Lesson 4: Enhanced Entity-Relationship Model

CSC430/530 – DATABASE MANAGEMENT SYSTEMS

DR. ANDREY TIMOFEYEV

OUTLINE

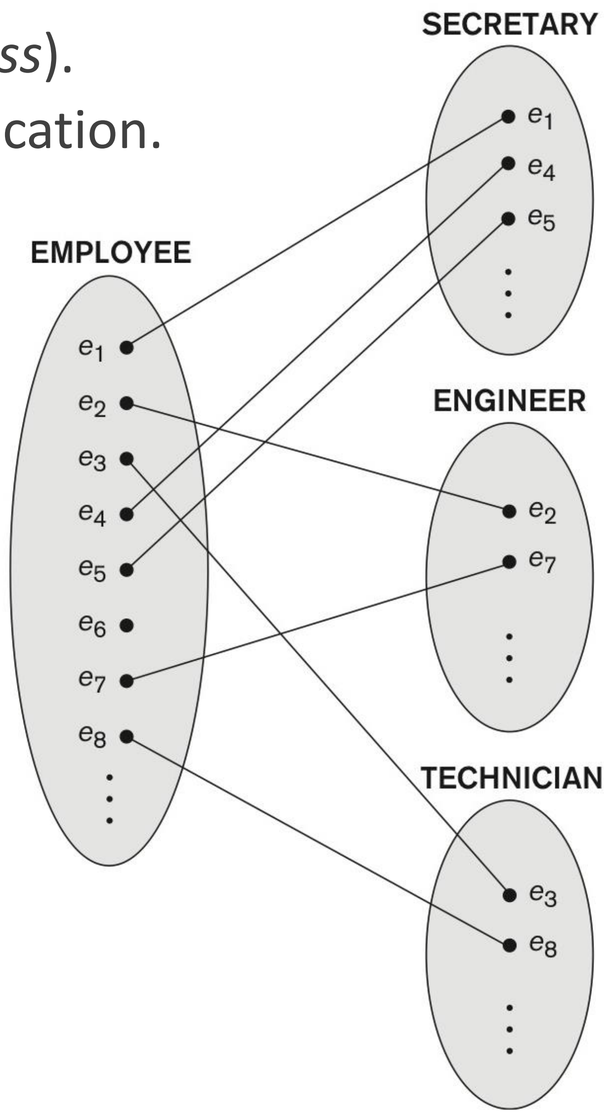
- Enhanced entity relationship model.
- Subclasses & superclasses.
- Specialization & generalization.
- Specialization & generalization constraints.
- Hierarchies & lattices.
- Categories.

INTRODUCTION

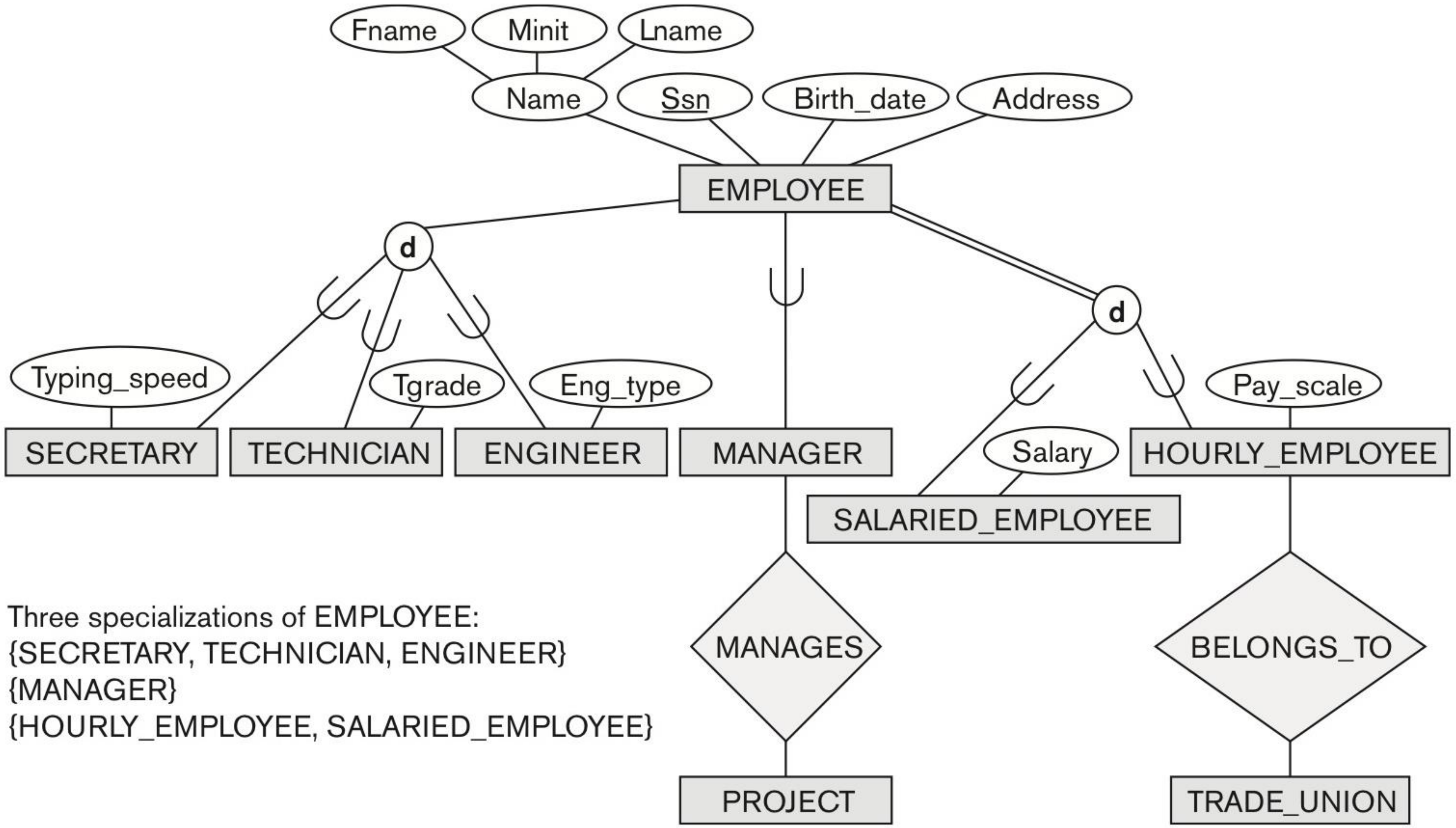
- **Enhanced** (*extended*) **ER model** aims to design more **accurate** database schemas.
  - Allows more **complex requirements** to reflect data properties and constraints more **precisely**.
- Includes **all** modeling **concepts** of basic **ER model**.
- Additional concepts:
  - **Subclasses** & **superclasses**.
  - **Inheritance** of attributes & relationships.
  - **Specialization** & **generalization**.
  - **Categories** (*UNION type*).

SUBCLASSES & SUPERCLASSES (1)

- **Subclass** (*subtype*) - meaningful **subgrouping** of an entity type (*superclass*).
  - Represented **explicitly** because of the significance to the database application.
  - Inherits all **attributes** & **relationships** of superclass.
    - **Type inheritance**.
- **Example**:
  - *EMPLOYEE* entity type subdivided into:
    - *SECRETARY, ENGINEER, TECHNICIAN.*
      - Based on **job title**.
    - *MANAGER.*
      - Based on **role**.
    - *SALARIED_EMPLOYEE, HOURLY_EMPLOYEE.*
      - Based on **method of pay**.



SUBCLASSES & SUPERCLASSES (2)



Three specializations of EMPLOYEE:
{SECRETARY, TECHNICIAN, ENGINEER}
{MANAGER}
{HOURLY_EMPLOYEE, SALARIED_EMPLOYEE}

EER diagram of COMPANY database

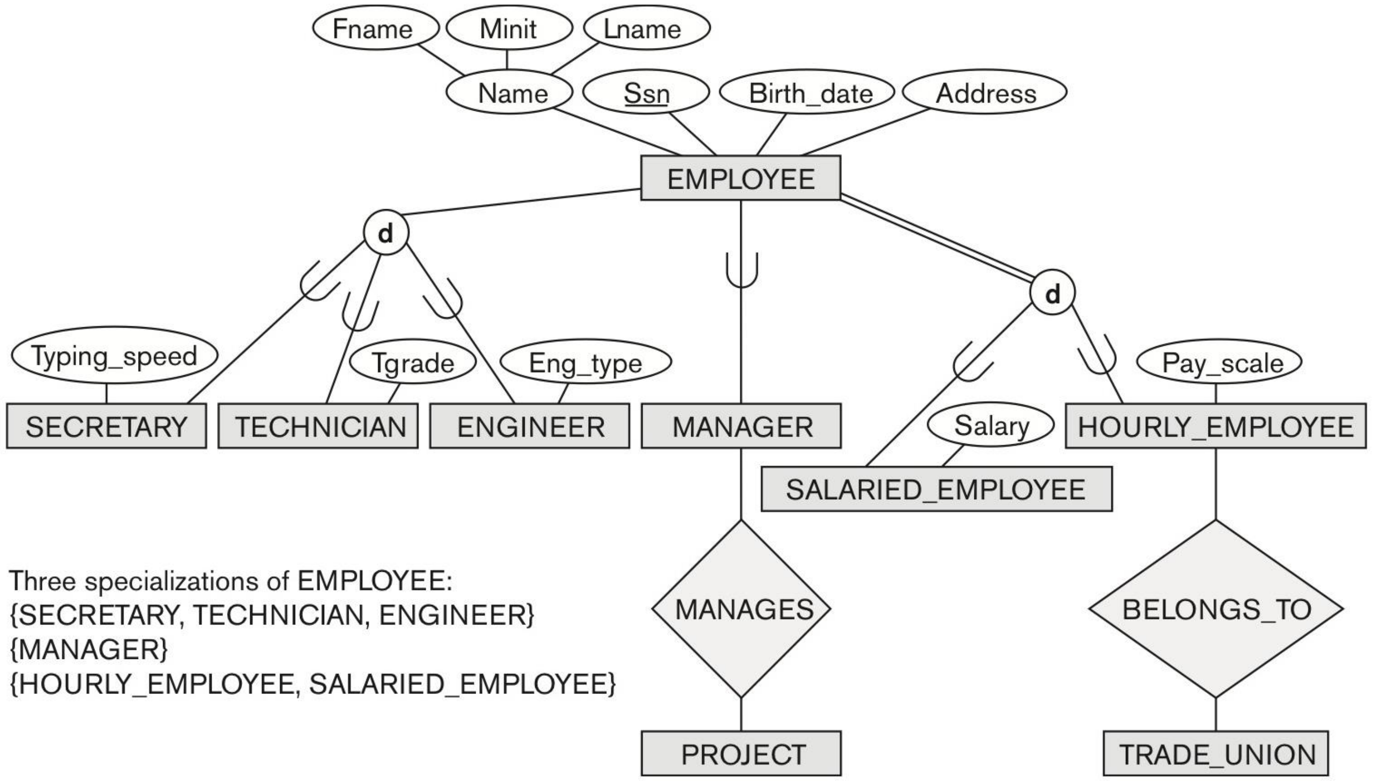
SUBCLASSES & SUPERCLASSES (3)

- **Subclass** entity represents the **same real-world entity** as members of the **superclass**.
  - **Subclass member** is the same entity, but 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**.
    - It is not necessary that *every* entity in a superclass be a member of some subclass.

SPECIALIZATION

- **Specialization** is the process of defining a set of *subclasses* of a *superclass*.
  - Based on some **distinguishing characteristics** of the entities in the superclass.

- Examples of *EMPLOYEE* specializations:
  - *{SECRETARY, ENGINEER, TECHNICIAN}.*
    - Based on **job type**.
  - *{MANAGER}.*
    - Based on **role***.*
  - *{SALARIED_EMPLOYEE, HOURLY_EMPLOYEE}.*
    - Based on **method of pay**.

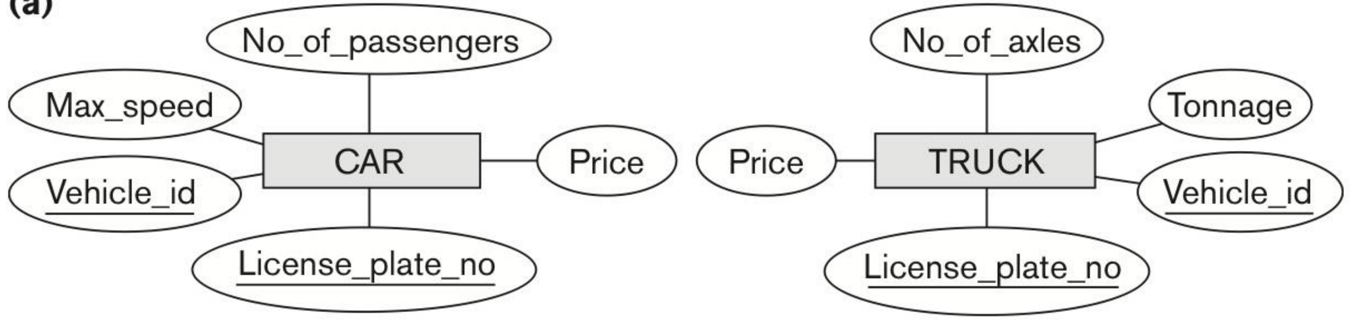


EMPLOYEE entity type specializations

GENERALIZATION

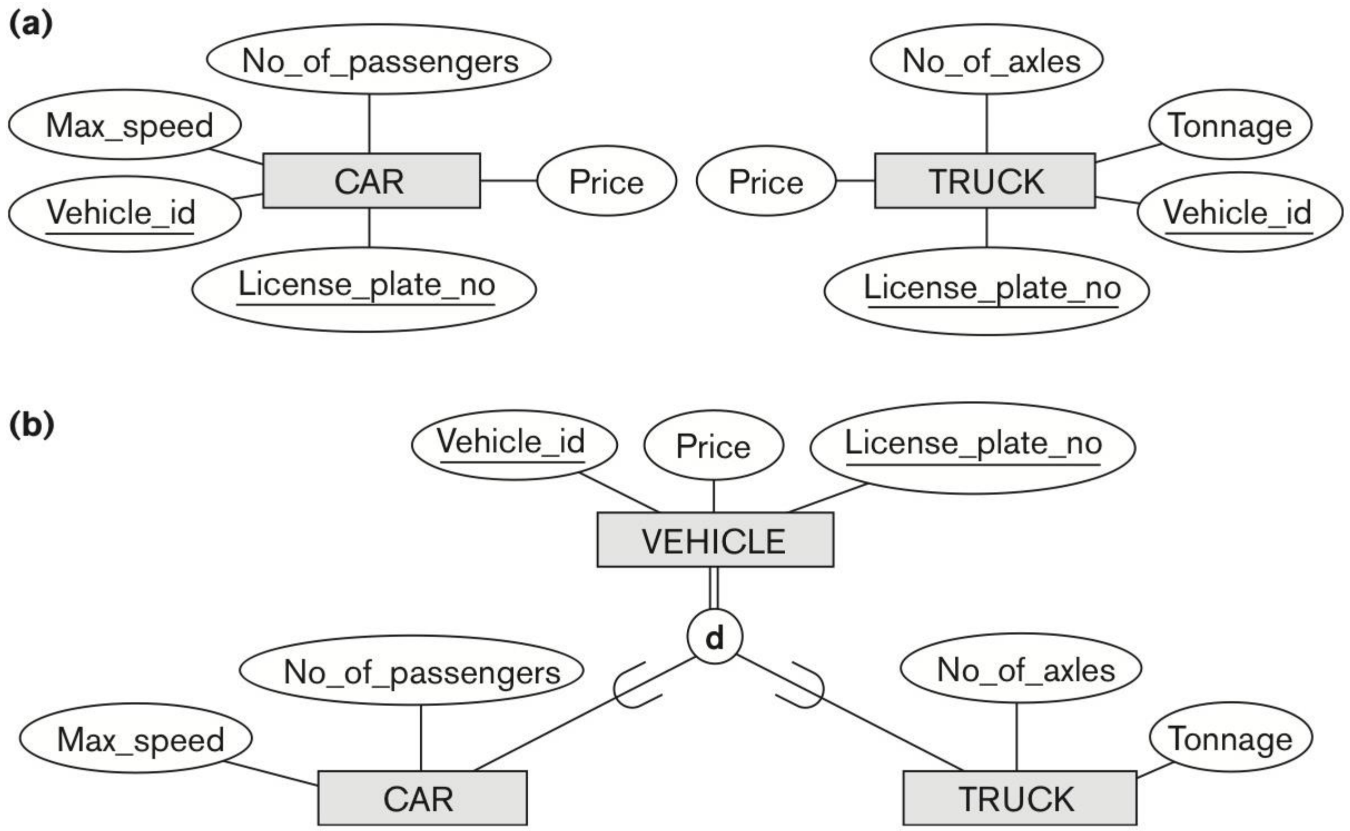
- **Generalization** is the **reverse** of the *specialization* process.
- Several classes with **common features** can be **generalized** into a *superclass*.
  - Original classes become *subclasses* of *superclass*.

(a)



GENERALIZATION

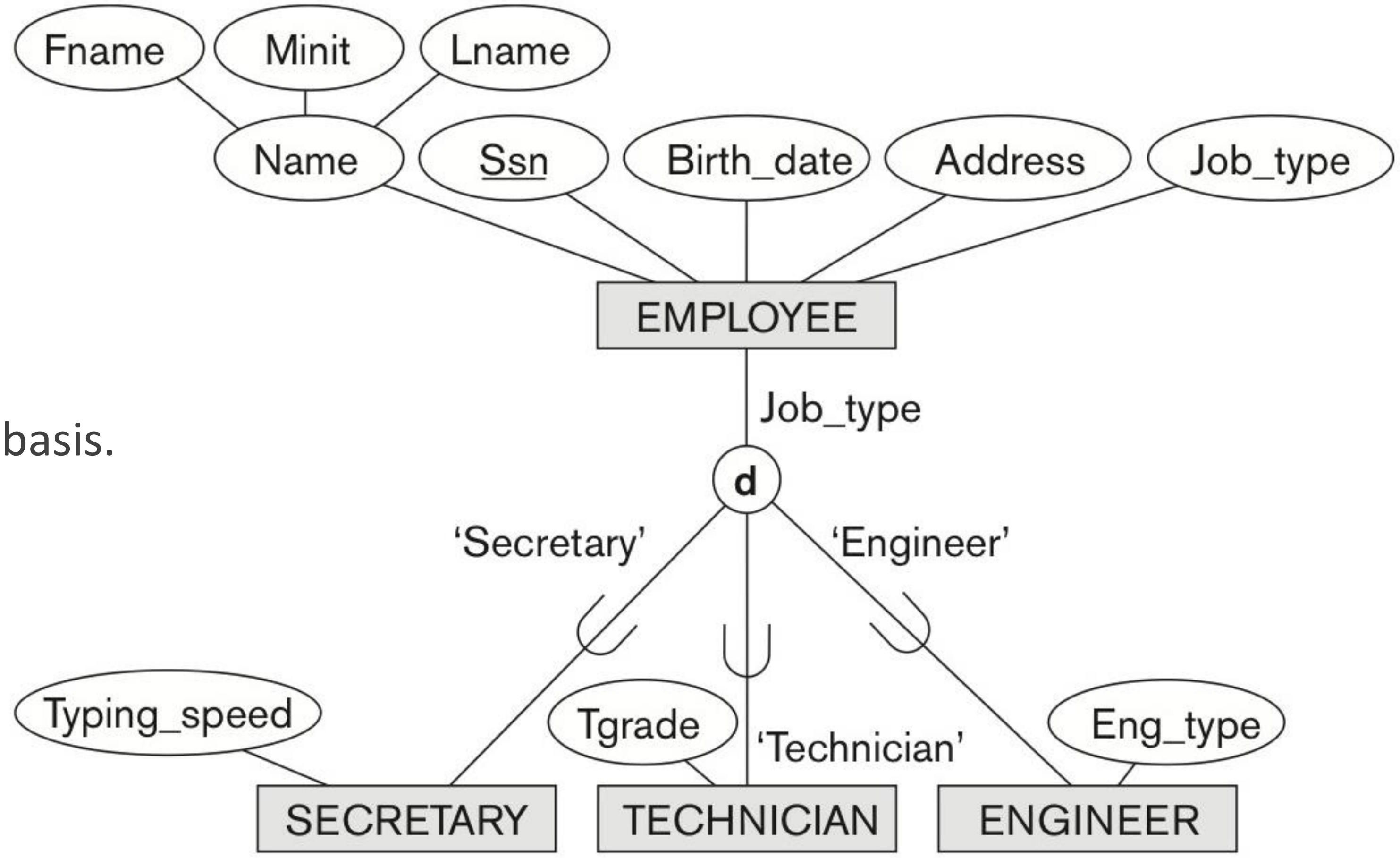
- **Generalization** is the **reverse** of the *specialization* process.
- Several classes with **common features** can be **generalized** into a *superclass*.
  - Original classes become *subclasses* of *superclass.*
- **Example**:
  - *CAR, TRUCK* generalized into *VEHICLE*.
    - Both *CAR, TRUCK* become **subclasses** of the **superclass** *VEHICLE*.
    - *VEHICLE* is a **generalization** of *CAR* and *TRUCK*.
    - *{CAR, TRUCK}* is a **specialization** of *VEHICLE*.



CAR and TRUCK Generalization

TYPES OF SPECIALIZATION & GENERALIZATION

- Specialization/generalization **types**:
  - **Predicate-defined** (*condition-defined*).
    - Based on defining predicate.
      - Job_type = ‘Secretary’.
  - **Attribute-defined**.
    - Based on defining attribute.
  - **User-defined**.
    - Defined by the user on an entity by entity basis.



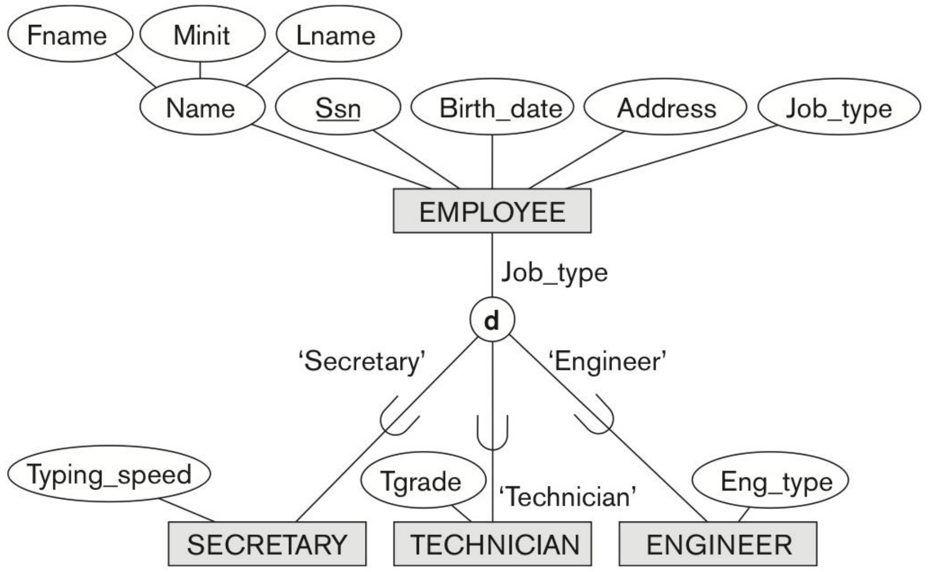
EMPLOYEE entity type specializations

SPECIALIZATION & GENERALIZATION CONSTRAINTS (1)

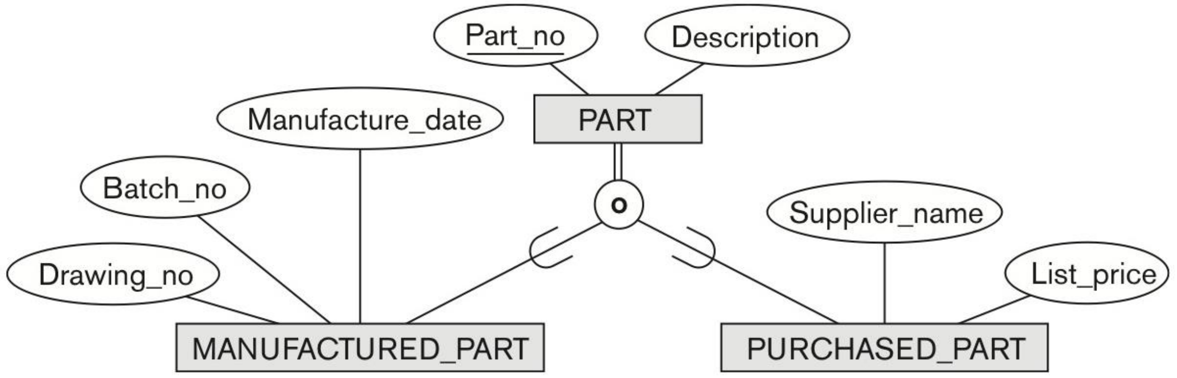
- Specialization & generalization have two types of **constraints**:
  - **Disjointness constraint**.
    - Specialization/generalization can be **disjoint** or **overlapping**.
  - **Completeness constraint**.
    - Specialization/generalization can be **total** or **partial**.

SPECIALIZATION & GENERALIZATION CONSTRAINTS (2)

- **Disjointness constraint**.
  - **Disjoint** sets.
    - Entity can be a member of **at most one** of the subclasses of the specialization.
    - Specified by ***d*** in EER diagram.
  - **Overlapping** sets.
    - Entity may be a member of **more than one** subclass of the specialization.
    - Specified by ***o*** in EER diagram.



Disjoint specialization



Overlapping specialization

SPECIALIZATION & GENERALIZATION CONSTRAINTS (3)

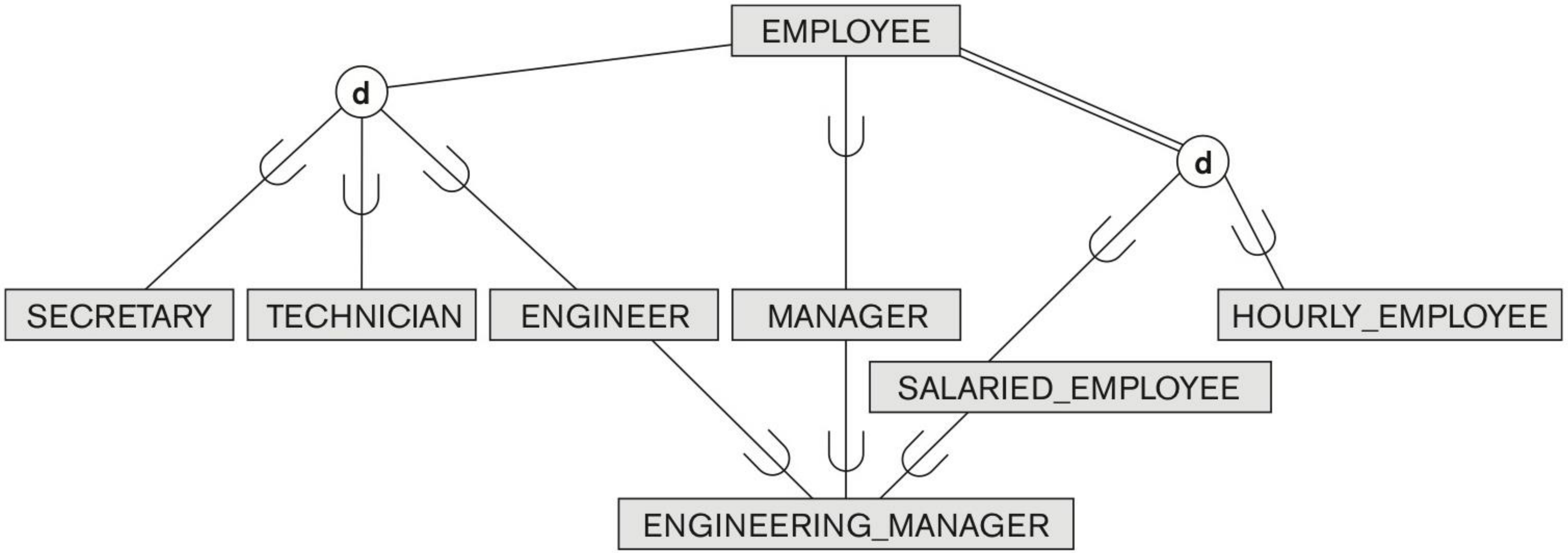
- **Completeness constraint**.
  - **Total**.
    - **Every** entity in the *superclass* must be a member of some *subclass* in the specialization/generalization.
    - Shown in EER diagrams by a **double line.**
  - **Partial**.
    - Allows an entity **not to belong** to any of the *subclasses*.
    - Shown in EER diagrams by a **single line**.
- *Disjointness* and *completeness* constraints are **independent**.
  - Disjoint total.
  - Disjoint partial.
  - Overlapping total.
  - Overlapping partial.

HIERARCHIES & LATTICES (1)

- A subclass may have its *own* subclasses.
  - Forms a **hierarchy** or a **lattice**.
- **Hierarchy.**
  - Every subclass has **only one** superclass.
  - **Single** inheritance.
  - *Tree-like* structure.
- **Lattice.**
  - Subclass can have **more than one** superclass.
    - Subclass that has more than one superclass is called a **shared subclass**.
  - **Multiple** inheritance.
  - *Graph-like* structure.
- In a **lattice** or **hierarchy**, a subclass inherits **attributes** not only of its direct superclass, but also of **all** its predecessor superclasses.

HIERARCHIES & LATTICES (2)

- **Example**:
  - *ENGINEERING_MANAGER* is a **shared subclass** that inherits from three superclasses *ENGINEER*, *MANAGER* and *SALARIED_EMPLOYEE*.

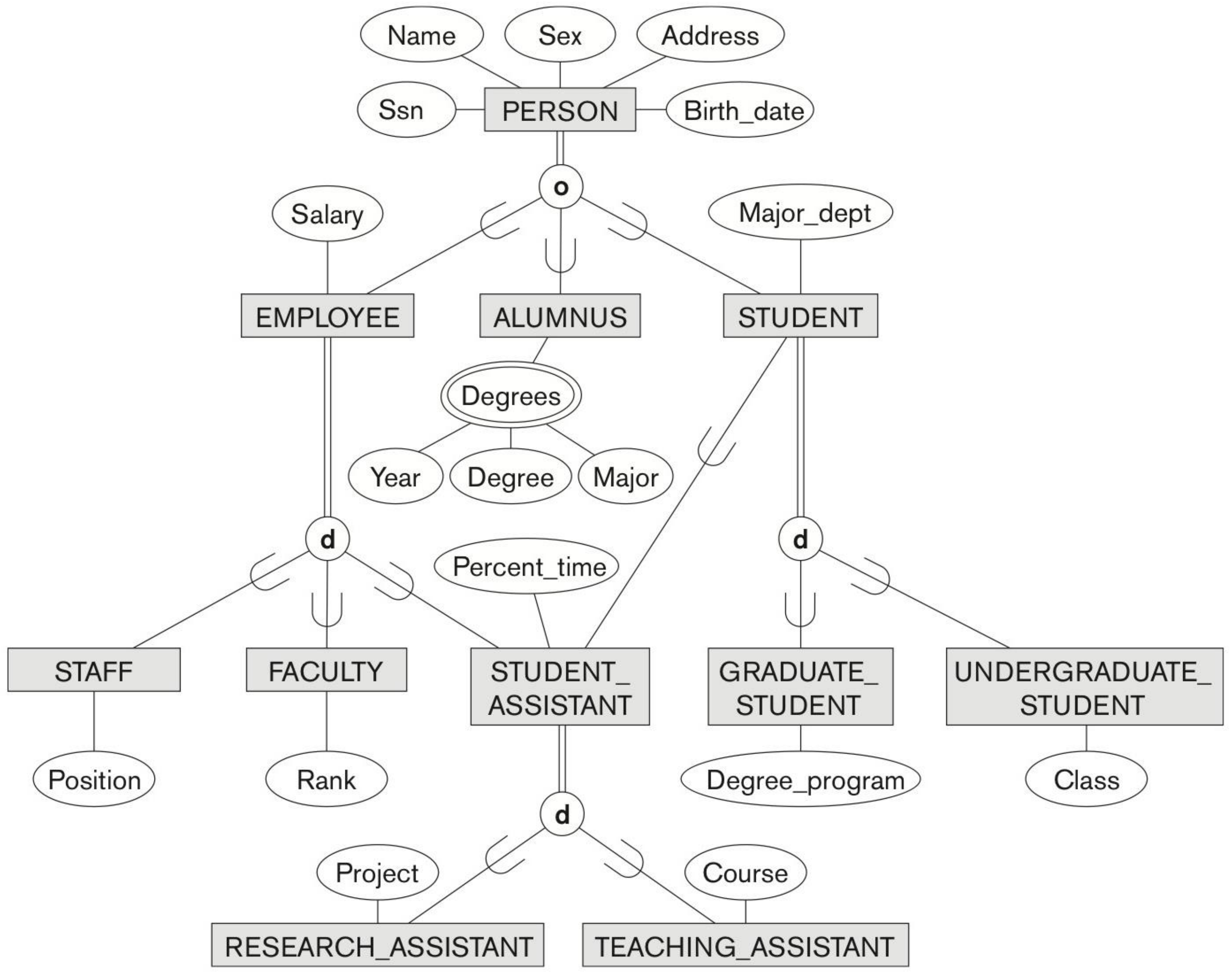


Specialization lattice with ENGINEERING_MANAGER shared subclass

HIERARCHIES & LATTICES (3)

- ER schema can be further **refined** into EER schema in two ways:
  - **Top-down** conceptual **refinement**.
    - Based on **specialization**.
      - Start with an entity type and then define subclasses of the entity type by successive *specialization*.
  - **Bottom-up** conceptual **synthesis**.
    - Based on **generalization**.
      - Start with many entity types and *generalize* those that have common properties.
- In practice, a **combination of both** processes is employed.

HIERARCHIES & LATTICES (4)



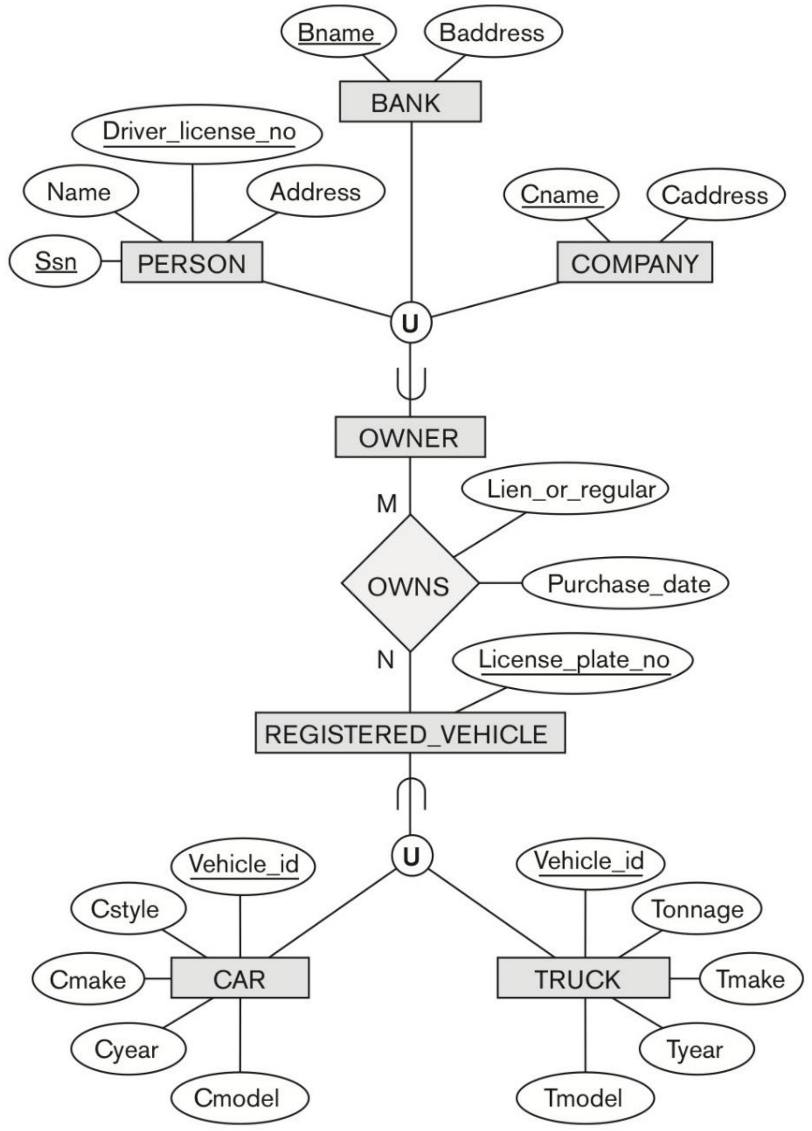
Specialization lattice with multiple inheritance for a UNIVERSITY database

CATEGORIES / UNION TYPES (1)

- **Category** (*union type*).
  - In some cases it is necessary to represent a collection of entities from **different entity types**.
    - Subclass represents a collection of entities that is a **subset** of the *UNION* of entities from distinct entity types.
- **Example**.
  - In a database for vehicle registration, a vehicle owner can be a *PERSON*, a *BANK* or a *COMPANY*.
  - *OWNER* **category** (*union type*) 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** (typically *just* one) of its superclasses.

CATEGORIES / UNION TYPES (2)

- **Example** (cont.)
  - *OWNER* category (union type).
    - **Union** of *BANK*, *PERSON*, and *COMPANY* entity types.
  - *REGISTERED_VEHICLE* category (union type)
    - **Union** of *CAR* and *TRUCK* entity types.



EER DESIGN GUIDELINES

- **Guidelines** for the EER design process:
  - More *specializations* and *subclasses* = more **accurate** conceptual model.
    - **Drawback** – *cluttered* design.
  - Subclass with **few** specific (local) *attributes* / no specific *relationships* is **merged** into superclass.
    - Specific attributes = *NULL* values for entities that are **not members** of the subclass.
    - A *type* attribute can be used to specify the subclass.
  - The choice for **disjoint**/**overlapping** and **total**/**partial** constraints is driven by the *rules* in the mini-world.
    - **No** particular constraints = *overlapping* and *partial*.

SUMMARY

- Enhanced ER model.
- Subclasses & superclasses.
- Specialization & generalization.
  - Predicate-defined, attribute-defined, and user-defined.
- Constraints on specialization & generalization.
  - Disjointness & completeness.
- Hierarchies & lattices.
  - Single inheritance / multiple inheritance.
- Categories (union types).