Data Models

ER Model

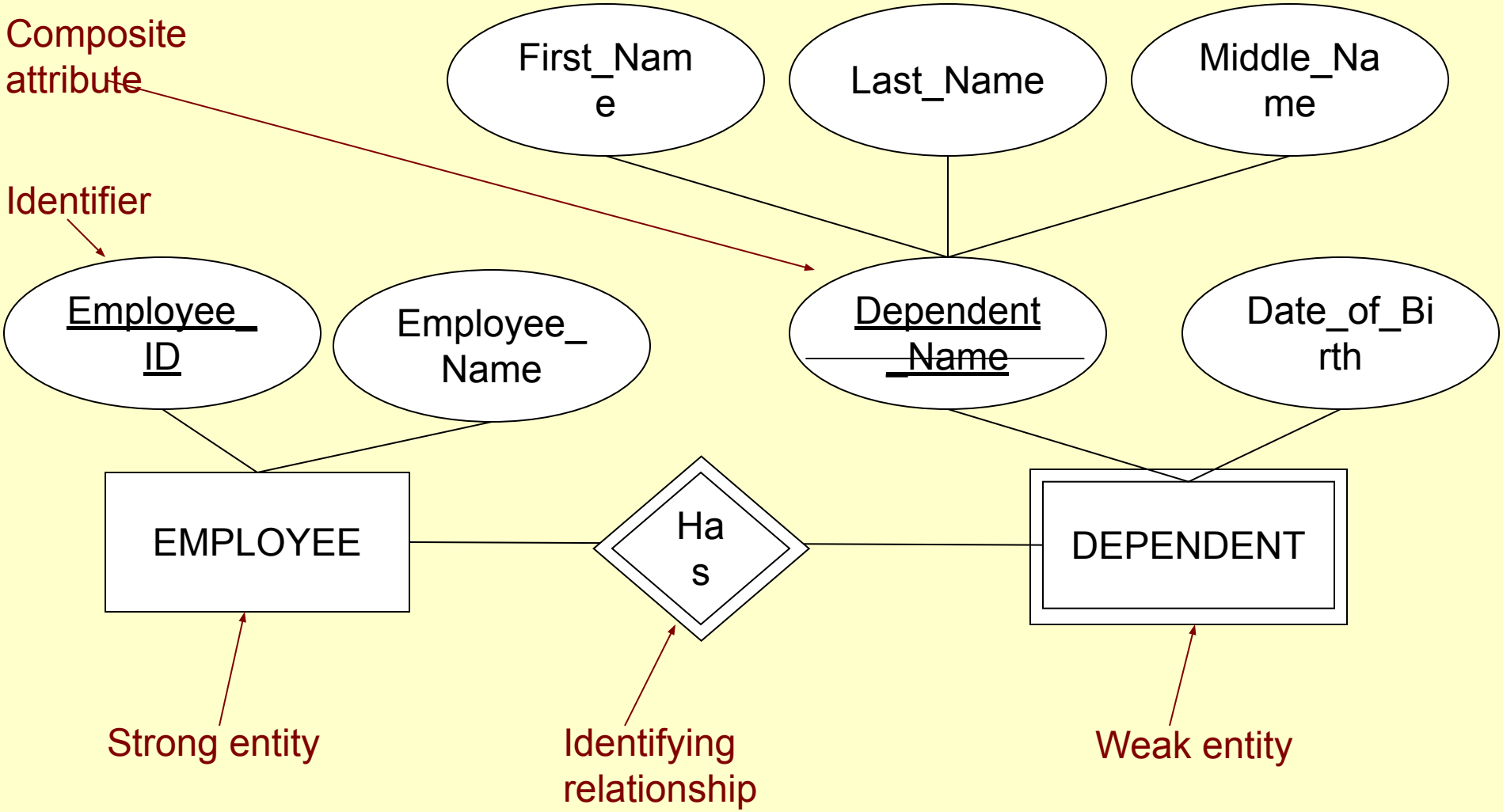
Existence Dependency

- An instance of one entity can't exist without the existence of some other related entity.
- An entity having an existence dependency is referred as Weak Entity and it may optionally be represented by double rectangle.
- Weak entities often don't have a natural identifier and the PK of the parent entity is used as part of the PK of the weak (child) entity.
- A relationship between a parent/strong entity and a weak entity is called Identifying Relationship.
- It offers the following benefits:

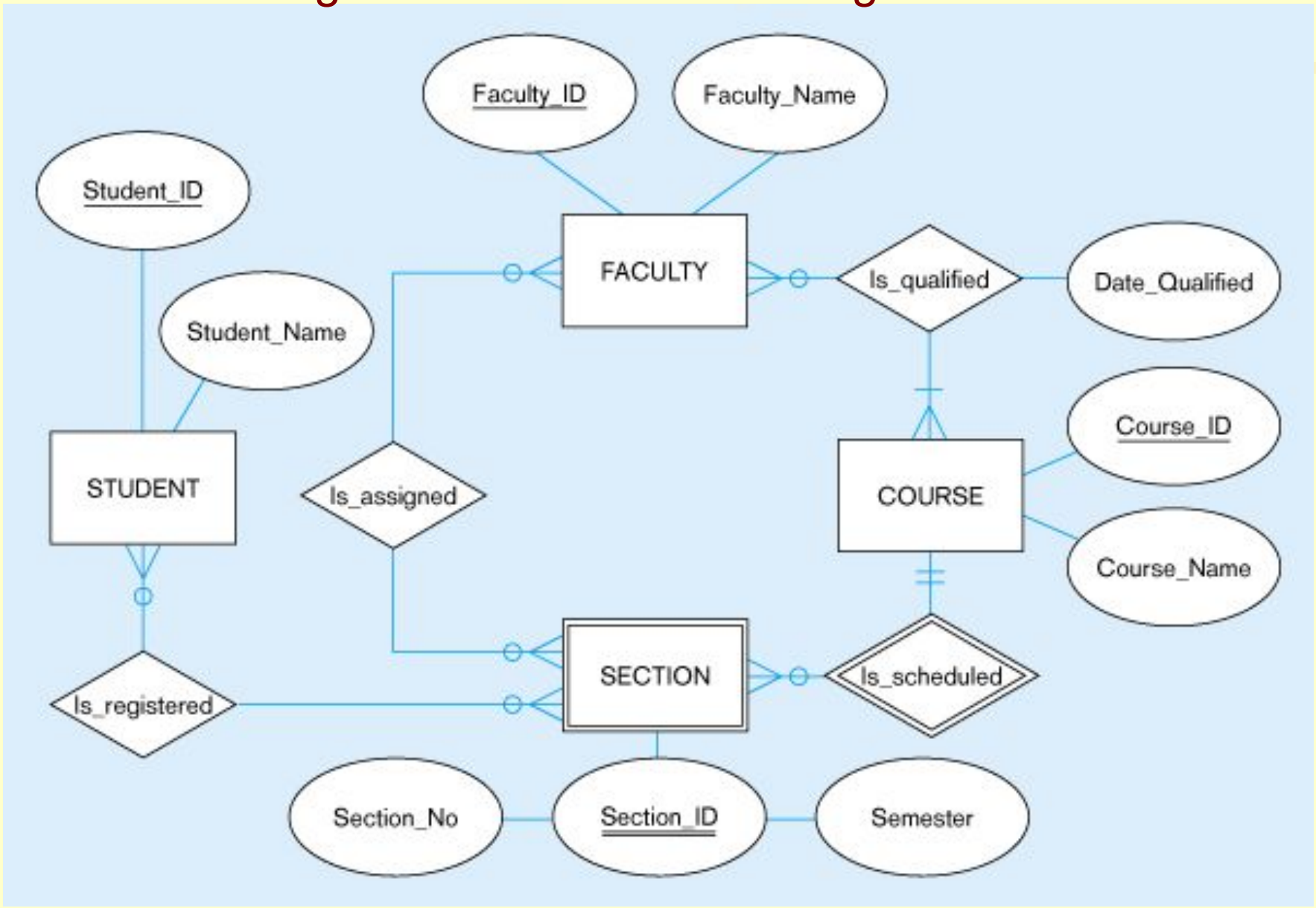
a) Data Integrity

b) Ease of Access of the Dependent Entity

Example


Composite attribute
First_Name
Last_Name
Middle_Name
Identifier
Employee_ID
Employee_Name
Dependent_Name
Date_of_Birth
EMPLOYEE
Has
DEPENDENT
Strong entity
Identifying relationship
Weak entity

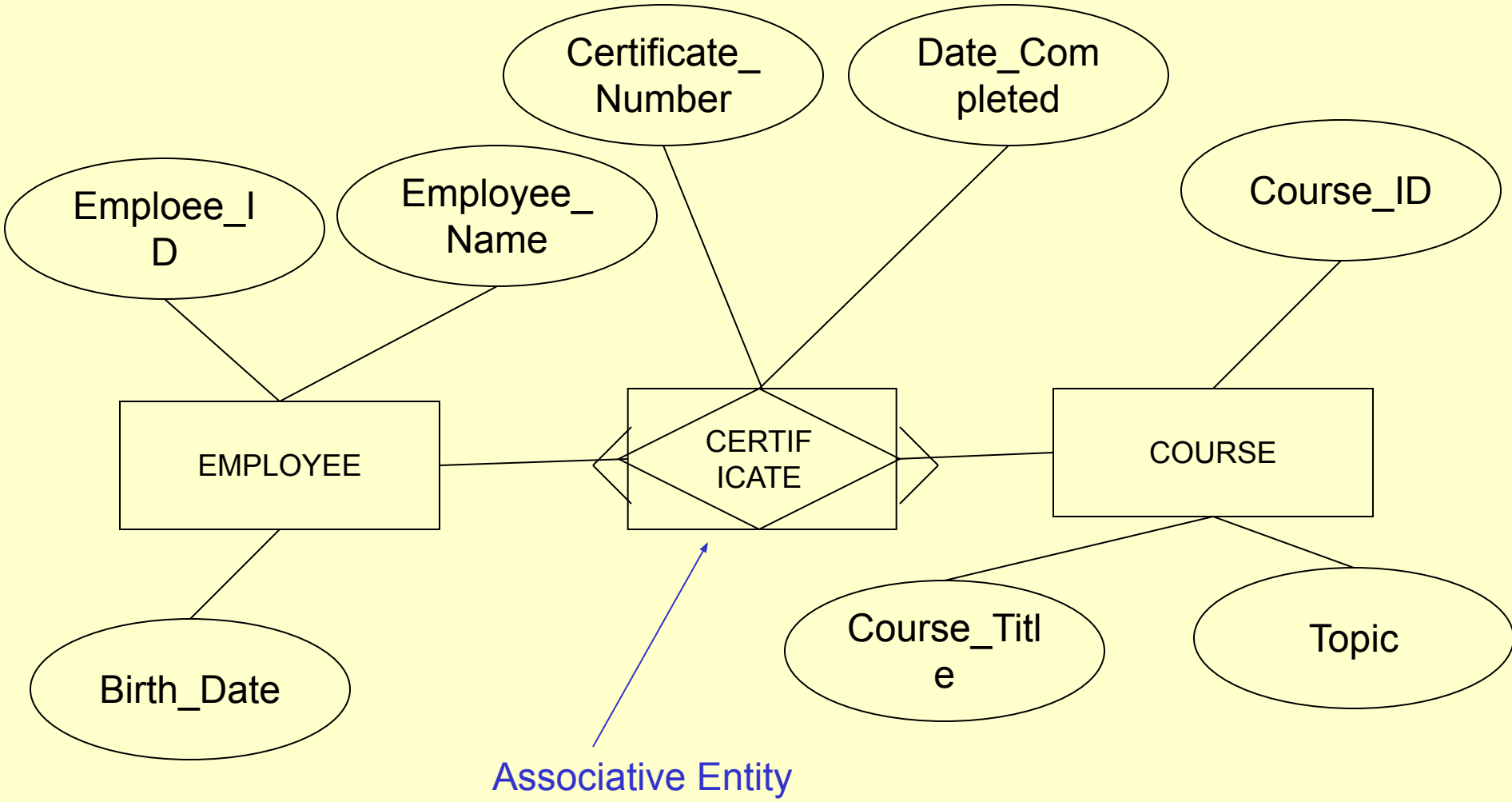
Data model segment for class scheduling



Gerund / Associative Entity

- A Gerund is a many to many relationship that is represented as an entity type.
- It is also called an entity in disguise or a Composite Entity.
- Gerund is an entity derived from a relationship.
- It is represented by and Enclosed diamond symbol within entity box
- Example…..

Gerund – Example



Converting Relationship to an Associative Entity

- Conditions are:
  - All of the relationships for the participating entity types are “many” relationships
  - The resulting associative entity has independent meaning to end users, & preferably can be identified with a single-attribute identifier
  - The associative entity has one or more attributes in addition to the identifier
  - The associative entity participates in one or more relationships independent of the entities related in associated relationship

Enhanced ER Model

- The model that has resulted from extending the original E-R model with new modeling constructs.
- Used to cope with the dynamic and complex business environment.

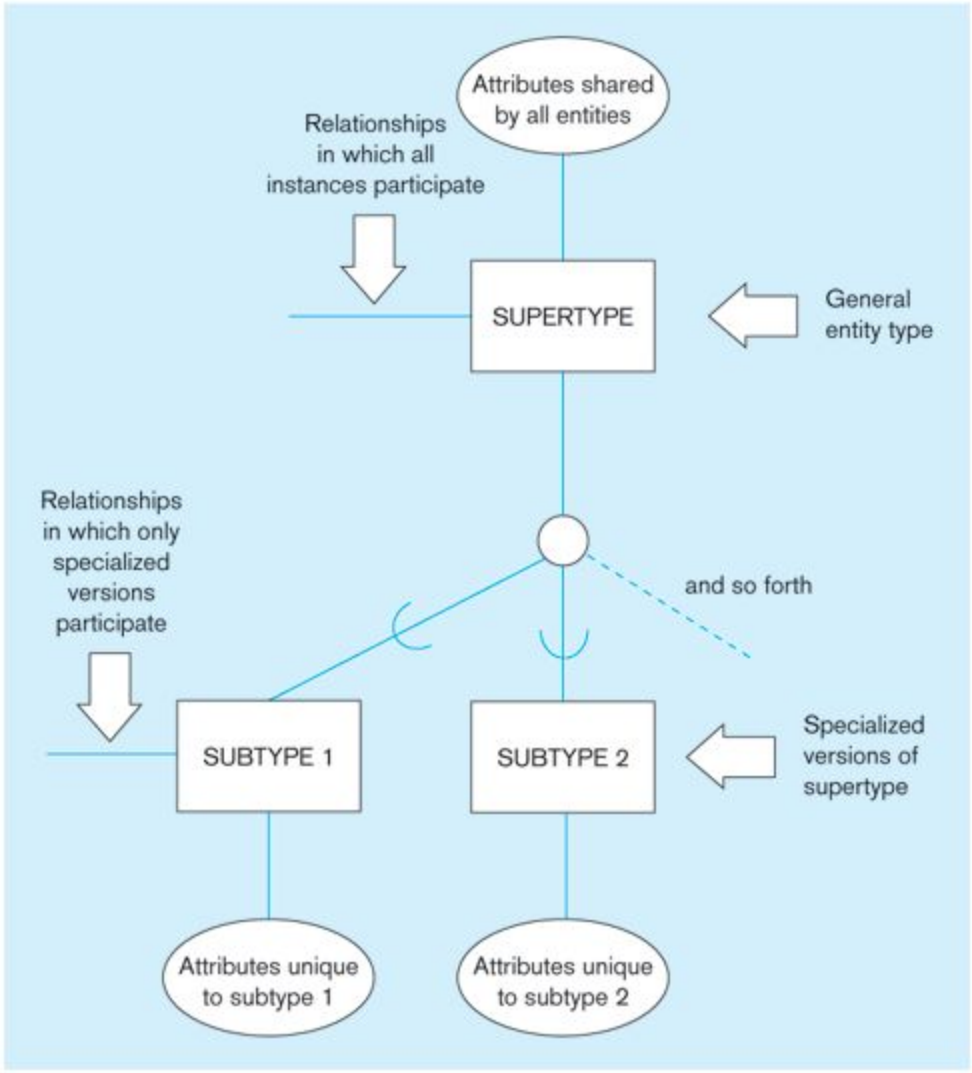
Super Types, Subtypes

- A Super type is a generic entity type that is subdivided into subtypes.
- A Subtype is a subset of a super type that shares the common attributes of the super type plus it has some of its own attributes that distinguish it from other subtypes.
- The super type and subtypes employ the concept of Generalization and Categorization.
- Generalization is the concept that some entities are subtypes of other more general entities. E.g. Bird, Vehicle etc.
- Categorization is the concept that an entity comes in various subtypes.
- Super type & Subtype representation is used to represent the entities that are very much similar.
- Attribute Inheritance:

– Subtype entities inherit values of all attributes of the supertype

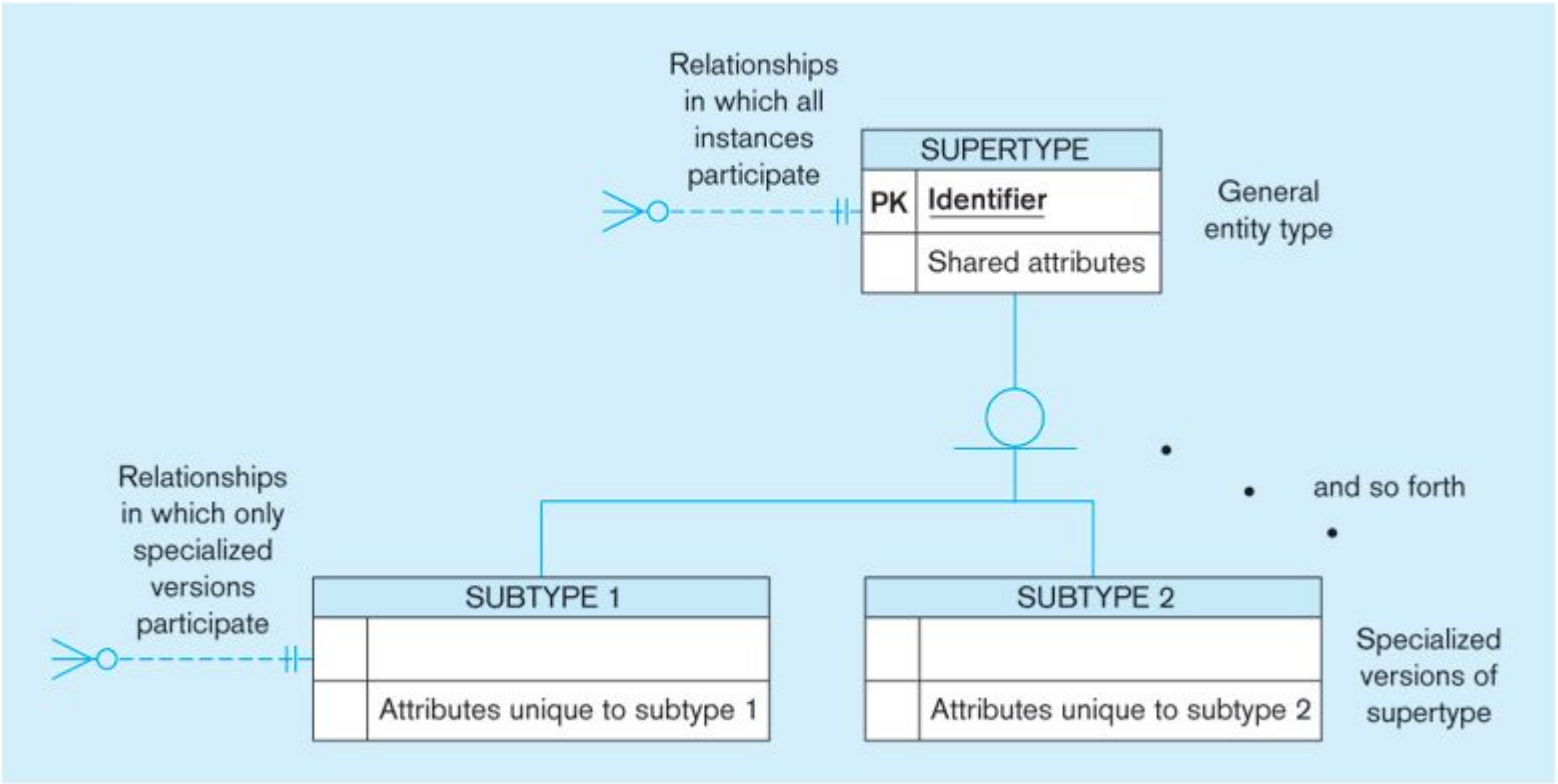
– An instance of a subtype is also an instance of the supertype

Figure 4-1a Basic notation for supertype/subtype relationships - Traditional EER notation



Basic Notations for Super Type/Subtype

Figure 4-1b Basic notation for supertype/subtype relationships - Microsoft Visio notation



Different modeling tools may have different notation for the same modeling constructs

Super Types, Subtypes - Example

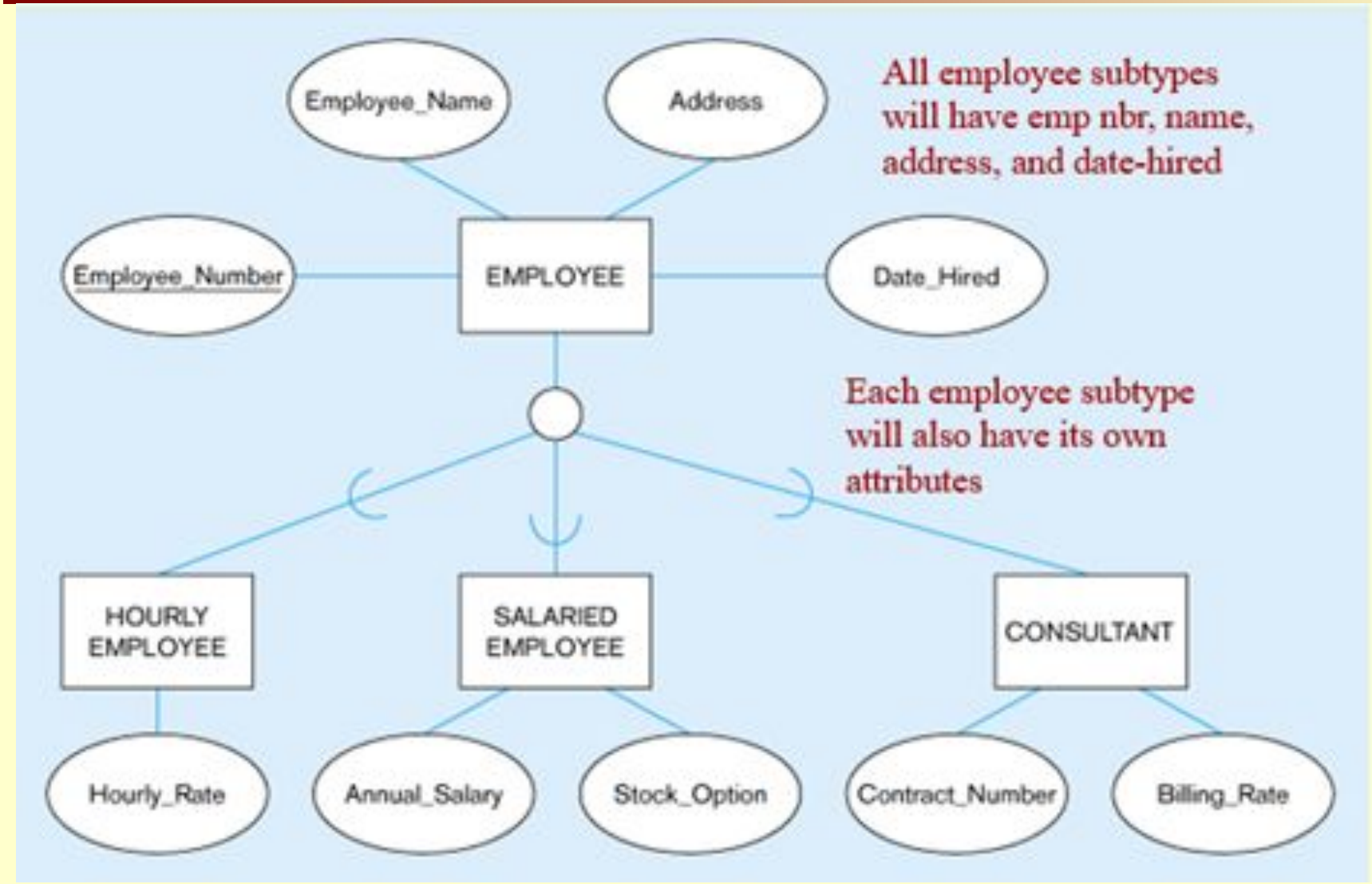
EMPLOYEE TYPE	ATTRIBUTES
1. Hourly	ENo, EName, Address, Hire_Date, Hourly_Rate
2. Salaried	ENo, EName, Address, Hire_Date, Salary
3. Contract	ENo, EName, Address, Hire_Date, Rate

- We have three choices for ER diagram:

1. Define a single entity type Employee with all the possible attributes in it.
2. Define a separate entity type for each employee type.
3. Define a supertype with all the common attributes and introduce its subtypes having only the applicable attributes.

ER Diagram

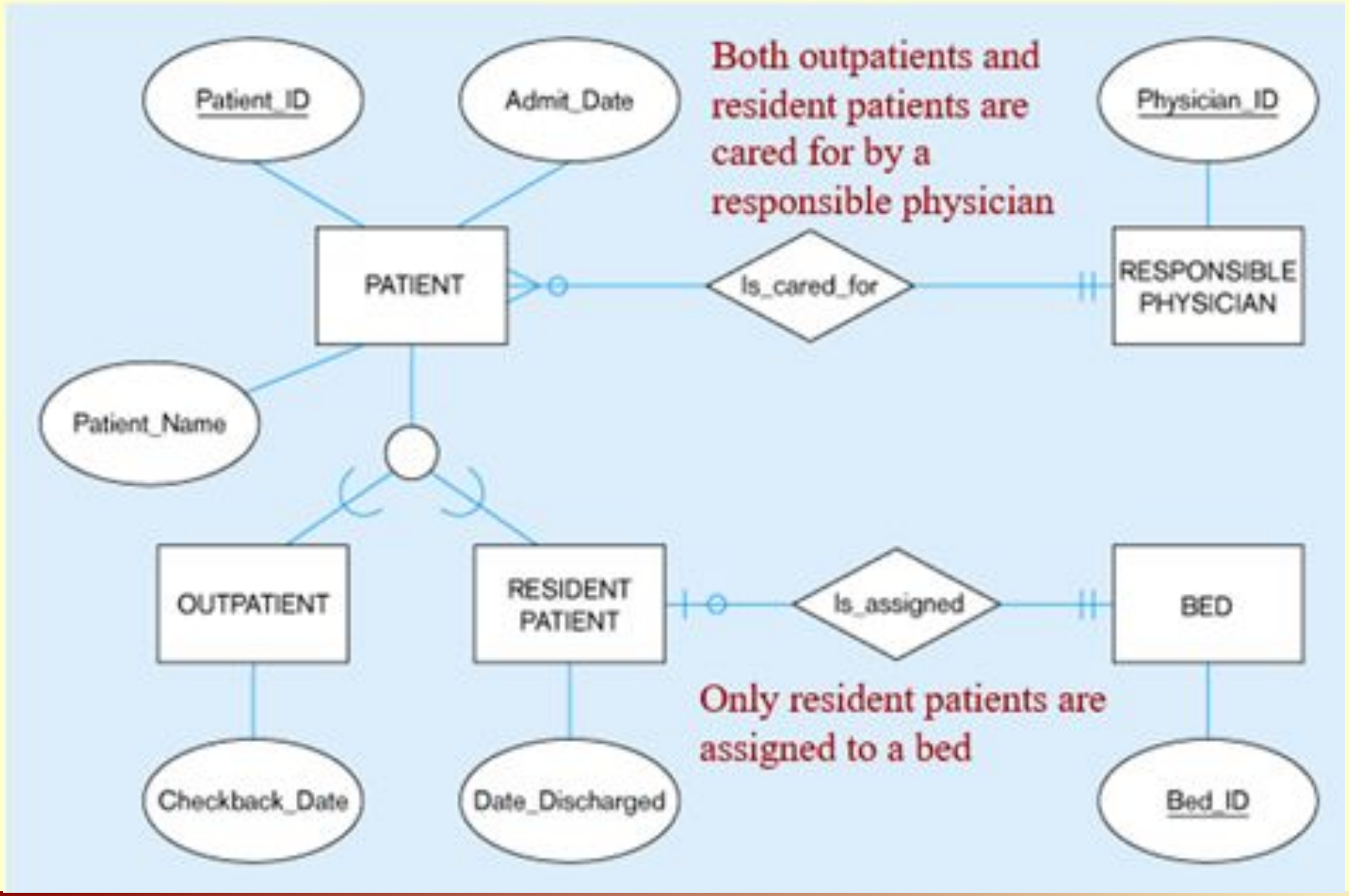
Figure 4-2 – Employee supertype with three subtypes



Super Types, Subtypes

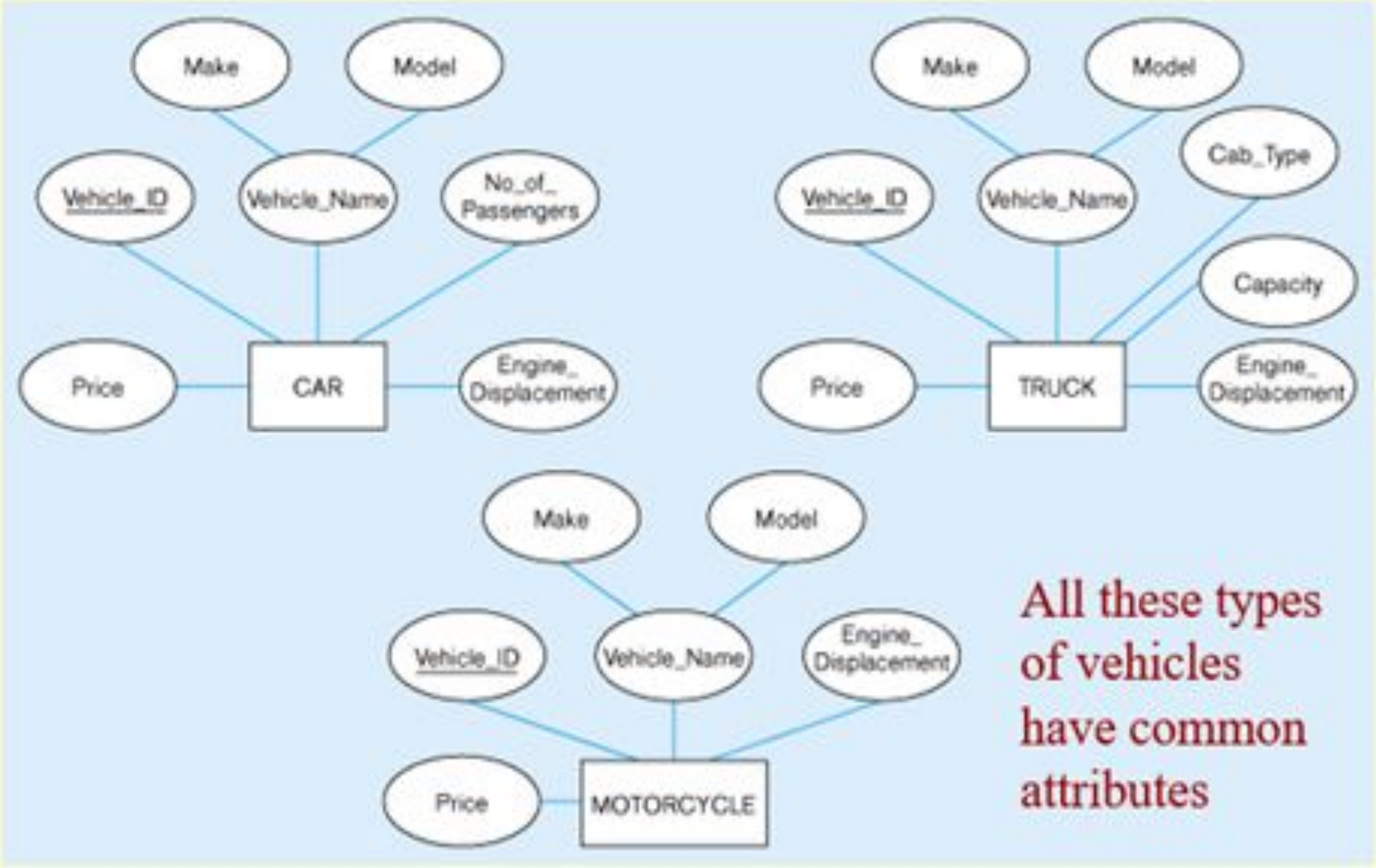
- The relationship between each subtype and its super type is called ISA relationship.
- The relationship is read from the subtype to the super type.
- The cardinality of the relationship from subtype to super type is always mandatory one and from super type to subtype is optionally one. Henceforth, cardinality may be omitted in diagram.

Supertype/subtype relationships in a hospital

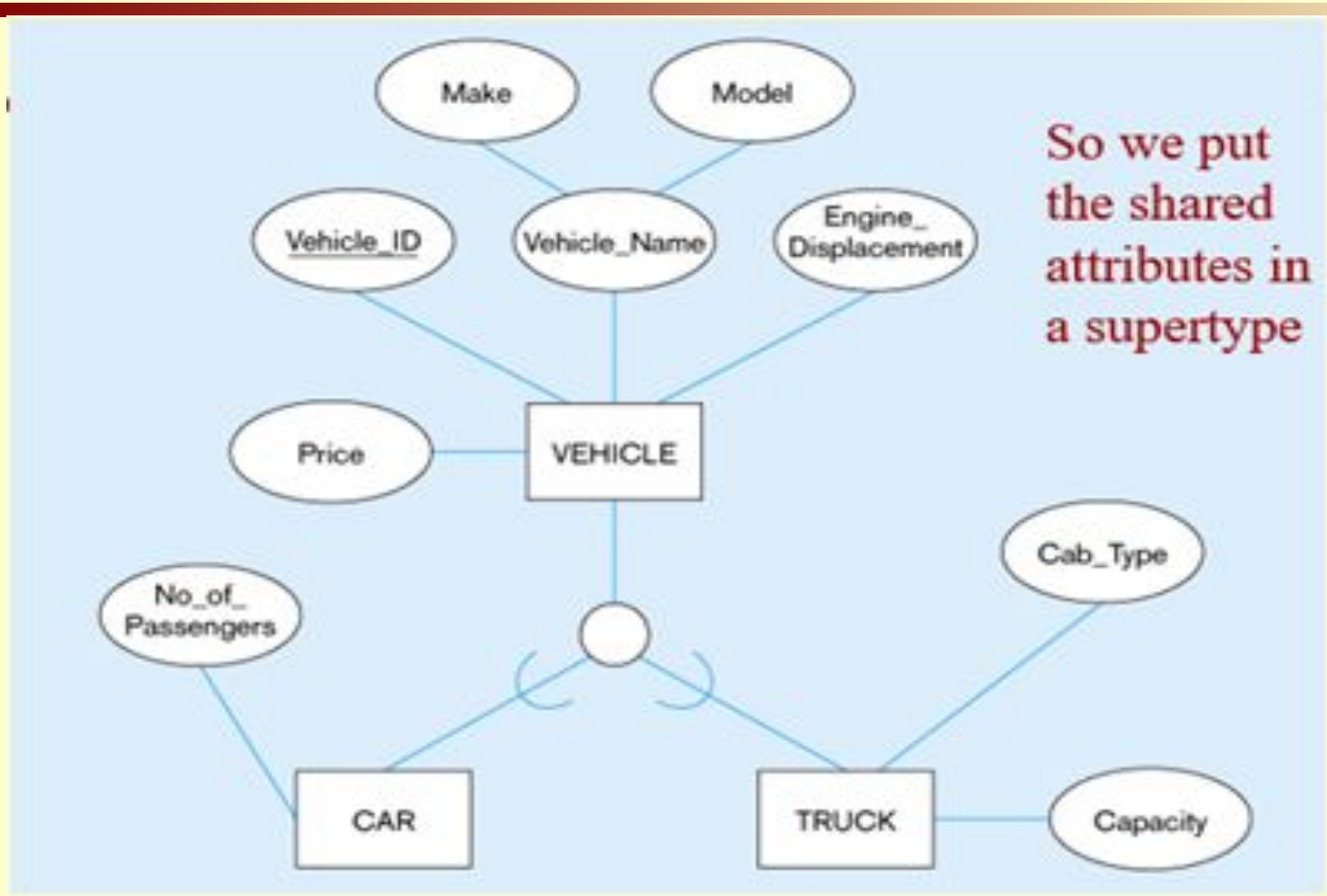


Example of generalization

Three entity types: CAR, TRUCK, and MOTORCYCLE



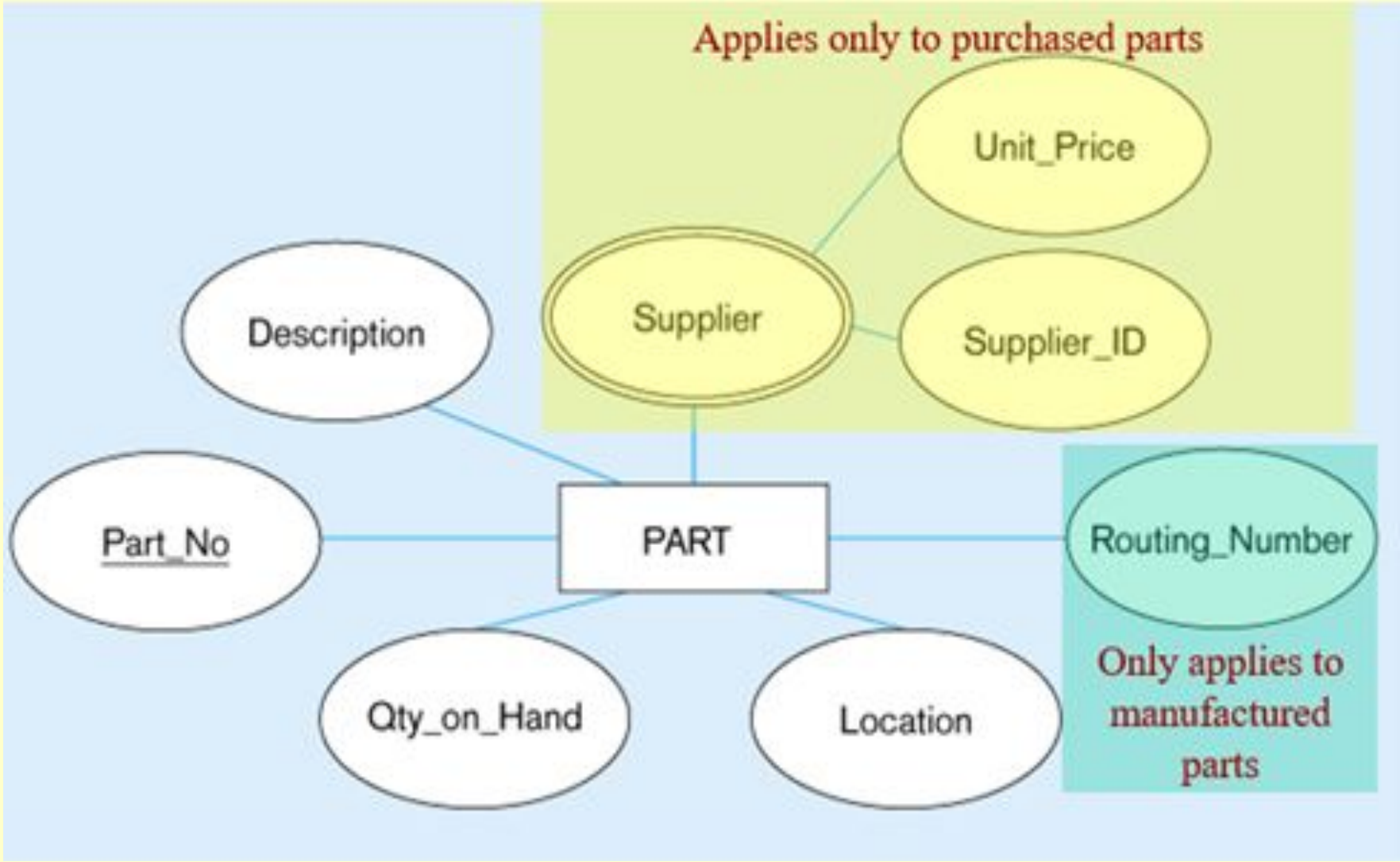
Generalization to VEHICLE supertype



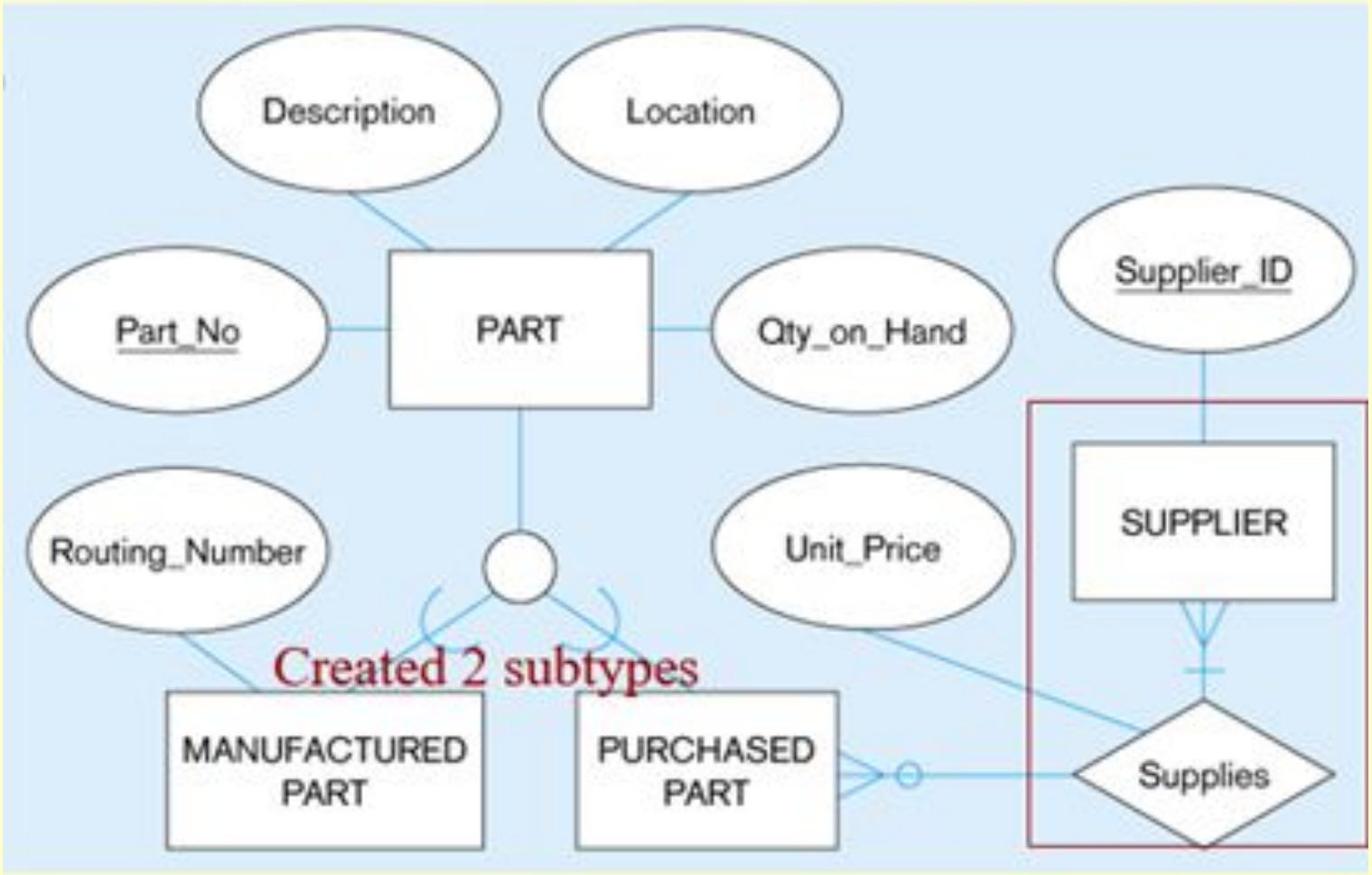
Note: no subtype for motorcycle, since it has no unique attributes

Example of specialization

Entity type PART



Specialization to MANUFACTURED PART and PURCHASED PART



Note: multivalued attribute was replaced by a relationship to another entity

Types of Subtypes

- There are the following types of subtypes:

a) Exclusive & Non-Exclusive subtypes.

b) Exhaustive & Non-Exhaustive subtypes (Completeness Constraint)

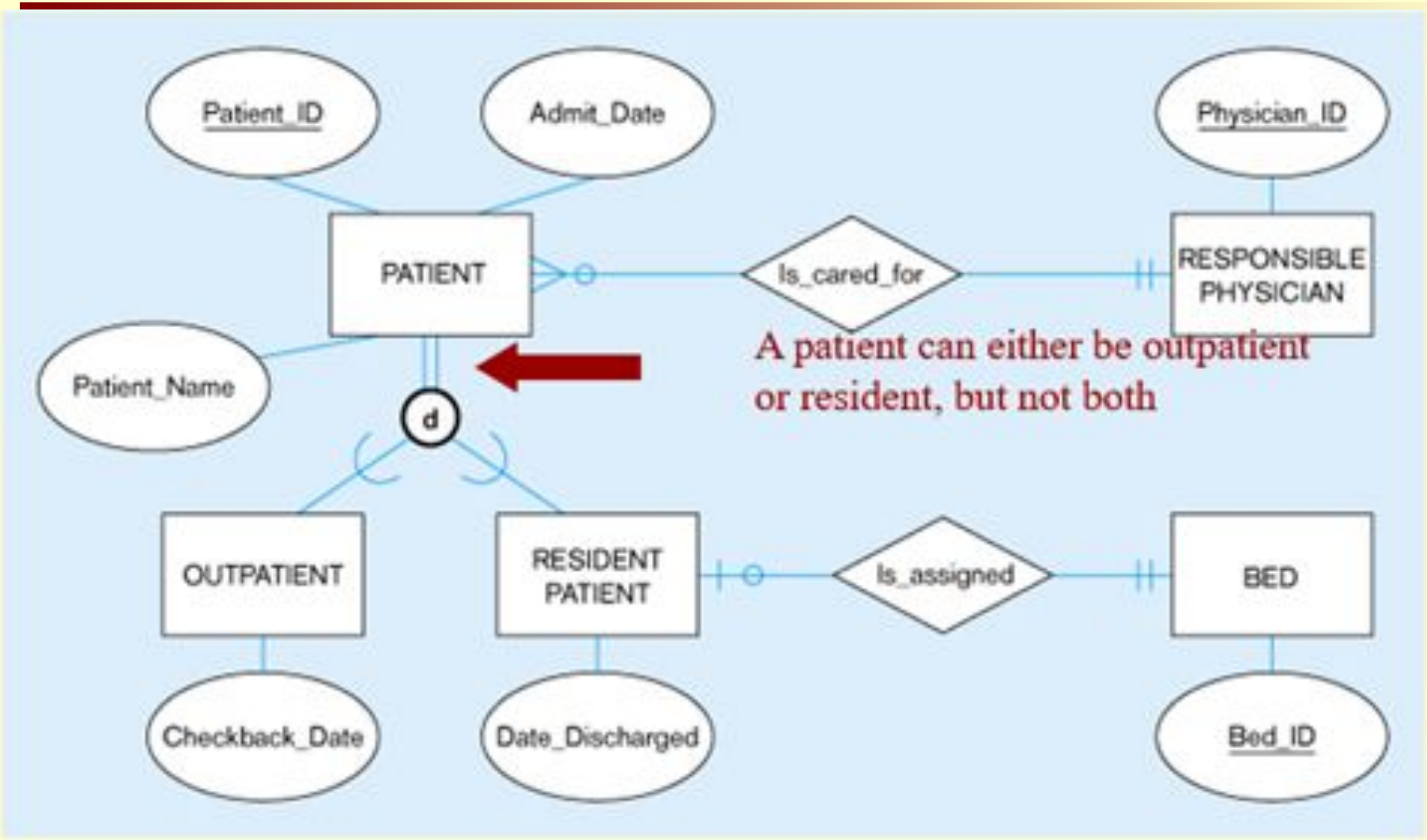
- Exclusive subtypes mean that each instance of a super type is categorized as exactly one subtype.
- Exhaustive subtypes mean that all the subtypes have been defined and there can't be any further subtypes.

Types of Subtypes

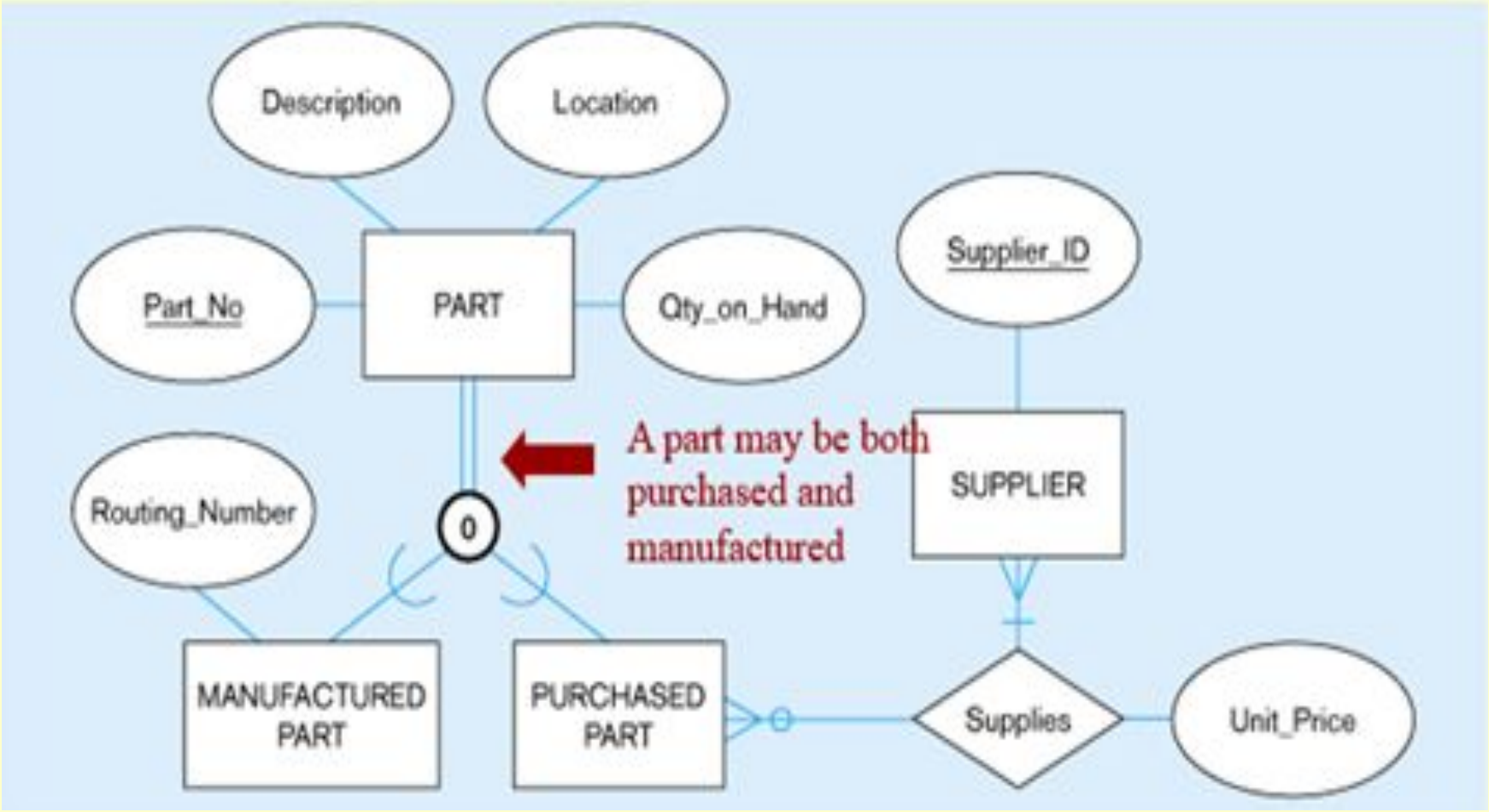
a) Exclusive & Non-Exclusive subtypes.

- Exclusive subtypes mean that each instance of a super type is categorized as exactly one subtype.
- It is used for enforcing Disjointness Constraints, which determines, Whether an instance of a super type may *simultaneously* be a member of two (or more) subtypes
  - Disjoint Rule: An instance of the super type can be only ONE of the subtypes
  - Overlap Rule: An instance of the super type could be more than one of the subtypes

Disjoint Rule



Overlap Rule

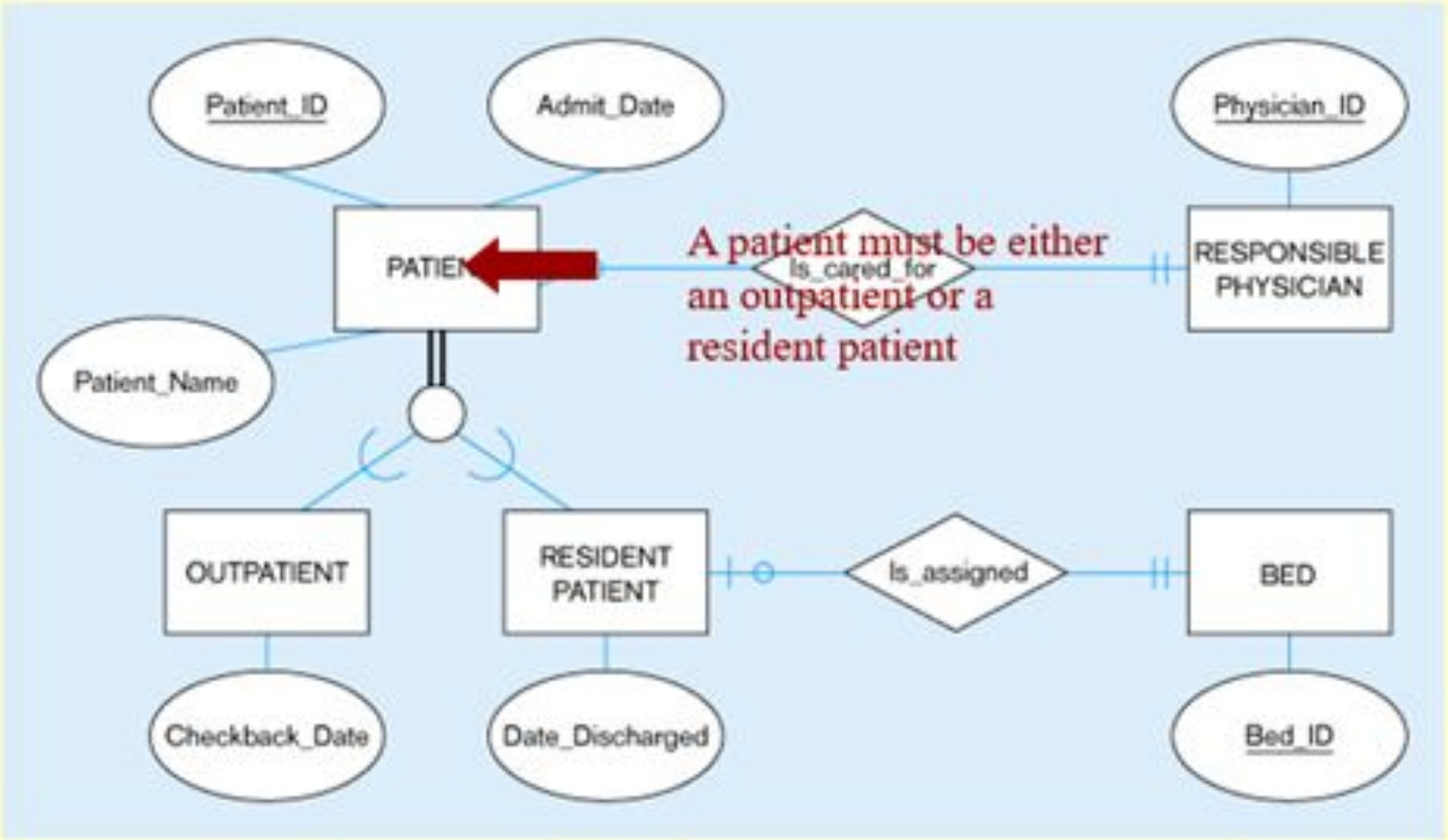


Types of Subtypes

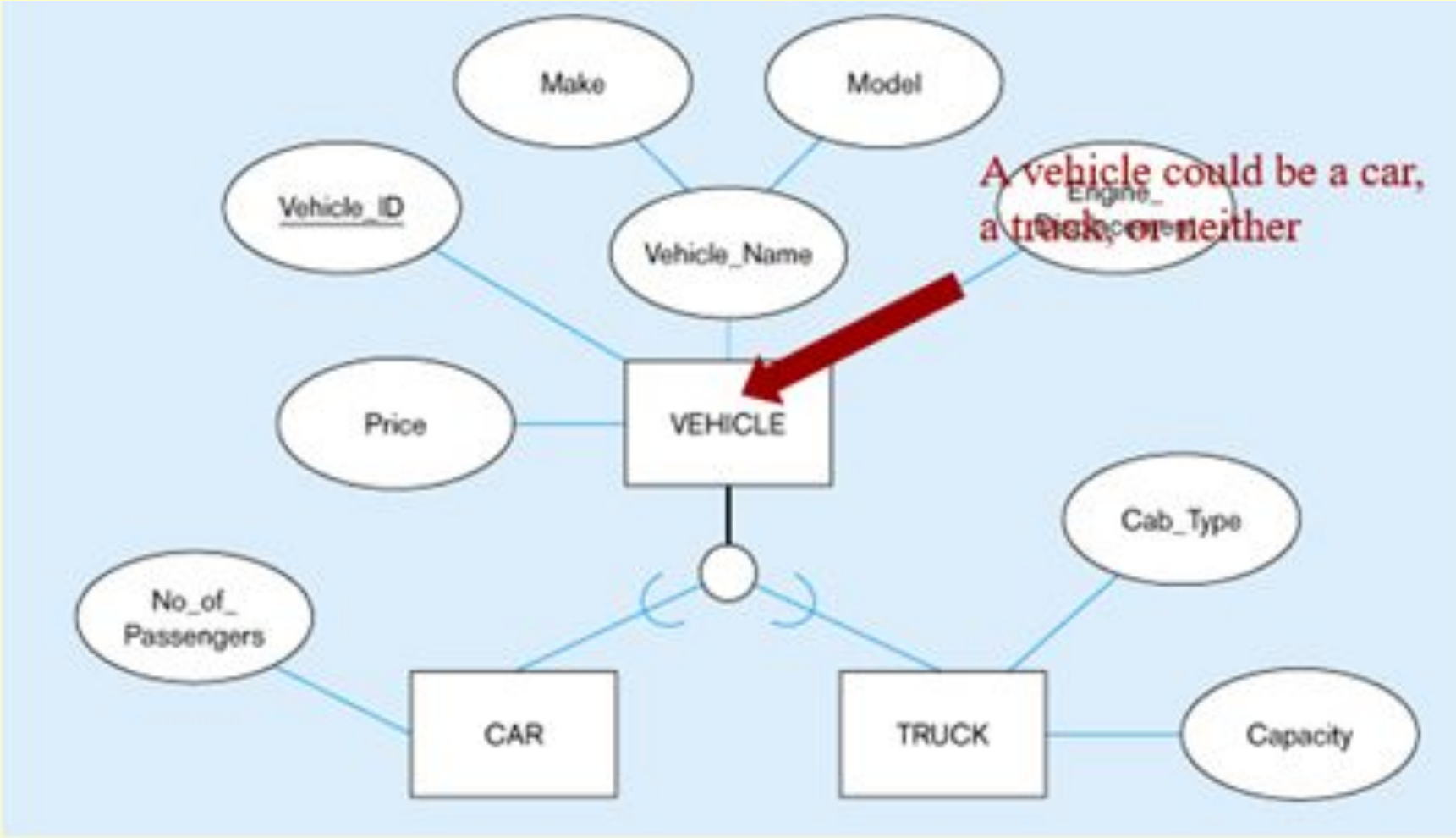
b) Exhaustive & Non-Exhaustive subtypes

- Exhaustive subtypes mean that all the subtypes have been defined and there can't be any further subtypes.
- It is used for enforcing completeness constraint, which determines, Whether an instance of a super type must also be a member of at least one subtype
  - Total Specialization Rule: Yes (double line)
  - Partial Specialization Rule: No (single line)

Total Specialization Rule



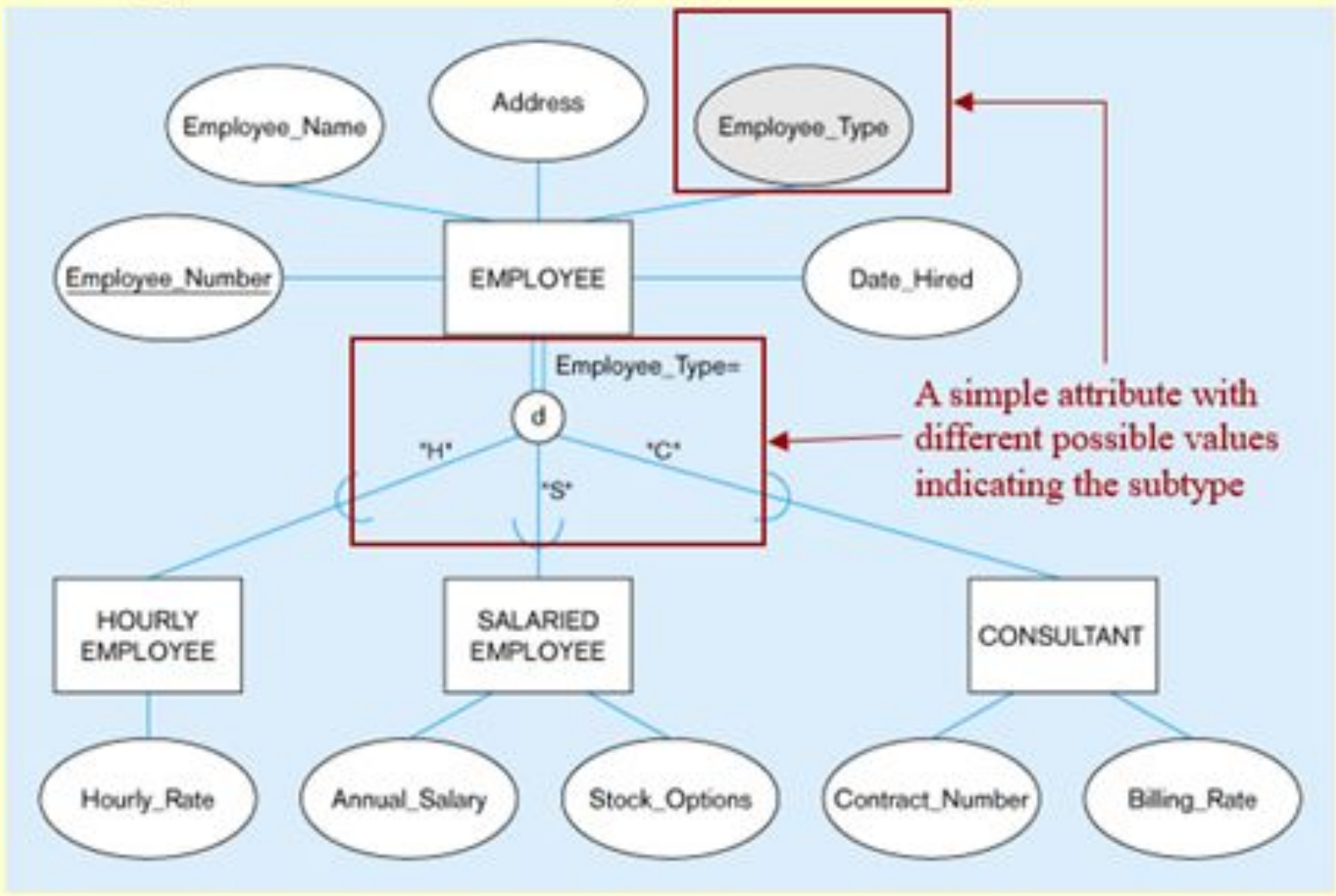
Partial Specialization Rule



Subtype Discriminators

- An attribute of the super type whose values determine the target subtype(s)
  - **Disjoint** – a *simple* attribute with alternative values to indicate the possible subtypes
  - **Overlapping** – a *composite* attribute whose subparts pertain to different subtypes. Each subpart contains a Boolean value to indicate whether or not the instance belongs to the associated subtype

Subtype Discriminator (***disjoint*** Rule)



Subtype discriminator (***Overlap*** Rule)

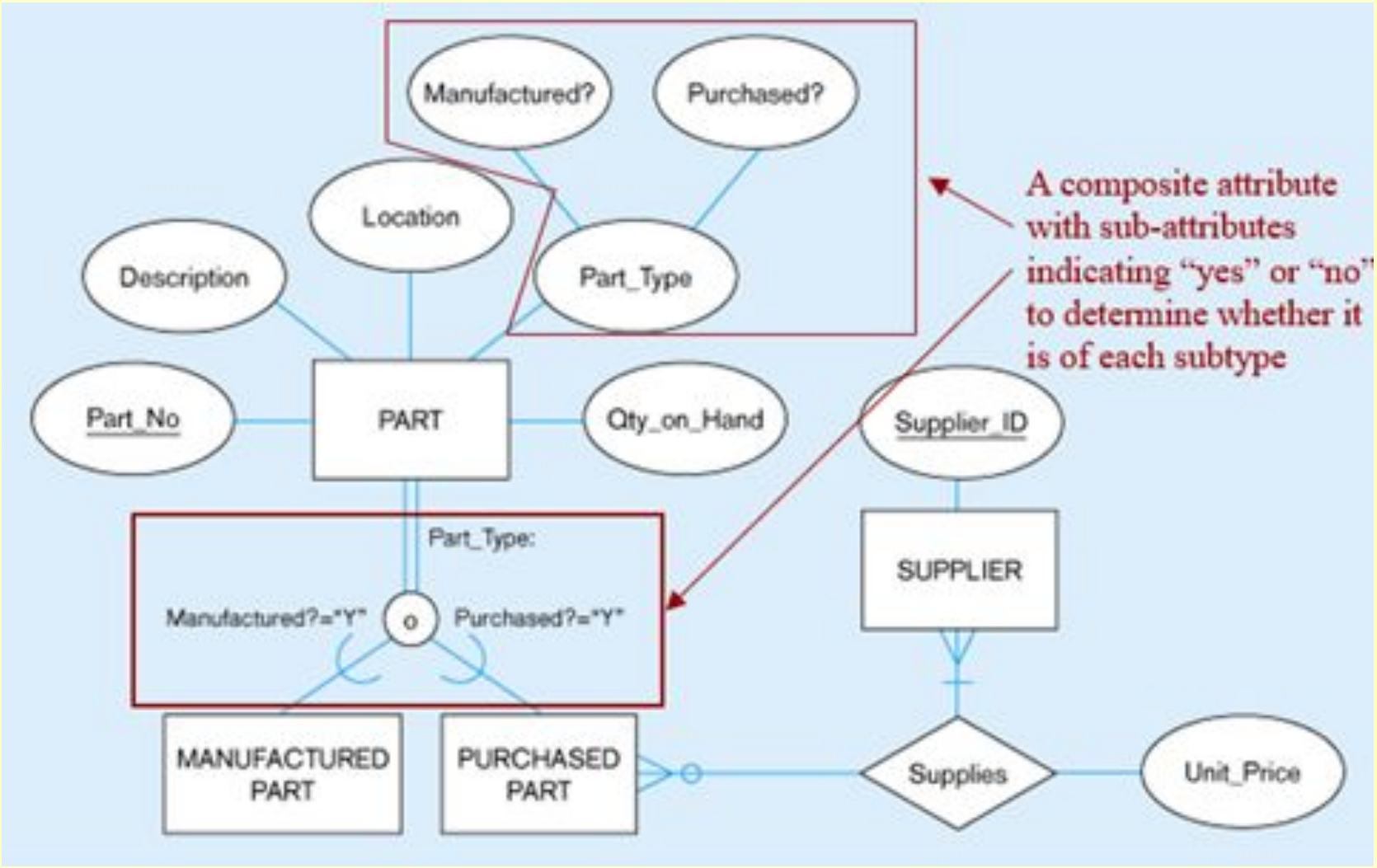


Figure 4-10 Example of supertype/subtype hierarchy