## Fundamentals of DATABASE SYSTEMS FOURTH EDITION

ELMASRI SON NAVATHE

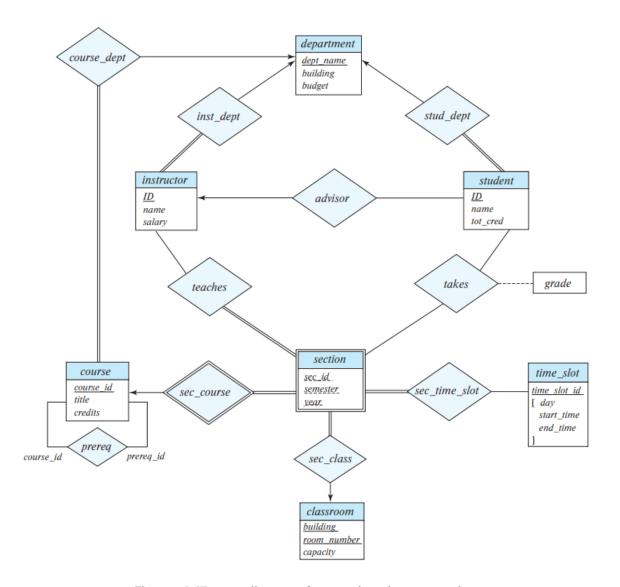


Figure 6.15 E-R diagram for a university enterprise.

### Chapter 4 - Part I

## Enhanced Entity-Relationship and UML Modeling



# Enhanced-ER (EER) Model Concepts

- Includes all modeling concepts of basic ER
- Additional concepts: subclasses/superclasses, specialization/generalization, categories, attribute inheritance
- The resulting model is called the enhanced-ER or Extended ER (E2R or EER) model
- It is used to model applications more completely and accurately if needed
- It includes some object-oriented concepts, such as inheritance

### Subclasses and Superclasses (1)

- An entity type may have additional meaningful subgroupings of its entities
- Example: EMPLOYEE may be further grouped into SECRETARY, ENGINEER, MANAGER, TECHNICIAN, SALARIED\_EMPLOYEE, HOURLY\_EMPLOYEE,...
  - Each of these groupings 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.
- Example: EMPLOYEE/SECRETARY, EMPLOYEE/TECHNICIAN

### Subclasses and Superclasses (2)

- 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
- Example: A salaried employee who is also an engineer belongs to the two subclasses ENGINEER and SALARIED\_EMPLOYEE
  - It is not necessary that every entity in a superclass be a member of some subclass

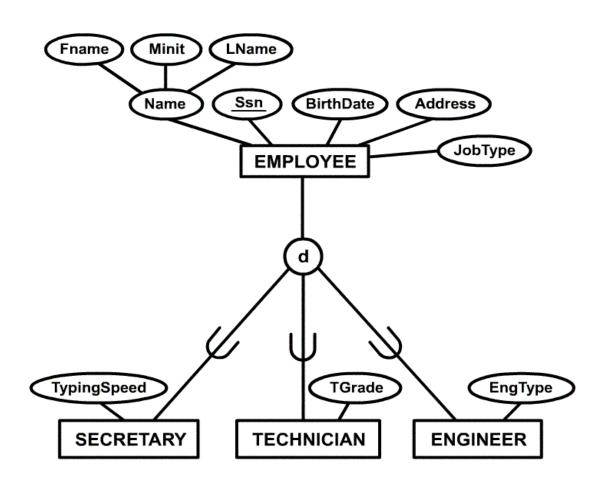
## Attribute Inheritance in Superclass / Subclass Relationships

- An entity that is member of a subclass *inherits* all attributes of the entity as a member of the superclass
- It also inherits all relationships

### 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
- Example: Another specialization of EMPLOYEE based in 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 attributes. For example, TypingSpeed of SECRETARY
  - The subclass can participate in specific relationship types.
     For example, BELONGS\_TO of HOURLY\_EMPLOYEE

### Example of a Specialization



#### Generalization

- 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, 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

### Generalization and Specialization

- Diagrammatic notation 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 in these situations
- Data Modeling with Specialization and Generalization
  - A superclass or subclass represents a set of entities
  - Shown in rectangles in EER diagrams (as are entity types)
  - Sometimes, all entity sets are simply called classes, whether they are entity types, superclasses, or subclasses

#### Constraints on Specialization and Generalization (1)

- 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
- If all subclasses in a specialization have membership condition on same attribute of the superclass, 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
- 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 (2)

- Two other conditions apply to a specialization/generalization:
- Disjointness Constraint:
  - Specifies that the subclasses of the specialization must be disjointed (an entity can be a member of at most one of the subclasses of the specialization)
  - Specified by d in EER diagram
  - If not disjointed, overlap; that is the same entity may be a member of more than one subclass of the specialization
  - Specified by o in EER diagram

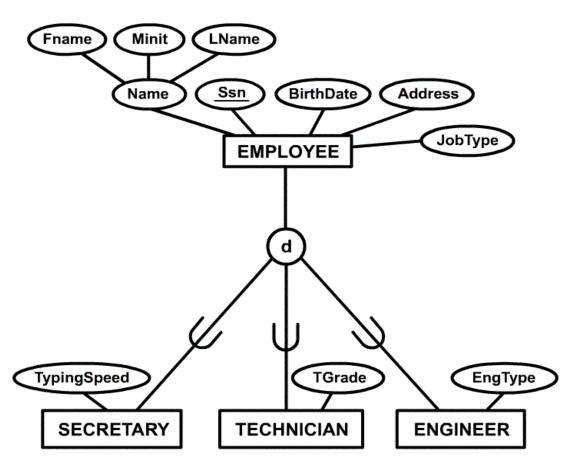
#### Completeness Constraint:

- 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 on Specialization and Generalization (3)

- 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.

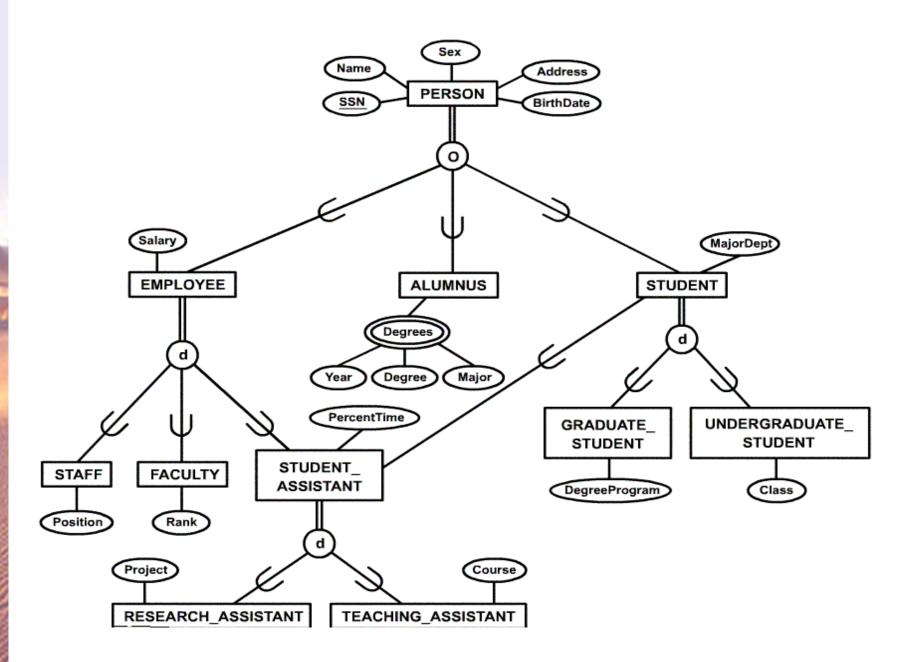
# Example of disjoint partial Specialization



### Specialization / Generalization Hierarchies,

### Lattices and Shared Subclasses A subclass may itself have further subclasses specified on it

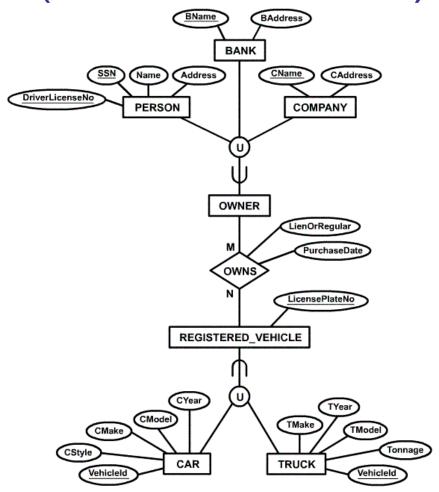
- Forms a hierarchy or a lattice
- Hierarchy has a constraint that every subclass has only one superclass (called *single inheritance*)
- In a lattice, a subclass can be subclass of more than one superclass (called *multiple inheritance)*
- In a lattice or hierarchy, a subclass inherits attributes not only of its direct superclass, but also of all its predecessor superclasses
- A subclass with more than one superclass is called a shared subclass
- Can have specialization hierarchies or lattices, or generalization hierarchies or lattices
- In specialization, start with an entity type and then define subclasses of the entity type by successive specialization (top down conceptual refinement process)
- In generalization, start with many entity types and generalize those that have common properties (bottom up conceptual synthesis process)
- In practice, the combination of two processes is employed



### Categories (UNION TYPES)

- All of the superclass/subclass relationships we have seen thus far have a single superclass
- A shared subclass is subclass in more than one distinct superclass/subclass relationships, where each relationships has a single superclass (multiple inheritance)
- In some cases, need to model a single superclass/subclass relationship with more than one superclass
- Superclasses represent different entity types
- Such a subclass is called a category or UNION TYPE
- Example: Database for vehicle registration, vehicle owner can be a person, a bank (holding a lien on a vehicle) or a company.
  - Category (subclass) OWNER is 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
- Note: The difference from shared subclass, which is subset of the intersection of its superclasses (shared subclass member must exist in all of its superclasses).

# Example of categories (UNION TYPES)



### Formal Definitions of EER Model (1)

- Class C: A set of entities; could be entity type, subclass, superclass, category.
- Subclass S: A class whose entities must always be subset of the entities in another class, called the superclass C of the superclass/subclass (or IS-A) relationship S/C:

 $S \subseteq C$ 

- Specialization Z: Z = {S1, S2,..., Sn} a set of subclasses with same superclass G; hence, G/Si 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  $\cup$  S2  $\cup$  ...  $\cup$  Sn = G; Otherwise, Z is partial.

– Z is disjoint if we always have:

Si  $\cap$  S2 empty-set for  $i \neq j$ ;

Otherwise, Z is overlapping.

#### Formal Definitions of EER Model (2)

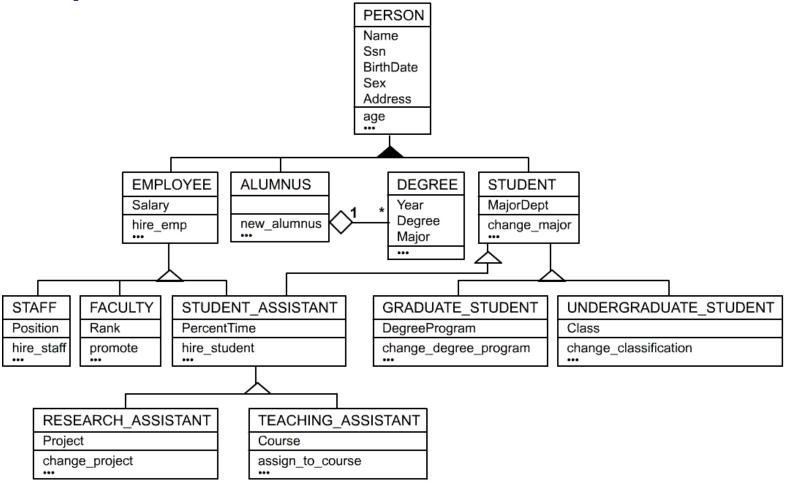
- Subclass S of C is predicate defined if predicate p on attributes of C is used to specify membership in S; that is, S = C[p], where C[p] is the set of entities in C that satisfy p
- A subclass not defined by a predicate is called user-defined
- Attribute-defined specialization: if a predicate A = ci (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  $ci \neq cj$  for  $i \neq j$ , and A is single-valued, then the attribute-defined specialization will be disjoint.
- Category or UNION type T
  - A class that is a subset of the union of n defining superclasses D1, D2,...Dn, n>1:

 $T \subseteq (D1 \cup D2 \cup ... \cup Dn)$ 

A predicate pi on the attributes of T.

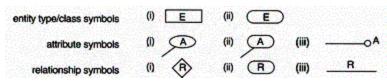
- If a predicate pi on the attributes of Di can specify entities of Di that are members of T.
- If a predicate is specified on every Di: T = (D1[p1] ∪ D2[p2] ∪...∪
   Dn[pn]
- Note: The definition of relationship type should have 'entity type' replaced with 'class'.

## UML Example for Displaying Specialization / Generalization

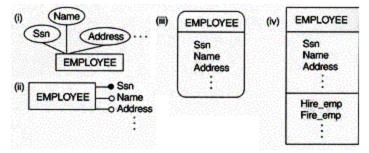


## Alternative Diagrammatic Notations

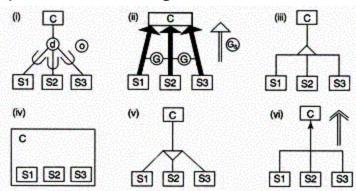
Symbols for entity type / class, attribute and relationship



Displaying attributes



Notations for displaying specialization / generalization



Various (min, max) notations

Displaying cardinality ratios

