CMPE 138/180B
Database System I
Enhanced Entity-Relationship (EER) Model

Instructor: Kong Li

Outline

- Subclasses, Superclasses, and Inheritance
- Specialization and Generalization
- Constraints
  - Participation: total vs partial
  - Disjoint vs overlap
- Design Choice

Subtype, Subclass

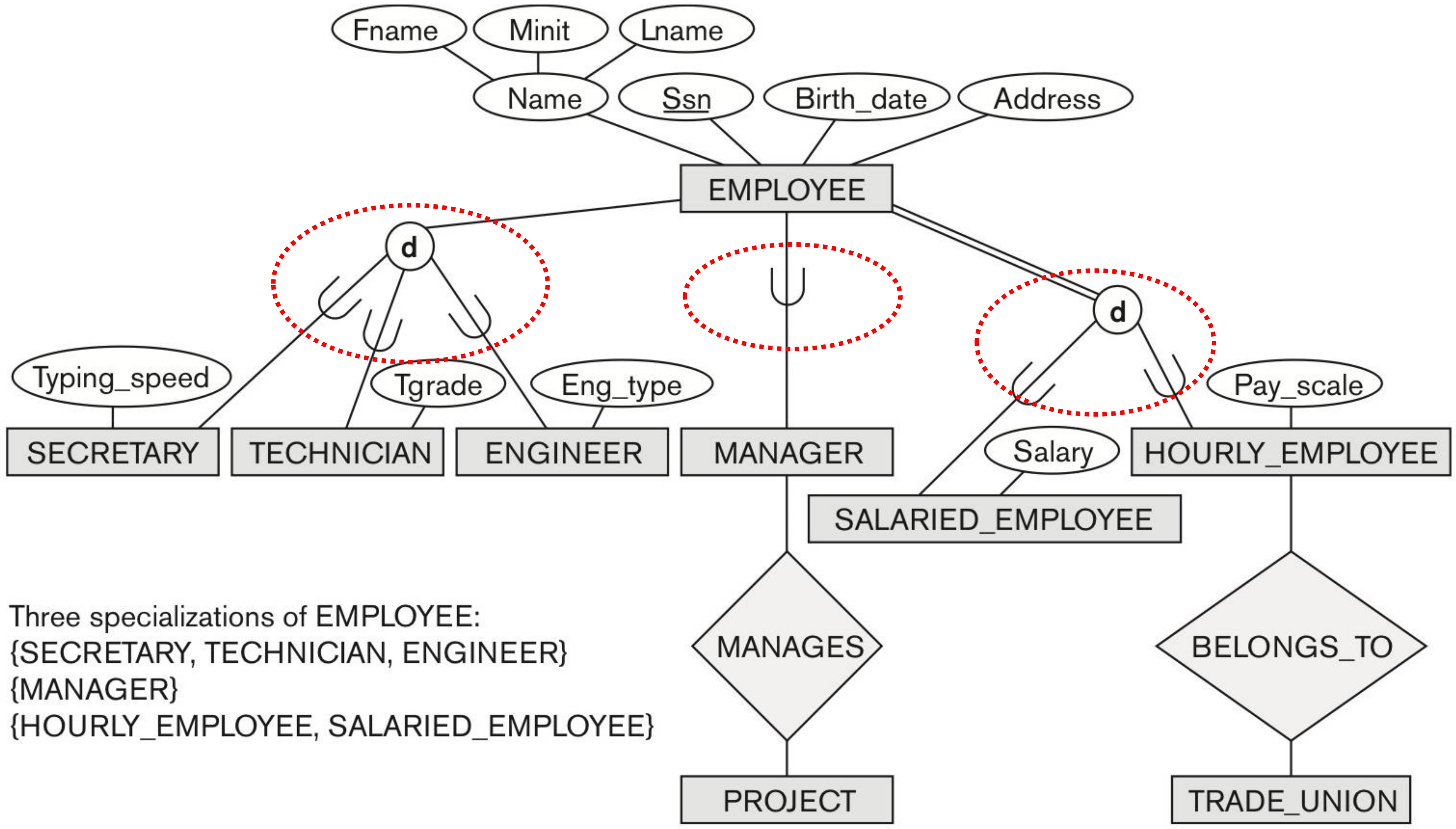
- **Subtype** or **subclass** of an entity type
  - Subgroupings of entities that are meaningful
    - w/ specific role, etc
  - Represented explicitly due to significance to DB apps
  - E.g., EMPLOYEE ➔ SECRETARY, ENGINEER, MANAGER, TECHNICIAN, SALARIED_EMPLOYEE, HOURLY_EMPLOYEE
- Terminology: relationship b/w a superclass and any one of its subclasses
  - **Superclass/subclass**
  - **Supertype/subtype**
  - **Class/subclass** relationship

IS-A relationship:
A SECRETARY *is an* EMPLOYEE
A TECHNICIAN *is an* EMPLOYEE

Enhanced Entity-Relationship (EER) Model

- Purposes
  - more accurate database schemas
    - Reflect properties and constraints more precisely
  - More complex requirements than traditional apps
- EER == ER +
  - subclass and superclass
  - specialization and generalization
  - category or union type
  - attr and relationship inheritance

EER Diagram (EERD)



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

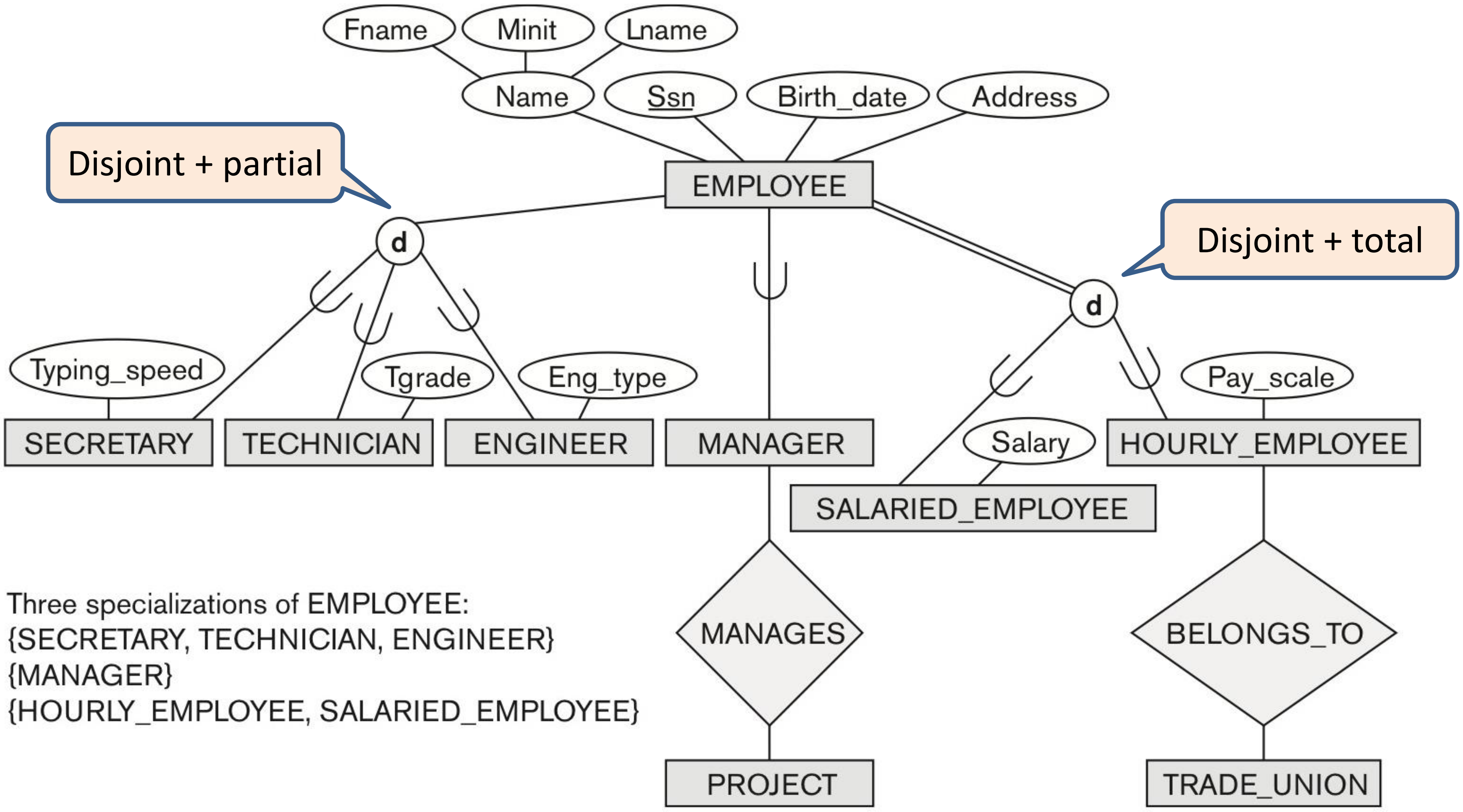
EERD: Subtype, Subclass

- Membership
  - membership in subclass ➔ membership in superclass
    - Any member in SECRETARY implies membership in EMPLOYEE
  - Completeness/Participation
    - Total (double line): *every* entity in superclass belong to *at least one* subclass
    - Partial (single line): *not* every entity in superclass belong to some subclass
  - Disjointness
    - Disjoint (d): An entity in superclass can be a member of *at most one* of the subclasses
      - A EMPLOYEE can be SECRETARY or ENGINEER, but not both
    - Overlap (o): An entity in superclass *may* belong to *multiple* subclasses
      - One PART may belong to both MANUFACTURED_PART and PURCHASED_PART
- Type inheritance
  - Subclass inherits all attrs and relationships of superclass
    - SECRETARY also has Ssn, Birth_day, Address attrs inherited from superclass

Specialization

- Process of defining a set of subclasses of an entity type
  - Based on distinguishing characteristic in the superclass
  - EMPLOYEE (job type) ➔ {SECRETARY, ENGINEER, TECHNICIAN}
  - EMPLOYEE (method of pay) ➔ {SALARIED_EMPLOYEE, HOURLY_EMPLOYEE}
- Subclass inherits all attrs and relationships of its superclass
- Subclass can define:
  - Specific (local) attrs
    - SECRETARY has Typing_speed attr, ENGINEER has Eng_type attr
  - Specific (local) relationship types
    - HOURLY_EMPLOYEE participates in the BELONGS_TO relationship
- EER Diagram:
  - an arc pointing to subclass
  - Subclass: rectangle

EER Diagram



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

Why Specialization?

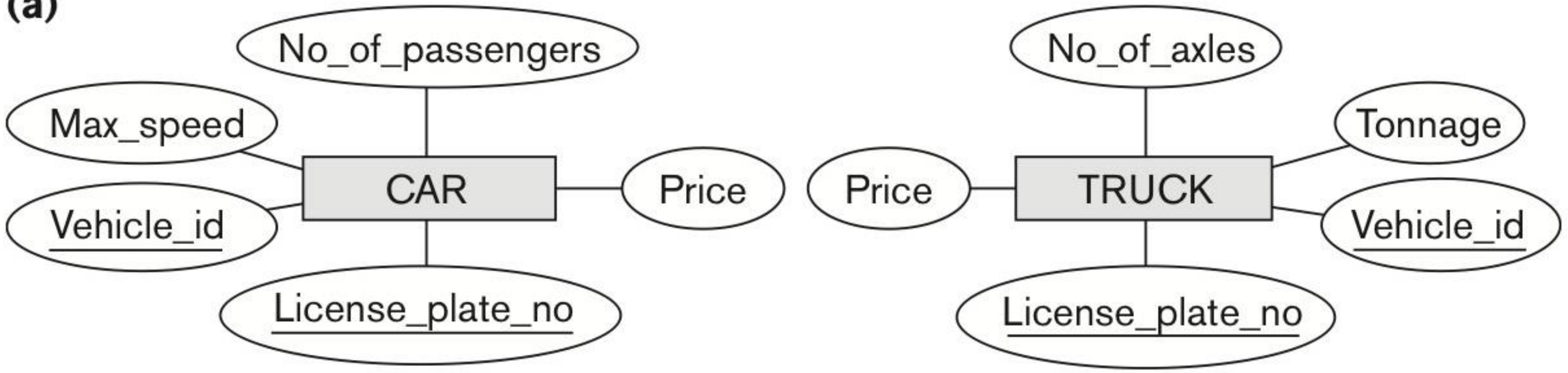
- Reasons: be more precise
  - Certain attrs may apply to some but not all entities of the superclass
  - Some relationship types may be participated in only by members of the subclass
- Summary: Specialization allows us to
  - Define a set of subclasses of an entity type
  - Establish additional specific attrs with each subclass
  - Establish additional specific relationship types b/w each subclass and other entity types

Generalization

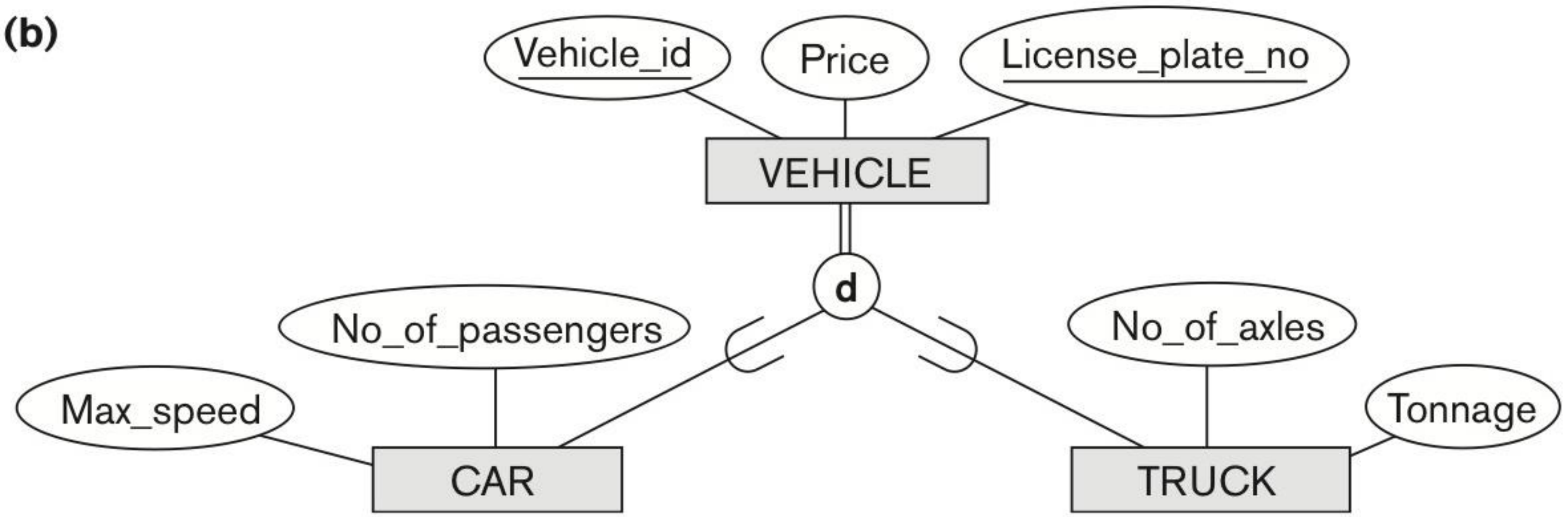
- Process of defining a generalized entity type from given entity types
  - Suppress differences among several entity types
  - Identify common features
  - **Generalize** into a single **superclass**
    - Original/source entity types are subclasses
- Reverse process of specialization
- Generalization: subclasses ➔ superclass
  - {CAR, TRUCK} ➔ VEHICLE
- Specialization: superclass ➔ subclass
  - VEHICLE ➔ {CAR, TRUCK}
- EER Diagram:
  - an arc pointing to subclass
  - Subclass: rectangle

Generalization (cont'd)

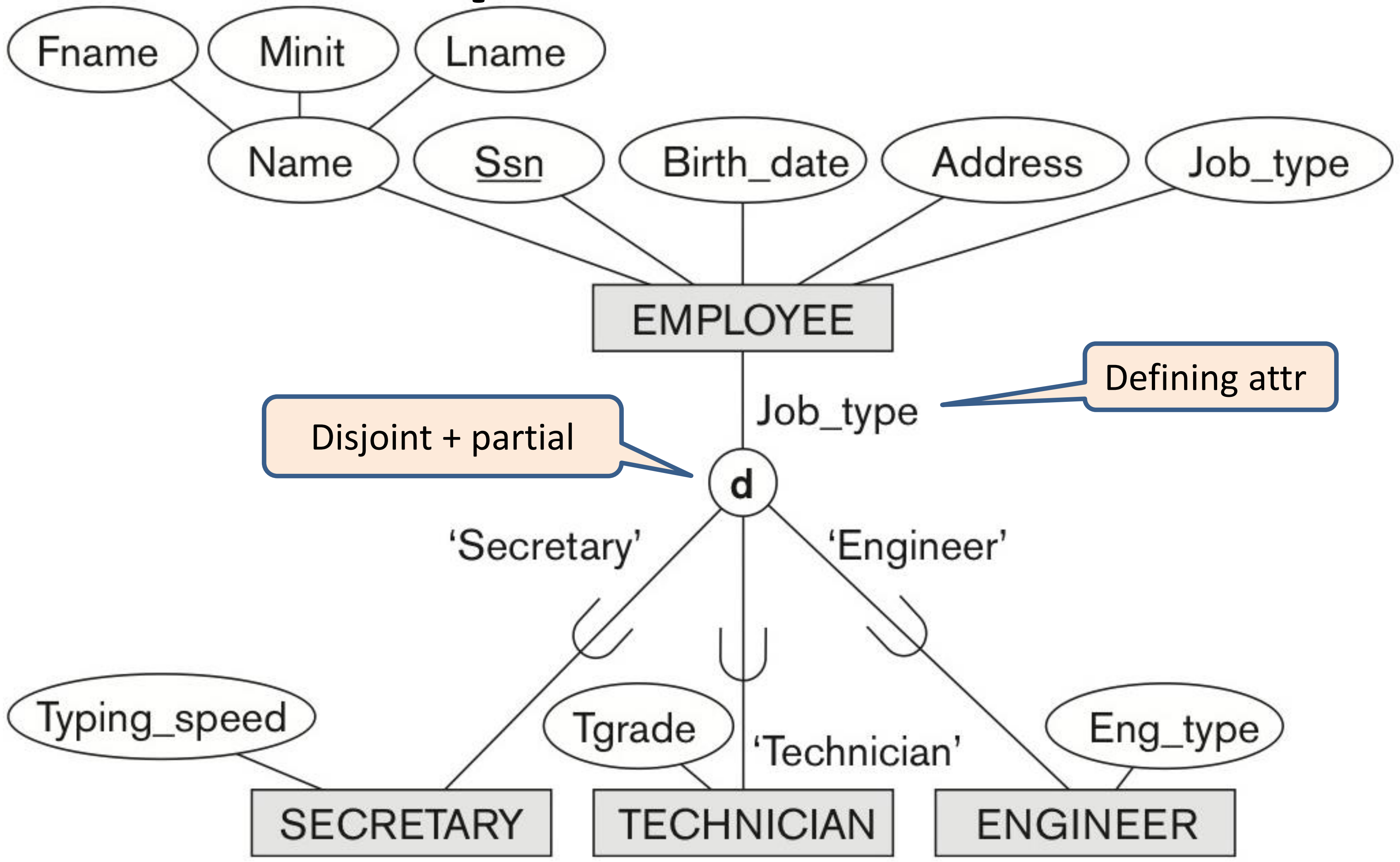
(a)



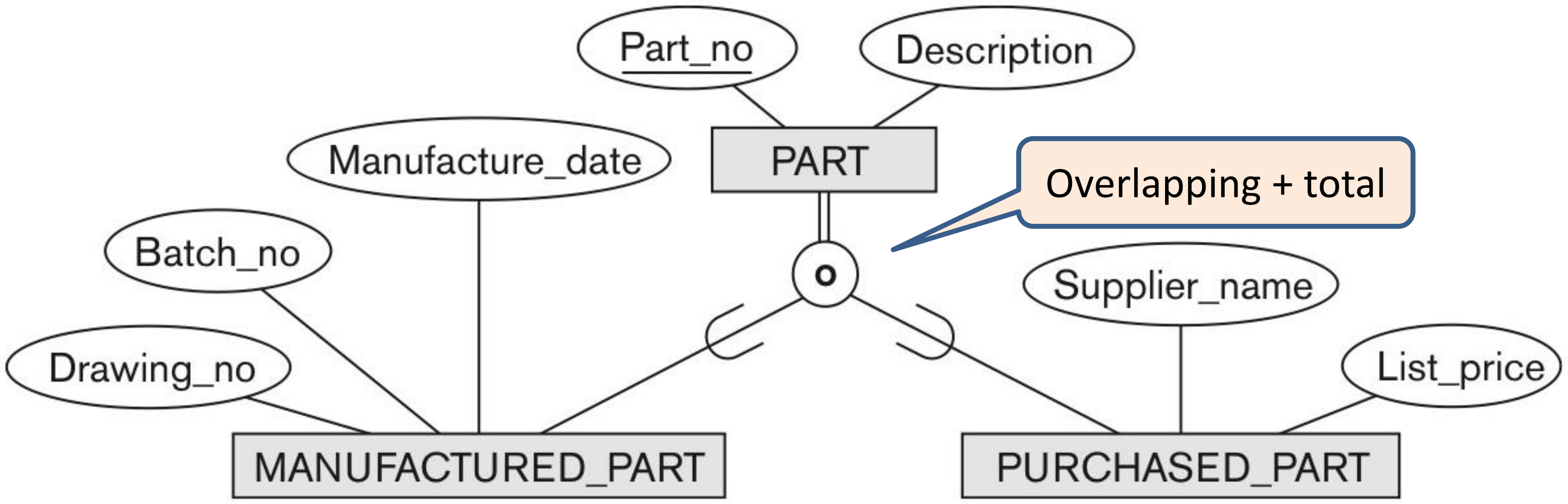
(b)



EER Diagram: attr-defined specialization



EER Diagram: overlapping specialization



Constraints on Specialization

- Disjointness and completeness constraints are *independent*
  - Disjoint + total
  - Disjoint + partial
  - Overlapping + total
  - Overlapping + partial
- Constraints on specialization also apply to generalization
- In general, a superclass through generalization process usually is total
- Insert/delete rules apply to specialization/generalization
  - Delete entity from superclass ➔ delete from *all* subclasses it belong to
  - Insert entity in superclass ➔ insert in *all* predicate-defined (or attr-defined) subclasses it belong to
  - Insert an entity in superclass of a *total specialization* ➔ inserted in at least one of the subclasses of the specialization

Refining Conceptual Schemas w/ Specialization and Generalization

- **Top-down conceptual refinement process**
  - Specialization process
  - Start from entity type and then subclasses (successive specialization)
- **Bottom-up conceptual synthesis**
  - Generalization process
  - Start from the bottom and work the way up
- Hybrid

Design Choices for Specialization, Generalization

- Specializations + subclasses
  - Pros: make conceptual model accurate
  - Cons: cluttered design
- A subclass w/ few attrs and no relationships
  - Can be merged into superclass (w/ one NULLable type attr)
- All subclasses of specialization/generalization w/ few attrs and no relationships
  - Can be merged into superclass (w/ multiple NULLable type attr)
- Specialization/generalization
  - Disjoint vs overlapping ?
  - Total vs partial ?
  - driven by requirements in miniworld being modeled
    - If no constraints specified by requirements ➔ default: overlapping + partial

Summary

- Superclass, Subclass, inheritance
- Specialization
- Generalization
- Constraints
  - Participation: total vs partial
  - Disjoint vs overlap
- Design choice

Self Exercises

- 7/E: Exercise 4.17, 4.18, 4.19, 4.21, 4.22, 4.27
- 6/E: Exercise 8.17, 8.18, 8.19, 8.21, 8.22, 8.27