Enhanced Entity-Relationship and UML Modeling

Handout 3

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 sub-groupings 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 that is also a member of super class, but playing specific role
  - An entity cannot exist in the database only as a member of a subclass
  - A member of the superclass can be optionally included as a member of any number of its subclasses

Attribute Inheritance in Superclass / Subclass Relationships

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
- 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 are derived based upon some distinguishing characteristics of the entities in the superclass

  Example 1: {SECRETARY, ENGINEER, TECHNICIAN} is a specialization of EMPLOYEE based upon *job type*.

- May have several specializations of the same superclass

Specialization

Example 2: 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

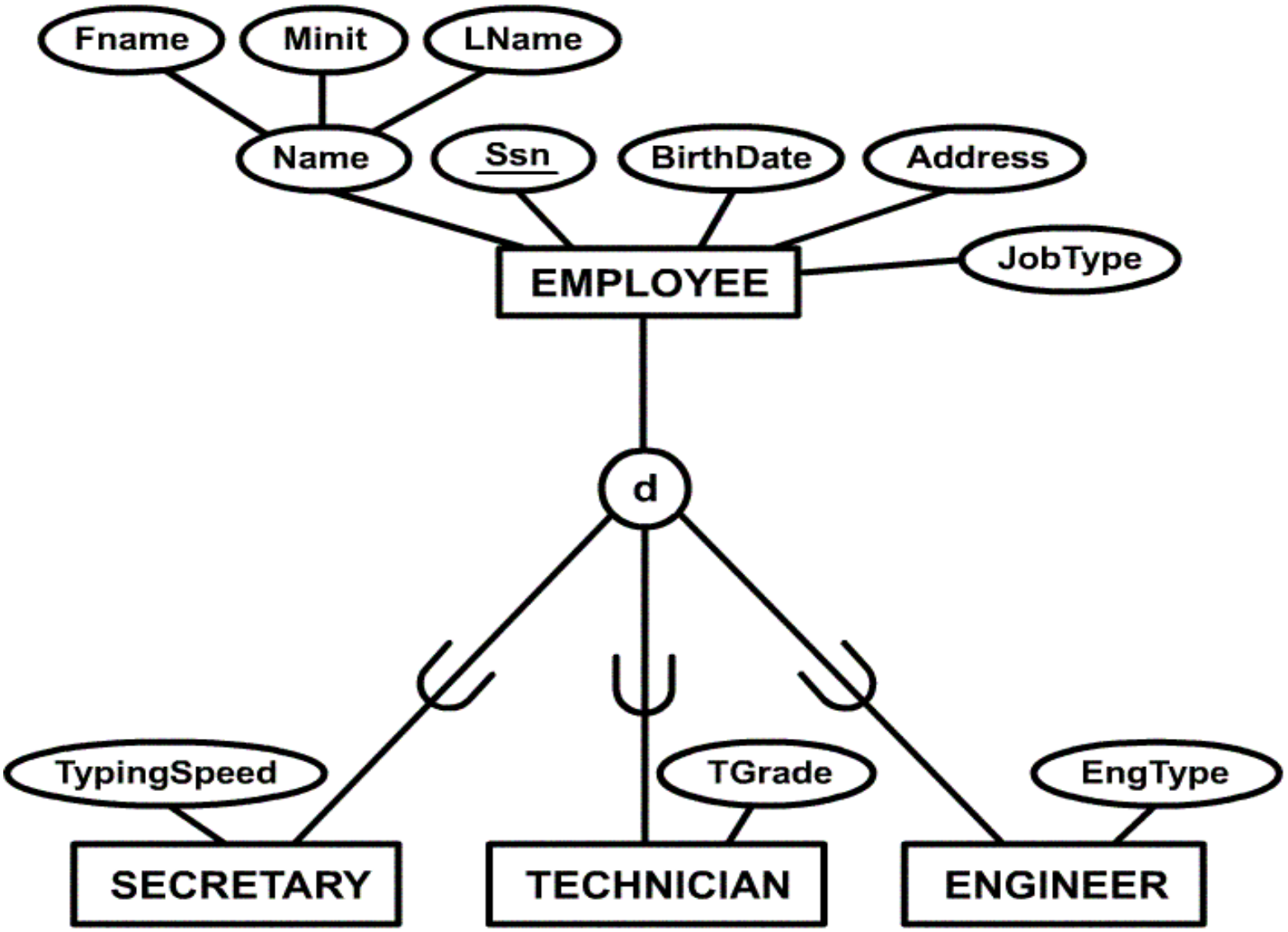
Properties:

- Attributes of a subclass are called specific attributes. For example, TypingSpeed of SECRETARY

- The subclass can participate in specific relationship types.
  SALARIED_EMPLOYEE **enjoys** BENEFITS

  (Only salaried employees enjoys benefits)

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

Specialization, Generalization and Inheritance

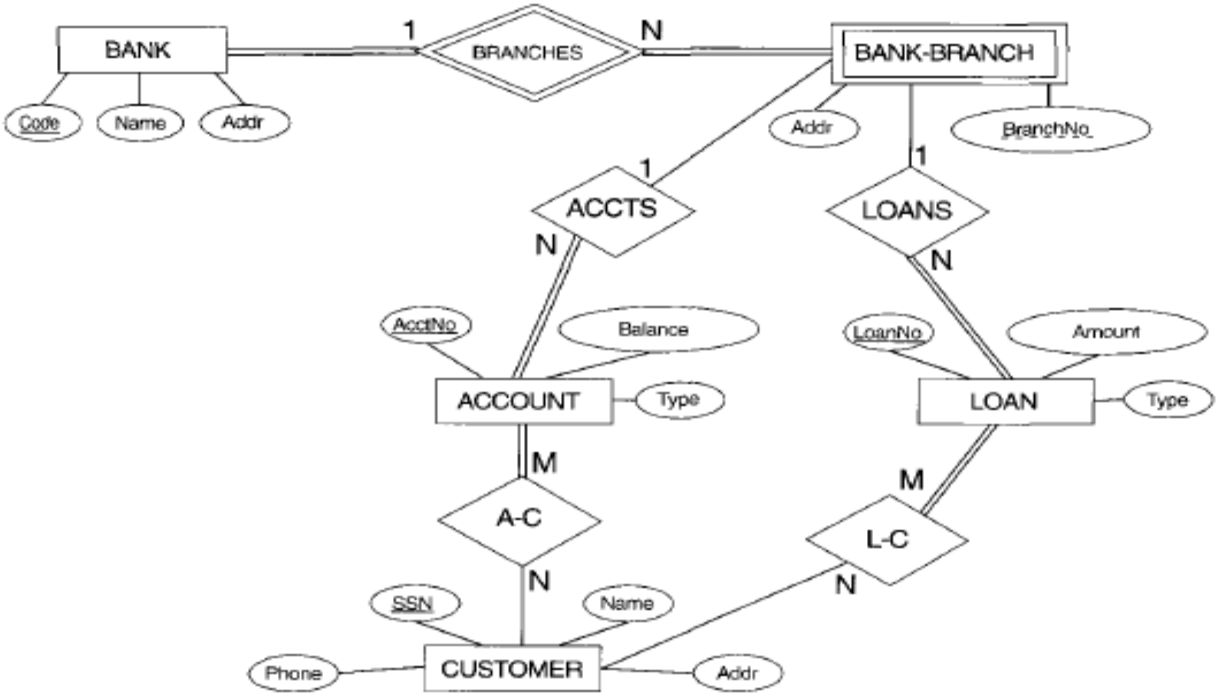
- Specialization: identifying subclasses, and their distinguishing characteristics (attributes & relationships) (Top-down design)
- Generalization: aggregate entities to a superclass entity type by identifying their common characteristics (Bottom-up design)
- Inheritance: IS_A (instance) relationship that supports attribute inheritance and relationship participation
  - Single inheritance results in a hierarchy
  - Multiple inheritance results in a lattice

    E.g., EMPLOYEE → STUDENT-ASSISTANT ←STUDENT

Generalization and Specialization

- 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

Exercise: Generalization and Specialization



Consider the BANK ER schema above, and suppose that it is necessary to keep track of different types of ACCOUNTS (SAVINGS_ACCTS, CHECKING_ACCTS, •.•) and LOANS (CAR_LOANS, HOME_LOANS, •••). Also, it is desirable to keep track of each account's TRANSACTIONS (deposits, withdrawals, checks, ...) and each loan's PAYMENTS; both of these include the amount, date, and time. Modify the BANK schema, using ER and EER concepts of specialization and generalization. State any assumptions you make about the additional requirements.

Constraints on Specialization and Generalization

- Conditions apply to a specialization/generalization:
- **Disjointness / Overlap Constraint**:
  - ***Disjoint*** (An entity can be a member of at most one of the subclasses of the specialization. Shown by a ***d*** in EER diagram)
  - ***Overlap*** (Same entity may be a member of more than one subclass of the specialization , Specified by ***o*** in EER diagram)

Constraints on Specialization and Generalization

- **Completeness Constraint (Total/Partial)**:
  - ***Total*** specifies that every entity in the superclass must be a member of some subclass in the specialization/ generalization. (Symbol: double line)
  - ***Partial*** allows an entity not to belong to any of the subclasses
    (Symbol: single line)

Inclusion Constraints on Specialization and Generalization

- The *disjoint* constraint: the subclasses of a superclass are disjoint.
  - This means that an entity can be a member of only one subclass.
  - The entities for each class can be *user-defined* or specified with a *predicate-defined subclass.*
  - In a predicate-defined subclass, we use a selection condition on one or more attributes to define the entities of the subclass. E.g., MembershipStatus
- The *non-disjoint* constraints: specify that the subclasses are overlapping and an entity may be a member of more than one subclass.

Completeness Constraints on Specialization and Generalization

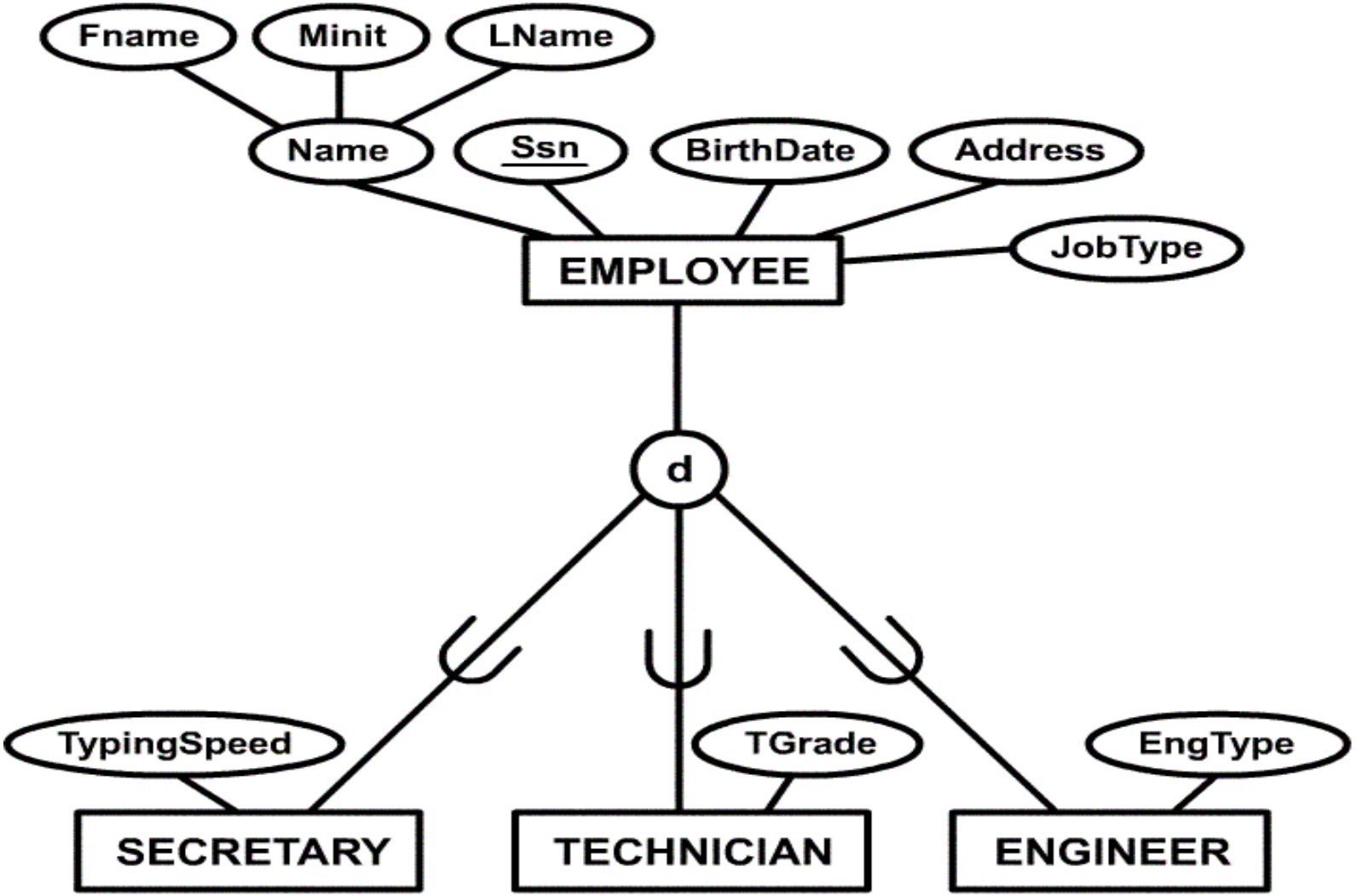
- A *total* specialization: specifies that every entity in the superclass must be a member of some of its subclasses
  - E.g., a librarian must belong to one of the subclasses of LIBRARIAN.
- A *partial* specialization: specifies that an entity may not belong to any subclass
  - E.g. , an honorary member may not belong to any of the specializations (subclasses) of MEMBER.
- Superclass via generalization is always total

Constraints on Specialization and Generalization

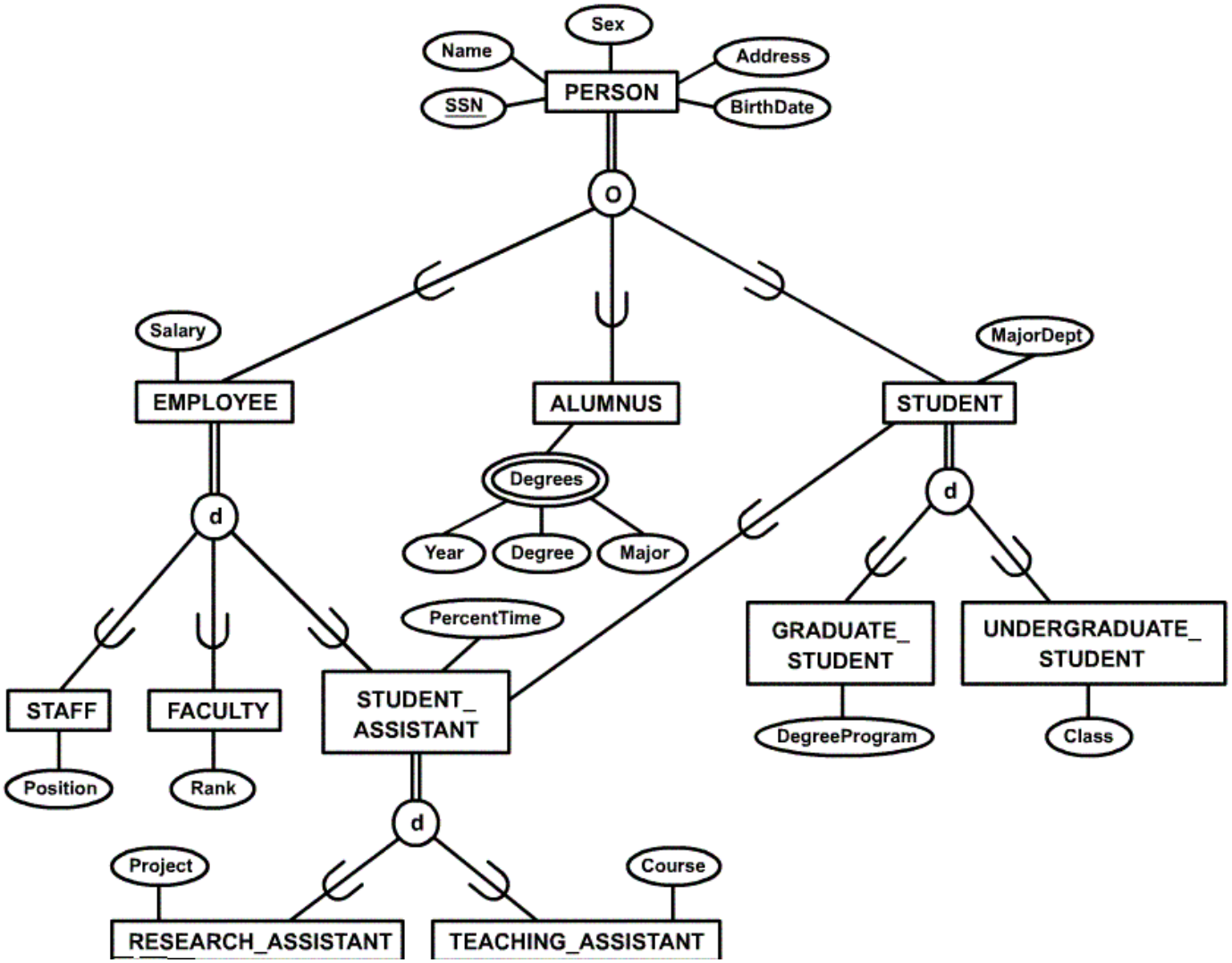
- Hence, we have four types of specialization/generalization:
  - Disjoint, total
  - Disjoint, partial
  - Overlapping, total
  - Overlapping, partial

- Note: Generalization usually is total because the superclass is derived from the subclasses.

Example of disjoint partial Specialization



Specialization / Generalization Lattice Example (UNIVERSITY)



Categories (UNION TYPES)

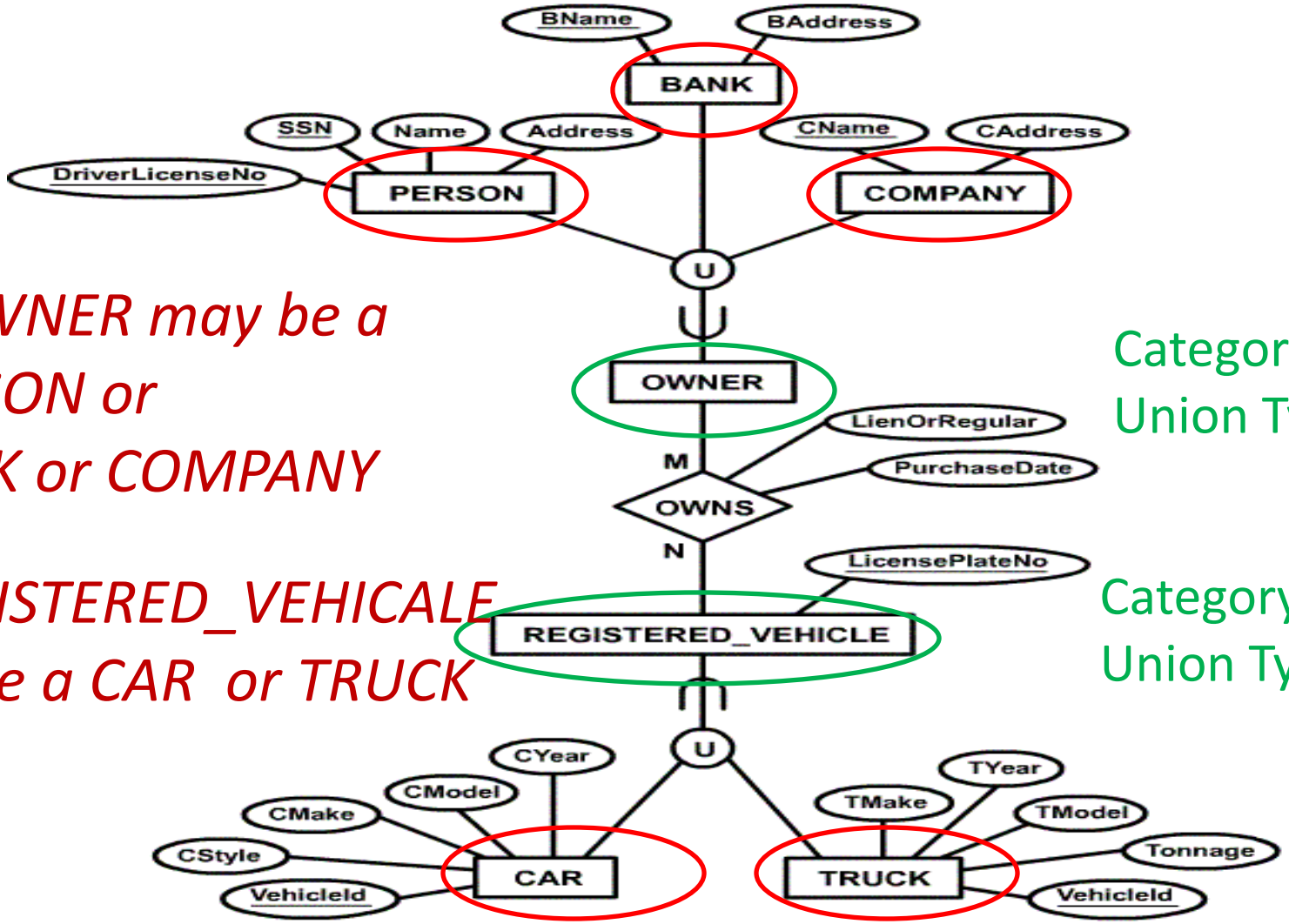
- A shared subclass is a subclass having more than one distinct superclass/subclass relationships (multiple inheritance)
- Superclasses represent different entity types
- Such a subclass is called a category or UNION TYPE

Union Types or Categories

- Category is a subclass of a union of two or more super classes representing *different entity types* playing a common role.

- A category is diagrammatically shown by a " U " in the circle that attaches the category to the super classes

Example of categories (UNION TYPES)



A OWNER may be a PERSON or BANK or COMPANY

Category/ Union Type

A REGISTERED_VEHICALE may be a CAR or TRUCK

Category/ Union Type

Union Types or Categories

- Collection of entities of distinct entity types
  - E.g., Vehicle owner: person, bank, company
- Multiple Inheritance with superclasses of different types
- Category OWNER is a subclass of the set **union** of the entity types: PERSON, BANK, COMPANY
- An instance in category must exists only in one of the superclasses
- Category can be:
  - total
  - partial (with predicate definition)

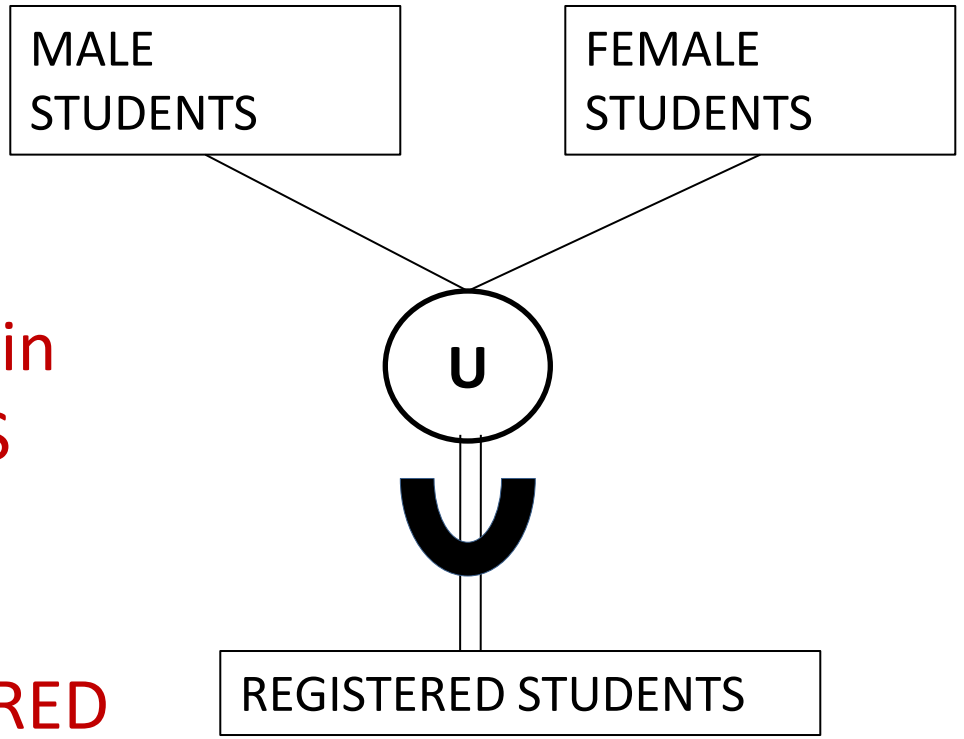
Categories (UNION TYPES)

- ***Categories*** can also have ***total*** (or full) participation or ***partial*** participation.

- ***Total*** participation means that the category holds the union of *all entities in its superclasses*

- ***Partial*** participation means that a category holds only a subset of the union of all entities in its superclasses.

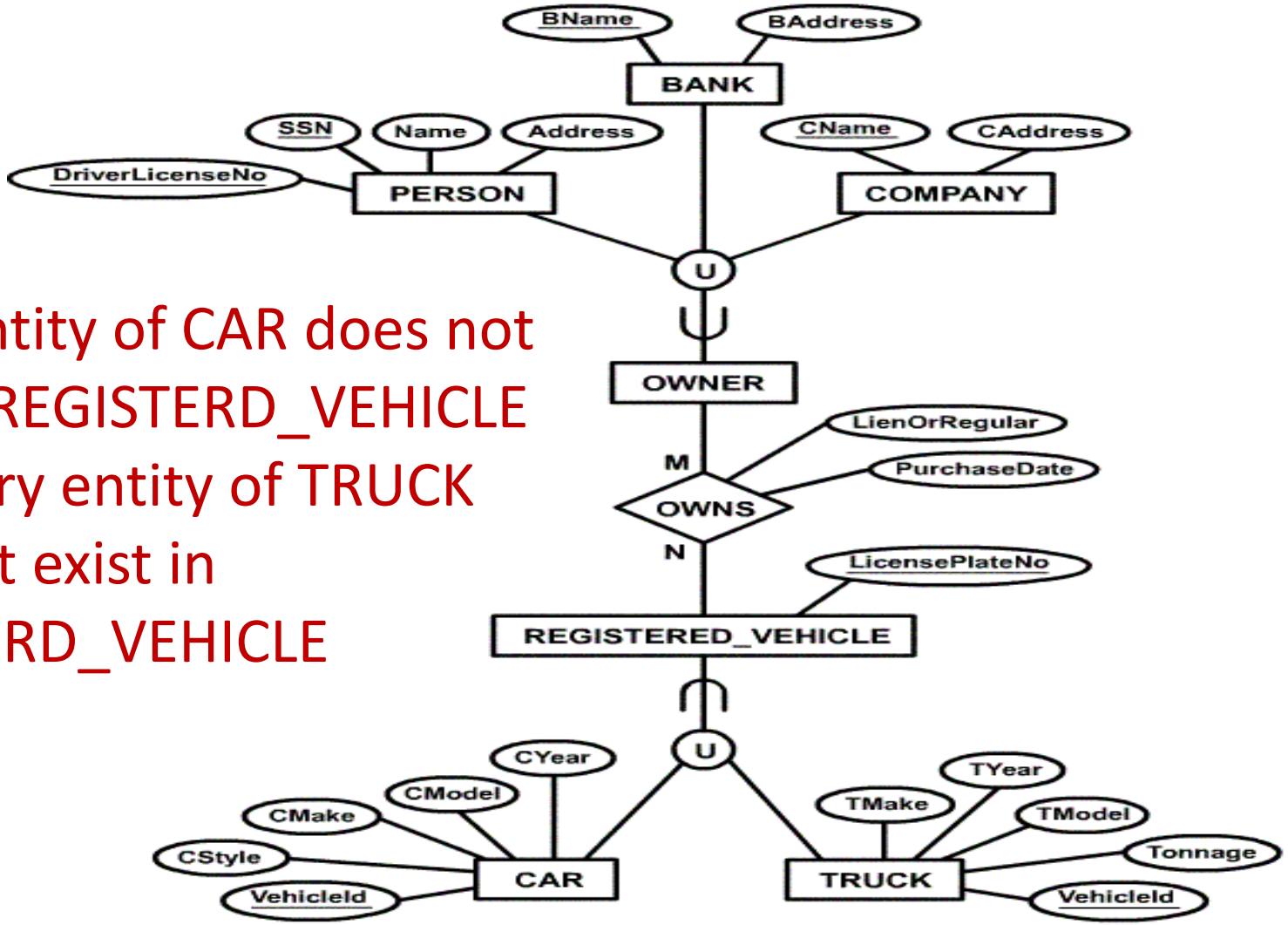
Categories (UNION TYPES)



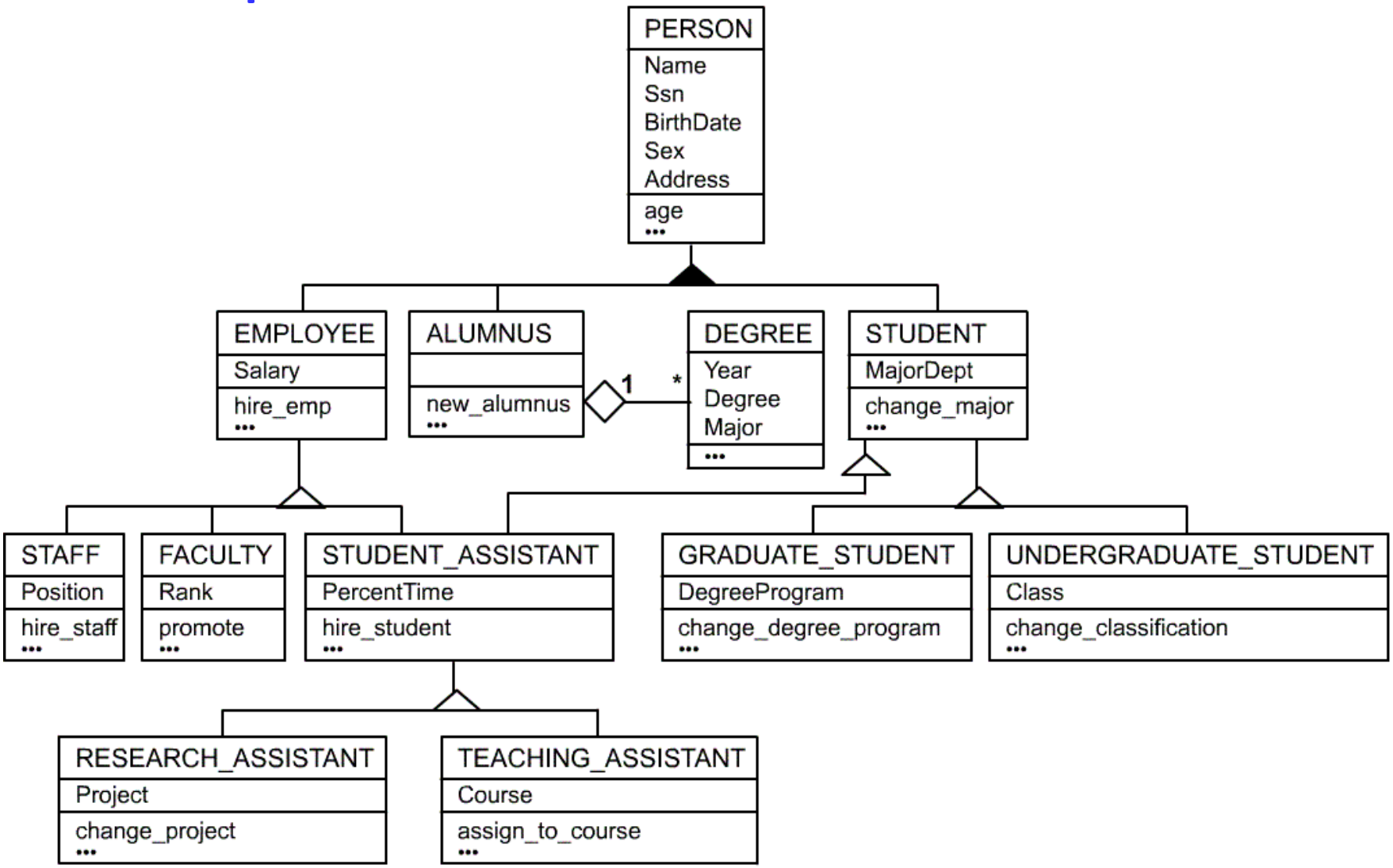
"Every entity of MALE STUDENTS would exist in REGISTERED STUDENTS and every entity of FEMALE STUDENTS would exist in REGISTERED STUDENTS"

Categories (UNION TYPES)

Every entity of CAR does not exist in REGISTERD_VEHICLE and every entity of TRUCK does not exist in REGISTERD_VEHICLE

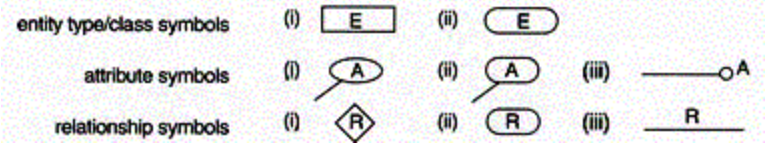


UML Example for Displaying Specialization / Generalization

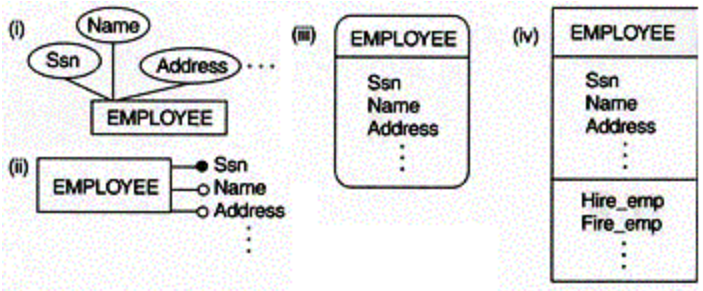


Alternative Diagrammatic Notations

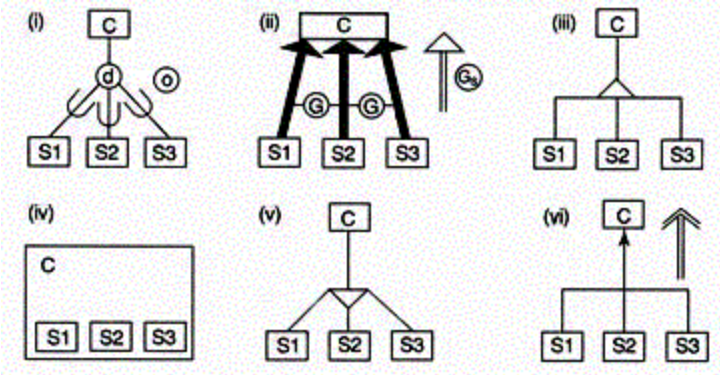
Symbols for entity type / class, attribute and relationship



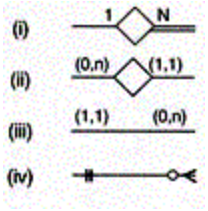
Displaying attributes



Notations for displaying specialization / generalization



Various (min, max) notations



Displaying cardinalitv ra