Database Principles

CCS3400

Database Design

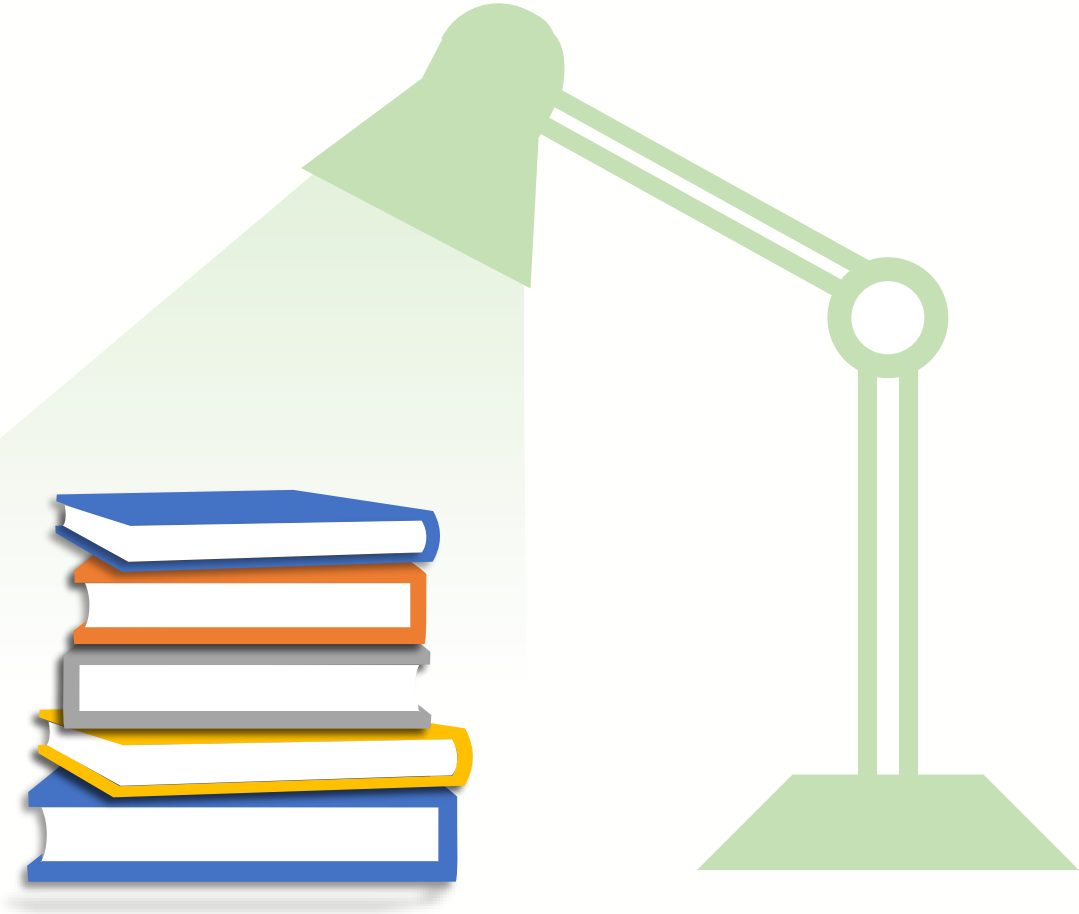
Learning Objectives

01 Describe the database design stages

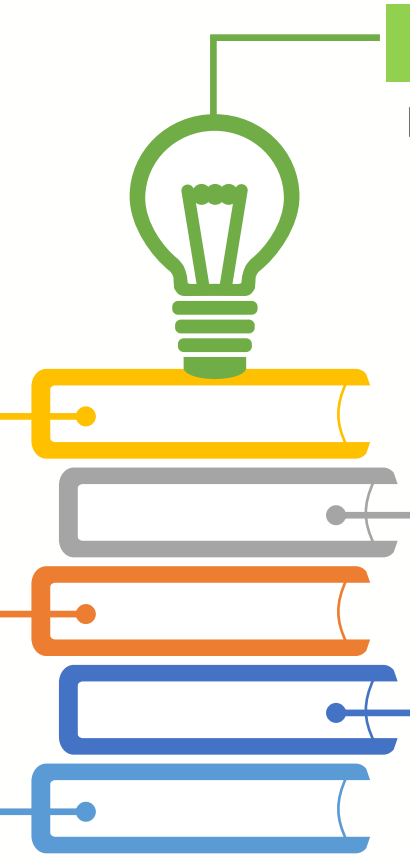
02 Explain the construction of ER Diagram

03 Explain the construction of the Enhanced ER Diagram

04 Describe the concept of generalization and specialization

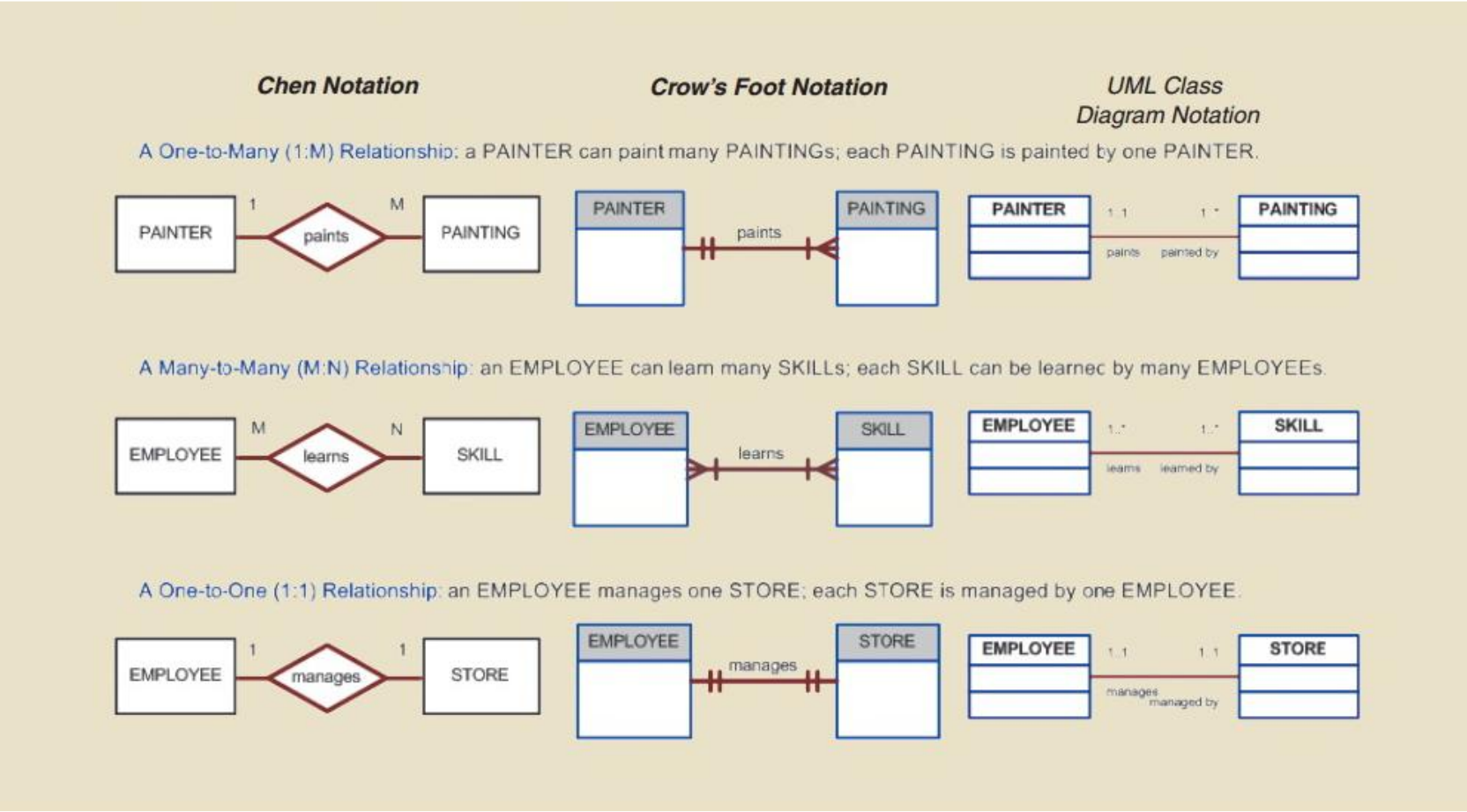


Database Design Stage

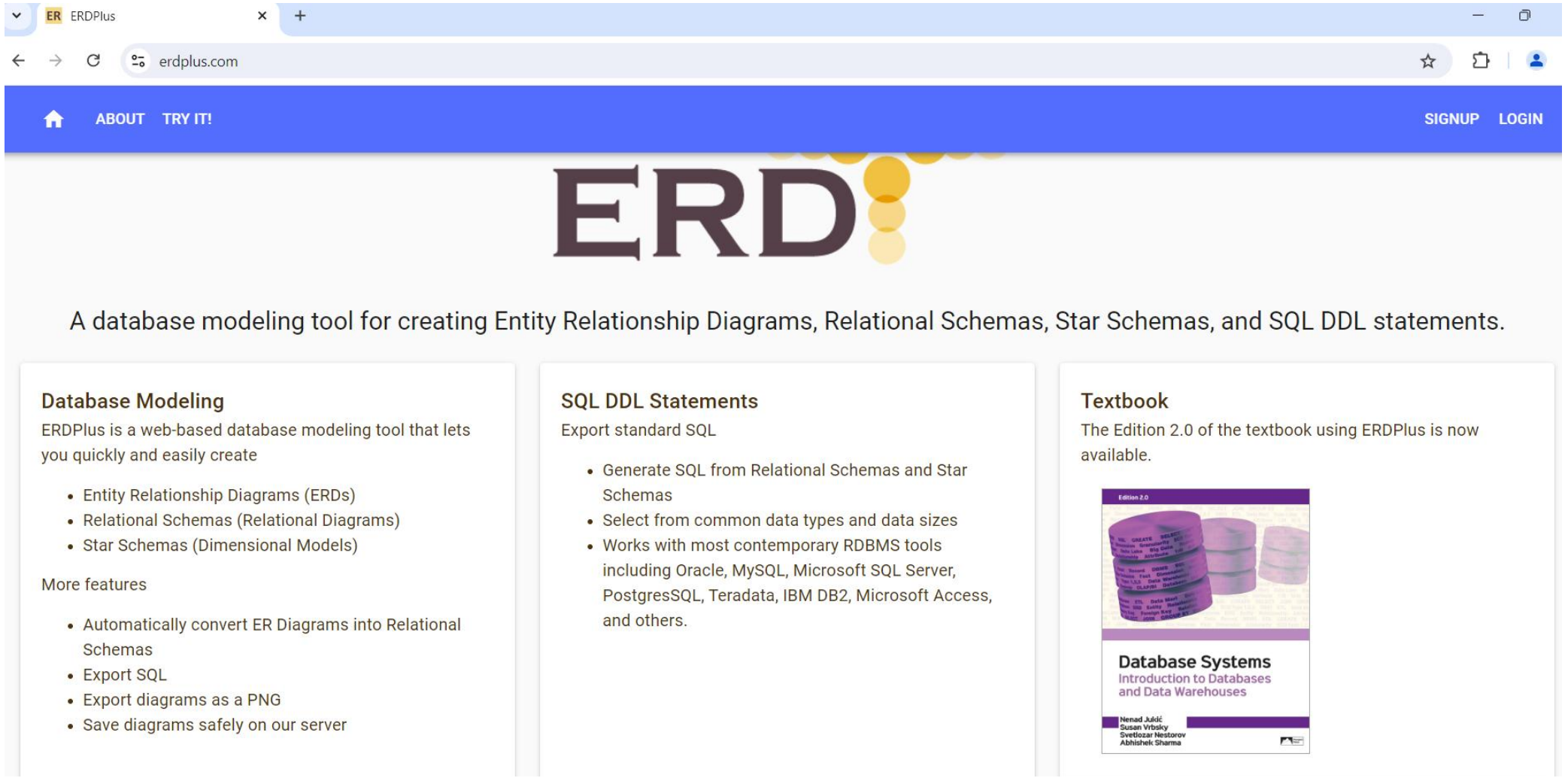


1. Identify the exact goals of the system
2. Talk to the users to identify the basic forms and reports
3. Identify the data items to be stored.
4. Design the tables and relationships
5. Identify any business constraints.
6. Verify the design matches the business rules

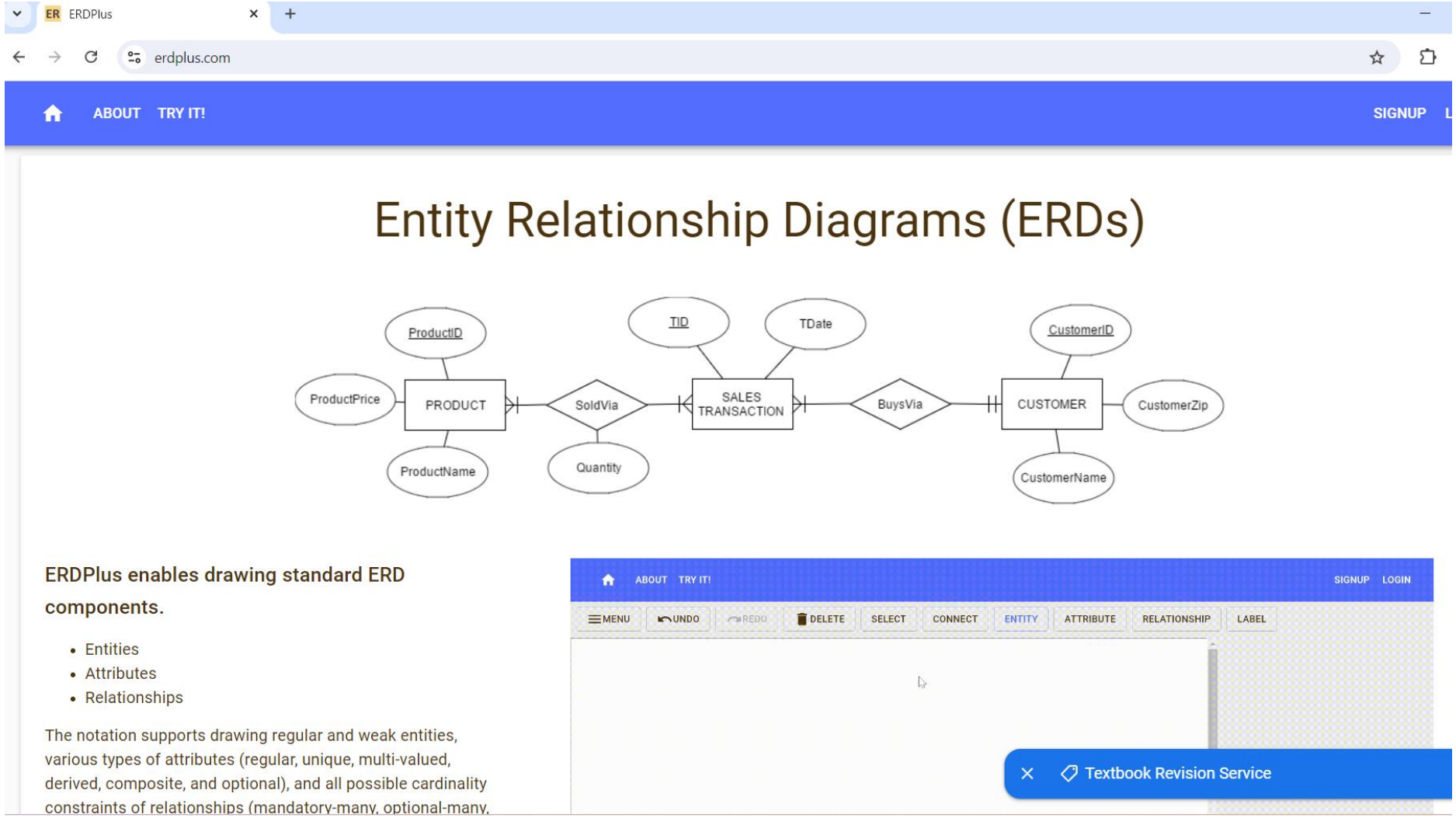
The **Entity Relationship (ER) Model** has become a widely accepted standard for data modeling and are normally represented in an **Entity Relationship Diagram (ERD**), which uses graphical representations to model database components. The three common ER notations are shown below: the original **Chen notation**, the **Crow's Foot notation**, and the newer class diagram notation, which is part of the **Unified Modeling Language (UML).**



In this course, we use ERDPlus which **combines** the Chen and Crow Foot notations

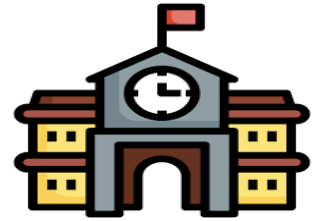


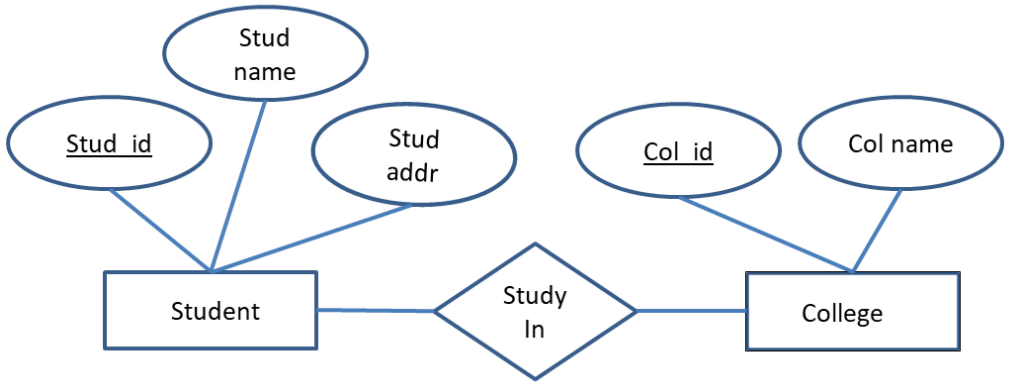
ERDPlus combines the Chen's entity, relationship and attributes notations with the Crow's Foot cardinality notations.



ER Model

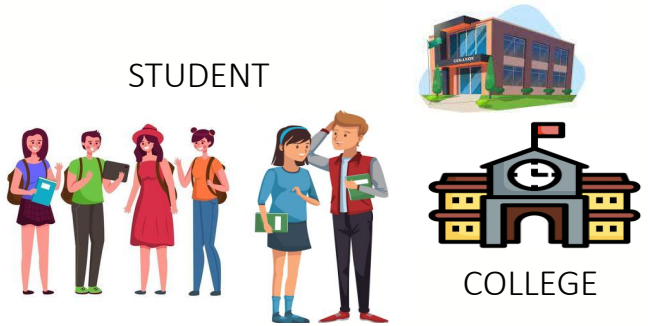
- ER model – a logical representation of the data for an organization or a business area
- ER diagram – a graphical representation of an entity-relationship model

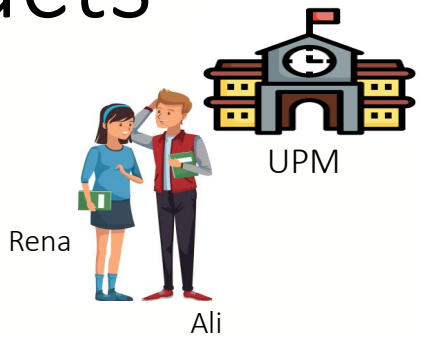


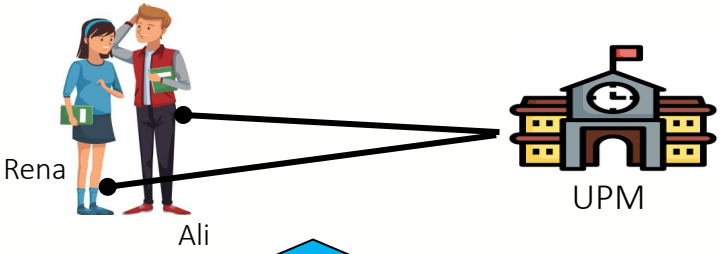


ER Diagram

ER Model Constructs







ENTITY

ENTITY

person, place, object, event, or concept in the user environment about which data is to be maintained.

Example: Student, College

ATTRIBUTE

ATTRIBUTE

property or characteristic of an entity type.

Example: Name of Student Entity, Name of College Entity

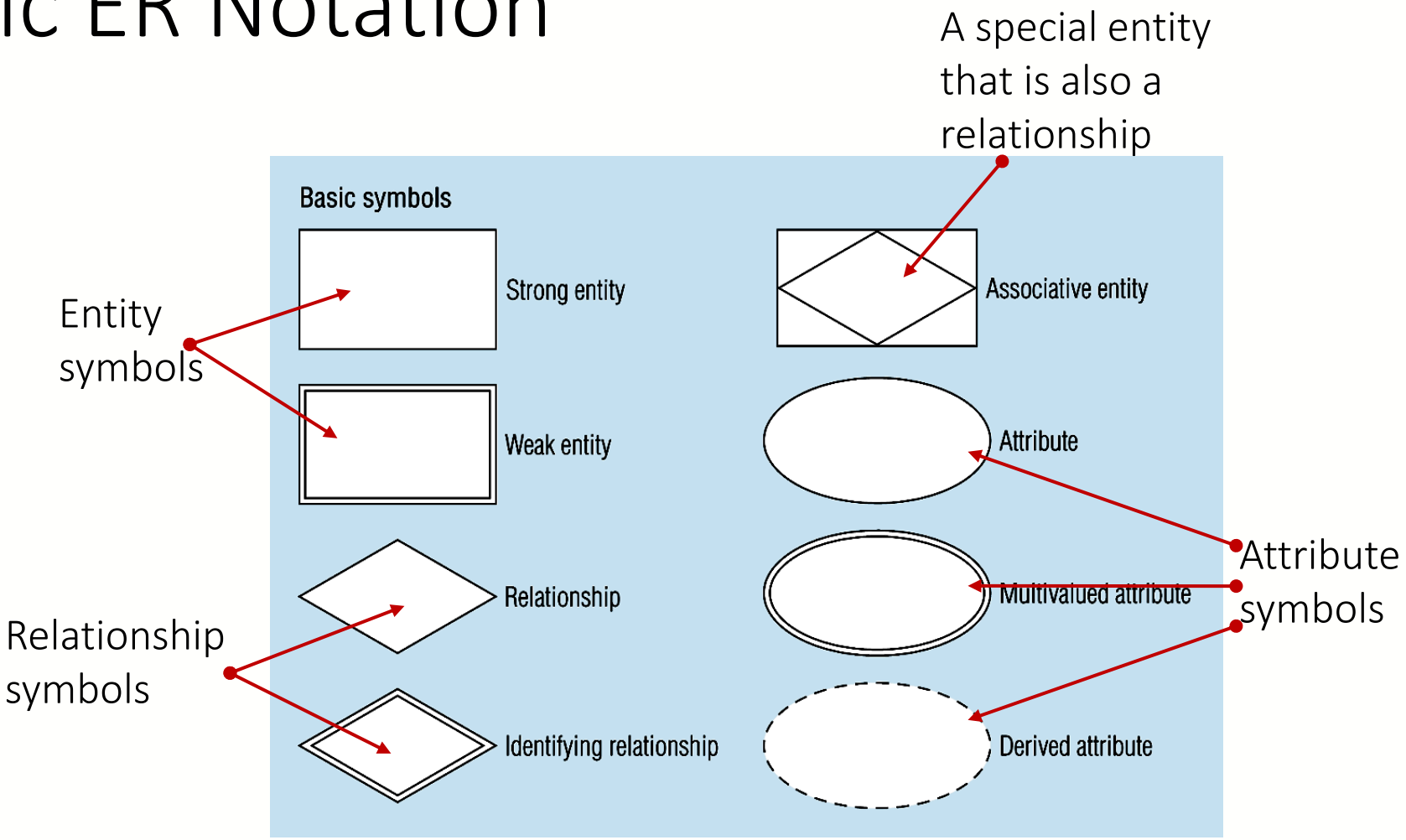
RELATIONSHIP

RELATIONSHIP

association between the instances of one or more entity types, i.e., link between entities

Example: Rena study in UPM, Ali study in UPM

Basic ER Notation



Entity

- Entity type represents a set or collection of objects in the real world that share the same properties.

Entity Identity	Example of Entity Type
Person	STAFF, STUDENT, LECTURER
Place	DISTRICT, TOWN, STATE
Object	BUILDING, TOOL, PRODUCT
Event	REGISTRATION, APPLICATION
Concept	COURSE, ACCOUNT

Example of Entity types according to its identity

Entity

- Entity Type vs. Entity Instance
  - Entity Type - a collection of a similar type of entities (often corresponds to **a table**)
  - Entity instance - a single occurrence of an entity type (often corresponds to **a row in a table**)
- Strong vs. Weak Entity Types
  - Strong Entity Type – an entity that exists **independently** of other entity types
  - Weak Entity Type – an entity type whose existence **depends on** some other entity type

Example of Entity Type vs. Entity Instances

Avinesh Damansara | Sandra Petaling Jaya | Alisya Ampang | Rena Cheras | Amirul Serdang

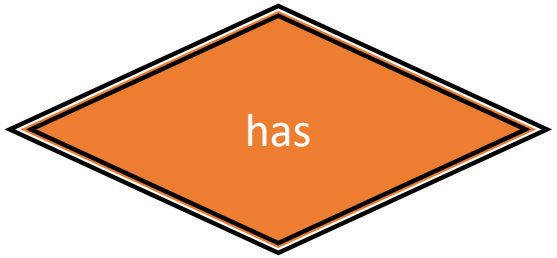
Entity Type: STUDENT

NAME	ADDRESS
Avinesh	Damansara
Sandra	Petaling Jaya
Alisya	Ampang
Rena	Cheras
Amirul	Serdang

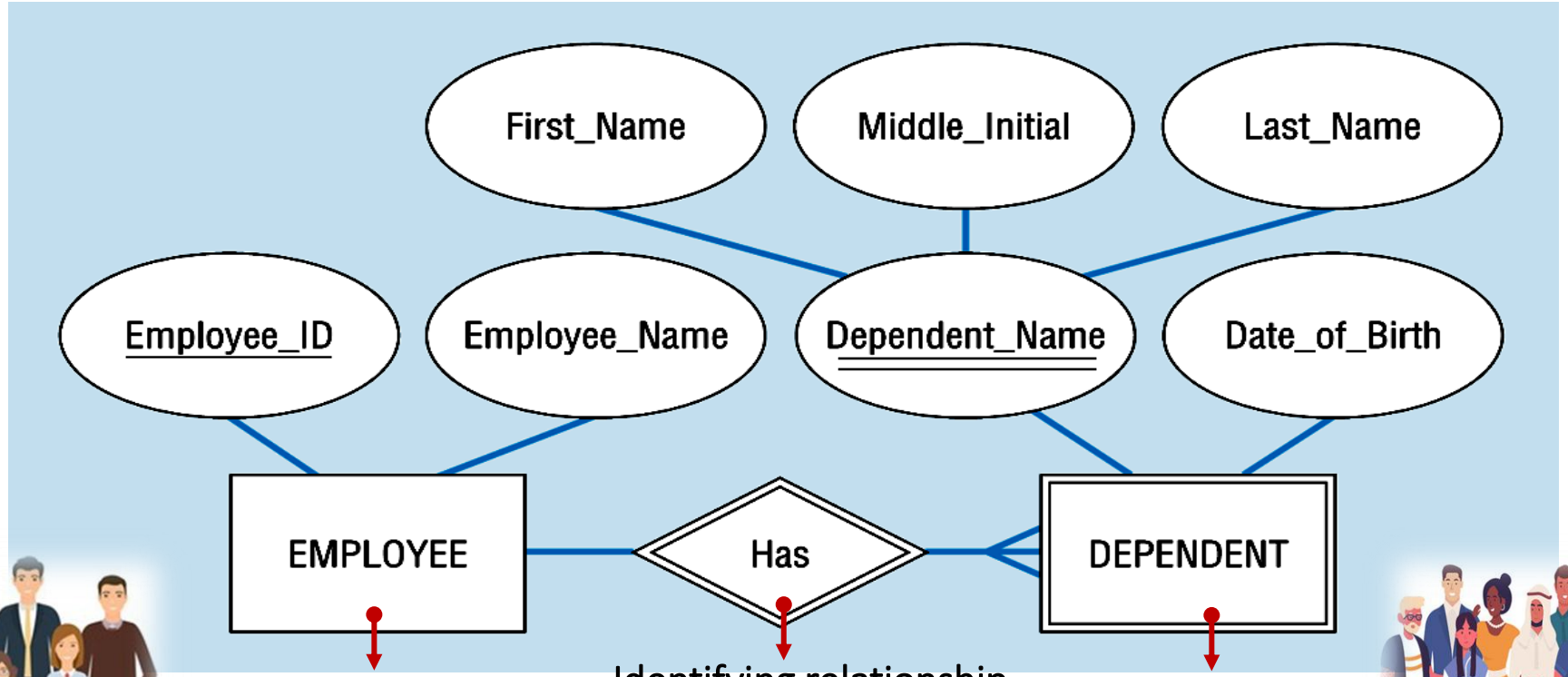
Each row represented an entity instance of the entity type. Hence, each student is an entity instance for the STUDENT entity.

Strong vs. Weak Entities, and Identifying Relationships

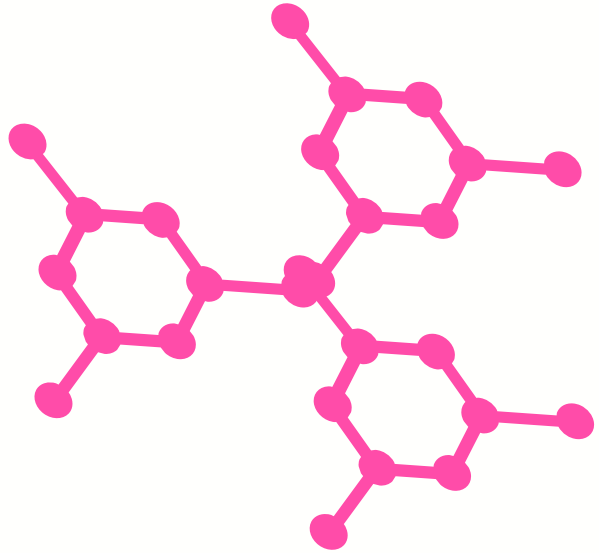
- Strong entities
  - exist independently of other types of entities
  - has its own unique identifier
  - represented with a single-line rectangle
- Weak entity
  - dependent on a strong entity…cannot exist on its own
  - Does not have a unique identifier
  - represented with a double-line rectangle
- Identifying relationship
  - links strong entities to weak entities
  - represented with double line diamond



Strong and weak entities, Identifying relationship



Naming and Defining Entity

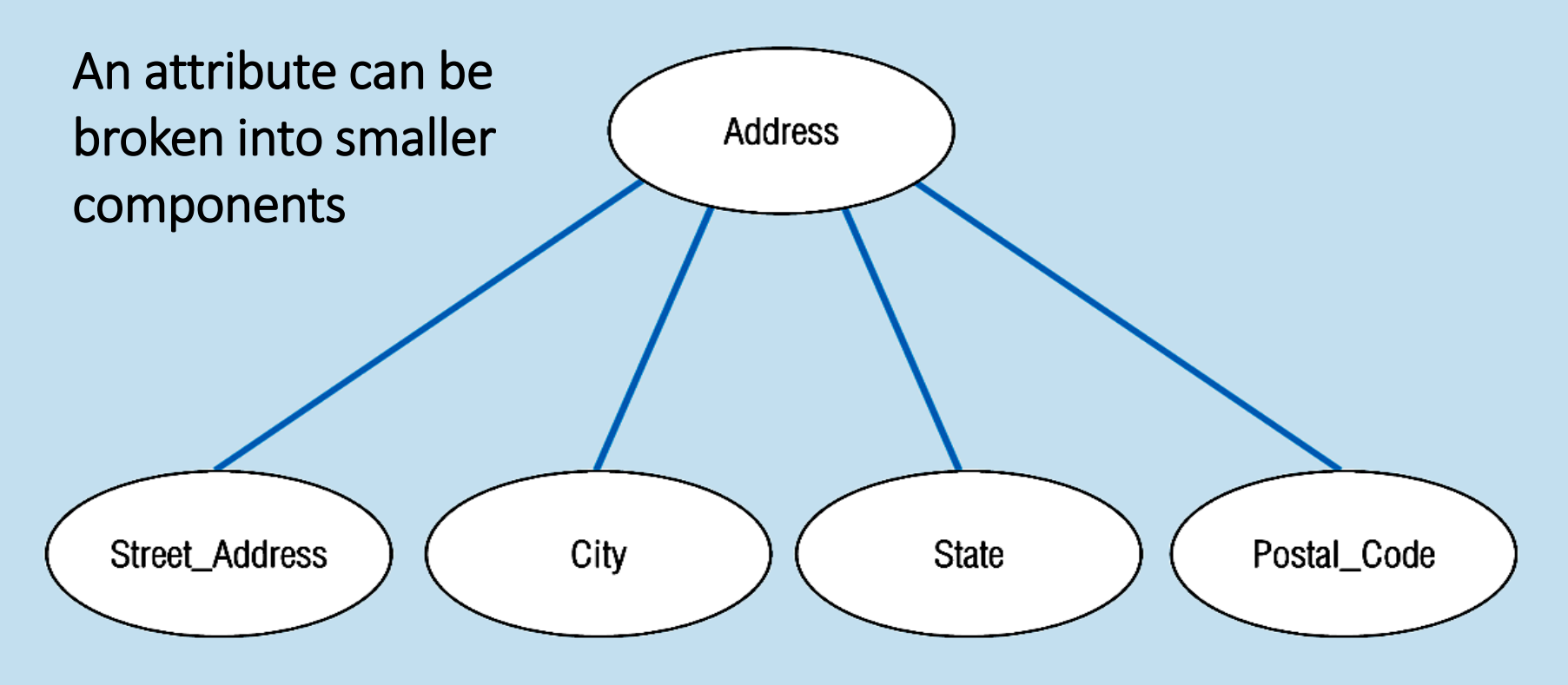


- Use singular nouns.
- It should be descriptive and specific.
  - For example, PURCHASE_ORDER and CUSTOMER_ORDER cannot be named ORDER1 and ORDER2.
- It should be concise.
  - For example, REGISTRATION_FOR_STUDENT_IN_CLASS
- The event's entity type should be named for the event's result, not the event's activity or process.
  - For example, a project manager assigning an employee to work on a project should be named ASSIGNMENT.

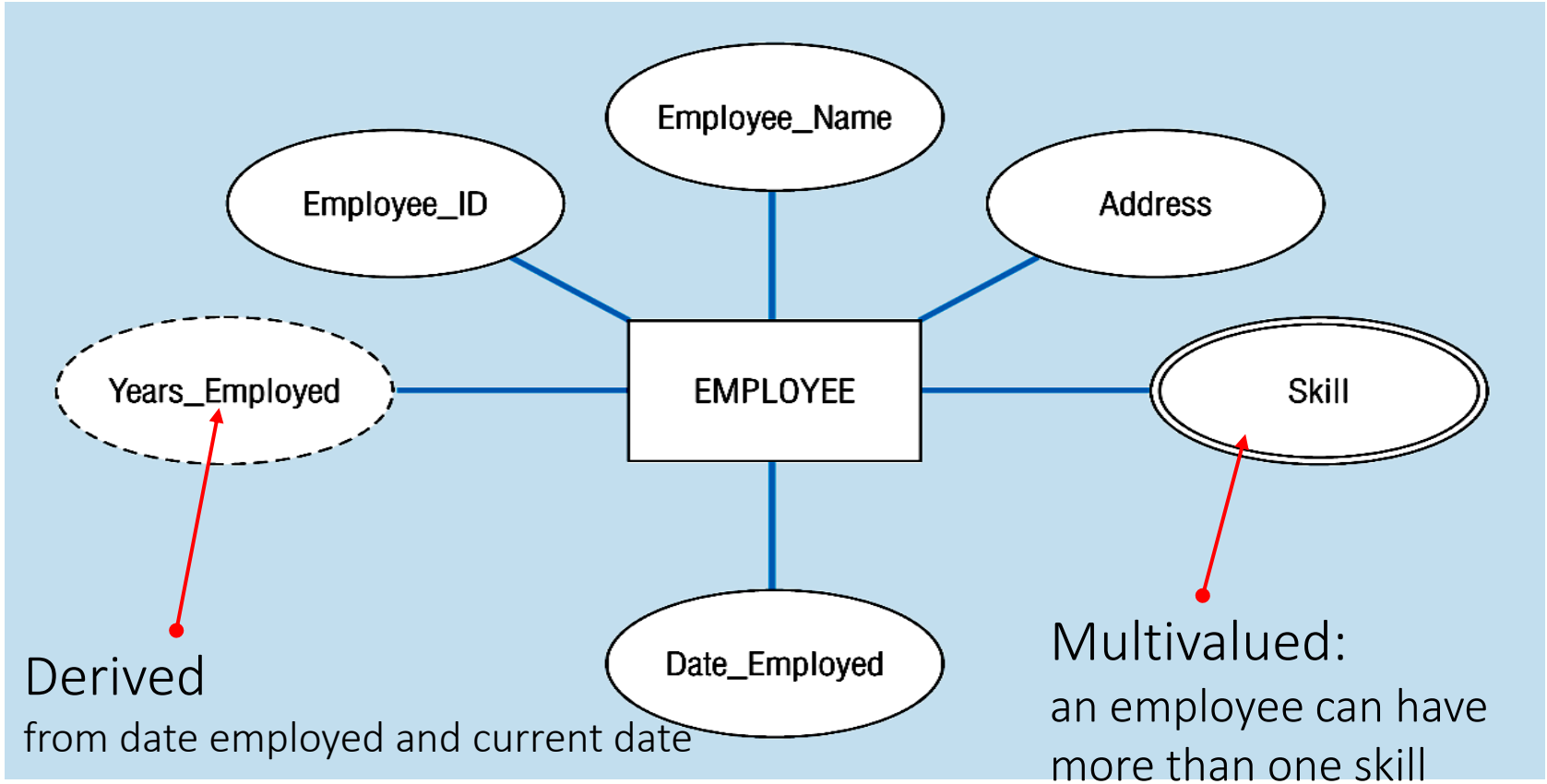
Attribute

- **Simple vs. Composite Attribute**
  - Simple attribute – cannot be broken into smaller components
  - Composite attribute – can break into smaller components
- **Single-Valued vs. Multivalued Attribute**
  - Single-Valued – each of the attributes has one value
  - Multivalued – attribute more than one value
- **Stored vs. Derived Attributes**
  - Stored attribute – data input or set
  - Derived Attribute – attribute whose values can be calculated from related attribute values.
- **Identifier Attributes**

An Example of Composite Attribute



An entity with a multivalued attribute (Skill) and derived attribute (Years_Employed)



Identifier (Key)

- Identifier (Primary Key) - An attribute (or combination of attributes) that uniquely identifies individual instances of an entity type
- Composite Identifier – an identifier that consists of a composite attribute.
- Candidate Key – an attribute that could be a key…satisfies the requirements for being a key

Simple Identifier: Employee Id

Employee Id	Name	Contact no.	Salary
101	Hitesh	5555555555	250000
102	Gaurav	4444444444	40000
103	Anjali	8888888888	25000
104	Gaurav	6666666666	60000
105	Poonam	2222222222	25000

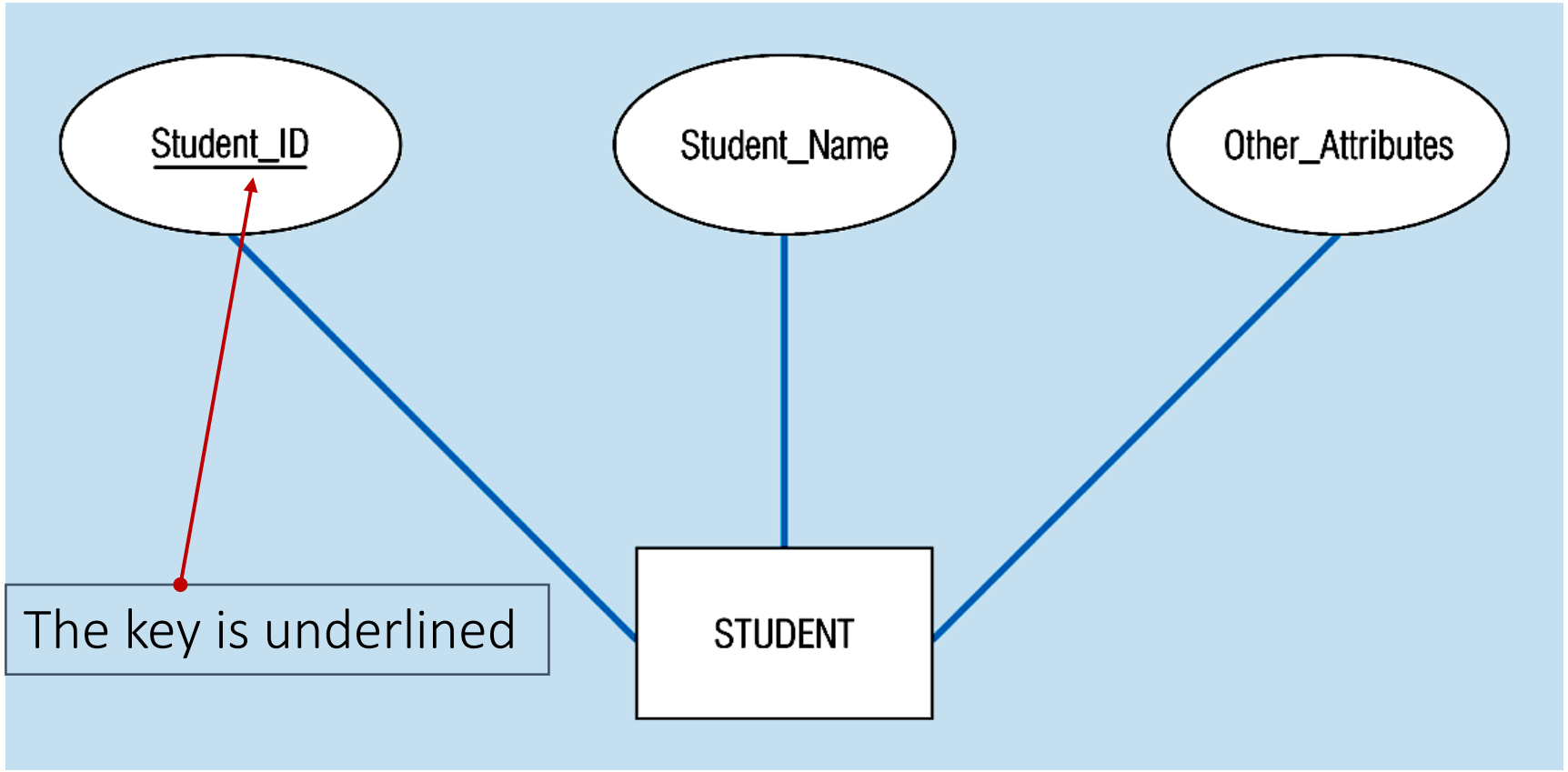
Composite Identifier: CourseCode , Date

Course code	Date	Cname	Seat	Remain	Room	Rcapa
C125	12-01-2014	C Programming	12	5	101	15
DS144	12-01-2014	Data Structure	45	22	102	50
C125	21-07-2014	C Programming	15	11	101	15
J678	08-11-2014	Java Programming	15	2	102	50

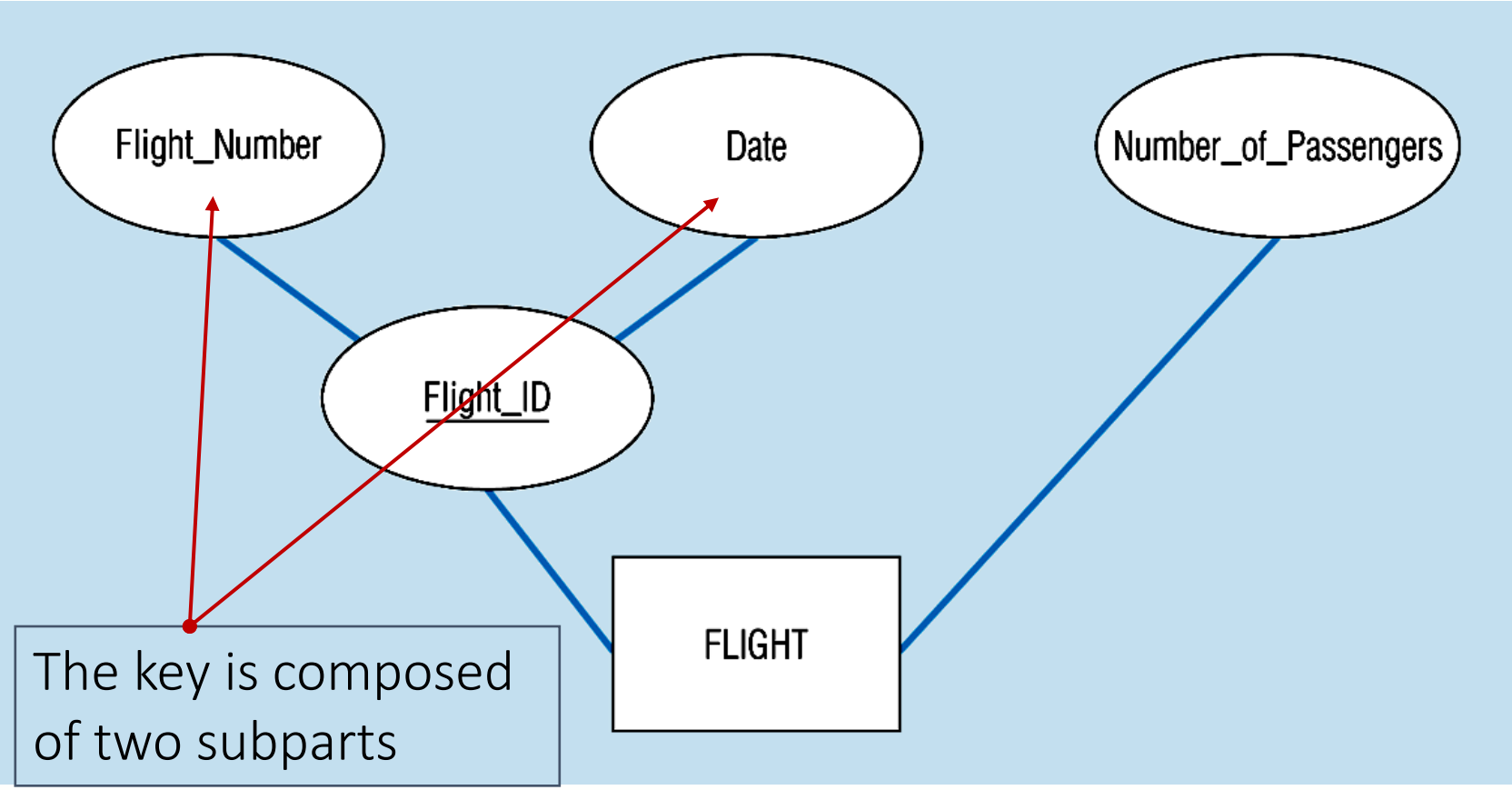
Characteristic of Identifier

- Will not change in value
- Will not be null
- No intelligent identifiers whose structure indicates classification, locations, or people that might change
- Substitute new, simple keys for long, composite keys

Simple Key Attribute

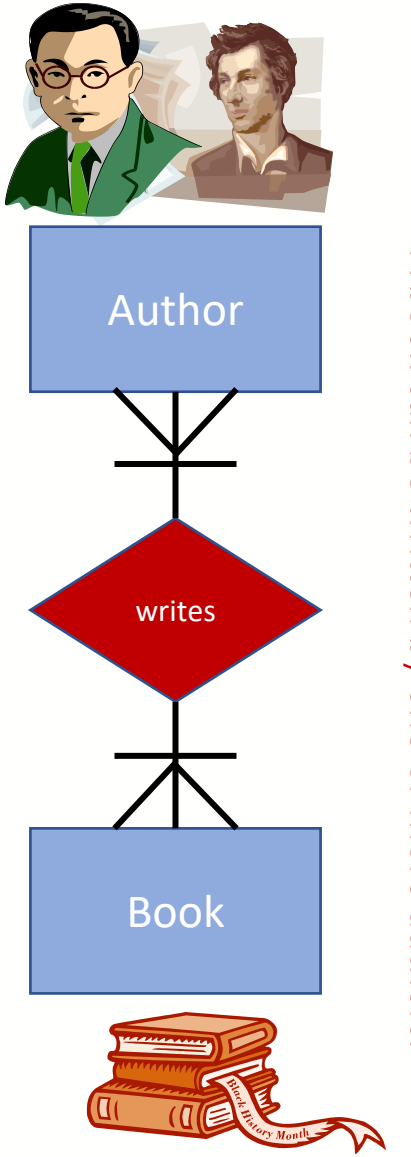


Composite Key Attribute

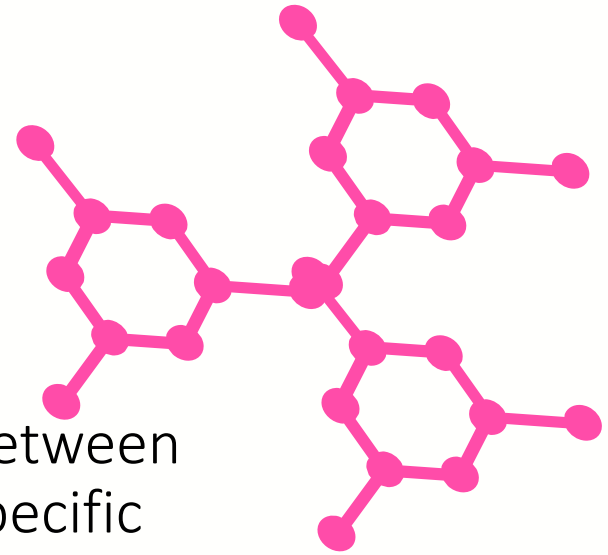


Relationship

- Associations between instances of one or more entity types that is of interest
- Given a name that describes its function.
  - relationship name is an active or a passive verb.

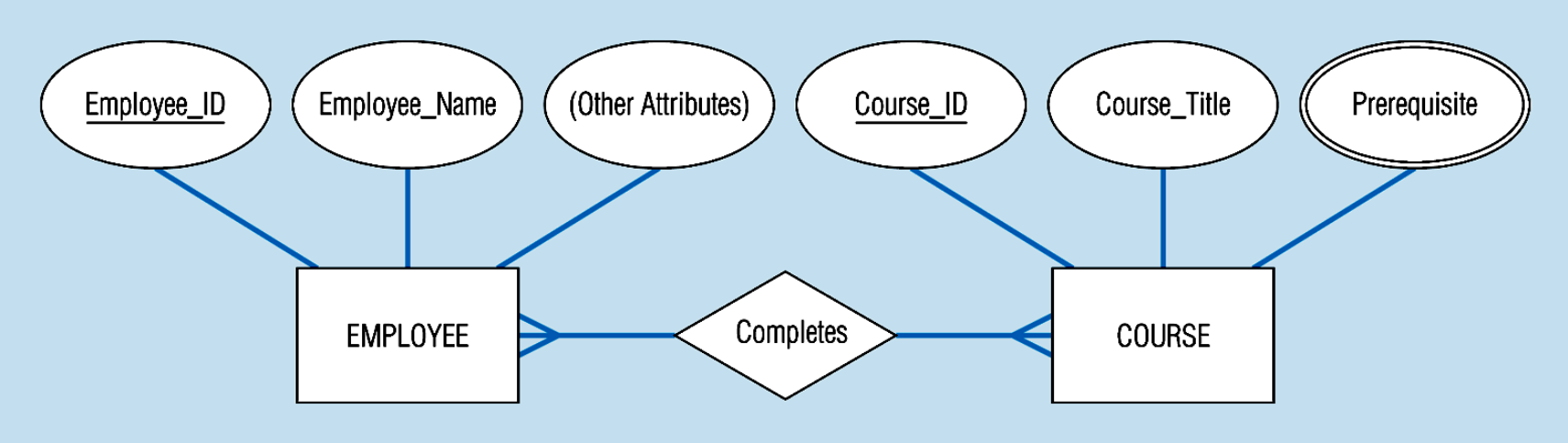


Relationship

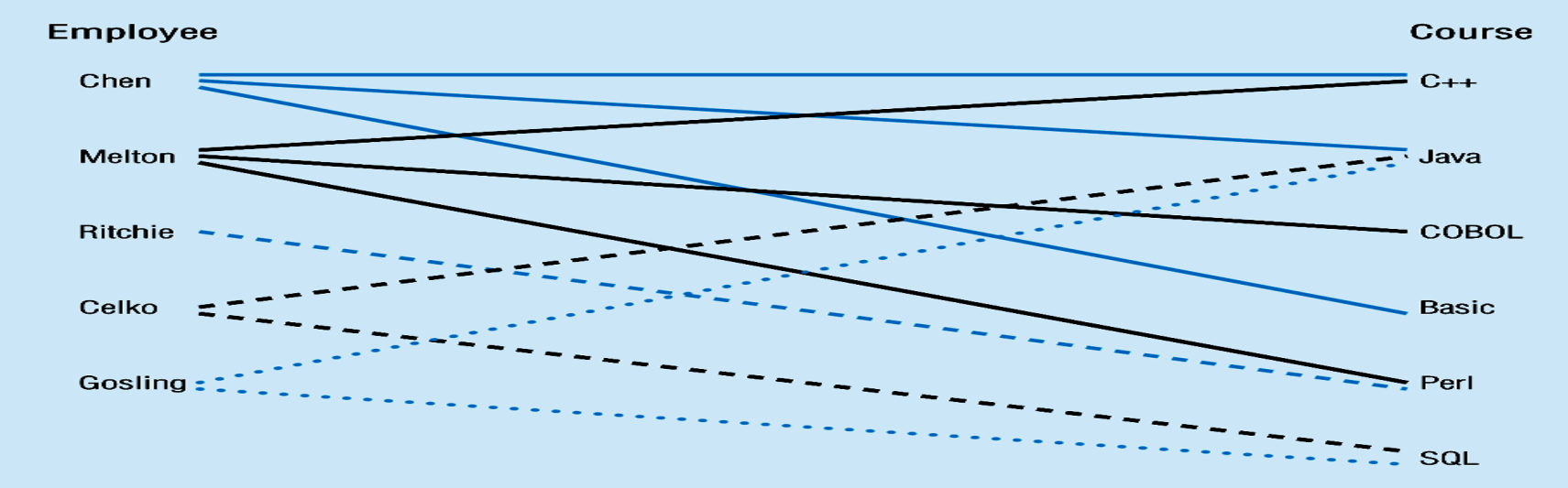


- Relationship Types vs. Relationship Instances
  - The relationship type is modeled as the diamond and lines between entity types, whereas the relationship instance is between specific entity instances
- Relationships can have attributes.
  - These describe features pertaining to the association between the entities in the relationship.
- Two entities can have more than one type of relationship between them (multiple relationships)
- Associative Entity = combination of relationship and entity

Relationship type

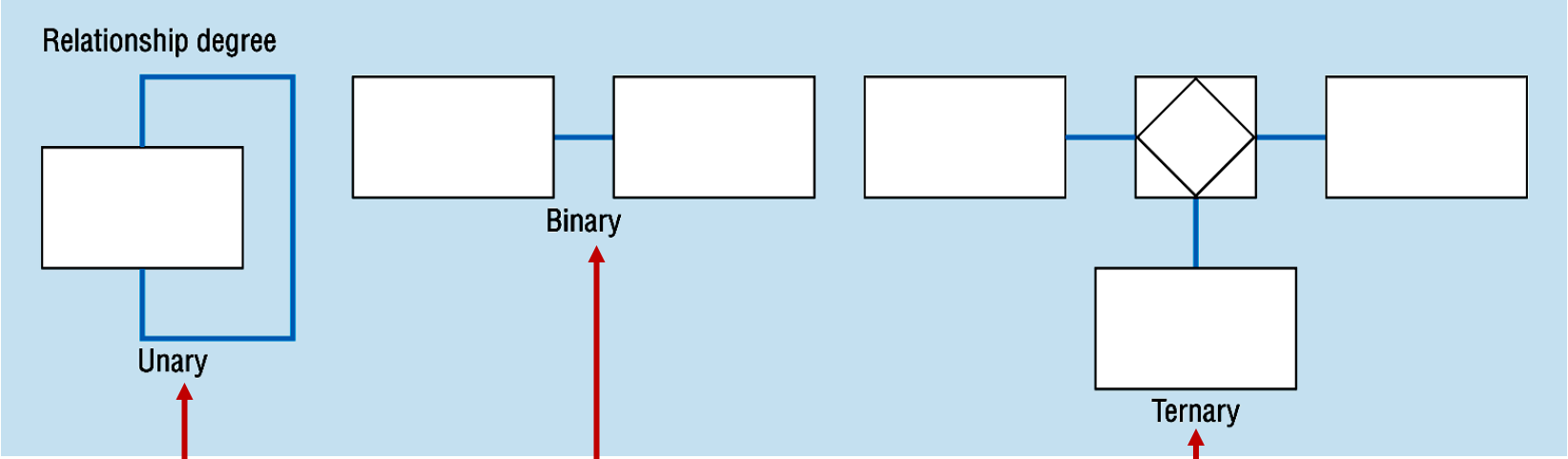


Entity and Relationship instances



Degree of Relationship

- The degree of a relationship is the number of entity types that participate in it
  - Unary Relationship
  - Binary Relationship
  - Ternary Relationship



One entity related to another of the same entity type

Entities of two different types related to each other

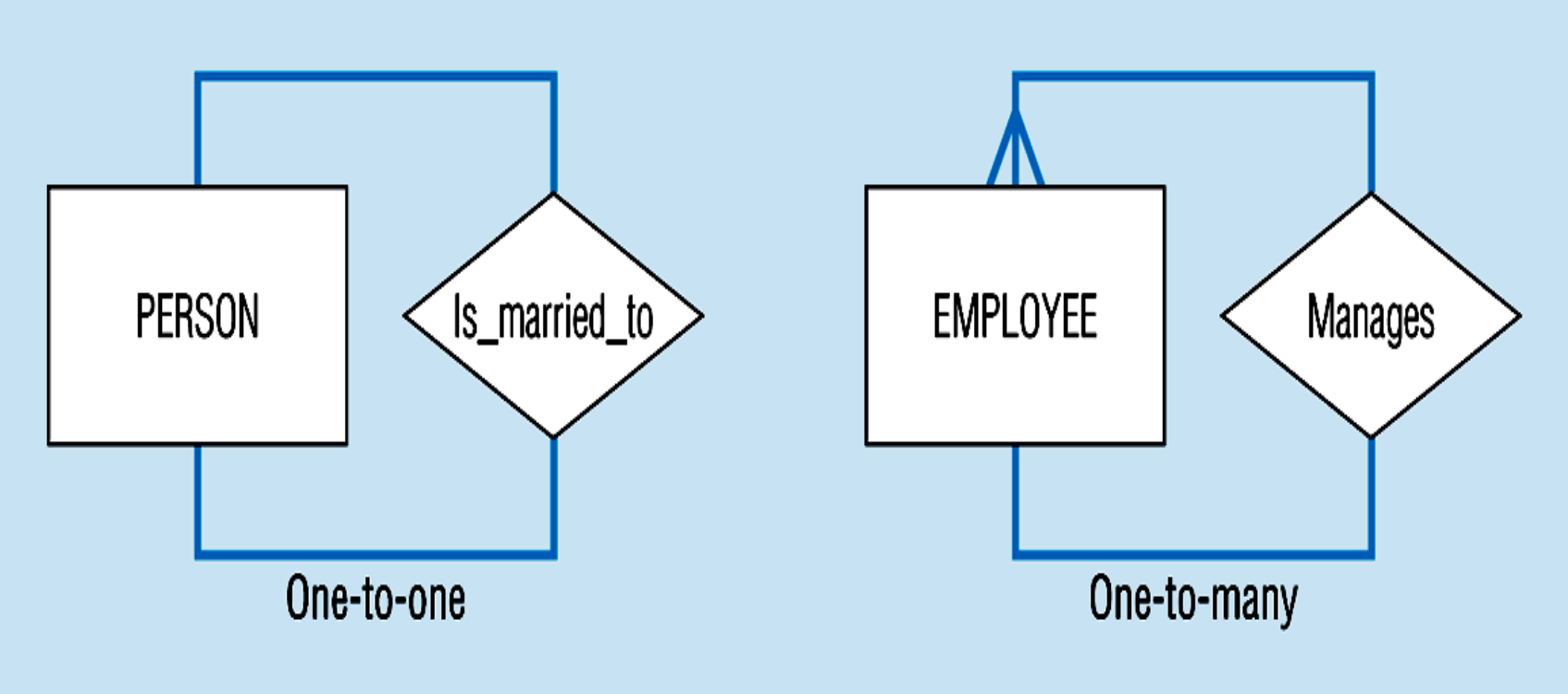
Entities of three different types related to each other

Cardinality of Relationship

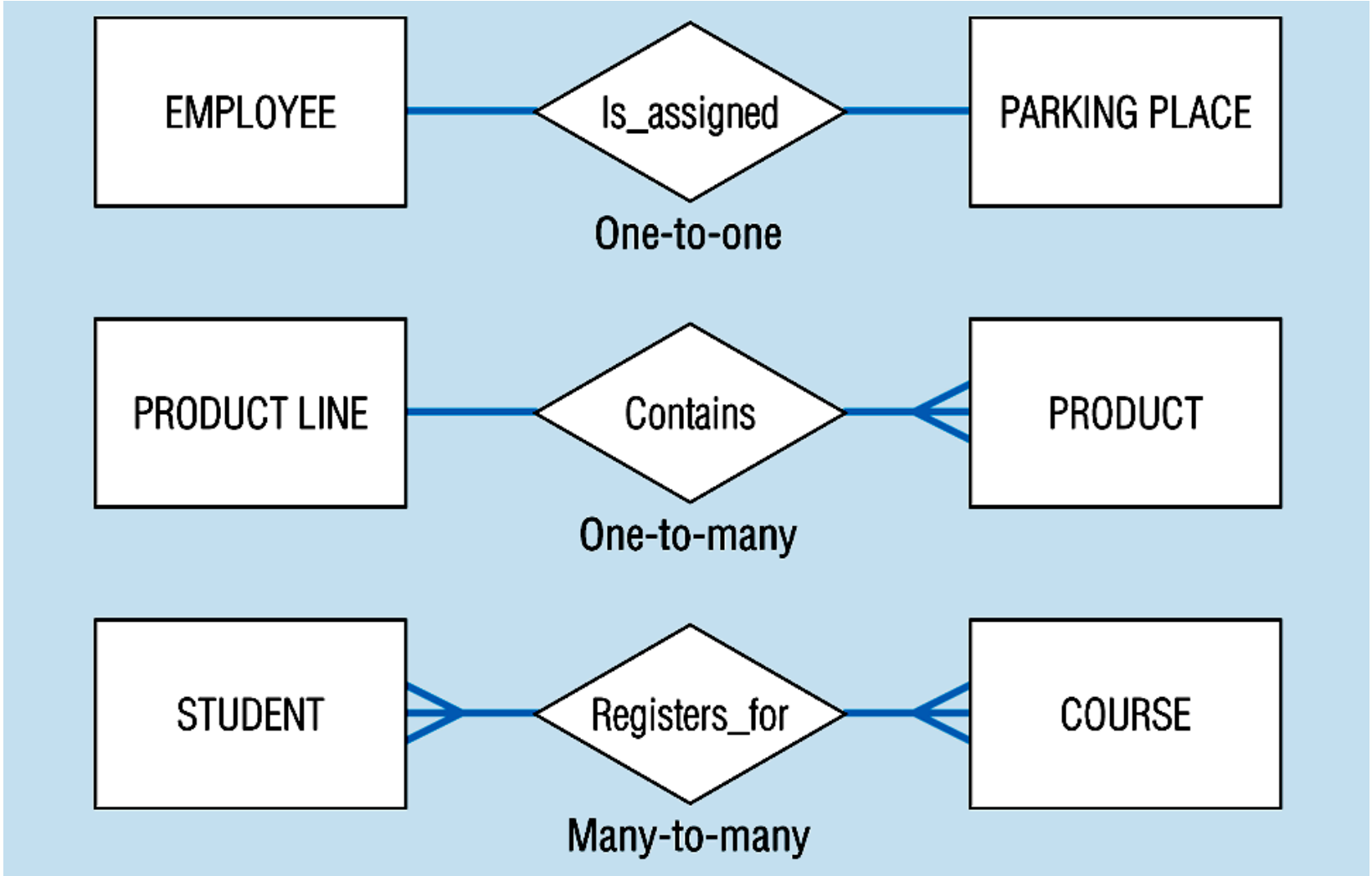
- One–to–One
  - Each entity in the relationship will have exactly one related entity instance.
- One–to–Many
  - An entity on one side of the relationship can have many related entity instances, but an entity on the other side will have a maximum of one related entity-instance
- Many–to–Many
  - Entities on both sides of the relationship can have many related entity instances on the other side.

Degree of relationships and Cardinality

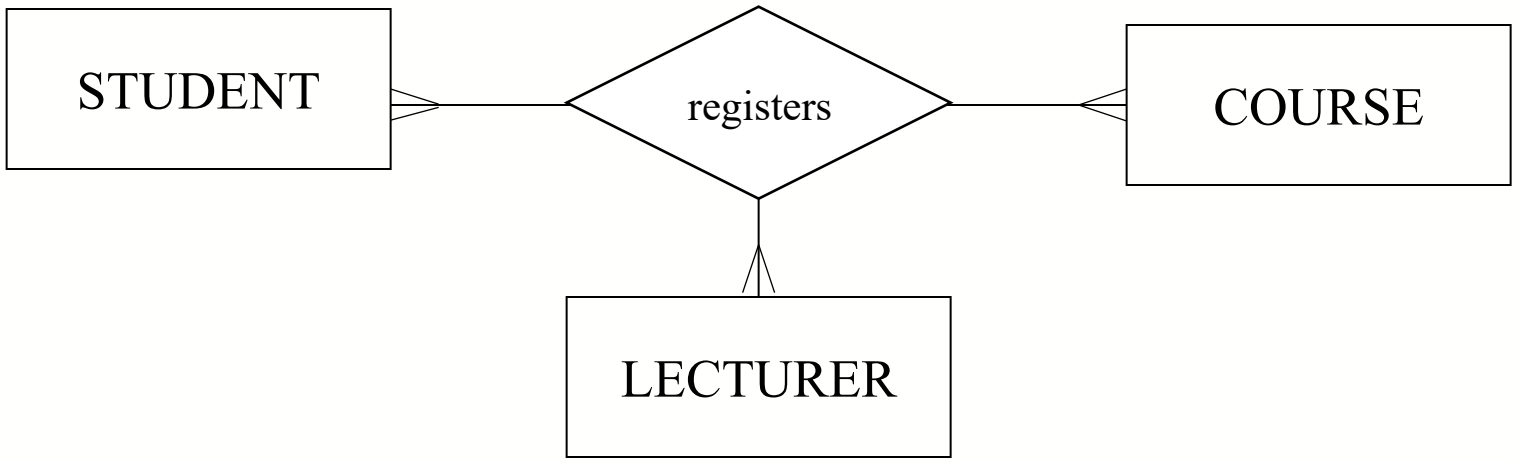
(a) **Unary relationships**



(b) Binary relationships

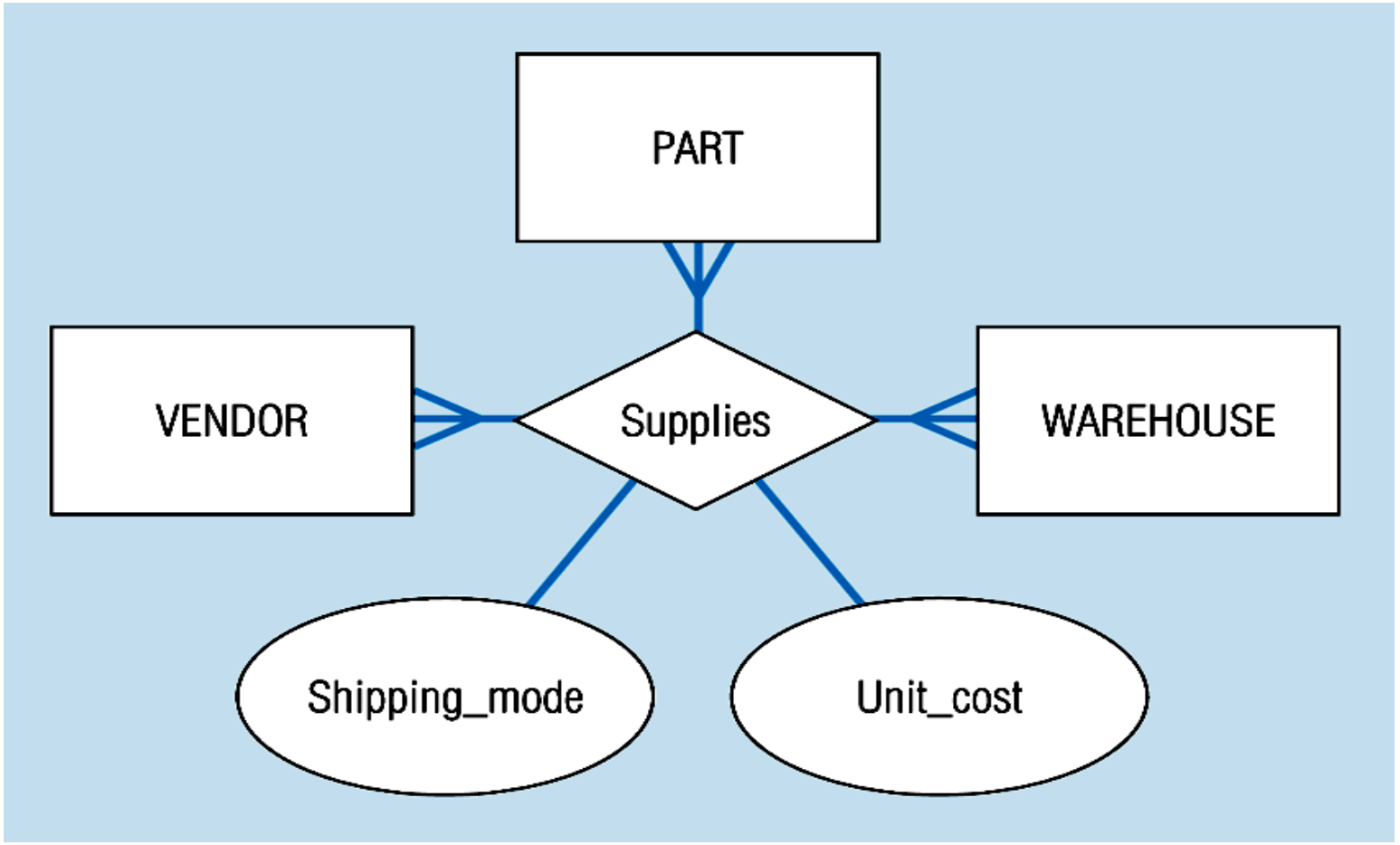


(c) Ternary relationships (1)



- A simultaneous relationship between instances of three entity types.
- The above E-R diagram shows that:
  - A student can register for many courses, and a course can be registered by many students.
  - A student can have many lectures, and a lecture can have many students.
  - A lecturer can teach many courses, and a course can be taught by many lecturers.

(c) Ternary relationships (2)

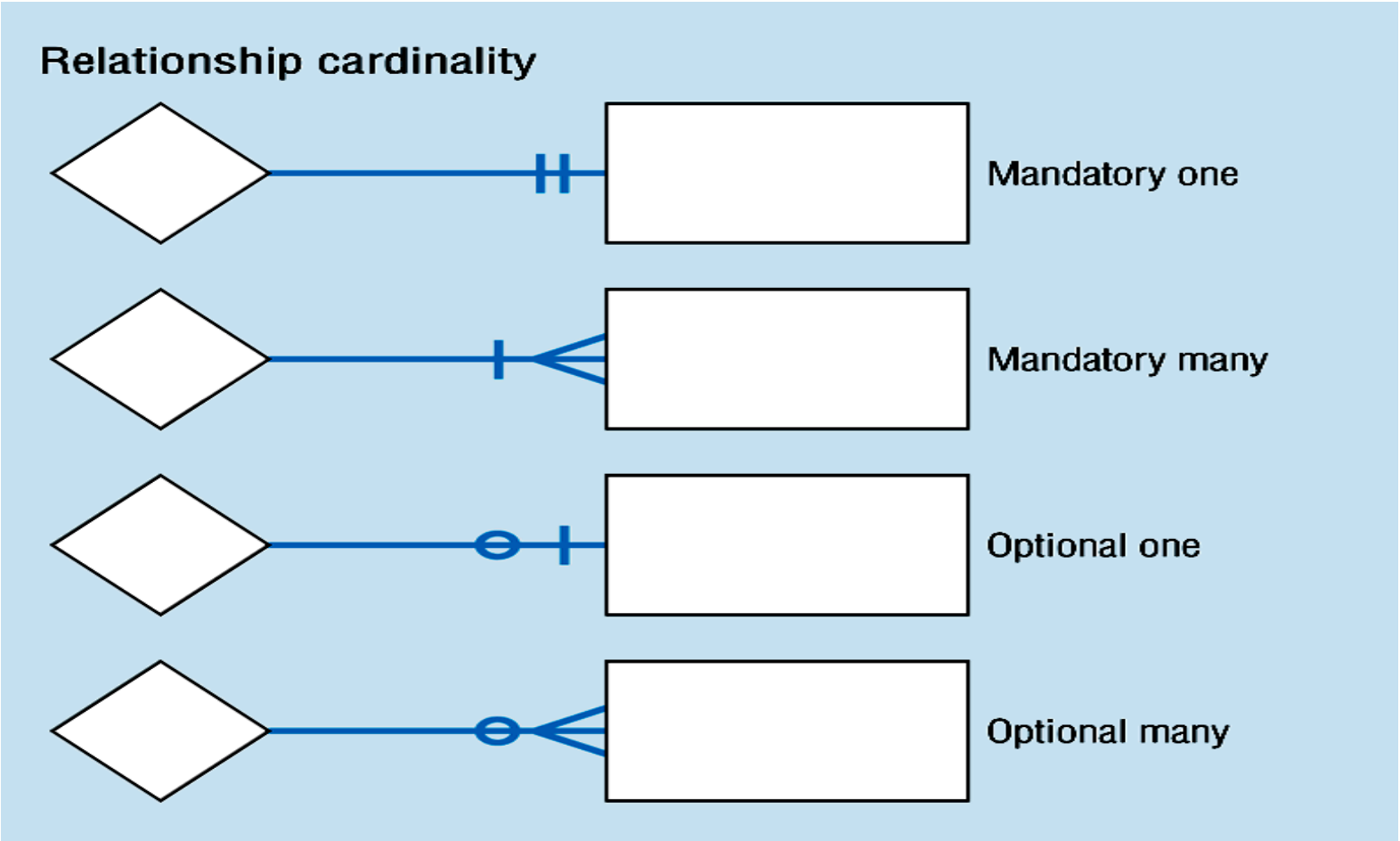


Note: a relationship can have attributes of its own

Cardinality Constraint

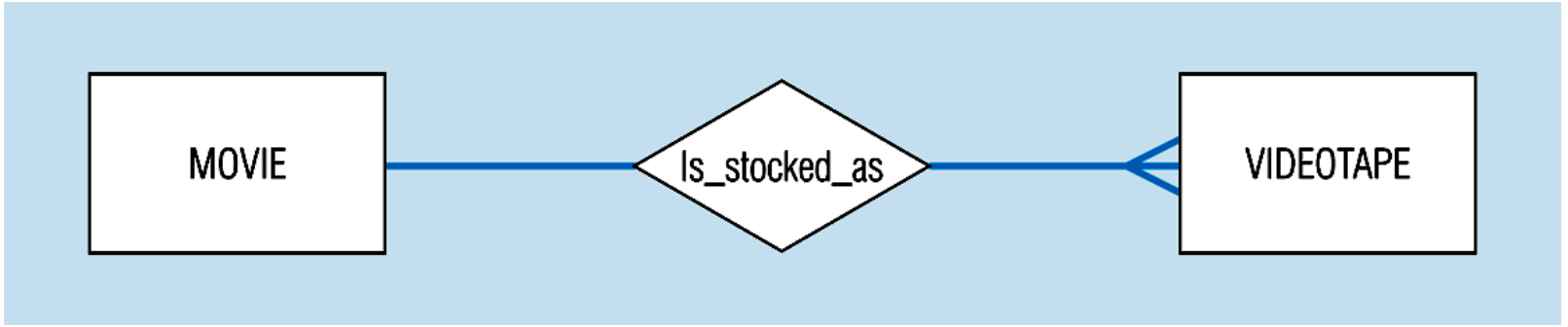
- Cardinality Constraints - the number of instances of one entity that can or must be associated with each instance of another entity.
- Minimum Cardinality
  - If zero, then optional
  - If one or more, then mandatory
- Maximum Cardinality
  - The maximum number

Cardinality Mandatory and Optional

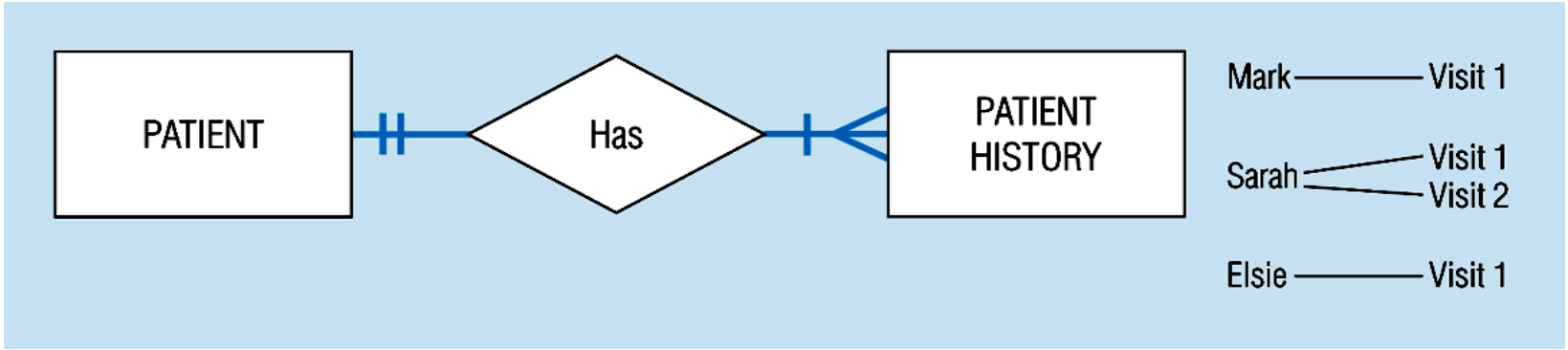


Cardinality Constraint

(a) Basic relationship with only maximum cardinalities showing

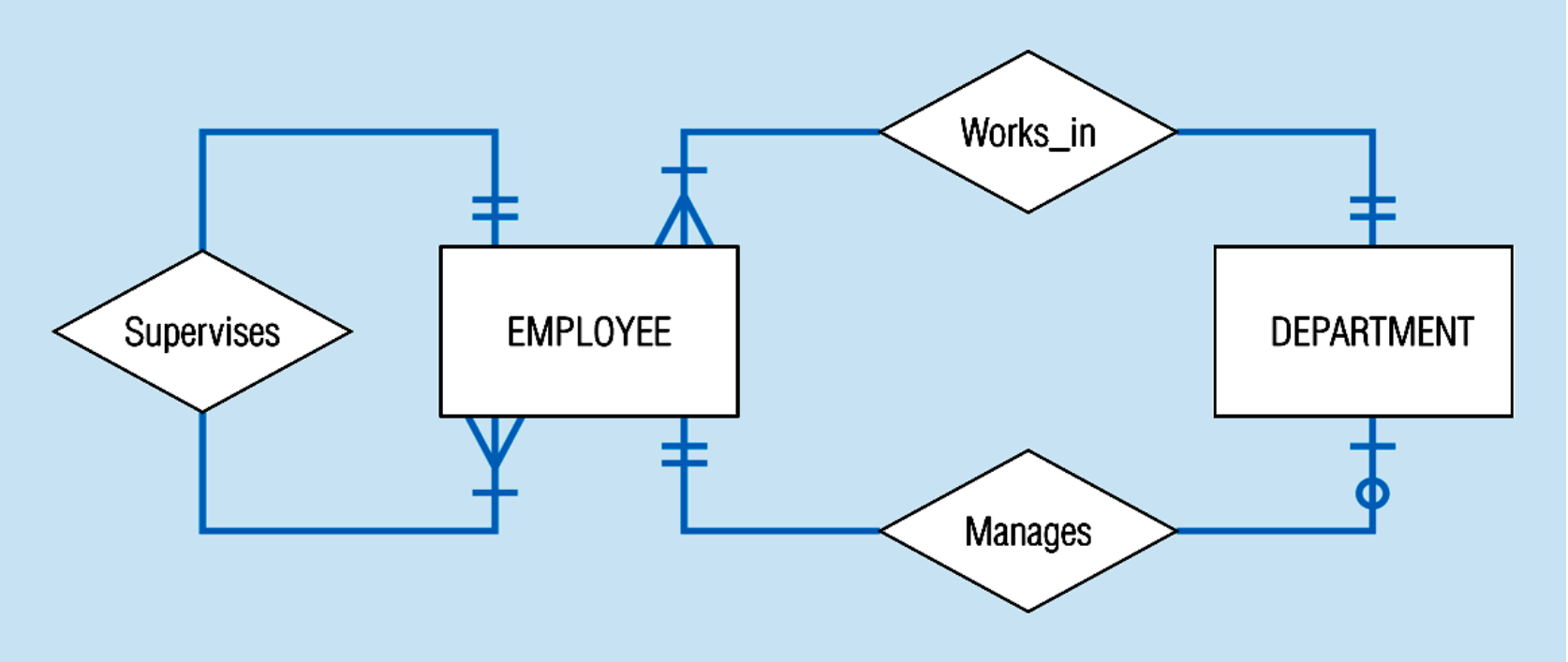


(b) Mandatory minimum cardinalities



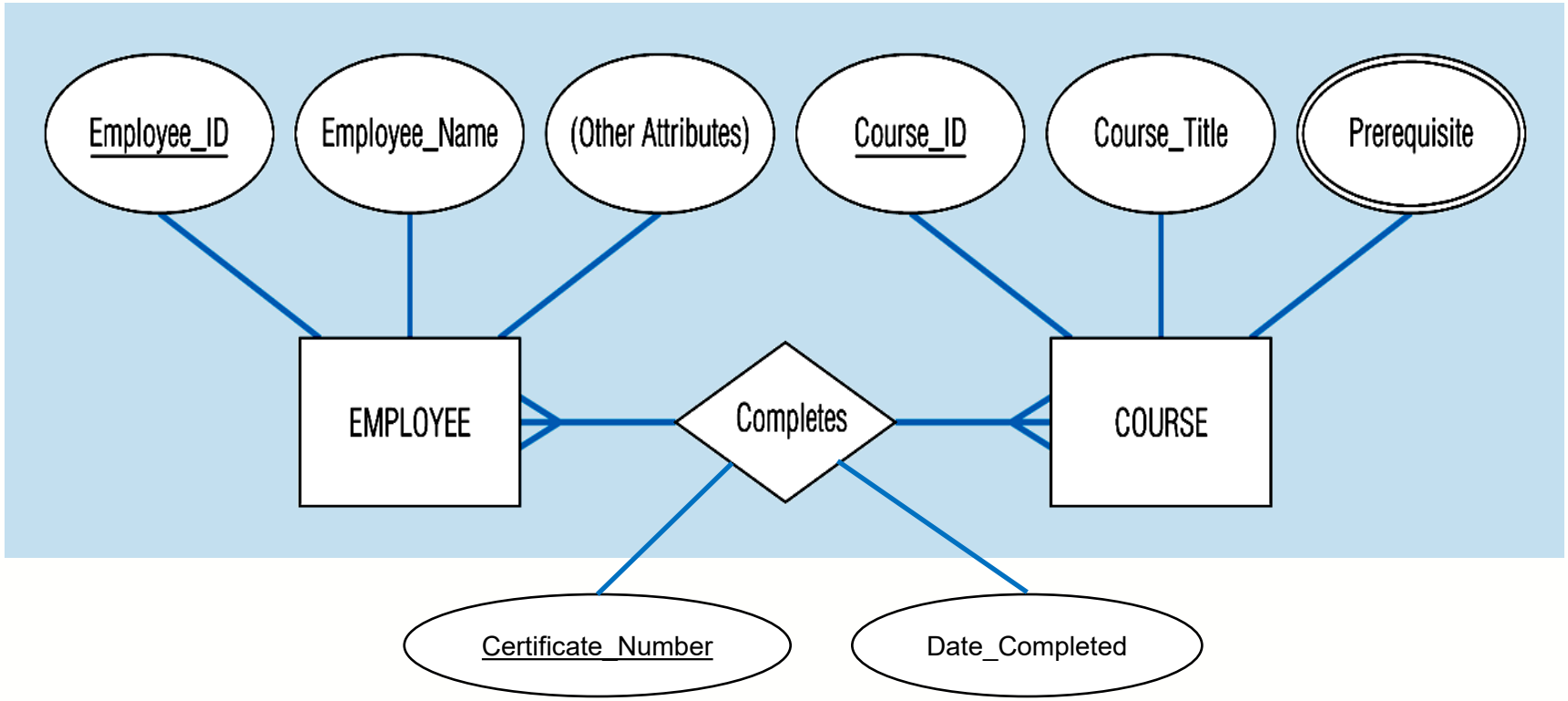
Example of multiple relationships

Employees and departments – entities can be related to one another in more than one way

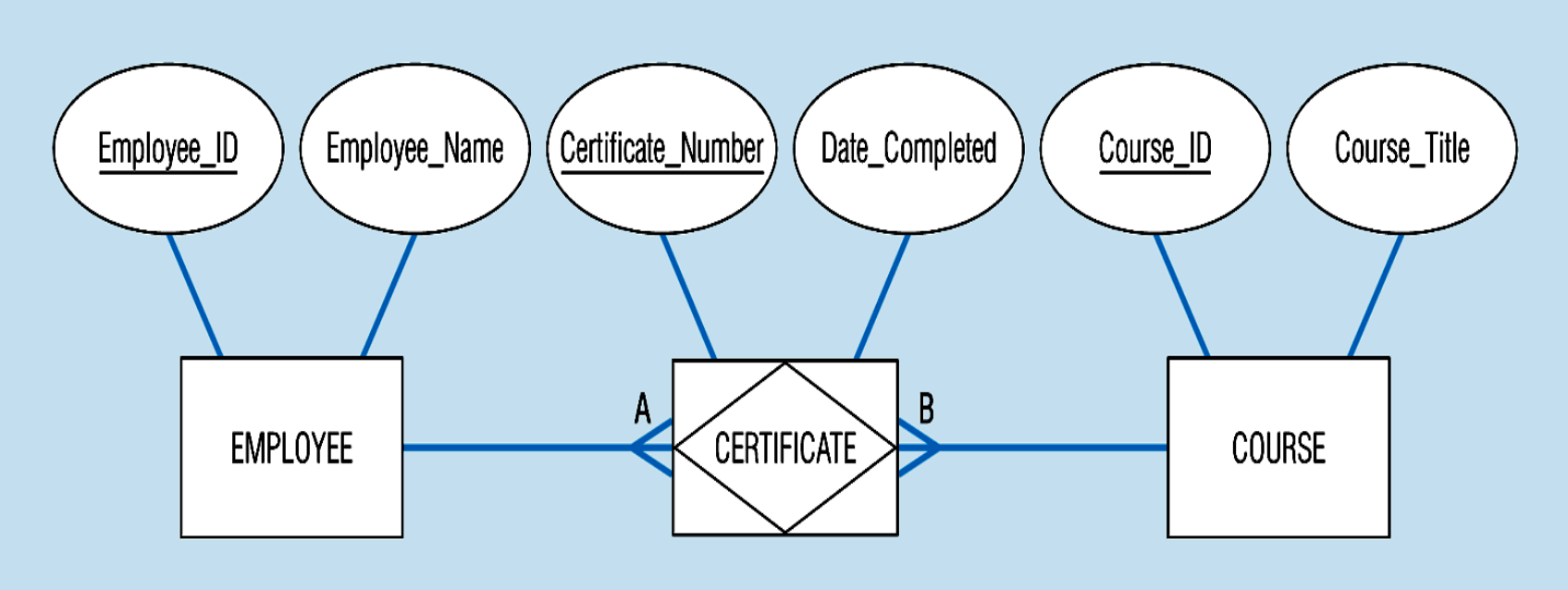


Associative Entity

- It's an entity – it has attributes.
- Also, it's a relationship – it links entities together.
- When should a *relationship with attributes* instead be an *associative entity*?
  - All relationships for the associative entity should be many
  - The associative entity could have meaning independent of the other entities
  - The associative entity preferably has a unique identifier and should also have other attributes
  - The associative may be participating in other relationships other than the entities of the associated relationship
  - Ternary relationships should be converted to associative entities

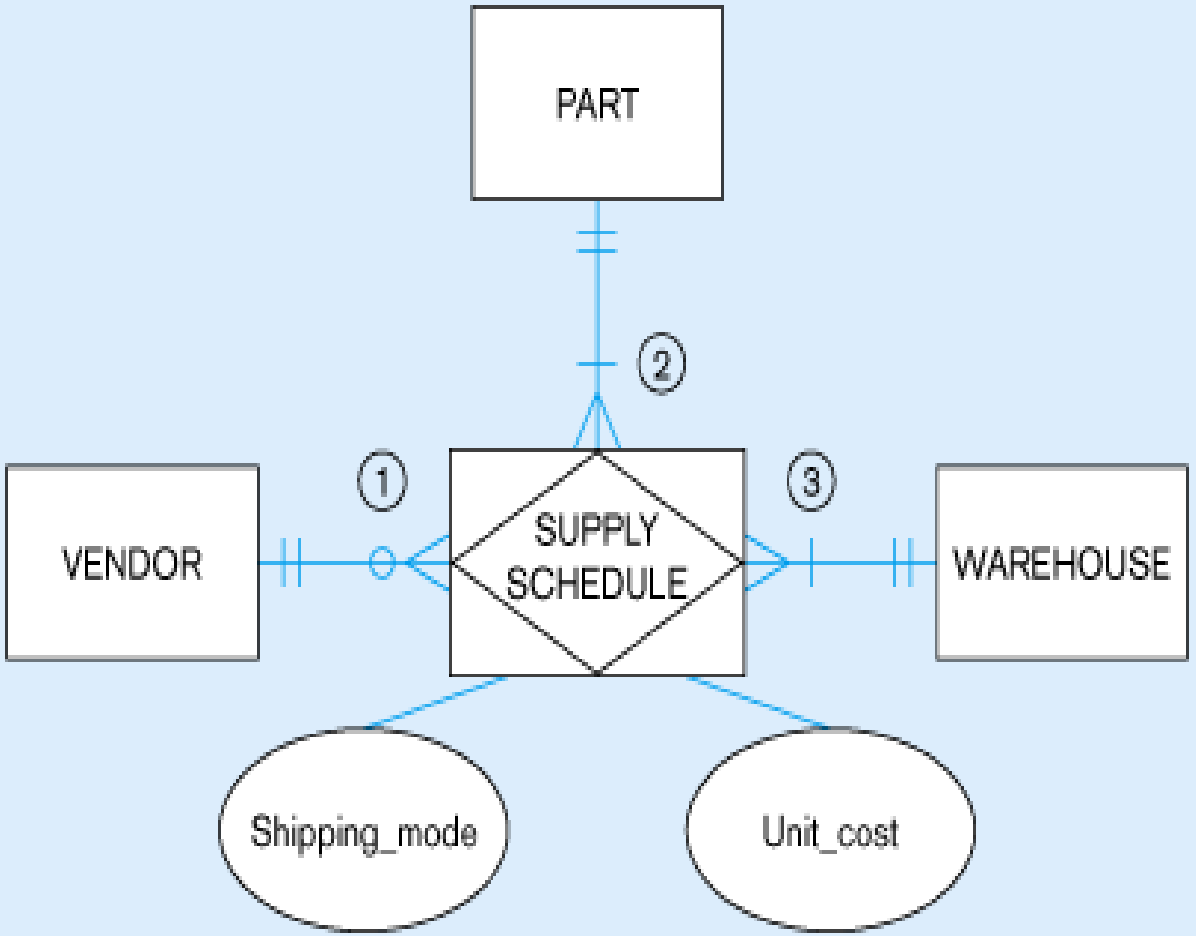

Employee_ID
Employee_Name
(Other Attributes)
Course_ID
Course_Title
Prerequisite
EMPLOYEE
Completes
COURSE
Certificate_Number
Date_Completed

An Example of Associative Entity - CERTIFICATE



An associative entity involves a rectangle with a diamond inside. Note that the many-to-many cardinality symbols face toward the associative entity and not toward the other entities.

(b)Ternary relationship as an associative entity



Business Rules

1. Each vendor can supply many parts to any number of warehouses, but need not supply any parts.
2. Each part can be supplied by any number of vendors to more than one warehouse, but each part must be supplied by at least one vendor to a warehouse.
3. Each warehouse can be supplied with any number of parts from more than one vendor, but each warehouse must be supplied with at least one part.

ERD Practice (1)

For each pair of sentences, identify entities and the relationship representing the sentences.

- "Each student must take one or more modules."
  "Each module must be taken by one or more student(s)."
- "Each customer must receive at least one delivery."
  "Each delivery must be only for one customer."
- "A client may have an account manager."
  "Each account manager has only one client."

ERD Practice (2)

An automobile insurance company needs to keep track of information about vehicle policyholders. The company has to store information about customers, cars, and accidents. Customer information is License_no, name, and address. Customers can own one or more cars, where the car information is Plate_no, model, and year. A car may be involved in many accidents or none. If the car is involved in an accident, the information stored is Report_number, location, and date. For each car that is involved in the accident, the damage amount will be estimated.
Draw the ERD.

ERD Practice (3)

A small company that rents canoes need a database to track basic information about the rental and the canoes. Eventually, the firm wants to identify customers who cause problems by damaging the canoes, but for now, the managers just want to track the costs. The managers have outlined the data as a form.

Draw an ER diagram for this case.

RentalID	**Canoe Rental**	Rent Date
Customer LastName, First Name E-Mail Phone Address City, State Postal Code Country		Credit Card Number Expiration Date Name on Card Deposit Amount

Number	Description: Length, Material	Returned Date	Fee	Damage Charge	Total

		Total

Enhanced ER (EER) Modeling

- EER is a high-level data model incorporating extensions to the original ER model.
- It is a diagrammatic technique for displaying the subclass and superclass; specialization and generalization; union or category; Aggregation, etc.
- Generalization and specialization are common relationships found in real entities.
- Specialized classes are often called subclass, while a generalized class is called a superclass.

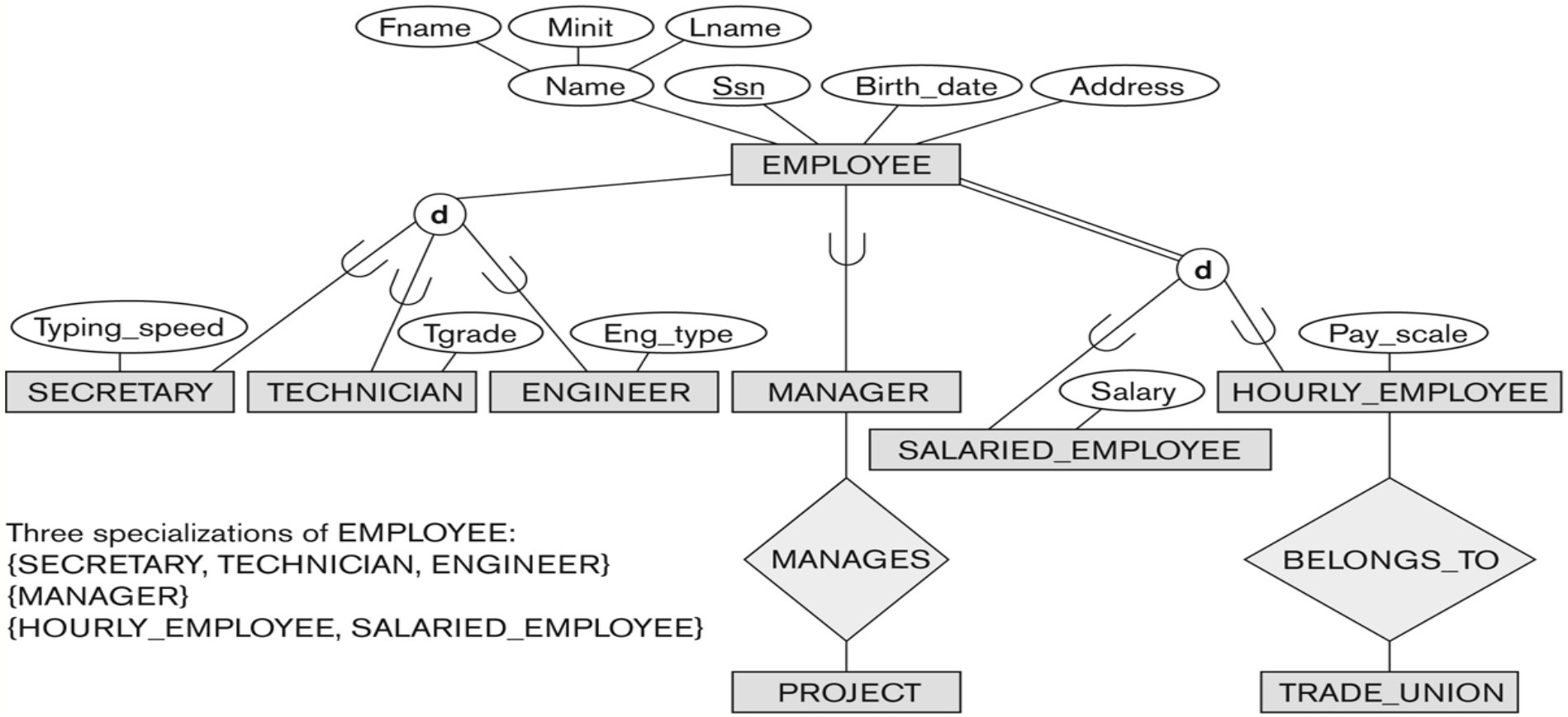
The concept of Generalization/Specialization

- Specialization is the process of defining a set of subclasses of a superclass.
- The set of subclasses is based upon some distinguishing characteristics of the entities in the superclass.

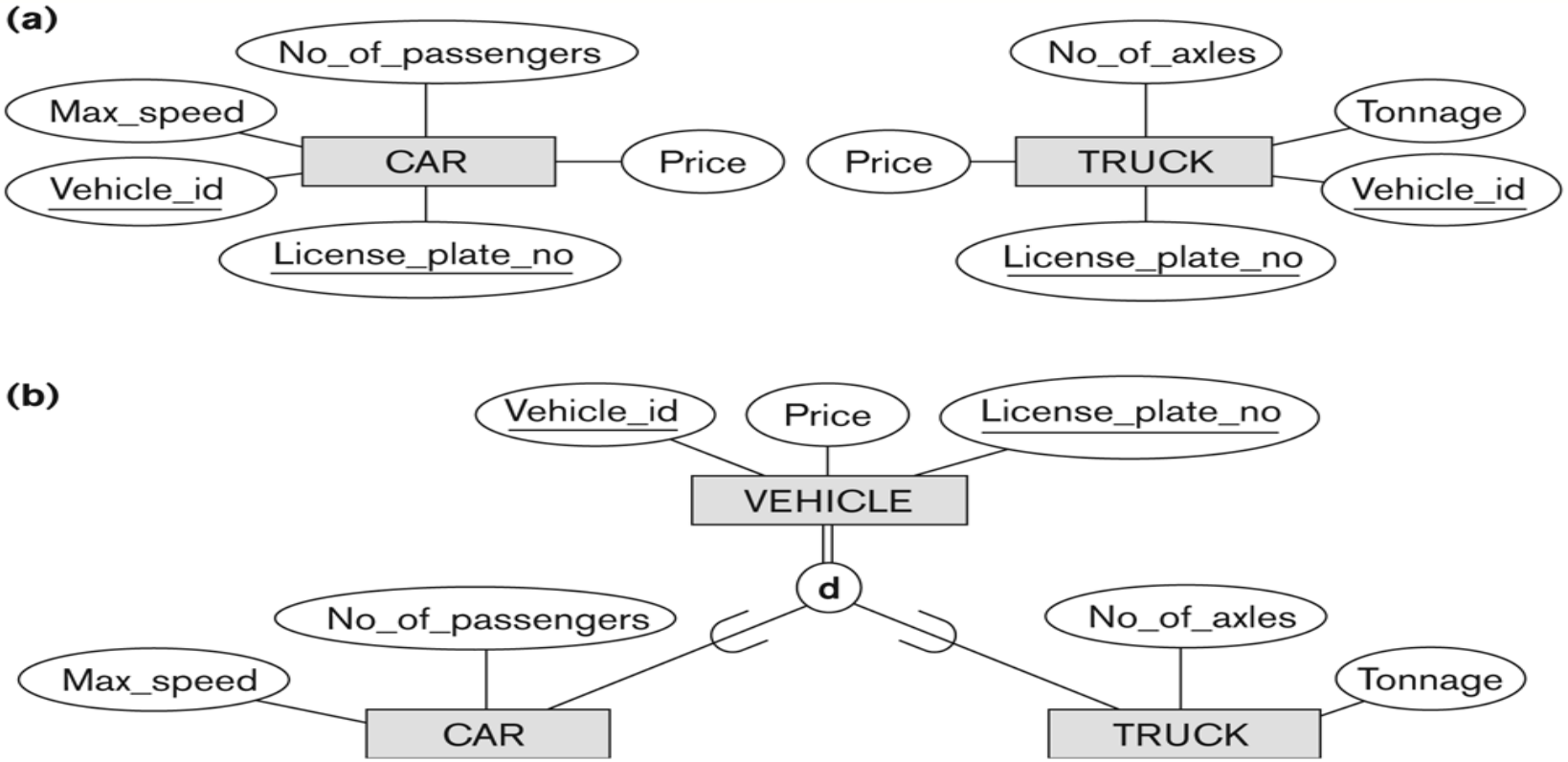
  Example: {SECRETARY, ENGINEER, TECHNICIAN} is a specialization of EMPLOYEE based upon *job type.*
- Generalization is the reverse of the specialization process.
- Several classes with common features are generalized into a superclass;
  - original classes become its subclasses

  Example: CAR, TRUCK generalized into VEHICLE;
    - both CAR, and TRUCK become subclasses of the superclass VEHICLE.
    - We can view {CAR, TRUCK} as a specialization of VEHICLE
    - Alternatively, we can view VEHICLE as a generalization of CAR and TRUCK

EER Diagram: Specialization



EER Diagram: Generalization



Generalization. (a) Two entity types, CAR and TRUCK.
(b) Generalizing CAR and TRUCK into the superclass VEHICLE.

Subclass

- It is best understood by ’IS-A analysis’.
- For example:
  - Student IS-A Person
  - Laptop IS-A Computer
  - Technician IS-A Employee
- An entity is a specialized class of another entity. For example, a technician is a special employee in an organization. Hence, the technician is a special class of Employee.
- A subclass entity inherits all attributes of its superclass.

Illustration of subclass

SUPERCLASS: EMPLOYEE

EMPID	NAME	ADDRESS	HP_NO
1001	SALLY	PENANG	0193323407
1005	AHMAD	SELANGOR	0132283838
1007	CHEN	KEDAH	0119292929
1008	RAVI	PUTRAJAYA	0111239939
1009	CHERYL	MELAKA	0139292930

SUBCLASS: SECRETARY

EMPID	TYPING SPEED
1001	68

SUBCLASS: TECHNICIAN

EMPID	TGRADE
1005	ELECTRICIAN
1007	AUTOMOBILE
1008	WELDER

SUBCLASS: ENGINEER

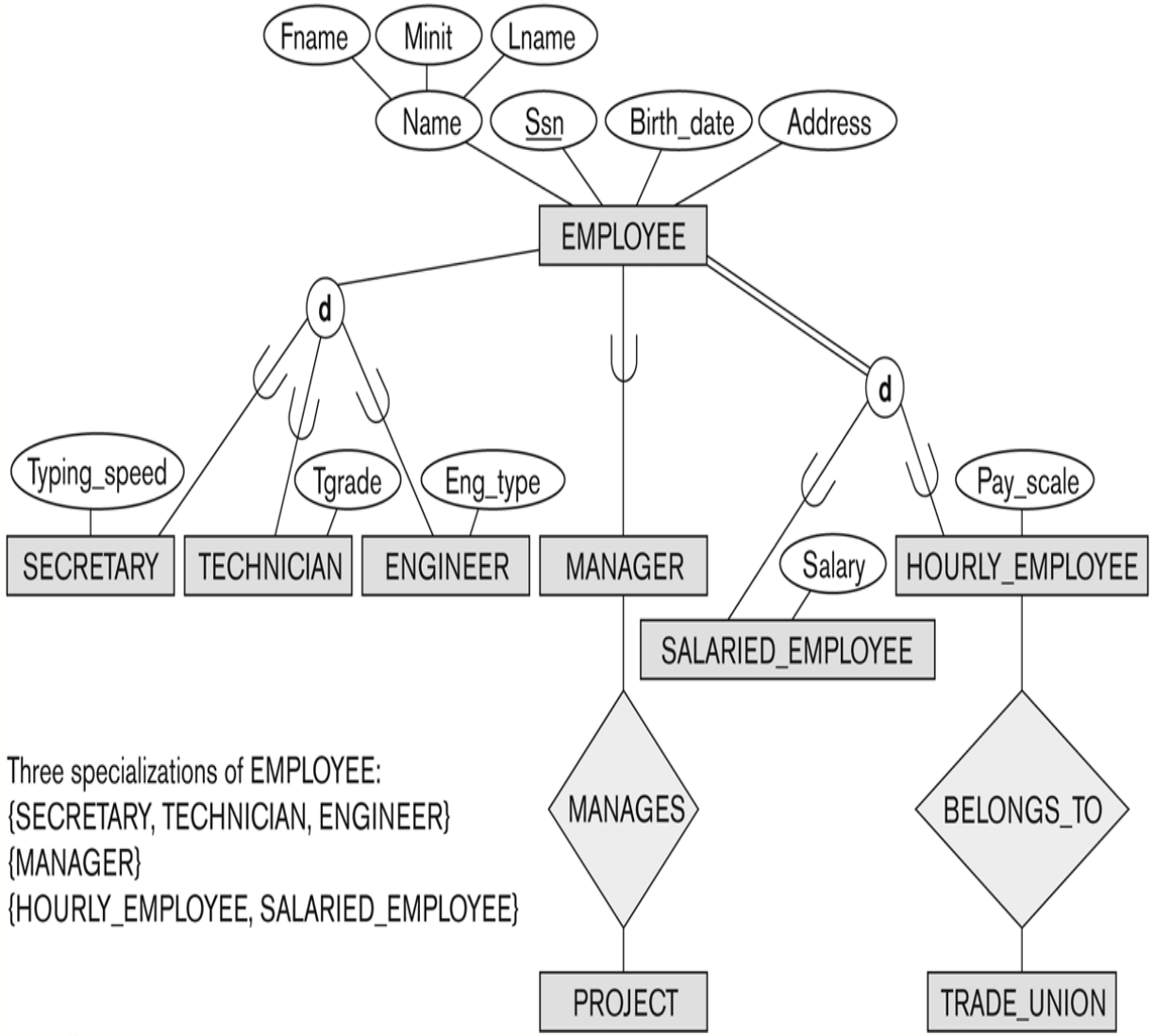
EMPID	TYPE
1009	MECHANICAL

CONSTRAINT OF SUBCLASS

- There are two types of constraints on the subclass relationship.
  - Total or Partial
  - Overlapped or Disjoint
- These constraints are independent of each other. Hence, it can be:
  - Overlapped and Total
  - Overlapped and Partial
  - Disjoint and Total
  - Disjoint and Partial

Total or Partial

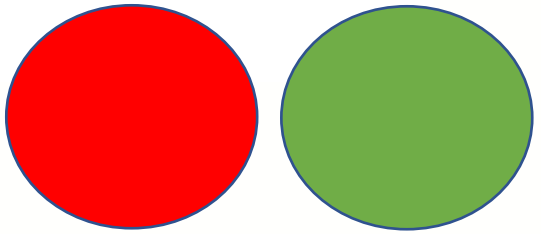
- A subclass relationship is total if every superclass entity is to be associated with some subclass entity. Otherwise, partial.
- For example, the subclass of Secretary, Technician, and Engineer is partial because not every employee is one of them.
- On the other hand, the subclass of Salaried_Employee and Hourly_Employee is total because every employee must be one of them.
- The total constraint is shown in the EER diagram by a double line.
- The partial constraint is shown in the EER diagram by a single line.



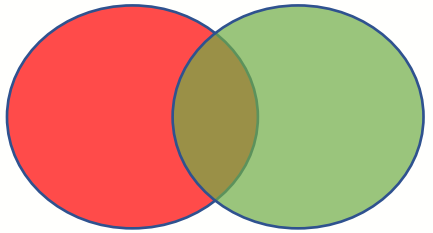
Overlapped or Disjoint

- If an entity from a superclass can be related to more than one subclass, then it is an overlapping specialization. Otherwise, it is disjoint.
- For example, both job-type and salary-type subclasses in the previous example are disjoint because an employee can only hold one job, either secretary, technician, or engineer. The same goes for the salary type.

Disjoint

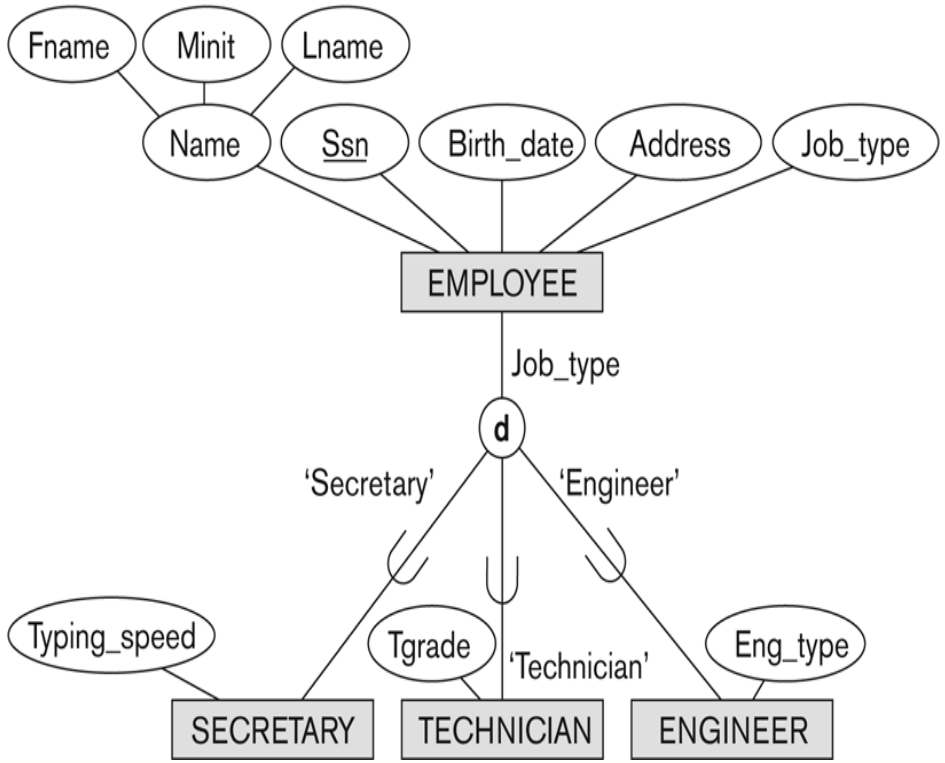


Overlapping

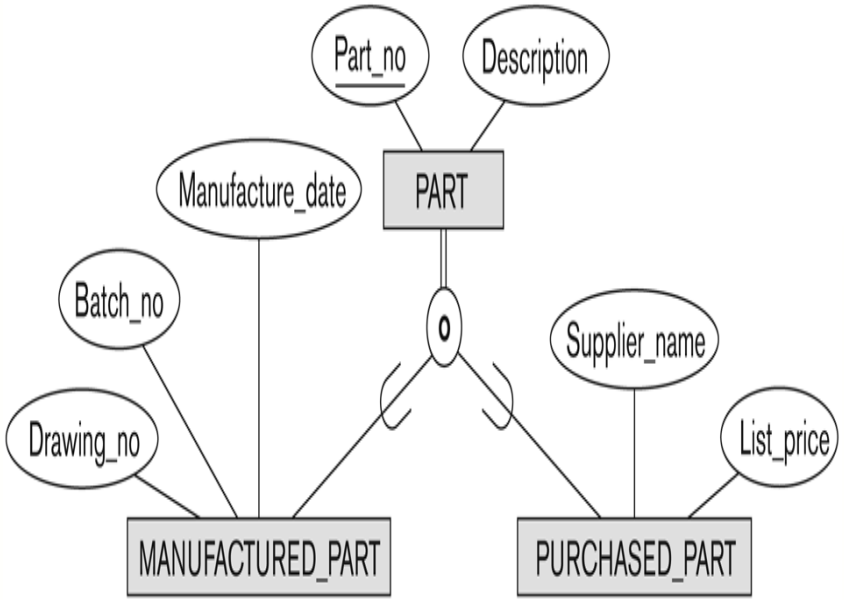


Example of Overlapped and Disjoint

Disjoint



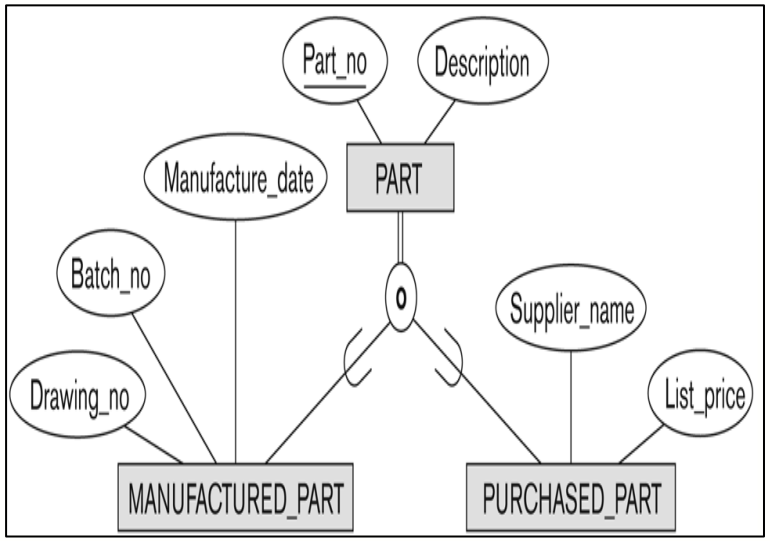
Overlapping

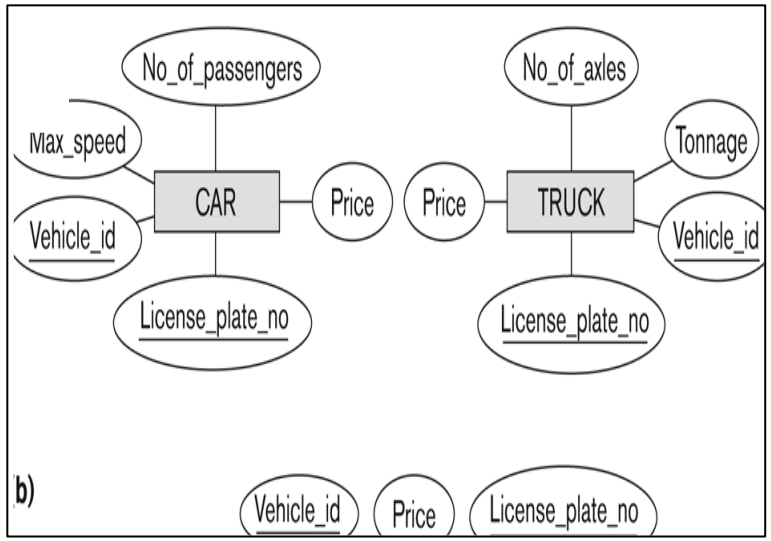


Overlapping

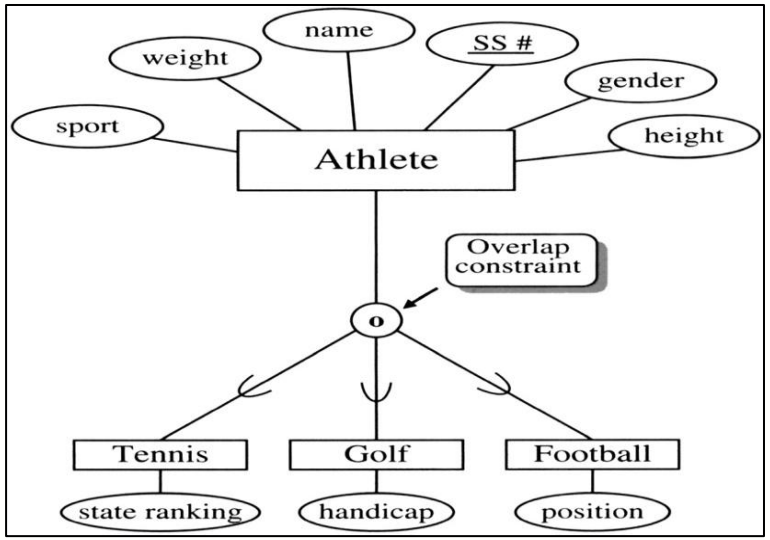
Disjoint

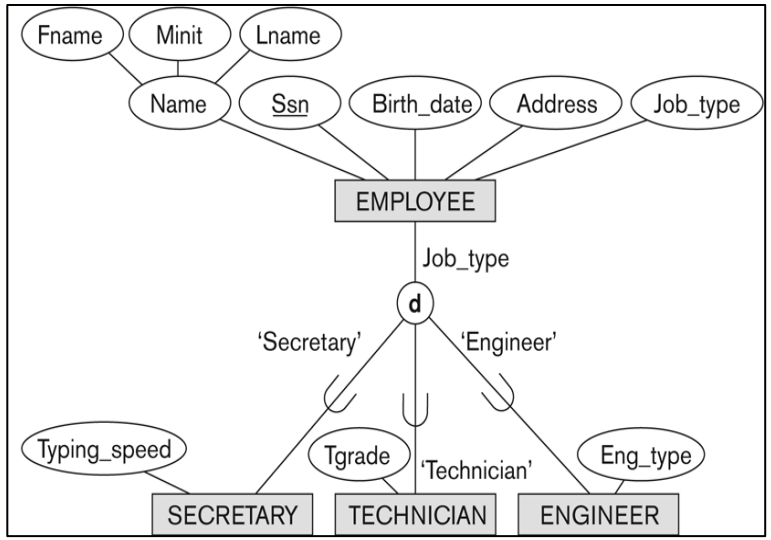
Total





Partial





Hierarchies, Lattices & Shared Subclasses (1)

A subclass may itself have further subclasses specified on it

- forms a hierarchy or a lattice

Hierarchy has a constraint that every subclass has only one superclass (called ***single inheritance***); this is basically a ***tree structure***

In a ***lattice***, a subclass can be subclass of more than one superclass (called ***multiple inheritance***)

Hierarchies, Lattices & Shared Subclasses (2)

In a lattice or hierarchy, a subclass inherits attributes not only of its direct superclass, but also of all its predecessors' superclasses

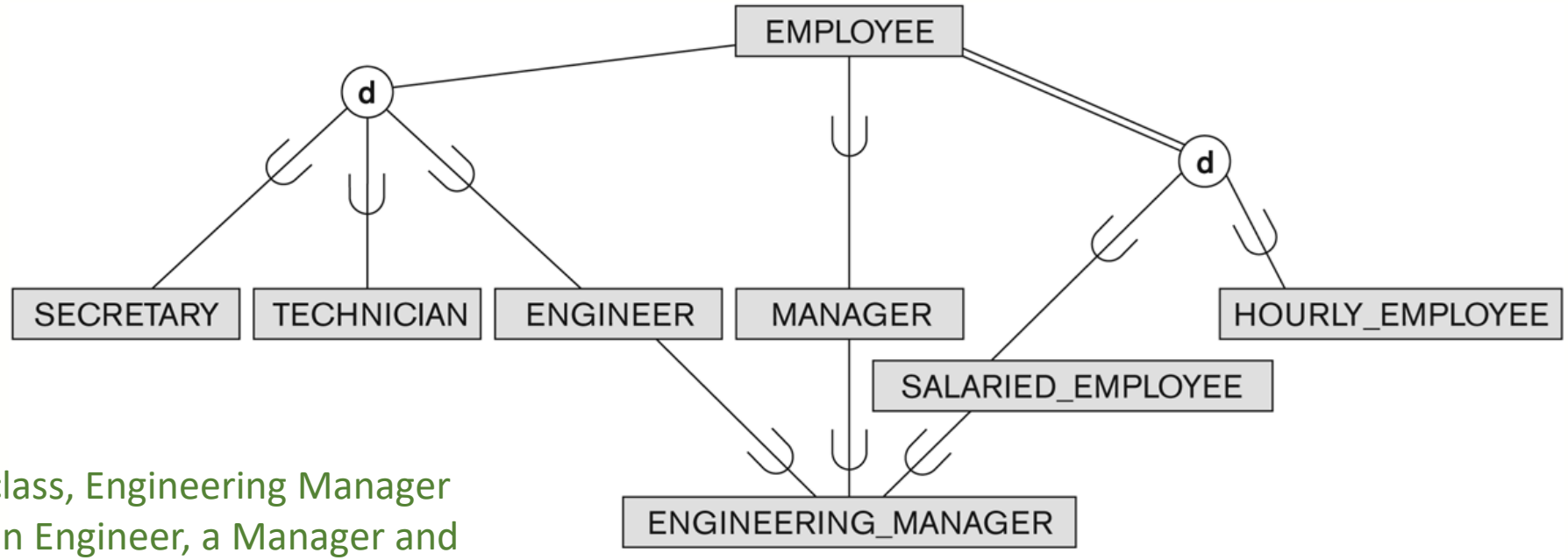
A subclass with more than one superclass is called a **shared subclass** (multiple inheritance)

Can have:

- *specialization* hierarchies or lattices, or
- *generalization* hierarchies or lattices,
- depending on how they were *derived*

We just use *specialization* (to stand for the end result of either specialization or generalization)

Shared Subclass: Engineering_Manager



A specialization lattice with shared subclass ENGINEERING_MANAGER.

A shared subclass, Engineering Manager must be ALL an Engineer, a Manager and a Salaried-Employee

Hierarchies, Lattices & Shared Subclasses (3)

In *specialization*, start with an entity type and then define subclasses of the entity type by successive specialization

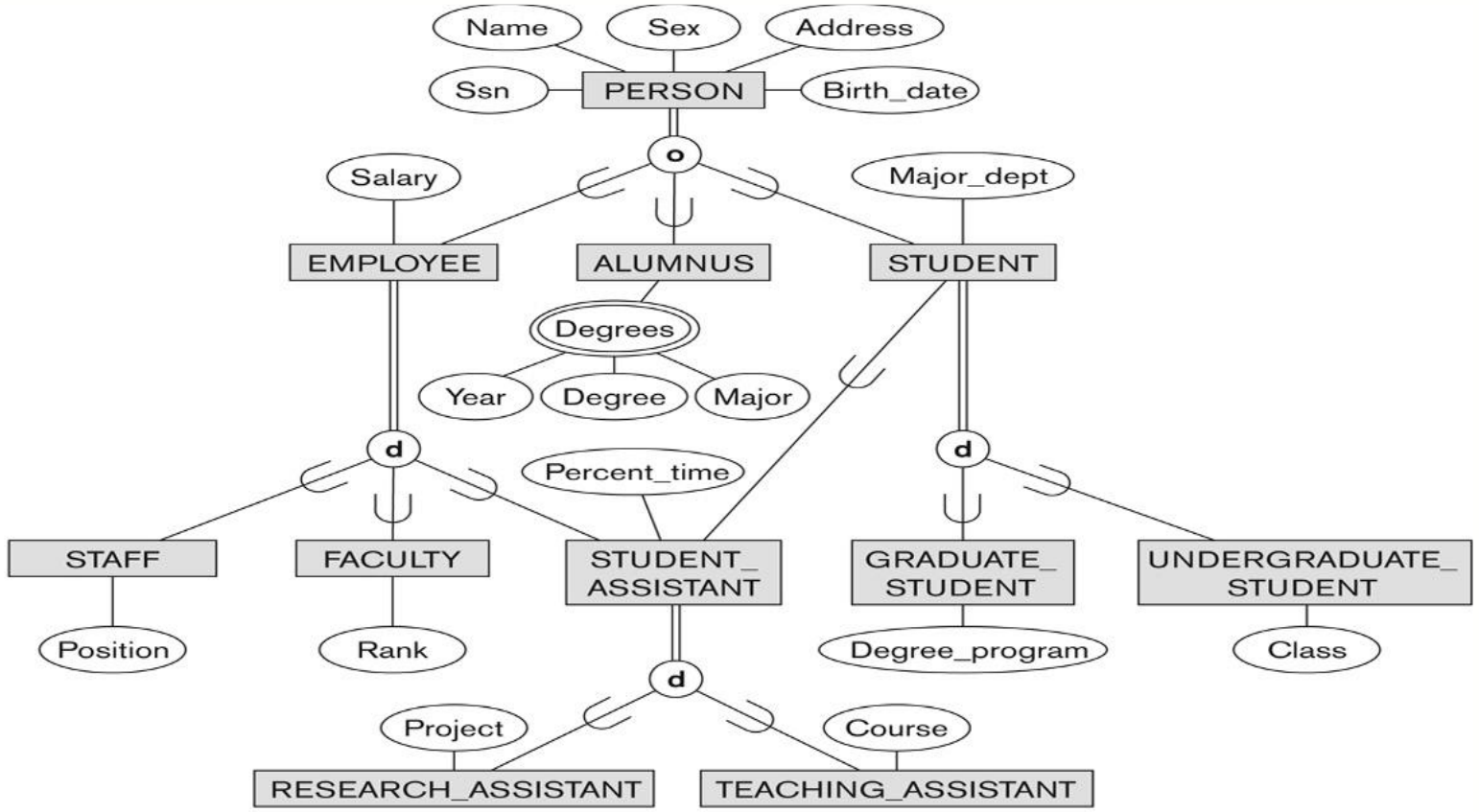
- called a *top down* conceptual refinement process

In *generalization*, start with many entity types and generalize those that have common properties

- Called a *bottom up* conceptual synthesis process

In practice, a *combination of both processes* is usually employed

EER Diagram: Lattice Example



A specialization lattice with multiple inheritance for a UNIVERSITY database.

EERD Practice (1)

A bank has three types of accounts which are Current, Savings, and Loan. The following are attributes for each type of account.

Current: account_No, open_date, balance, service_charge

Savings: account_No, open_date, balance, interest_rate

Loan: account_No, open_date, balance, interest_rate, payment

Assume that each bank account must be a member of exactly one of these subtypes.

Draw the EERD.

EERD Practice (2)

Each trainee is identified by a trainee code, and we want to store his/her name, age, sex, and place of birth. We also want to store each trainee's previous employers, which contains information on the employer's name, employer address, and employer telephone number beside the trainees' periods employed (can be determined by checking between the past employment start date and end date). To work with us, each trainee must have at least one employment history as a basis. We also notice there are some trainees who worked at the same companies previously, especially from Oracle and Microsoft companies. The trainee also needs to be classified as either self-employed or an employee. If a trainee is self-employed (i.e., professional), we need to know her area of expertise and, if appropriate, her professional title. For somebody who works for a company, we store the level and position held.

EERD Practice (3)

A non-profit organization depends on a number of different types of persons for its successful operation. The organization is interested in the following attributes for all of these persons: Social Security Number, Name, Address, City, State, and Telephone. Three types of persons are of interest: employees, volunteers, and donors. Employees have only a Date_Hired attribute, and volunteers have only a Skill attribute. Donors have a relationship (named Donates) with an Item. A donor must have donated one or more Items, and an Item can only be donated by one donor. Attributes of an item include an identity and a description. There are persons other than employees, volunteers, and donors who are of interest to the organization, so a person does not have to belong to one of these groups. A person may also belong to one or more of these groups, at any time.

EERD Practice (4)

Attic Antiques buys and sells one-of-a-kind antiques of all kinds (e.g., furniture, China, clothing, etc.). Each item is uniquely identified by a serial number and is characterized by the asking price and condition. Attic works with several individuals who sell and buy items from the store. Some clients only sell items to Attic, others only buy, and some both buy and sell. The Attic keeps track of its clients through the assigning of client numbers. They also keep track of clients' names and addresses. When Attic sells an item to a client, they need to keep track of the actual selling price, the date of the sale, and the sales tax. When Attic buys an item, they wish to track the purchase cost, condition at the time of purchase, and the date.

End of Chapter