CSIE30600/CSIEB0290
Database Systems

# Lecture 8: Enhanced Entity-Relationship(EER) Model

## Outline

- EER stands for Enhanced ER or Extended ER
- EER Model Concepts
  - Includes all modeling concepts of basic ER
  - Additional concepts:
    - subclasses/superclasses
    - specialization/generalization
    - categories (UNION types)
    - attribute and relationship inheritance
  - These are fundamental to conceptual modeling
- The additional EER concepts are used to model applications more completely and more accurately
  - EER includes some object-oriented concepts, such as inheritance

## The Enhanced Entity-Relationship (EER) Model

- **Enhanced ER (EER) model**
  - Created to design more accurate database schemas
    - Reflect the data properties and constraints more precisely
  - More complex requirements than traditional applications



## Subclasses, Superclasses, and Inheritance

- EER model includes all modeling concepts of the ER model
- In addition, EER includes:
  - **Subclasses** and **superclasses**
  - **Specialization** and **generalization**
  - **Category** or **union type**
  - **Attribute** and **relationship inheritance**
- EER includes some object-oriented concepts

# Subclasses and Superclasses (1)

- An entity type may have additional meaningful subgroupings of its entities
  - Example: EMPLOYEE may be further grouped into:
    - SECRETARY, ENGINEER, TECHNICIAN, …
      - Based on the EMPLOYEE's Job
    - MANAGER
      - EMPLOYEEs who are managers
    - SALARIED_EMPLOYEE, HOURLY_EMPLOYEE
      - Based on the EMPLOYEE's method of pay
- EER diagrams extend ER diagrams to represent these additional subgroupings, called subclasses or subtypes



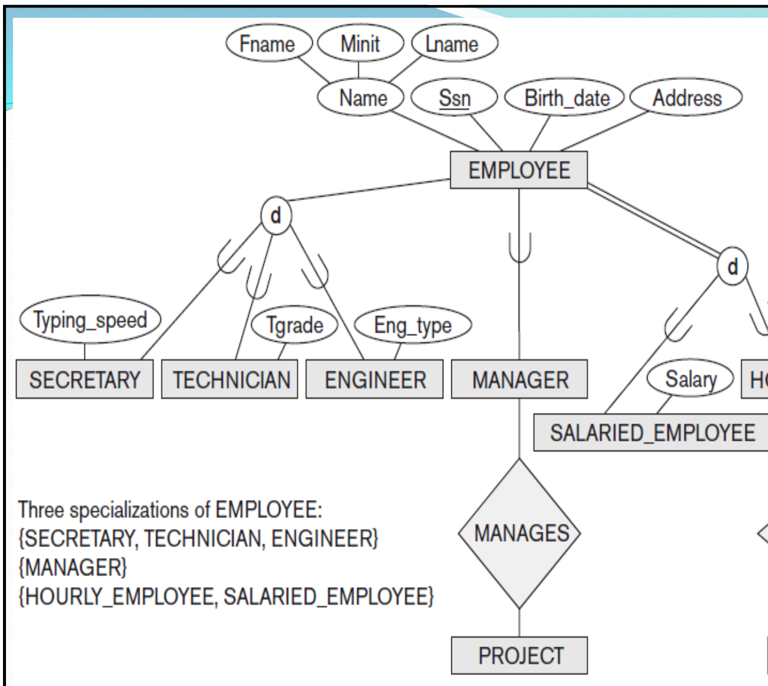



**Figure 4.1**
EER diagram notation to represent subclasses and specialization.

## Subclasses and Superclasses (2)

- Each of these subgroupings is a subset of EMPLOYEE entities
- Each is called a subclass of EMPLOYEE
- EMPLOYEE is the superclass for each of these subclasses
- These are called superclass/subclass relationships:
  - EMPLOYEE/SECRETARY
  - EMPLOYEE/TECHNICIAN
  - EMPLOYEE/MANAGER
  - …



## Subclasses and Superclasses (3)

- These are also called IS-A relationships
  - SECRETARY IS-A EMPLOYEE, TECHNICIAN IS-A EMPLOYEE, ….
- Note: An entity that is member of a subclass represents **the same** real-world entity as some member of the superclass:
  - The subclass member is the same entity in a *distinct specific role*
  - An entity cannot exist in the database merely by being a member of a subclass; it must also be a member of the superclass
  - A member of the superclass can be optionally included as a member of any number of its subclasses

## Subclasses and Superclasses (4)

- Examples:
  - A salaried employee who is also an engineer belongs to the two subclasses:
    - ENGINEER, and
    - SALARIED_EMPLOYEE
  - A salaried employee who is also an engineering manager belongs to the three subclasses:
    - MANAGER,
    - ENGINEER, and
    - SALARIED_EMPLOYEE
- It is not necessary that every entity in a superclass be a member of some subclass



## Specialization (1)

- 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***.
    - May have several specializations of the same superclass

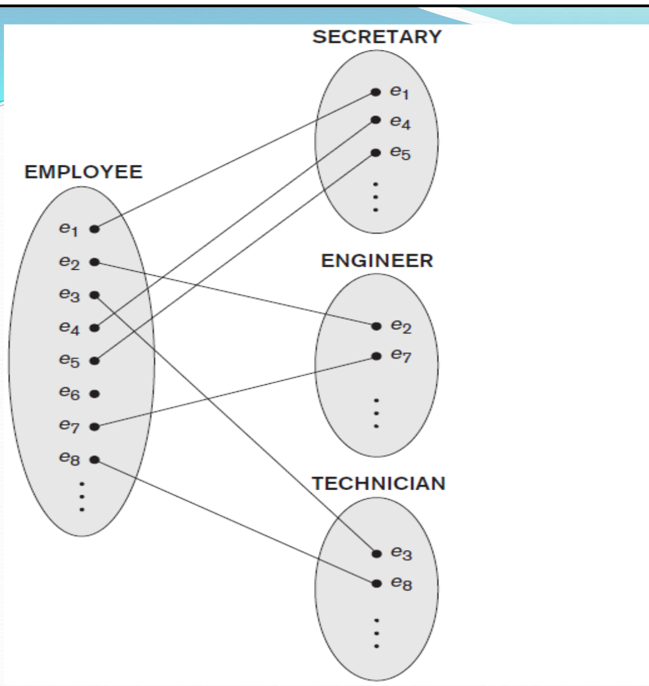



**Figure 4.2**
Instances of a specialization.



# Representing Specialization in EER Diagrams

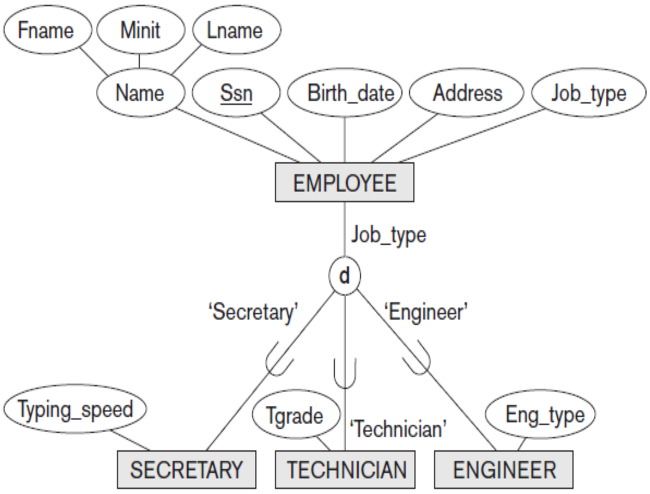



**Figure 4.4**
EER diagram notation for an attribute-defined specialization on Job_type.

## Specialization (2)

- Example: Another specialization of EMPLOYEE based on *method of pay* is {SALARIED_EMPLOYEE, HOURLY_EMPLOYEE}.
  - Superclass/subclass relationships and specialization can be diagrammatically represented in EER diagrams
  - Attributes of a subclass are called specific or local attributes.
    - For example, the attribute TypingSpeed of SECRETARY
  - The subclass can also participate in specific relationship types.
    - For example, a relationship BELONGS_TO of HOURLY_EMPLOYEE



## Attribute Inheritance

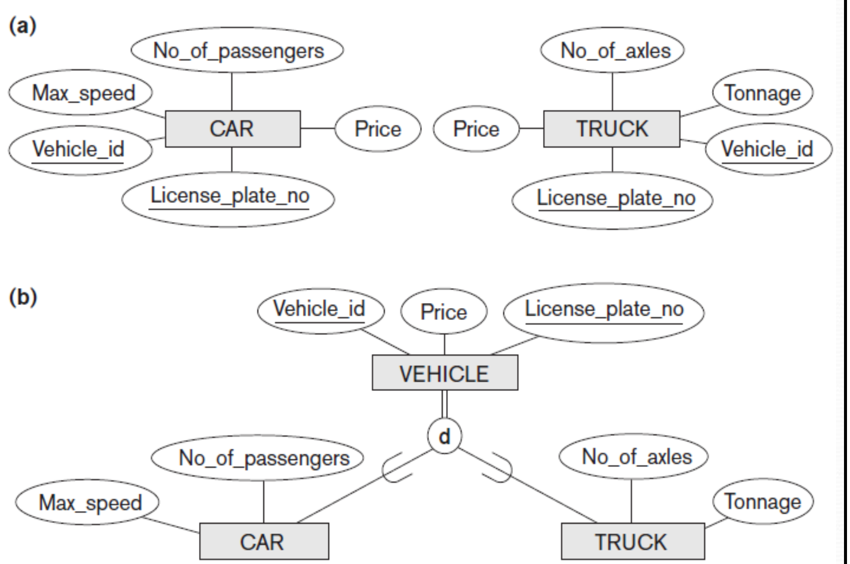
- A subclass entity inherits
  - All attributes of the superclass
  - All relationships of the superclass
- Example:
  - In the previous slide, SECRETARY (as well as TECHNICIAN and ENGINEER) inherit the attributes Name, SSN, ..., from EMPLOYEE
  - Every SECRETARY entity will have values for the inherited attributes

# Generalization

- Generalization
  - Process of defining a generalized entity type from the given entity types
  - The reverse of specialization
- Several classes with common features are generalized into a superclass;
  - original classes become its subclasses
- Example: CAR, TRUCK generalized into VEHICLE;
  - both CAR, TRUCK become subclasses of VEHICLE.
  - We can view {CAR, TRUCK} as a specialization of VEHICLE
  - Alternatively, we can view VEHICLE as a generalization of CAR and TRUCK



**Figure 4.3**
Generalization. (a) Two entity types, CAR and TRUCK.
(b) Generalizing CAR and TRUCK into the superclass VEHICLE.



## Generalization and Specialization (1)

- Diagrammatic notation are sometimes used to distinguish between generalization and specialization
  - Arrow pointing to the generalized superclass represents a generalization
  - Arrows pointing to the specialized subclasses represent a specialization
  - We *do not use* this notation because it is often subjective as to which process is more appropriate for a particular situation
  - We advocate not drawing any arrows



## Generalization and Specialization (2)

- Data Modeling with Specialization and Generalization
  - A superclass or subclass represents a collection (or set or grouping) of entities
  - It also represents a particular *type of entity*
  - Shown in rectangles in EER diagrams (as are entity types)
  - We can call all entity types (and their corresponding collections) classes, whether they are entity types, superclasses, or subclasses

## Constraints on Specialization and Generalization

- If we can determine exactly those entities that will become members of each subclass by a condition, the subclasses are called predicate-defined (or condition-defined) subclasses
  - Condition is a constraint that determines subclass members
  - Display a predicate-defined subclass by writing the predicate condition next to the line attaching the subclass to its superclass



## Attribute-Defined Constraints

- If all subclasses have membership condition on same attribute of the superclass, the specialization is called an attribute-defined specialization
  - Attribute is called the defining attribute of the specialization
  - Example: JobType is the defining attribute of the specialization {SECRETARY, TECHNICIAN, ENGINEER} of EMPLOYEE

# Attribute-defined Specialization

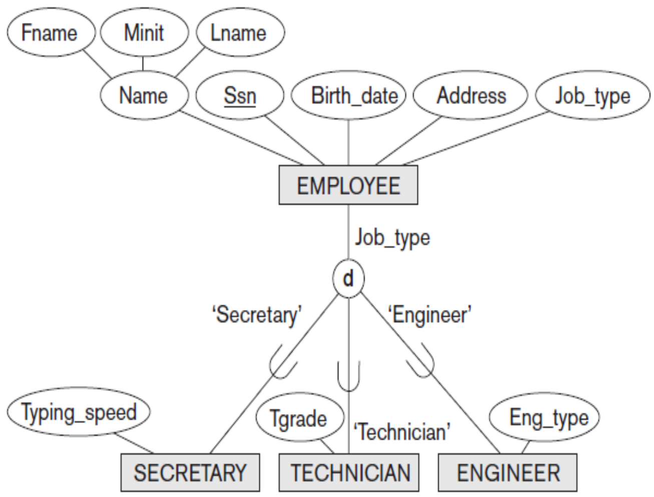



**Figure 4.4**
EER diagram notation for an attribute-defined specialization on Job_type.



# User-Defined Constraints

- If no condition determines membership, the subclass is called user-defined
  - Membership in a subclass is determined by the database users by applying an operation to add an entity to the subclass
  - Membership in the subclass is specified individually for each entity in the superclass by the user

## Constraints on Specialization and Generalization

- Two basic constraints can apply to a specialization/generalization:
  - Disjointness Constraint:
  - Completeness Constraint:



## Disjointness Constraint

- Specifies that the subclasses of the specialization must be disjoint:
  - an entity can be a member of at most one of the subclasses of the specialization
- Specified by ***d*** in EER diagram
- If not disjoint, specialization is overlapping:
  - that is the same entity may be a member of more than one subclass of the specialization
- Specified by ***o*** in EER diagram

## Completeness Constraint

- Total specifies that every entity in the superclass must be a member of some subclass in the specialization/generalization
- Shown in EER diagrams by a double line
- Partial allows an entity not to belong to any of the subclasses
- Shown in EER diagrams by a single line



## Constraints Combination

- Hence, we have four types of specialization/generalization:
  - Disjoint, total
  - Disjoint, partial
  - Overlapping, total
  - Overlapping, partial
- Note: Generalization usually is total because the superclass is derived from the subclasses.

# Example of Disjoint Partial Specialization

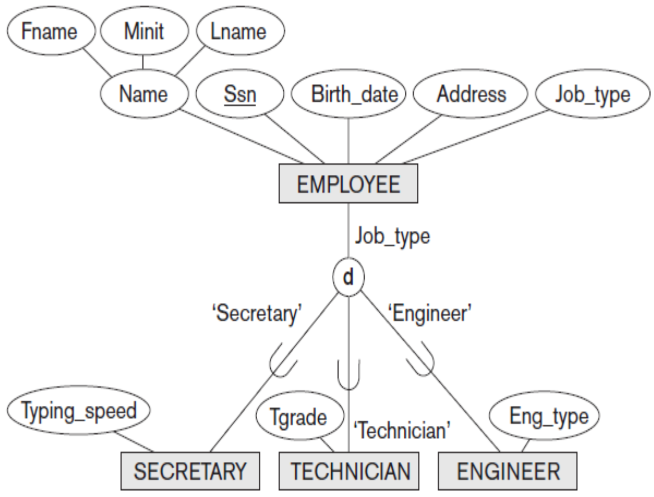



**Figure 4.4**
EER diagram notation for an attribute-defined specialization on Job_type.



# Example of Overlapping Total Specialization

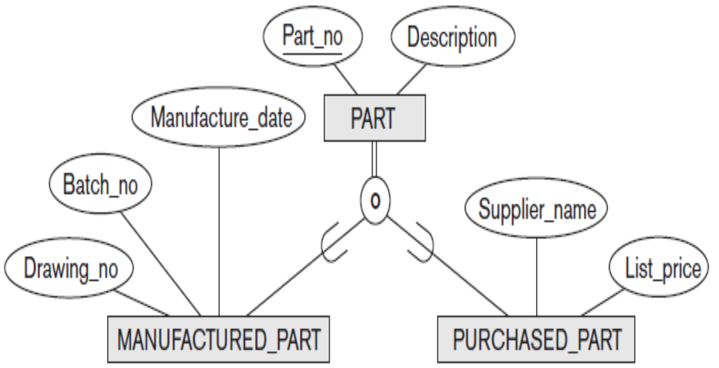



**Figure 4.5**
EER diagram notation for an overlapping (nondisjoint) specialization.

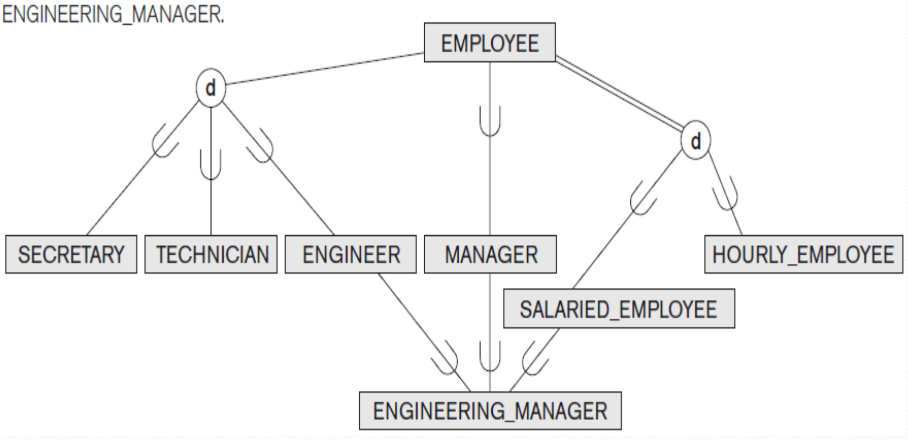
# Hierarchies, Lattices & Shared Subclasses (1)

- 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)



# Example

**Figure 4.6**
A specialization lattice with shared subclass ENGINEERING_MANAGER.



## 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 predecessor 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)



## 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

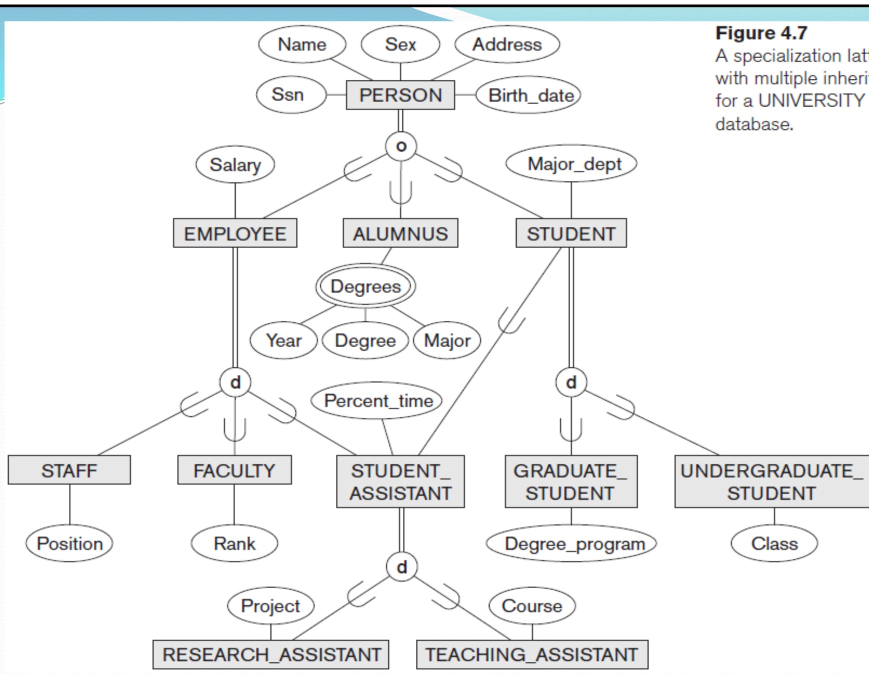



**Figure 4.7**
A specialization lattice with multiple inheritance for a UNIVERSITY database.



# Categories (UNION TYPES) (1)

- All of the *superclass/subclass relationships* we have seen thus far have a single superclass
- A shared subclass is a subclass in:
  - *more than one* distinct superclass/subclass relationships
  - each relationships has a single superclass
  - shared subclass leads to multiple inheritance
- In some cases, we need to model a *single superclass/subclass relationship* with *more than one* superclass
- Superclasses can represent different entity types
- Such a subclass is called a category or UNION TYPE

# Categories (UNION TYPES) (2)

- Example: In a database for vehicle registration, a vehicle owner can be a PERSON, a BANK (holding a lien on a vehicle) or a COMPANY.
  - A *category* (UNION type) called OWNER is created to represent a subset of the ***UNION*** of the three superclasses COMPANY, BANK, and PERSON
  - A category member must exist in ***at least one*** of its superclasses
- Difference from *shared subclass*, which is a:
  - subset of the ***intersection*** of its superclasses
  - shared subclass member must exist in ***all*** of its superclasses



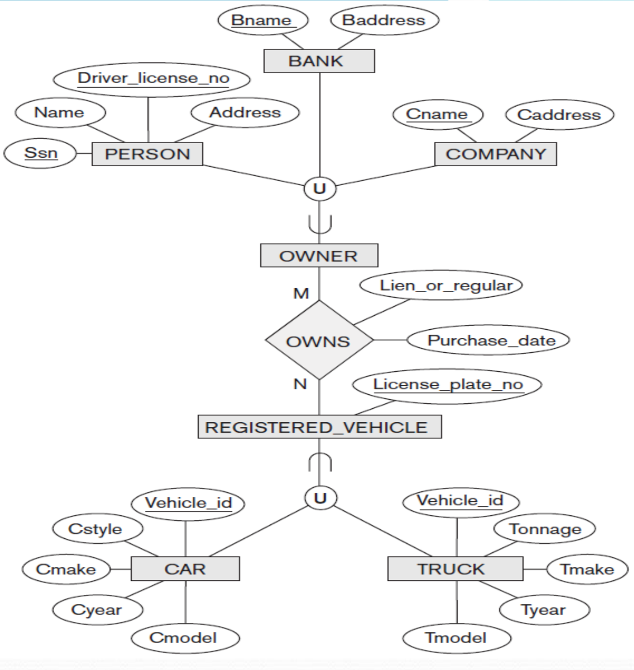



**Figure 4.8**
Two categories (union types): OWNER and REGISTERED_VEHICLE.

## Formal Definitions of EER (1)

- Class C: A type of entity with a corresponding set of entities:
  - could be entity type, subclass, superclass, or category
- Note: The definition of *relationship type* in ER/EER should have 'entity type' replaced with 'class‘ to allow relationships among classes in general



## Formal Definitions of EER (2)

- Subclass S is a class whose:
  - Type inherits all the attributes and relationship of class C
  - Set of entities must always be a subset of the set of entities of the other class C
    - $S \subset C$
  - C is called the superclass of S
  - A superclass/subclass relationship exists between S and C

## Formal Definitions of EER (2)

- Specialization Z: Z = {S1, S2,..., Sn} is a set of subclasses with same superclass G; G/Si is a superclass relationship for i = 1, ...., n.
  - G is called a generalization of the subclasses {S1, S2,..., Sn}
  - Z is total if we always have:
    - S1 ∪ S2 ∪ ... ∪ Sn = G;
    - Otherwise, Z is partial.
  - Z is disjoint if we always have:
    - Si ∩ Sj = empty-set for i ≠ j;
    - Otherwise, Z is overlapping.



## Formal Definitions of EER (3)

- Subclass S of C is predicate defined if predicate (condition) p on attributes of C is used to specify membership in S;
  - that is, S = C[p], where C[p] is the set of entities in C that satisfy condition p
- A subclass not defined by a predicate is called user-defined
- Attribute-defined specialization: if a predicate A = ci (where A is an attribute of G and ci is a constant value from the domain of A) is used to specify membership in each subclass Si in Z
  - Note: If ci ≠ cj for i ≠ j, and A is single-valued, then the attribute-defined specialization will be disjoint.

# Formal Definitions of EER (4)

- Category or UNION type T
  - A class that is a subset of the ***union*** of n defining superclasses D1, D2,…Dn, n>1:
    - $T \subset (D1 \cup D2 \cup \ldots \cup Dn)$
  - Can have a predicate pi on the attributes of Di to specify entities of Di that are members of T.
  - If a predicate is specified on every Di: $T = (D1[p1] \cup D2[p2] \cup \ldots \cup Dn[pn])$



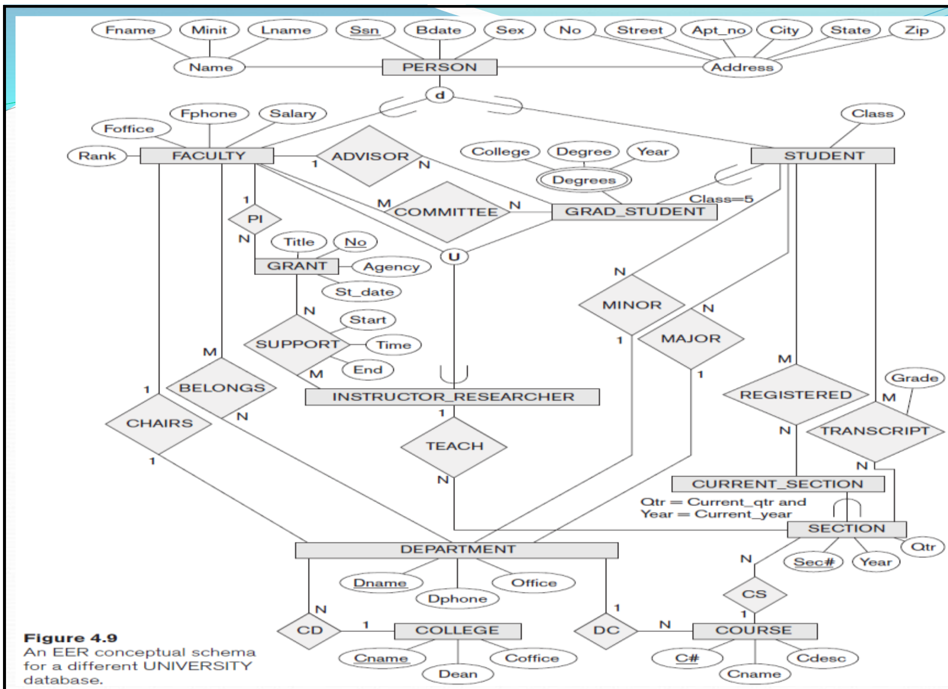



**Figure 4.9**
An EER conceptual schema for a different UNIVERSITY database.

## Alternative Diagrammatic Notations

- ER/EER diagrams are a specific notation for displaying the concepts of the model diagrammatically
- DB design tools use many alternative notations for the same or similar concepts
- One popular alternative notation uses UML class diagrams
- See next slides for UML class diagrams and other alternative notations



## Specialization/Generalization (UML)

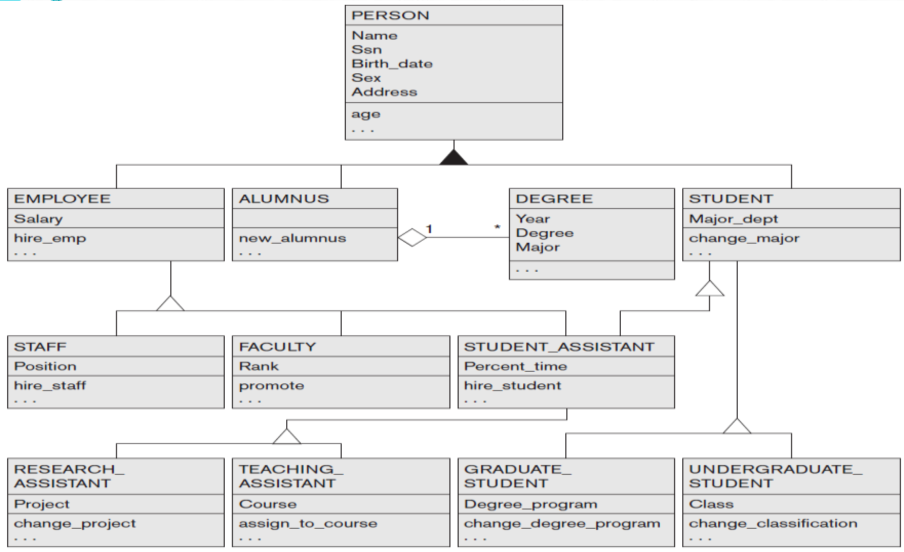
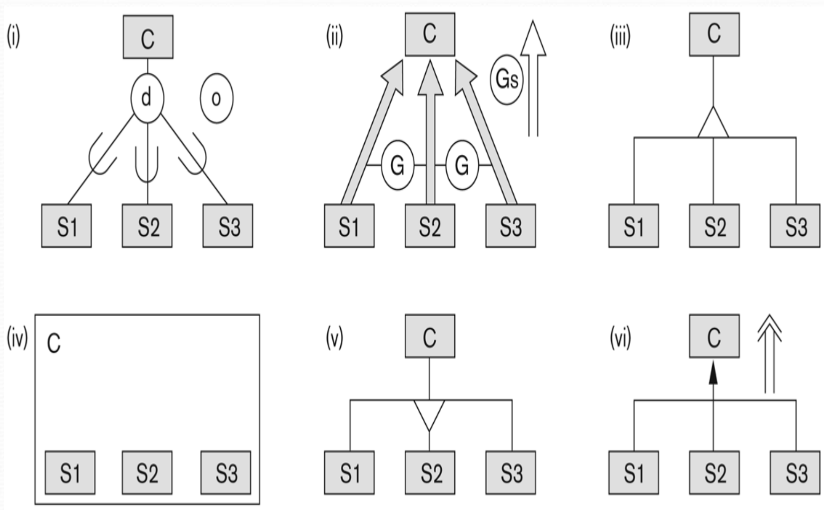



**Figure 4.10**
A UML class diagram corresponding to the EER diagram in Figure 4.7, illustrating UML notation for specialization/generalization.

## Specialization/Generalization






## General Conceptual Modeling Concepts

- General data abstractions
  - Classification and Instantiation
  - Aggregation and Association (relationships)
  - Generalization and Specialization
  - Identification
- Constraints
  - Cardinality (Min and Max)
  - Coverage (Total vs. Partial, and Exclusive (disjoint) vs. Overlapping)

## Ontologies

- Use conceptual modeling and other tools to develop "a specification of a conceptualization"
  - **Specification** refers to the language and vocabulary (data model concepts) used
  - **Conceptualization** refers to the description (schema) of the concepts of a particular field of knowledge and the relationships among these concepts
- Many medical, scientific, and engineering ontologies are being developed as a means of standardizing concepts and terminology



## Design Issues (1)

- Use of entity sets vs. attributes
  - Choice mainly depends on the structure of the enterprise being modeled, and on the semantics associated with the attribute in question
  - *E.g., should Phone be an attribute of Employee or a separate entity?*
- Use of entity sets vs. relationship sets
  - Possible guideline is to designate a relationship set to describe an *action that occurs between entities*

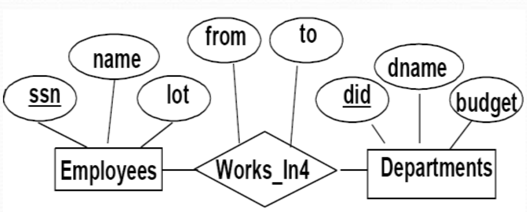
# Design Issues (2)

- Binary versus *n-ary relationship sets*
  - Although it is possible to replace any nonbinary (*n-ary, for n > 2)* relationship set by a number of distinct binary relationship sets, a *n-ary relationship set shows more clearly that several entities* participate in a single relationship
- Placement of relationship attributes
- The use of a strong or weak entity set
- The use of specialization contributes to modularity in the design

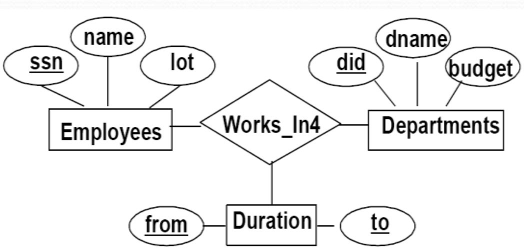


# Entity vs. Attribute

- Works_In4 does not allow an employee to work in a department for two or more periods.

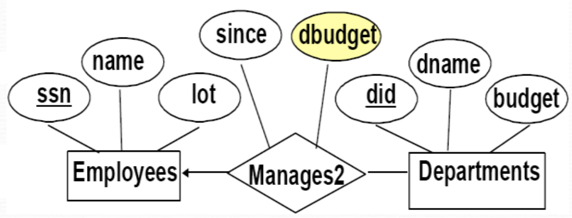



- What if we want to record all possible periods an employee worked in a particular department?

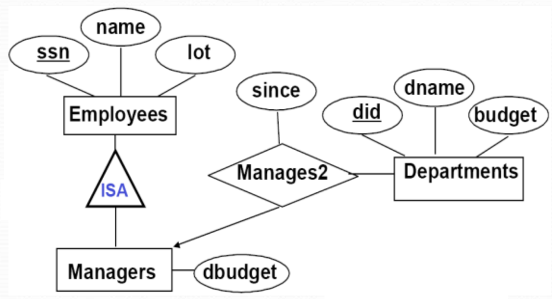


# Entity vs. Relationship

- Manager gets a separate budget for each dept.




- What if a manager gets a budget that covers *all managed depts?*






# Design Principles

- What makes a design good or bad?
- Design should be faithful to specifications
- Avoid redundancy – more on normalization later!
- Keep it simple
  - Avoid creating unnecessary entities/relationships
- Pick the right kind of element (see examples “Entity vs. Relationship” and “Entity vs. Attribute”)
  - Rule of thumb: if *thing has more info than just its name* make it an entity

## Summary

- Introduced the EER model concepts
  - Class/subclass relationships
  - Specialization and generalization
  - Inheritance
- These augment the basic ER model concepts introduced in Chapter 3
- EER diagrams and alternative notations were presented



## Assignment 4

- Textbook exercise 3-17, 18, 19, 23
- Textbook exercise 4-17, 20

- Due Date: May 14, 2019