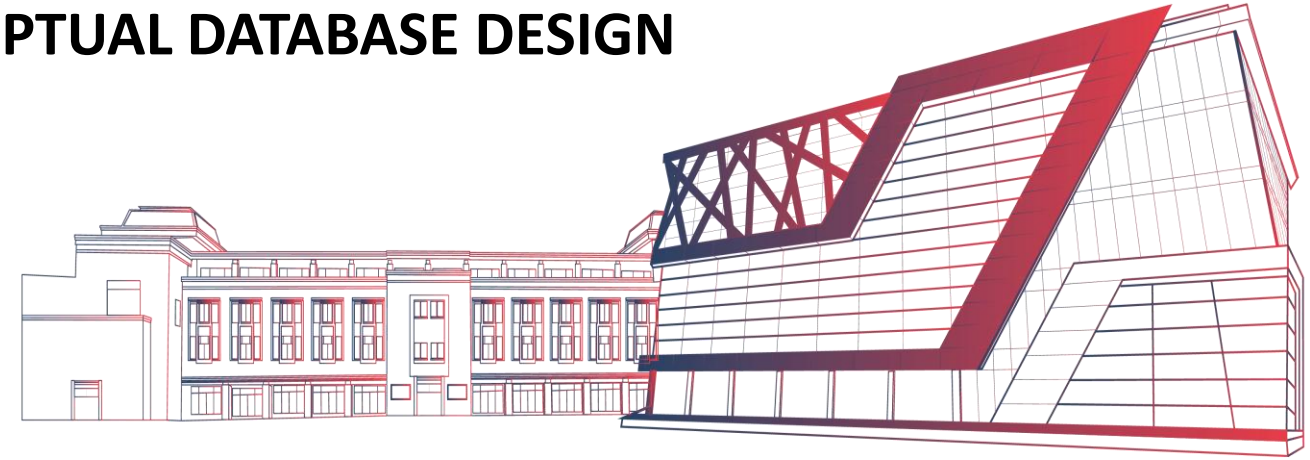
THE NORTHCAP UNIVERSITY

UNIT II

CONCEPTUAL DATABASE DESIGN



30 YEARS OF EXCELLENCE

NAAC Accredited

THE NORTHCAP UNIVERSITY

Constraints on Specialization & Generalization

Constraints on Specialization & Generalization

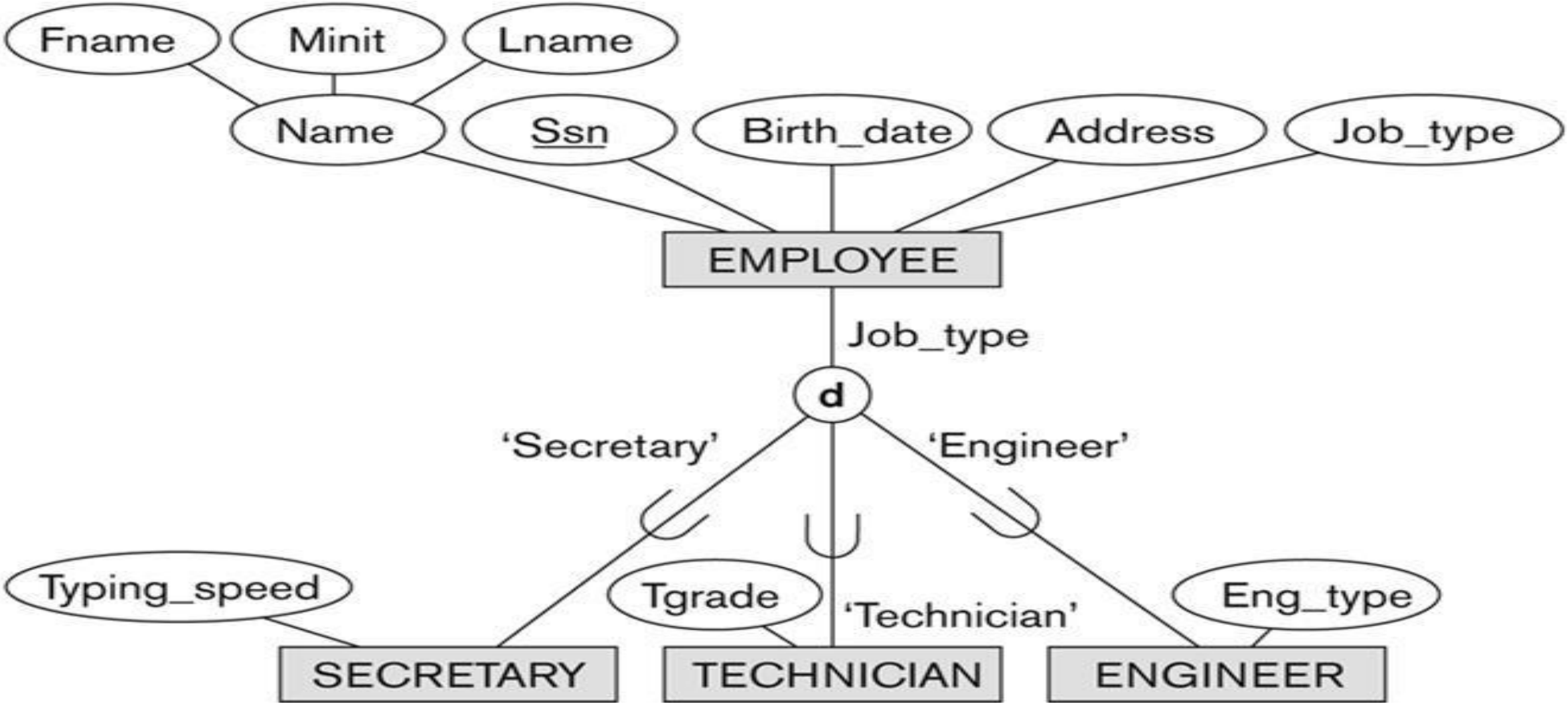
If we can determine exactly those entities that will become members of each subclass by a condition, the subclasses are called **predicate-defined (or condition-defined) subclasses**

- 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**

Constraints on Specialization & Generalization (Cont.)

- If all subclasses in a specialization have membership condition on same attribute
- of the superclass, specialization is called an attribute-defined 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 specialization
  - 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

Attribute Defined Specialization – Job-type



Constraints on Specialization & Generalization (Cont.)

Two basic constraints can apply to a specialization/generalization:

- **Disjointness Constraint**
- **Completeness Constraint**

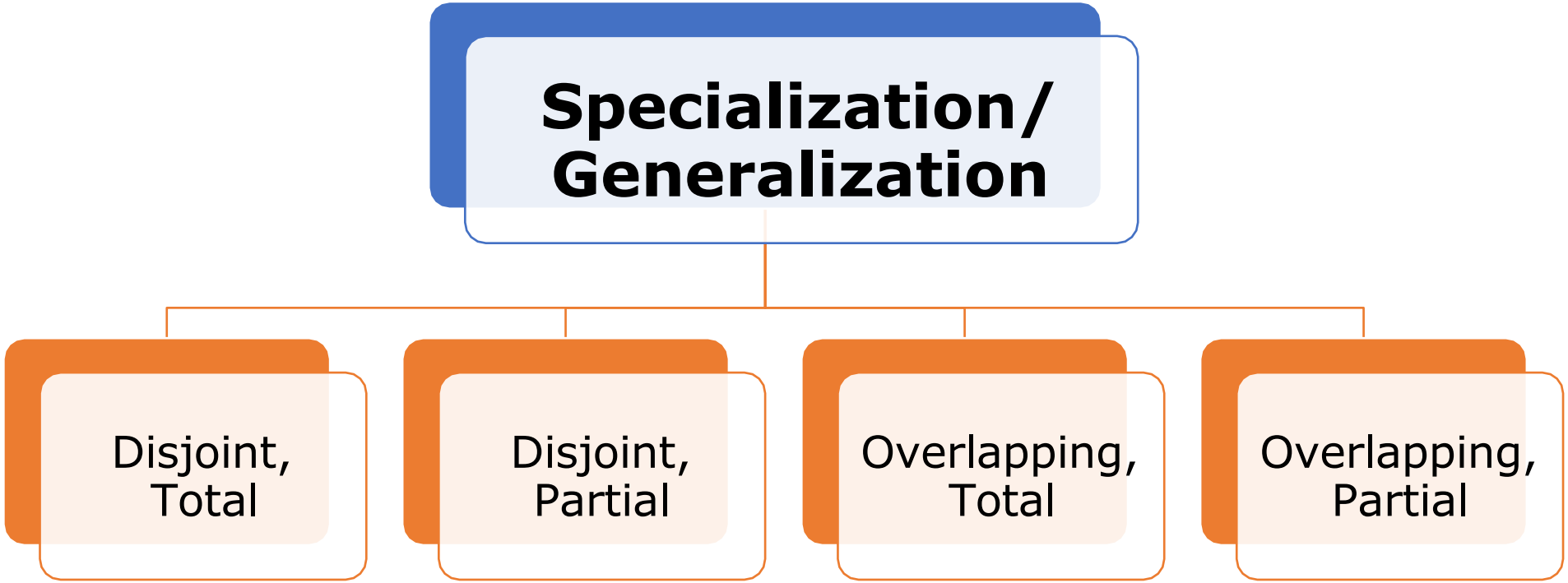
Constraints on Specialization & Generalization (Cont.)

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

COMPLETENESS Constraints - Specialization & Generalization

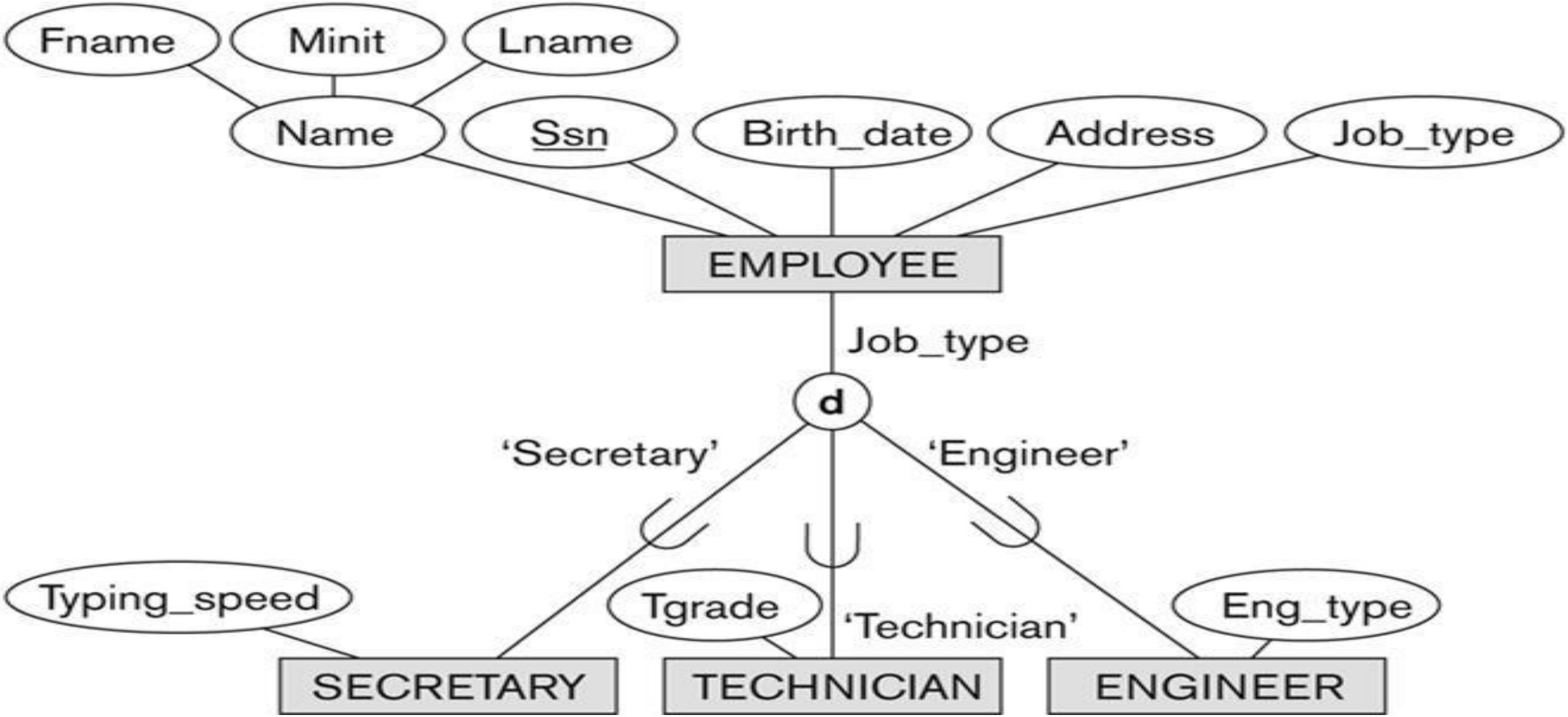
- ***Total*** specifies that every entity in the superclass must be a member of some subclass in the specialization/generalization
  - **Shown in EER diagrams by a *double line***
- ***Partial*** allows an entity not to belong to any of the subclasses
  - **Shown in EER diagrams by a single line**

Constraints on Specialization & Generalization

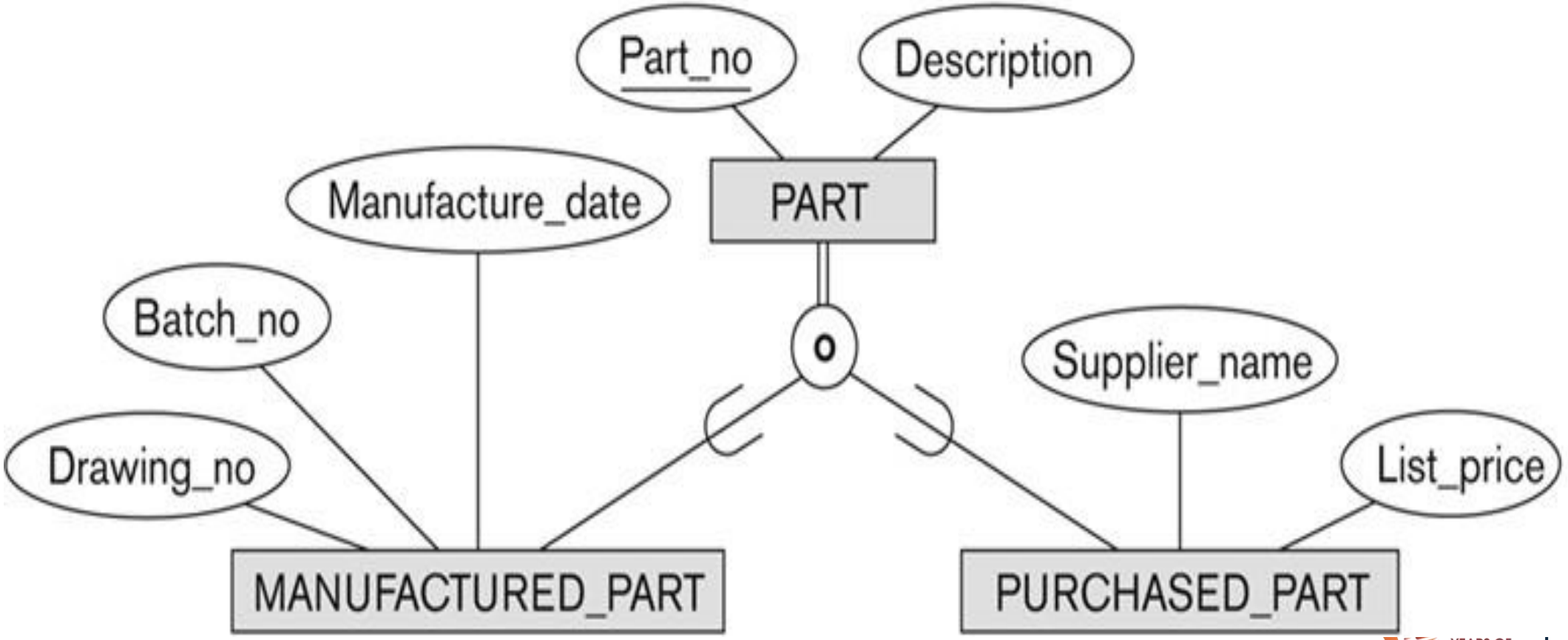


Note: Generalization usually is total because the superclass is derived from the subclasses.

Example of Disjoint Partial Specialization



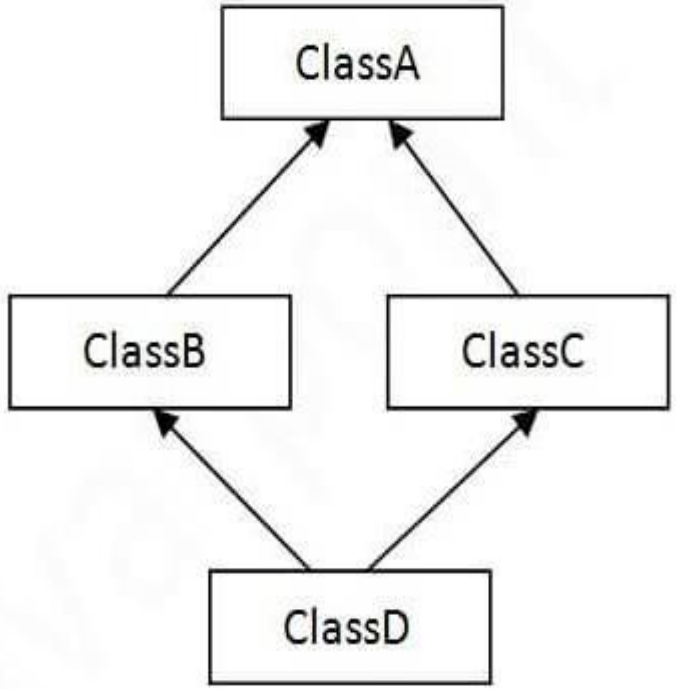
Example of Overlapping Total Specialization



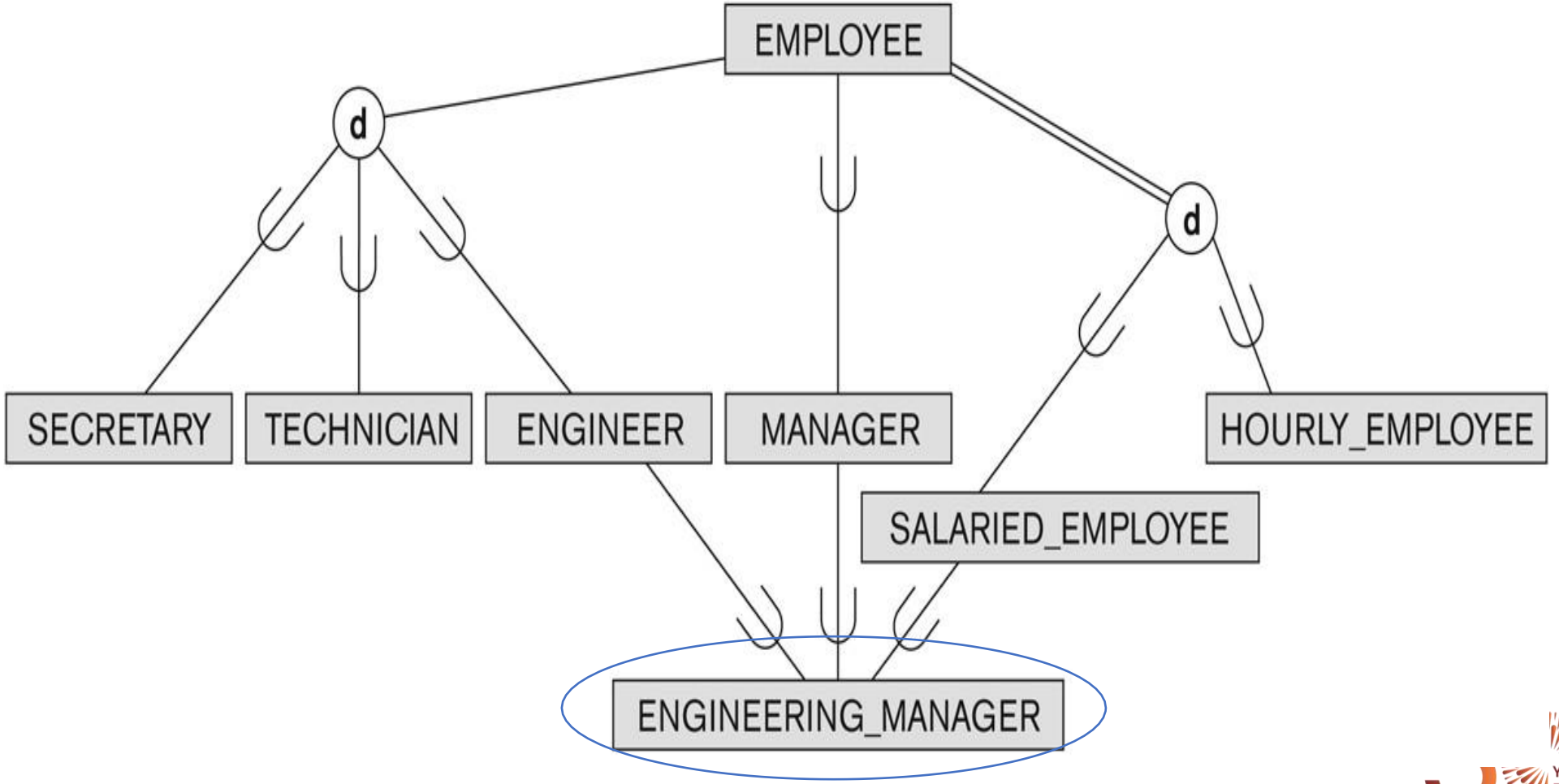
Specialization/Generalization -
Hierarchies, Lattices & Shared Subclasses

A subclass may itself have further subclasses specified on it i.e. **Hierarchy or 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***)



Shared Subclass “Engineering_Manager” (Lattice)



Specialization/Generalization - Hierarchies, Lattices & Shared Subclasses

- **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)**
  - *specialization* hierarchies or lattices, or
  - *generalization* hierarchies or lattices,

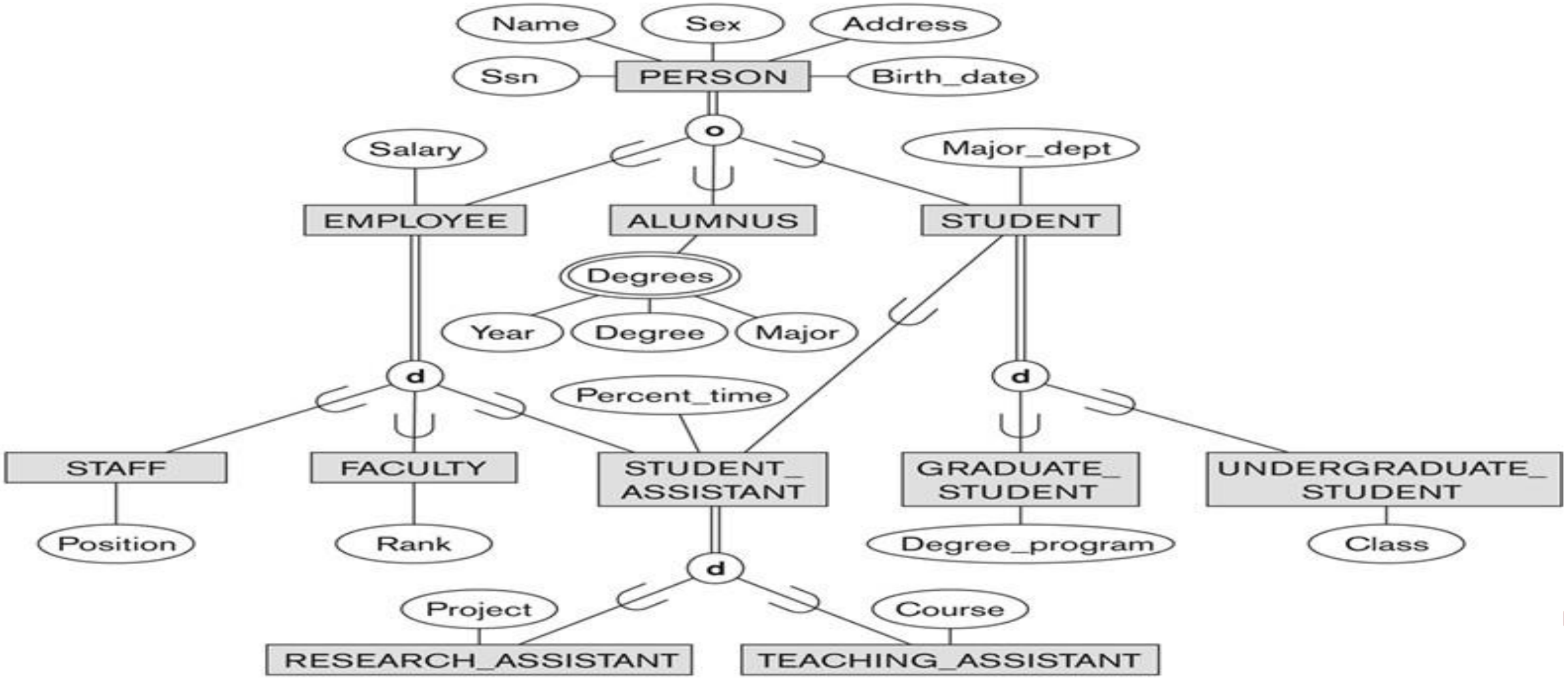
Specialization/Generalization - Hierarchies, Lattices & Shared Subclasses

THE NORTHCAP UNIVERSITY

- 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**

30 YEARS OF EXCELLENCE | NAAC Accredited

Example - Specialization/ Generalization Lattice (UNIVERSITY)



Categories (UNION Type)

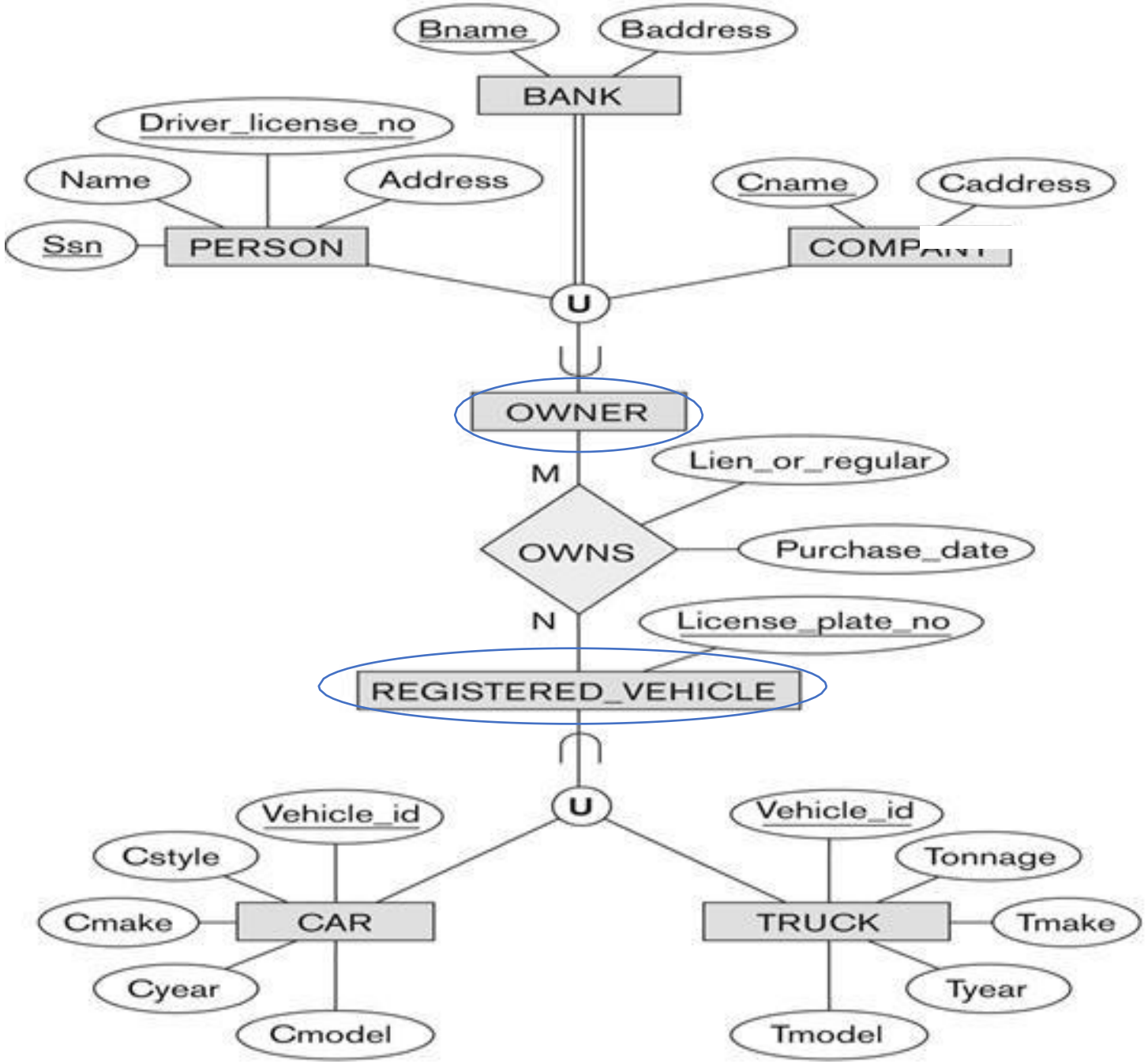
- All of the *superclass/subclass relationships* we have seen thus far have a single superclass
- A shared subclass is a subclass in:
  - *more than one* distinct superclass/ subclass relationships
  - shared subclass leads to multiple inheritance

- In some cases, we need to model a ***single superclass/ subclass relationship* with *more than one* superclass**
- Such a subclass is called a category or **UNION Type**

Example - Categories (UNION Type)

- 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

Two categories (UNION types): OWNER, REGISTERED_VEHICLE



Categories (UNION Type) Cont.

Attribute Inheritance works more selectively in case of categories in comparison to a shared subclass (inherit all attributes of all superclasses).

- A **total category** holds union of all entities in its superclasses. Represented by double line.
- A **partial category** can hold a subset of the union. Represented by single line.

THE NORTHCAP UNIVERSITY

Thanks!!