Database Management Systems (DBMS)

Lec 11: The Enhanced ER Model

Ramesh K. Jallu

IIIT Raichur

Date: 09/03/21

Recap

- Two more examples
  - University database
  - Bank database
- Relationship types of higher degree
- Common mistakes in designing ER diagrams

Today's plan

- Enhanced Entity-Relationship (EER) Model

- The concepts of EER model
  - Subclass and Superclass, Inheritance
  - Specialization and Generalization
  - Union or Category
  - Aggregation

Today's plan

- Enhanced Entity-Relationship (EER) Model

- The concepts of EER model
  - Subclass and Superclass, Inheritance
  - Specialization and Generalization
  - Union or Category
  - Aggregation

Traditional ER model

- The ER modeling concepts discussed so far good for traditional database applications
- Today's database are more complex and have requirements that are more complex than the traditional applications
  - e.g., databases for engineering design and manufacturing, telecommunications, complex s/w systems, GIS, etc.
- As the complexity of data increasing it becomes difficult to use the traditional ER model for database modeling

Enhanced Entity-Relationship (EER) model

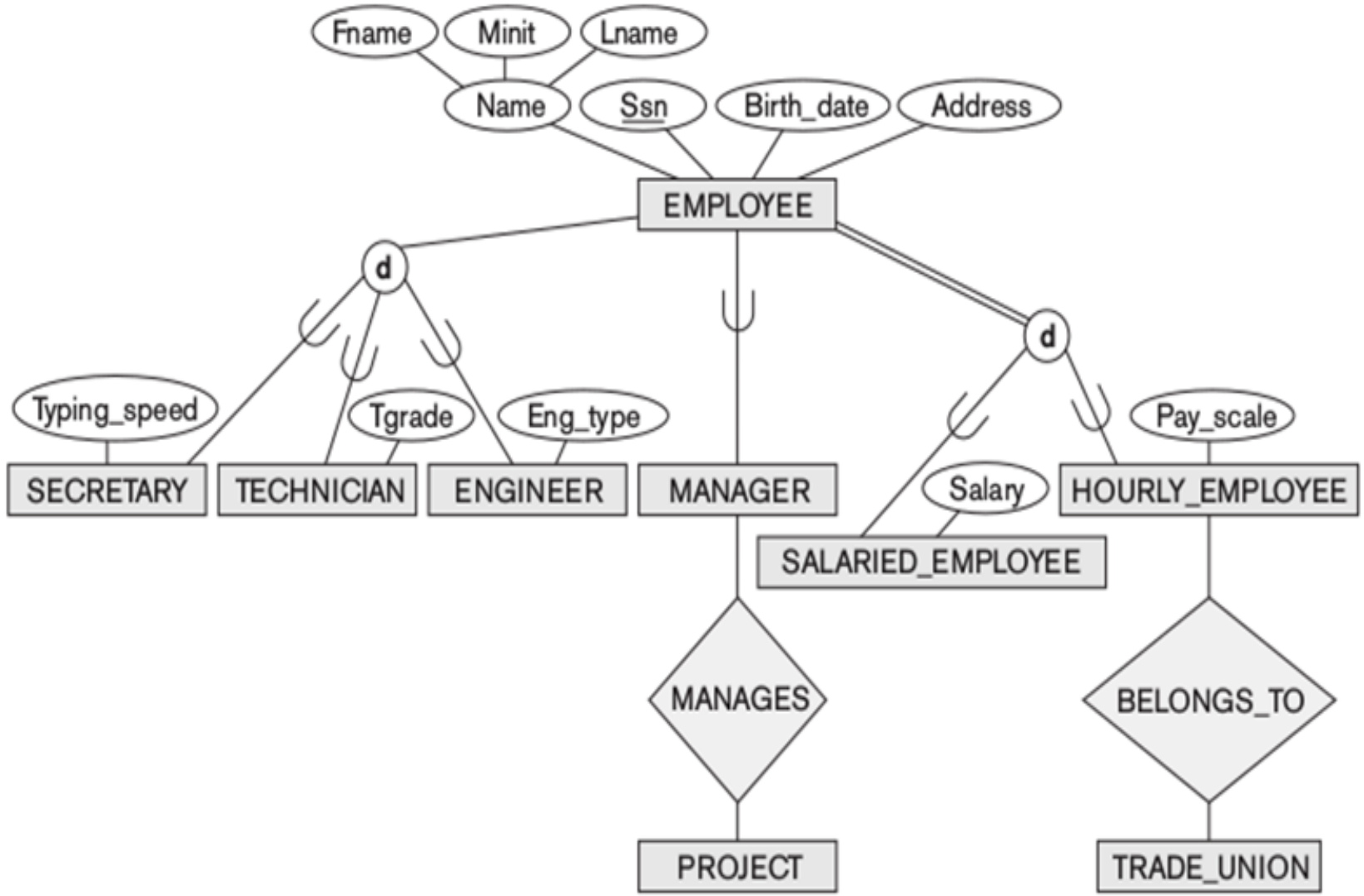
- To handle the complex applications in a better way improvements were made to the existing ER model
- EER diagrams are advanced database diagrams, very similar to regular ER diagrams, which represent requirements and complexities of complex databases
- EER model is a high-level data model that incorporates the extensions to the original ER model

Features of the EER model

- It includes all modeling concepts of the ER model
- EER creates a design more accurate to database schemas
- It reflects the data properties and constraints more precisely
- It is a diagrammatic technique for displaying the Subclass and Superclass, Inheritance; Specialization and Generalization; Union or Category; Aggregation etc.

Subclass, Superclass, and Inheritance

- ***Entity type***: a collection of entities that have the same attributes
- ***Subclass*** of an entity type is a specialized subgroup of entities
  - E.g., EMPLOYEE entity type may be grouped further into MANAGER, SECRETARY, TECHNICIAN, ENGINEER, etc.
- The entities of these belong to the EMPLOYEE entity type and are called subclasses of the EMPLOYEE entity type, and the EMPLOYEE entity type is called the ***superclass*** for each of these subclasses


Fname
Minit
Lname
Name
Ssn
Birth_date
Address
EMPLOYEE
d
d
Typing_speed
Tgrade
Eng_type
Pay_scale
SECRETARY
TECHNICIAN
ENGINEER
MANAGER
Salary
HOURLY_EMPLOYEE
SALARIED_EMPLOYEE
MANAGES
BELONGS_TO
PROJECT
TRADE_UNION

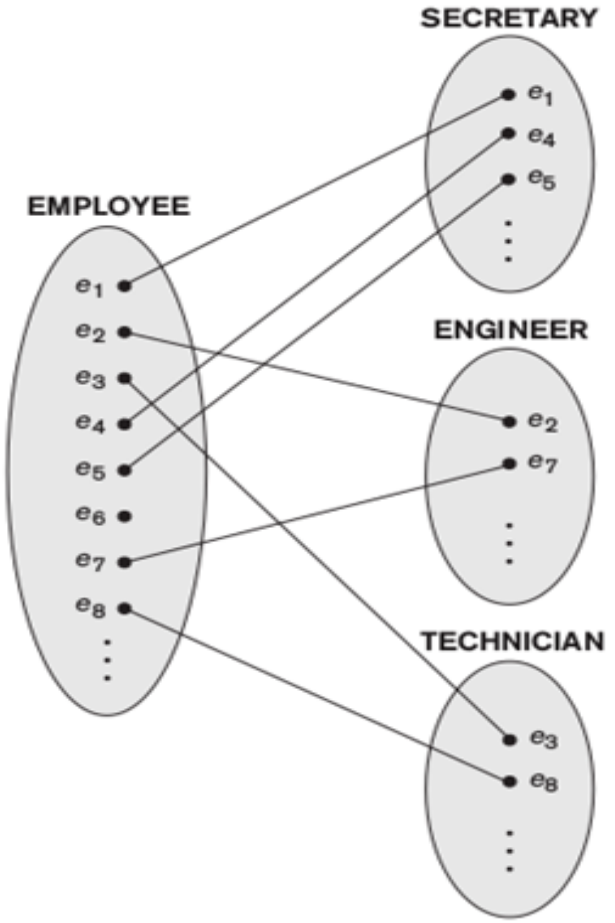
Subclass, Superclass, and Inheritance

- Every entity in any subclass is a member of the Superclass, but it is not necessary that every entity in a superclass is a member of some subclass
- An entity that is a member of a subclass ***inherits*** all the attributes and the relationships in which the superclass participates
- A subclass with its own specific (or local) attributes and relationships together with all the attributes and relationships it ***inherits*** from the superclass, can be considered an entity type in its own right

Relationship b/w Subclass and Superclass

- The relationship between a superclass and any one of its subclasses is called as ***superclass/subclass*** relationship or simply a ***class/subclass*** relationship
  - E.g, EMPLYOEE/MANAGER and EMPLYOEE/TECHNICIAN
- A class/subclass relationship is often called an IS-A (or IS-AN) relationship because of the way we refer to the concept
  - E.g., we say a SECRETARY is an EMPLOYEE, a TECHNICIAN is an EMPLOYEE, and so on

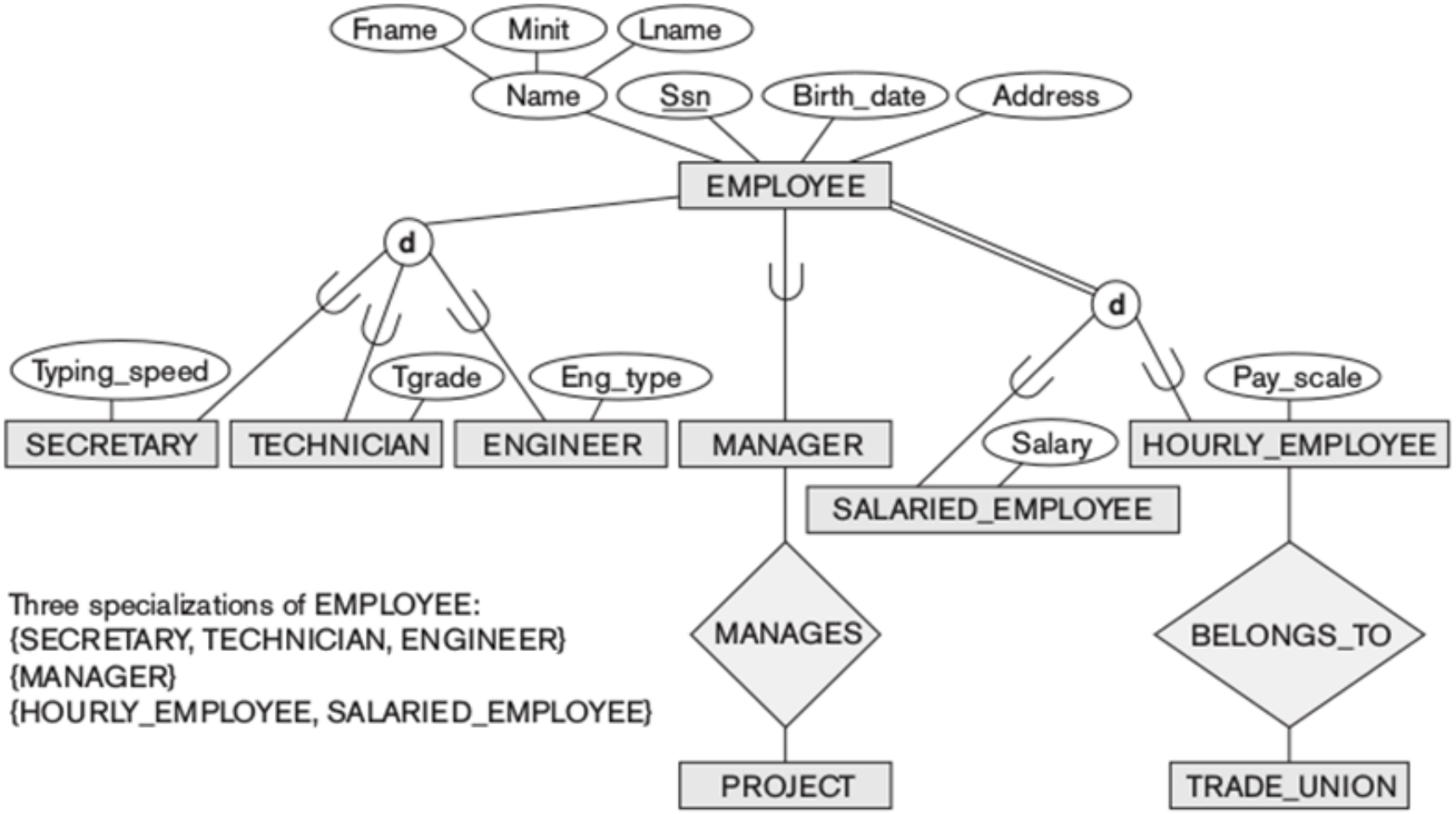
Relationship b/w subclass and superclass



Specialization and Generalization

- ***Specialization*** is the process of defining a set of subclasses of an entity type (i.e., Superclass) based on some characteristics the entities in the superclass
  - E.g., the set of subclasses {SECRETARY, ENGINEER, TECHNICIAN} is a specialization of the superclass EMPLOYEE that distinguishes among employee entities based on the job type of each employee
  - the set of subclasses {SALARIED_EMPLOYEE, HOURLY_EMPLOYEE} ; this specialization distinguishes among employees based on the method of pay

Diagrammatic representation



Why do we need Specialization?

- Certain attributes may apply to some but not all entities of the superclass entity type
  - A subclass is defined in order to group the entities to which these attributes apply
  - The members of the subclass may still share the majority of their attributes with the other members of the superclass
- Some relationship types may be participated in only by entities that are members of the subclass
  - Only HOURLY_EMPLOYEES can belong to a trade union

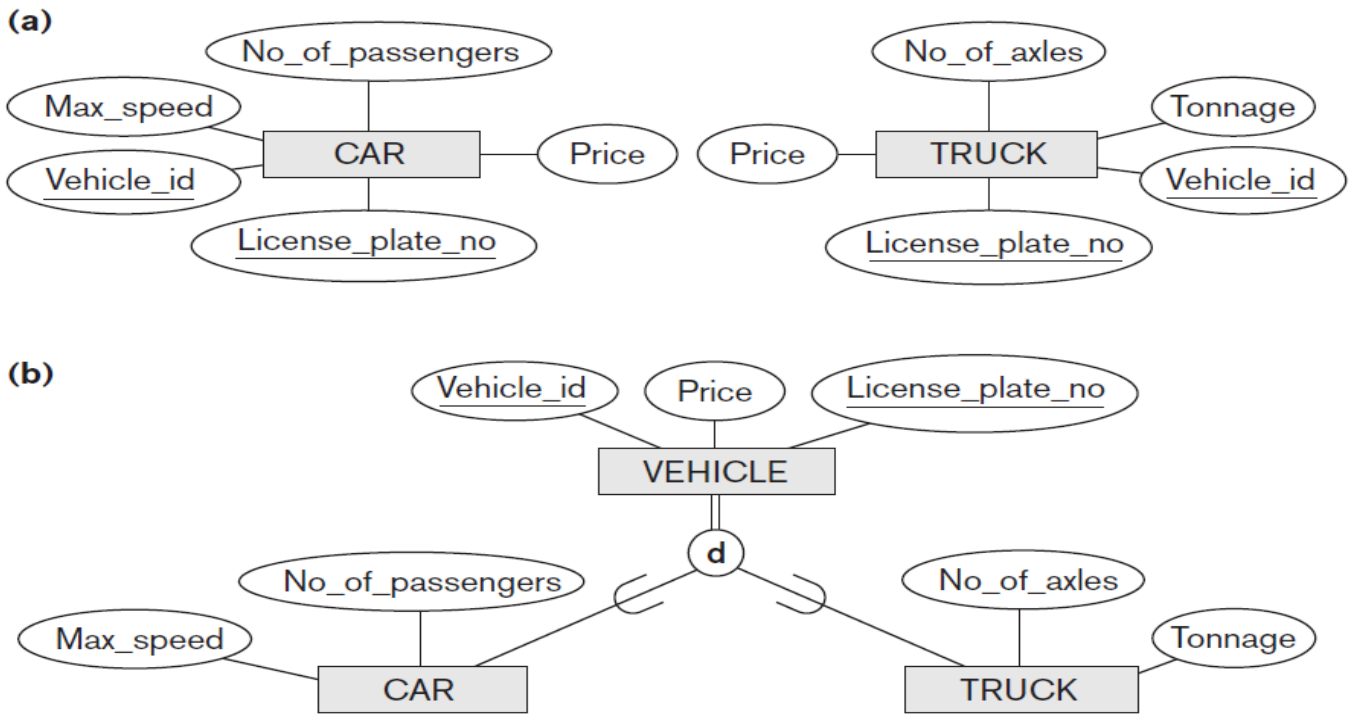
Features of Specialization

The specialization process allows us to do the following

- Define a set of subclasses of an entity type
- Establish additional specific attributes with each subclass
- Establish additional specific relationship types between each subclass and other entity types or other subclasses

Generalization

- ***Generalization*** is a process of abstraction in which we extract common features from a set of entities and create a *generalized single superclass* entity from it



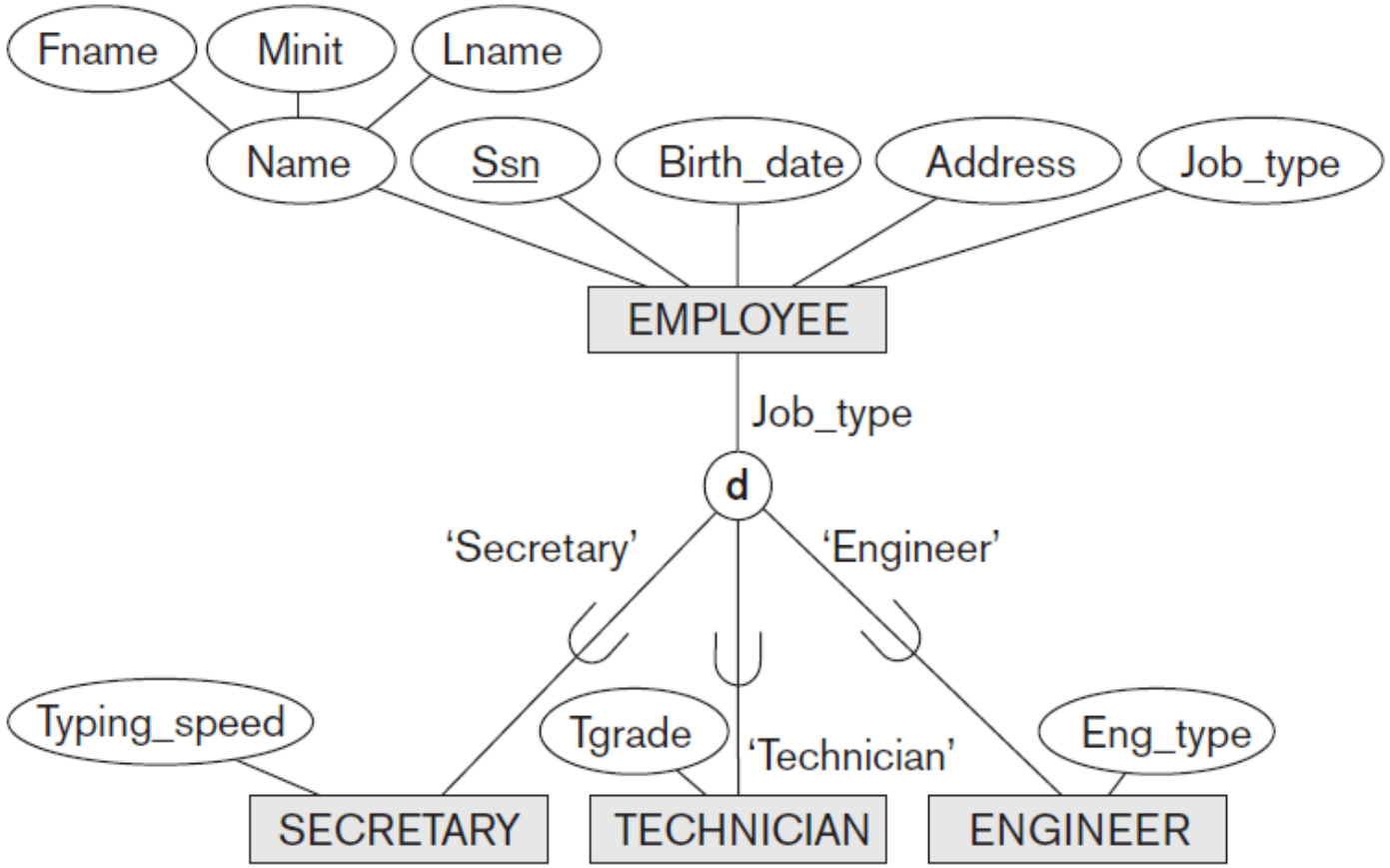
Constraints on Specialization and Generalization

- In general, we may have several specializations defined on the same superclass
- Entities may belong to subclasses in each of the specializations
- A specialization may also consist of a single subclass only
  - E.g., MANAGER subclass in previous example
- ***Condition-defined subclasses*** are subclasses in which we can determine exactly the entities that will become members of each subclass by placing a condition on the value of some attribute of the superclass

1. Predicate-defined constraint

- The condition that we define on the attribute is calles as the ***defining predicate***
- If all subclasses in a specialization have their membership condition on the same attribute of the superclass, the specialization itself is called an ***attribute-defined specialization***, and the attribute is called the ***defining attribute*** of the specialization
- When we do not have a condition for determining membership in a subclass, the subclass is called ***user-defined***
  - Membership in such a subclass is determined by the database users when they apply the operation to add an entity to the subclass

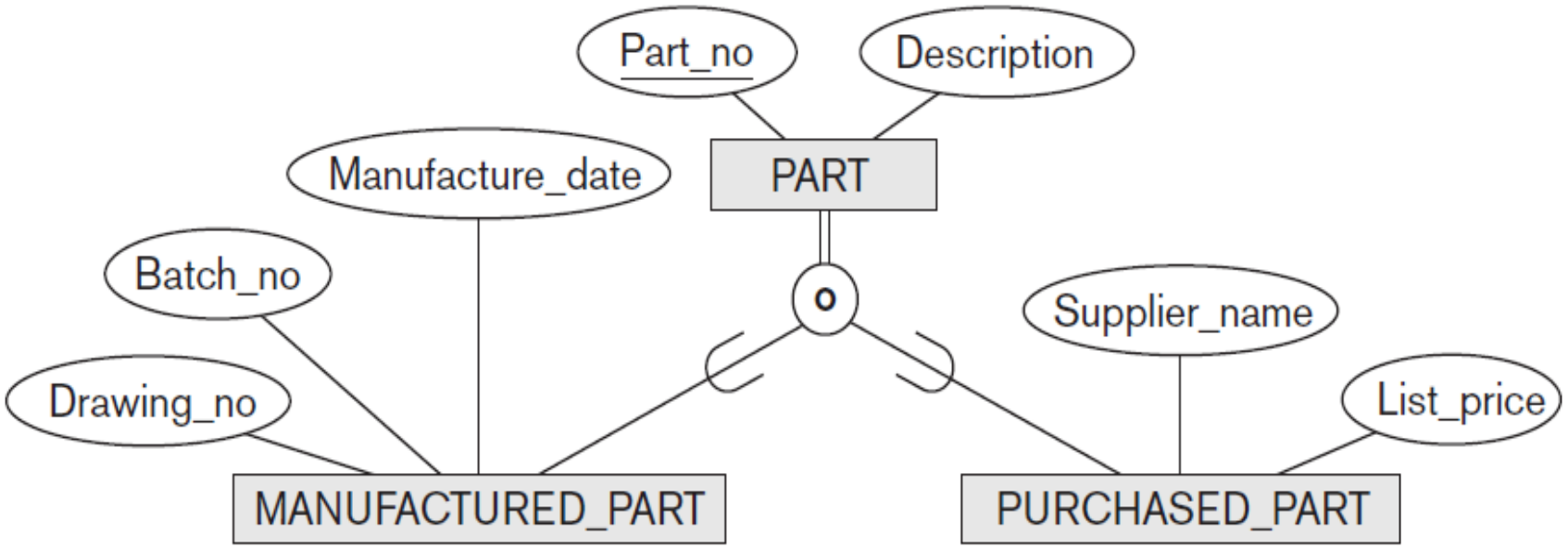
Example


Fname
Minit
Lname
Name
Ssn
Birth_date
Address
Job_type
EMPLOYEE
Job_type
d
'Secretary'
'Engineer'
'Technician'
Typing_speed
Tgrade
Eng_type
SECRETARY
TECHNICIAN
ENGINEER

2. Disjointness constraint

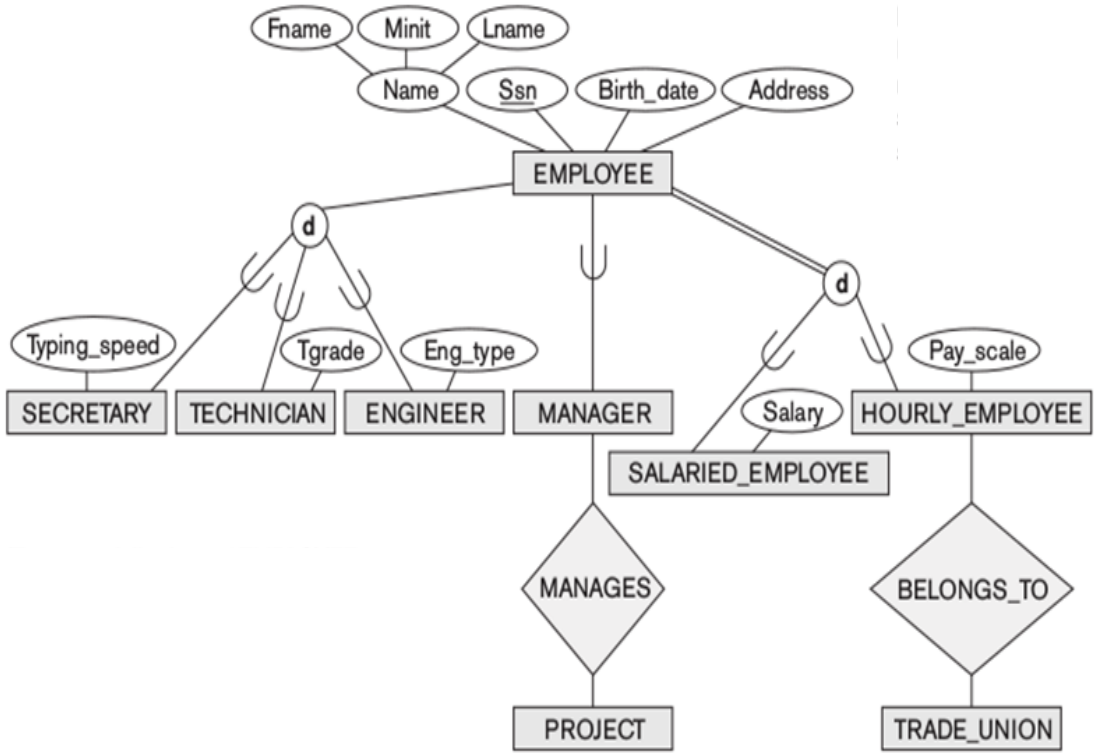
- This constraint specifies that the subclasses of the specialization must be disjoint
- A specialization that is attribute-defined implies the disjointness constraint, if the attribute used to define the membership predicate is single-valued
- In the EER diagram we denote this constraint by (d)
- The **d** notation also applies to user-defined subclasses of a specialization that must be disjoint
- Overlapping subclasses are denoted by ⊙

Example


Part_no
Description
Manufacture_date
PART
Batch_no
o
Supplier_name
Drawing_no
List_price
MANUFACTURED_PART
PURCHASED_PART

3. Completeness constraint

- A ***total specialization*** constraint specifies that every entity in the superclass must be a member of at least one subclass in the specialization
- a ***partial specialization*** constraint allows an entity not to belong to any of the subclasses



Combination of disjoint and completeness

- We have the following four possible constraints on a specialization:

  i. Disjoint and total
  ii. Disjoint and partial
  iii. Overlapping and total
  iv. Overlapping and partial

- The correct constraint is determined from the real-world meaning that applies to each specialization

- In general, a superclass that was identified through the generalization process usually is total

Insertion and deletion rules apply to specialization (and generalization)

- Deleting an entity from a superclass implies that it is automatically deleted from all the subclasses to which it belongs
- Inserting an entity in a superclass implies that the entity is mandatorily inserted in all predicate-defined (or attribute-defined) subclasses for which the entity satisfies the defining predicate
- Inserting an entity in a superclass of a total specialization implies that the entity is mandatorily inserted in at least one of the subclasses of the specialization

Takehome question

- Make a complete list of rules for insertions and deletions for the various types of specializations

Thank you!