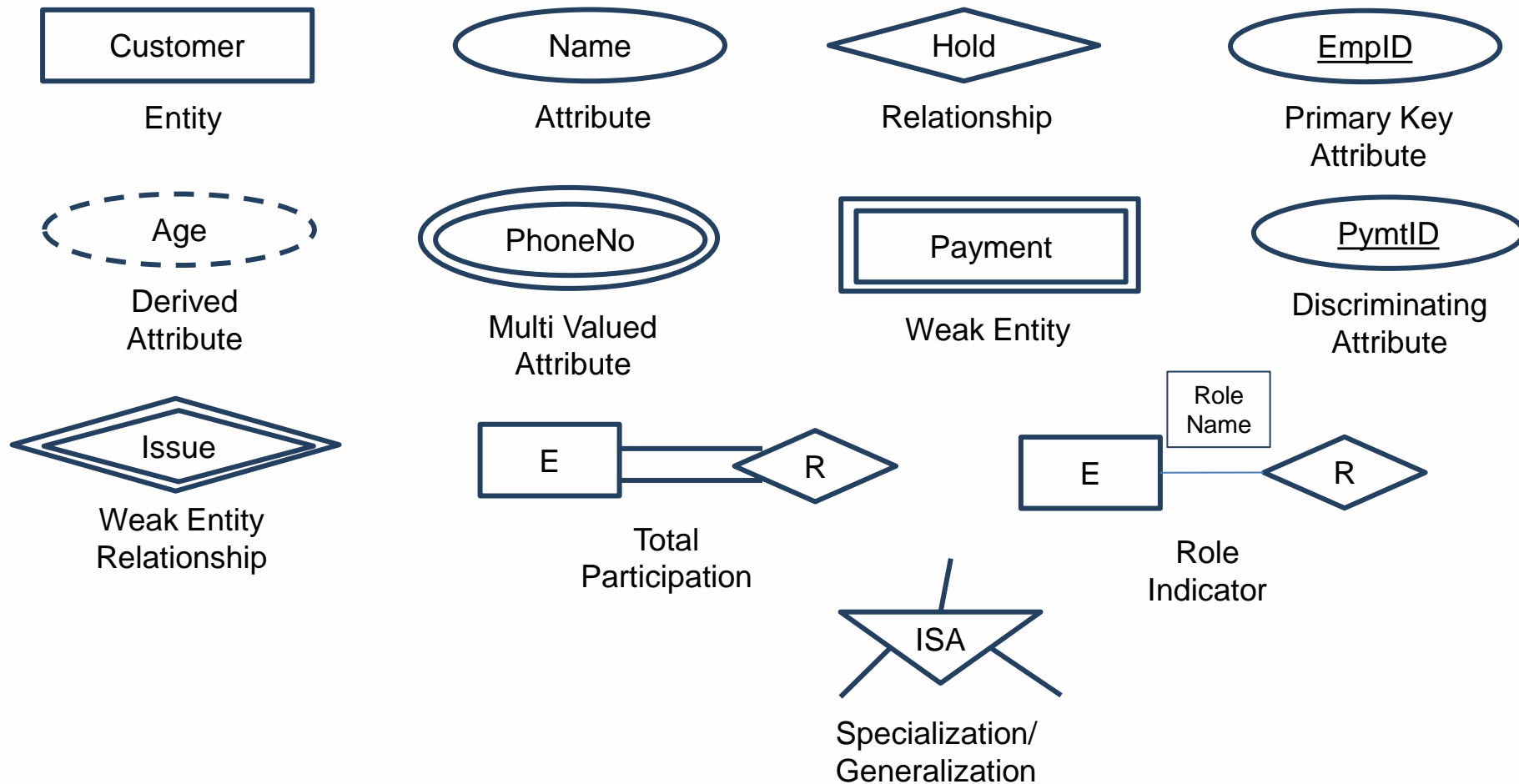


Figure: 1.55 E-R Model Symbols

Hierarchical Model

- It organize data into a tree like structure having one root or parent.
- Here data starts from room with one to many kind of relationship

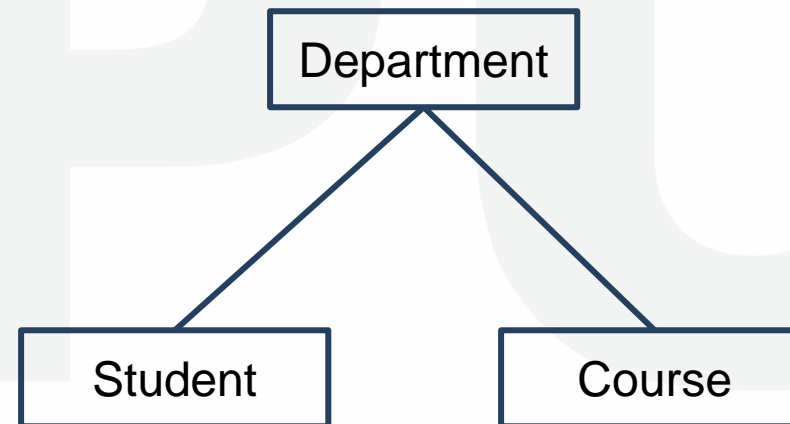


Figure: 1.56 Hierarchical Model

Network Model

- It is an extension of hierarchical model, with many to many kind of relationship in tree structure with multiple parents.

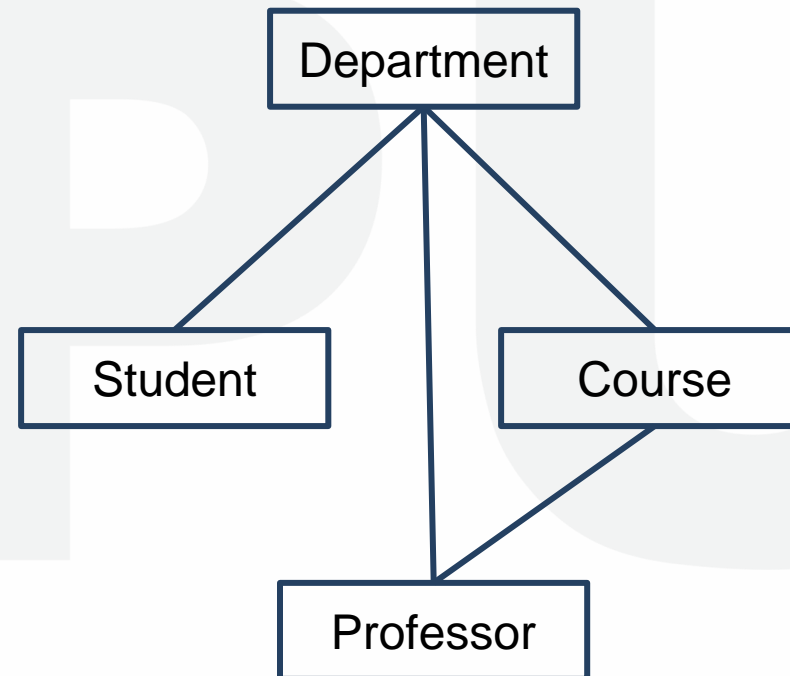


Figure: 1.57 Network Model

Relational Model

- In this model data is organized in two-dimensional tables and relationship is retained by storing a common attribute.
- **Tables:** Relations are saved in the format of tables. Table has rows and columns. Here rows denotes records and columns denotes attributes.
- **Tuple:** A single row of a table which has a single record value is known as tuple .

Relational Model

- **Relation Instance:** It is a finite set of tuples in a relation.
- **Relation schema:** It describes relation name (table name), attributes, and their names.
- **Relation Key:** Each tuple has one more attributes that identifies relation uniquely is known as relation key.
- **Attribute Domain:** It is a predefined value scope of a every attribute.

Relational Model

Table also called **Relation**

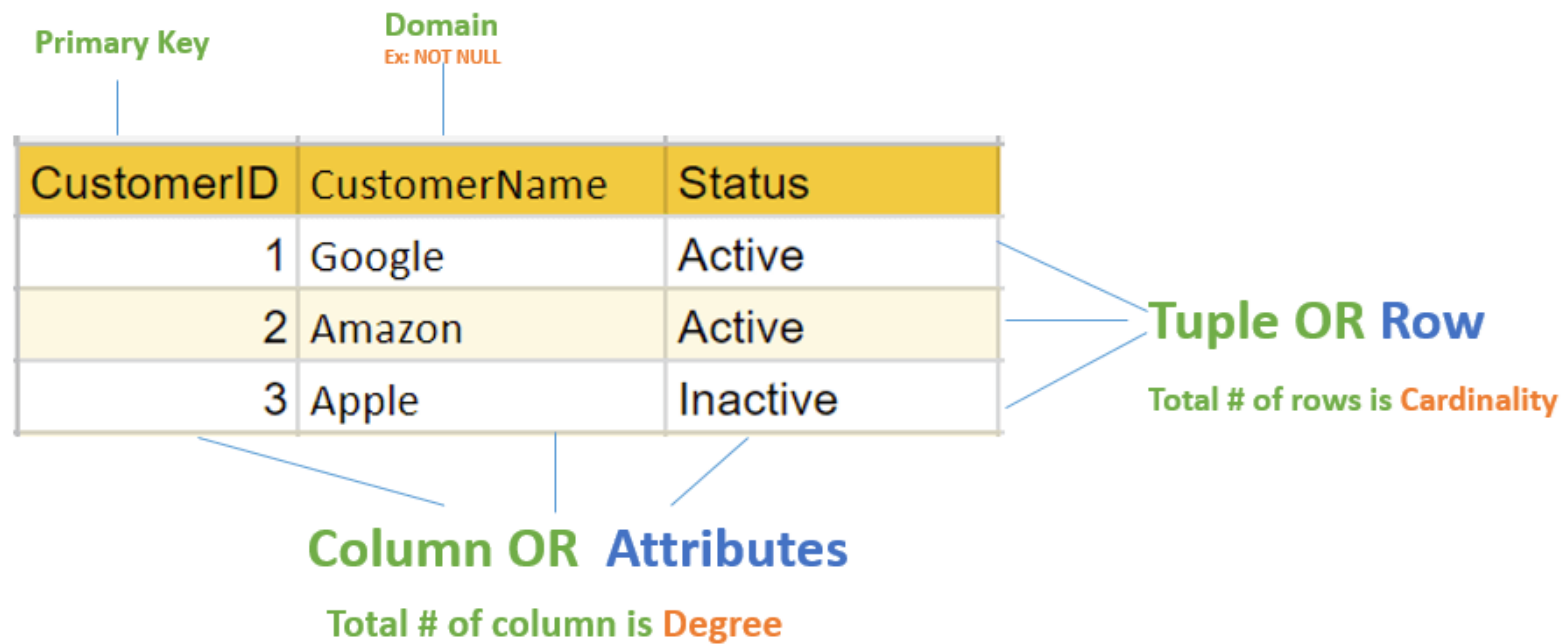


Figure: 1.58 Example Relational Model

(Image Source: <https://www.guru99.com/relational-data-model-dbms.html>)

Relational Model

<u>Rno</u>	Student_Name	Age
101	Dilen Panchal	22
102	Sanket Patel	21

<u>SubID</u>	Subject_Name	Teacher
1	DBMS	Kiran
2	OS	Abhijit

<u>ResID</u>	Rno	SubID	Marks
1	101	1	84
2	101	2	70
3	102	1	85
4	102	2	74

Foreign
Key

Foreign
Key



Object-Oriented Model

- This model follows the method of representing real world objects.
- Objects are derived from real world entities and situations.
- Each object has their own properties that are defined as attribute and their behavior is defined as methods.
- Object is like an instance of class. So similar attributes and methods are grouped together as a class.

Object-Oriented Model

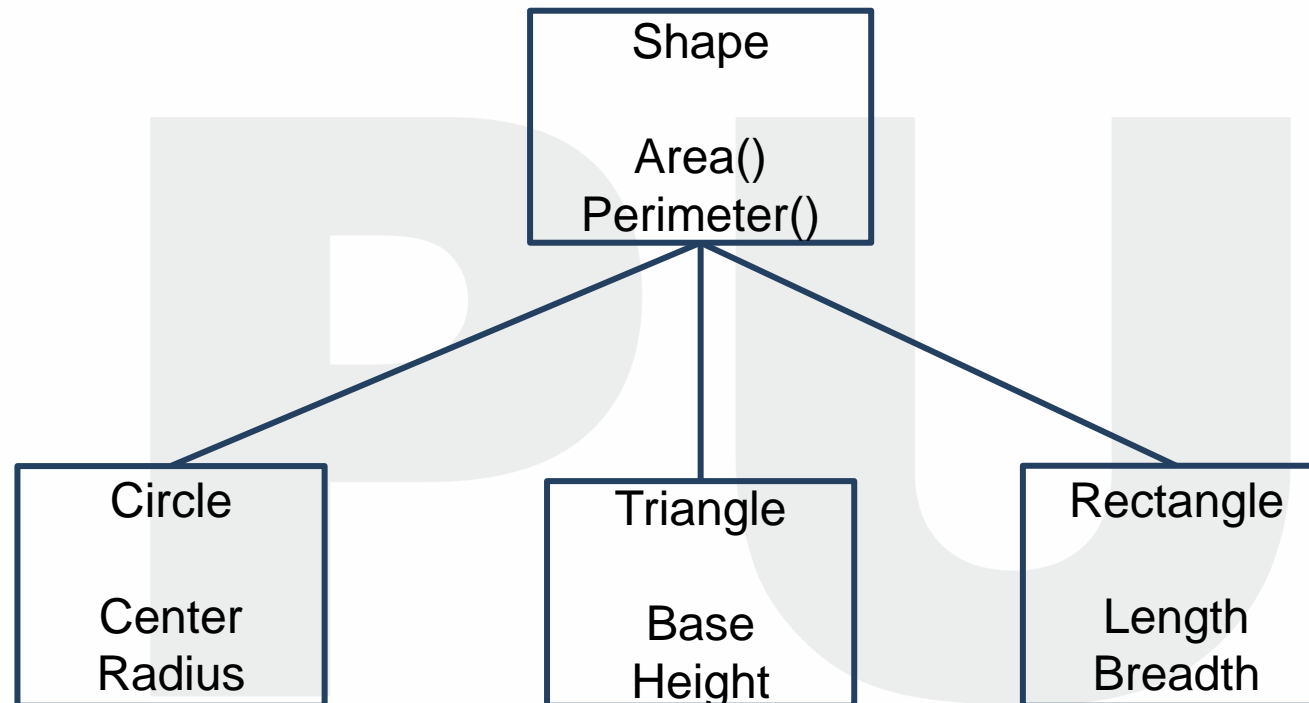


Figure: 1.60 Example Object-Oriented Model

Integrity Constraints

- Integrity constraints means set of rules defined in database system. It is used to maintain the quality of information.
- It ensures that any data related operation performed on database like, data insertion, updating does not affect integrity of data.
- It provide protection against any damage to database.
- Following are different types of Integrity Constraints:
 1. Check
 2. Not null
 3. Unique
 4. Primary key
 5. Foreign key

Integrity Constraints

1. Check:

- Check integrity constraint defines a specific rule for a column, all the rows of that column must have to satisfy it.
- Check controls the tuple values to some set, range or specific value.
- It can be applied on more than one column of a relation(table).
- Example: Tuple value of CGPA should be between 0 to 10



Integrity Constraints

2. Not Null

- As name suggests all the rows of relation(table) should be some definite value for the column on which Not null is applied.
- Simply when we apply this constraint on any column, it should have some value.
- Example: For relation *Student*, Name column should have some value.

Integrity Constraints

3. Unique


- As name defines it make sure that columns or group of columns in each row of table has unique (distinct) value.
- Columns can have null values, but they can't be duplicated.
- This constraint sometimes also referred as Unique Key.
- Example: For relation *Student*, Enrollmentno column should have unique value.

Integrity Constraints

4. Primary Key

- Primary key applies to column or combination of columns to uniquely identifies each row in the table.
- Primary Key = Unique key + Not null
- There can be only one primary key per table
- Example: For student relation table, **EnrollmentNo** column should have unique value and can't be null.

Primary Key

A blue curved arrow pointing from the text 'Primary Key' to the 'EnrollNo' column header.

<u>EnrollNo</u>	Student_Name	DeptID
001	Ramesh	1
002	Suresh	2
003	Mahesh	1

Figure: 1.61 Primary Key



Integrity Constraints

5. Foreign Key

- Foreign key is defined to link two tables(relation).
- Foreign key is set of one or more attributes whose value is derived from the primary key of another relation.
- Foreign key is also known as referential integrity.
- Attribute or column which is declared as foreign key in Table 1 is derived from primary key of Table 2 and every value of foreign key column in table 1 must be null or available in primary key of table 2.

Integrity Constraints

5. Foreign Key

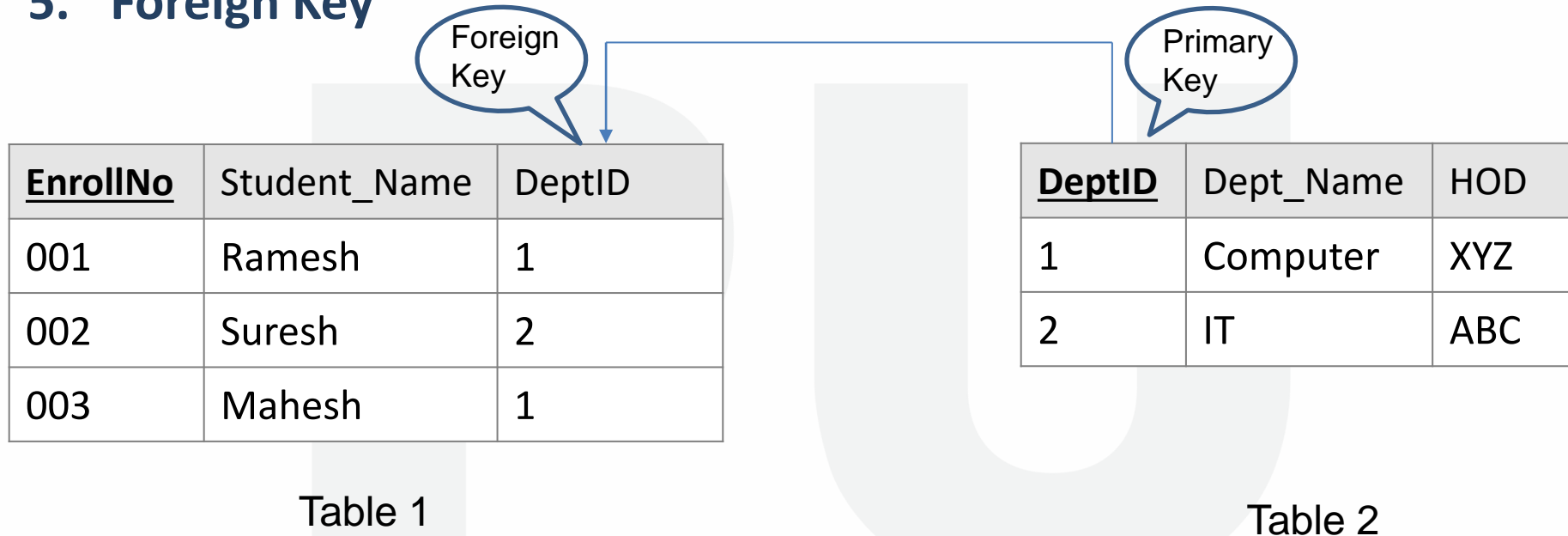


Figure: 1.62 Example Foreign Key

Some other keys

- **Super Key:** It is a set of one or more columns that identifies each tuple record uniquely in a relation.

Employee

Emp_SSN	Emp_Number	Emp_Name
123	101	Ramesh
456	102	Suresh
789	103	Dinesh
791	104	Mahesh

Figure: 1.63 Super Key

- Here entity sets {Emp_SSN}, {Emp_Number}, {Emp_SSN, Emp_Number}, {Emp_SSN, Emp_Name}, {Emp_SSN, Emp_Number, Emp_Name}, {Emp_Number, Emp_Name} are uniquely identifies rows so they all are super keys.

Some other keys

- **Candidate Key:** It is subset or part of the super key.
 - Sometimes they're also known as minimal super key with no repeated attributes.
 - Each table has minimum one candidate key, but there can be multiple candidate keys also.
 - One Primary key is selected from candidate keys.
 - Example: {Emp_SSN} , {Emp_Number}
- **Alternate Key:** Any candidate key that's not chosen as a primary key is alternate key.

References

- [1] Abraham Silberschatz, Henry F. Korth and S. Sudarshan, Database System Concepts, McGraw-Hill Education (Asia), Seventh Edition, 2019.
- [2] C. J. Date, A. Kannan and S. Swamynathan, An Introduction to Database Systems, Pearson Education, Eighth Edition, 2009.
- [3] Database Management Systems, CSE, DIET,
<https://www.darshan.ac.in/DIET/CE/GTU-Computer-Engineering-Study-Material>
- [4] Database management systems by Raghu Ramakrishnan and Johannes Gehrke
<http://pages.cs.wisc.edu/~dbbook/openAccess/thirdEdition/slides/slides3ed.html>
- [5] Database management system tutorial,
<https://www.tutorialspoint.com/dbms/index.htm>

× ○ DIGITAL LEARNING CONTENT



Parul[®] University



www.paruluniversity.ac.in