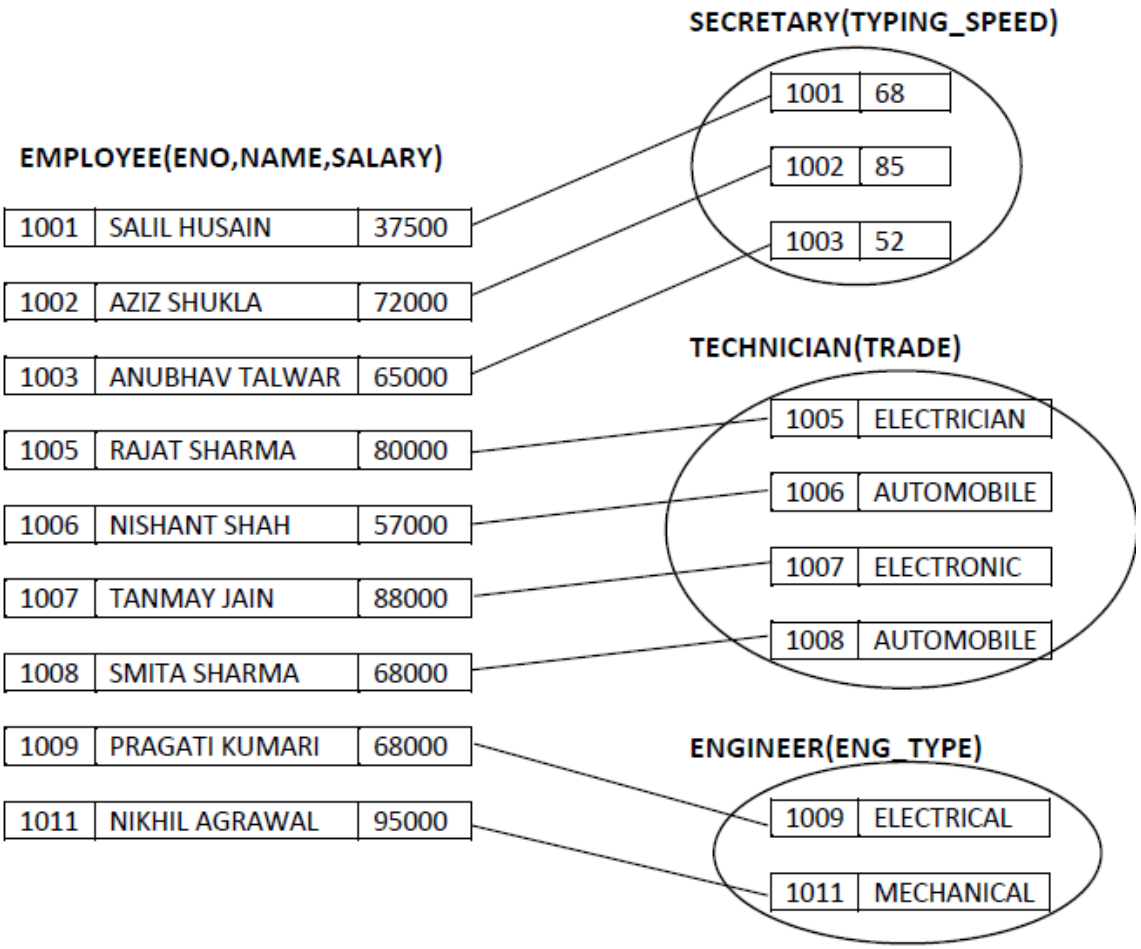
07. Enhanced Entity Relationship Modeling

[Most content has been taken from Elmasri/Navathe book]

Generalization/Specialization

- An entity is specialized type/class of other entity. For example Technician is special Employee; in a university system Faculty is special class of Employee. We call this phenomenon as generalization/specialization. In the example here, Employee is generalized entity class while Technician and Faculty are specialized class of Employee.
- Specialized classes are often called as sub-classes while generalized classes are called super-classes. A sub-class is best understood by "IS-A analysis". Following statements hopefully makes some sense to your mind
  "Technician IS-A Employee", "Laptop IS-A Computer".
- Generalization/specialization is not a very uncommon relationship found in real entities. However this kind of relationships was added later as Enhanced extension to classical ER model.
- There are different variations of this relationship, and have been sketched in following sections. Concepts presented here are primarily sourced from book Elmasri and Navathe.
- Figure below depicts instances of "sub-class" relationships.
- Here we have four sets Employee, Secretary, Technician, and Engineer. Employee is super-class of rest three; set of individual sub-class is subset of Employee set.



- An entity belonging to a sub-class is related with some super-class entity. For instance emp no 1001 is a secretary, and his typing speed is 68. Emp no 1009 is engineer (sub-class) and her trade is “Electrical”, so forth.
- Sub-class entity “inherits” all attributes of super-class; for example employee 1001 will have attributes eno, name, salary, and typing speed.
- ER Diagram below (taken from elmasri/navathe) depicts few examples of sub-classing relationships.

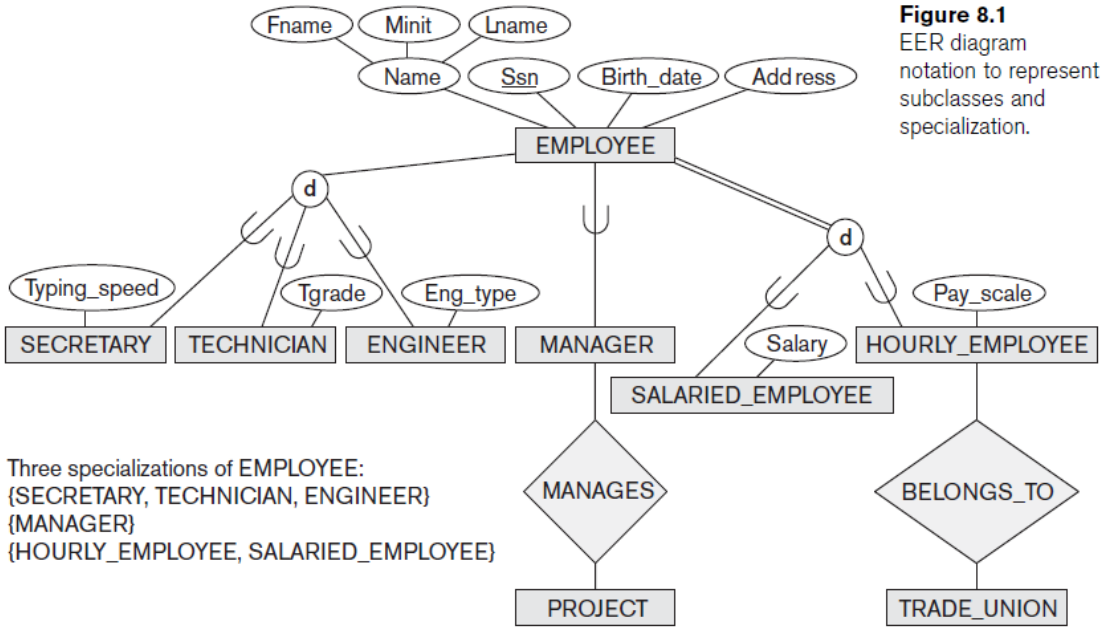


Figure 8.1
EER diagram notation to represent subclasses and specialization.

- Constraints on “Sub-class” relationship: there are of two types:

    - Total or Partial: A sub-classing relationship is total if every super-class entity is to be associated with some sub-class entity, otherwise partial. Sub-class “job type based employee category” is partial sub-classing – not necessary every employee is one of (secretary, engineer, and technician), i.e. union of these three types is proper subset of all employees. Whereas other sub-classing “Salaried Employee AND Hourly Employee” is total; union of entities from sub-classes is equal to total employee set, i.e. every employee necessarily has to be one of them.

    - Overlapped or Disjoint: If an entity from super-set can be related (can occur) in multiple sub-class sets, then it is overlapped sub-classing, otherwise disjoint. Both the examples: job-type based and salaries/hourly employee sub-classing are disjoint.

      Example of overlapping is not shown in diagram above; however consider example of movie crew has sub-classes cast, director, composer, and one crew member can be belonging to more than one sub-class – overlapped sub-classing. Note the letter (d) that represents disjoint sub-classing; for overlapping (o) is placed instead. Example: page#6.

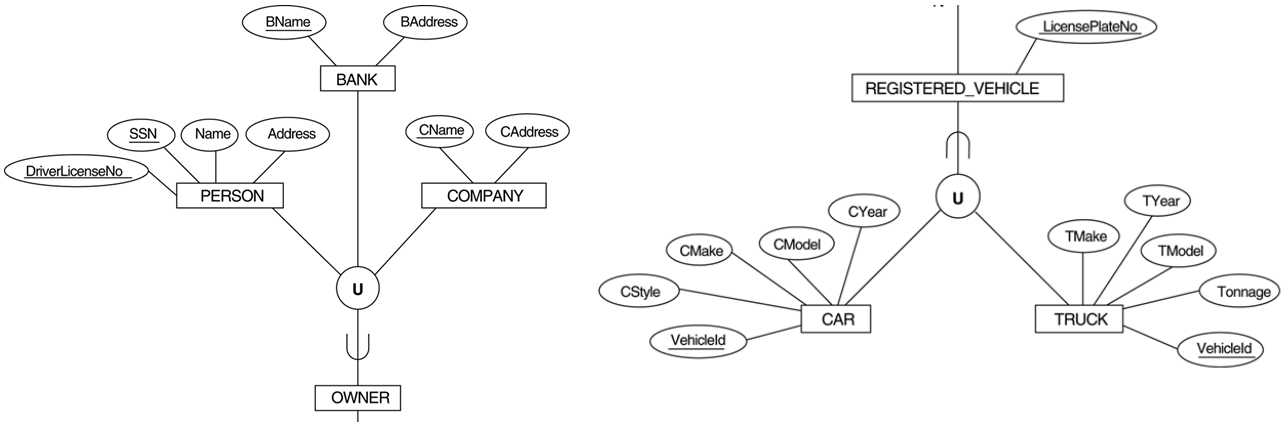
- Note that both above constraints are independent of each other: can be overlapped and total or partial or disjoint total and partial.
- Also sub-classing has transitive property.

Multiple Inheritance (sub-class of multiple super classes)

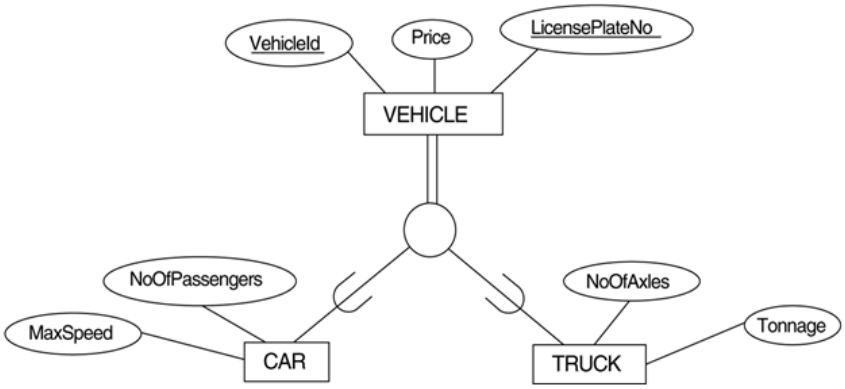
- An entity can be sub-class of multiple entity types; such entities are sub-class of multiple entities and have multiple super-classes; Teaching Assistant can subclass of Employee and Student both. A faculty in a university system can be sub-class of Employee and Alumnus both.
- In multiple inheritance, attributes of sub-class is union of attributes of all super-classes.

UNION

- Set of Libray Members is UNION of Faculty, Student, and Staff.
- A union relationship indicates either of type; for example: a library member is either Faculty or Staff or Student.
- Below are two examples from elmasri/navathe and shows how UNION can be depicted in ERD – Vehicle Owner is UNION of PERSON and Company, and RTO Registered Vehicle is UNION of Car and Truck.



- You might see some confusion in Sub-class and UNION; consider example in above figure in right – Vehicle is super-class of CAR and Truck; this is very much the correct example of sub-class as well but here use it different we are saying RTO Registered vehicle is UNION of Car and Vehicle, they do not inherit any attribute of Vehicle, attributes of car and truck are altogether independent set, where is in sub-classing situation car and truck would be inheriting the attribute of vehicle class. Below is Vehicle as modeled as class of Car and Truck-



  Here are two more confusions for your mind:
  - Employee set is union of secretary, technician, and engineer (or they sub-class of employee)?
  - Student, Staff and Faculty are sub-class of Library Member?

  Bottom line is in UNION you do not inherit attributes, different sets forming union have independent set of attributes.

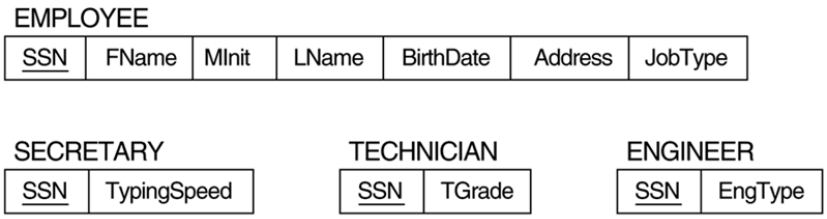
- You might also find confusion between Multiple Inheritance and UNION; Do you see in the left example in the figure above, Owner as Multiply inherited class of Person, Bank, and Company? But it is not; being multiply inherited mean, owner has all the attributes of Person and Company, where as in this case a owner entity is either of one, attributes of owner has attributes either of them. In multiple inheritance both sets of attributes are combined using AND, where as in UNION both sets of attributes are combined using OR.

EER to relational mapping rules

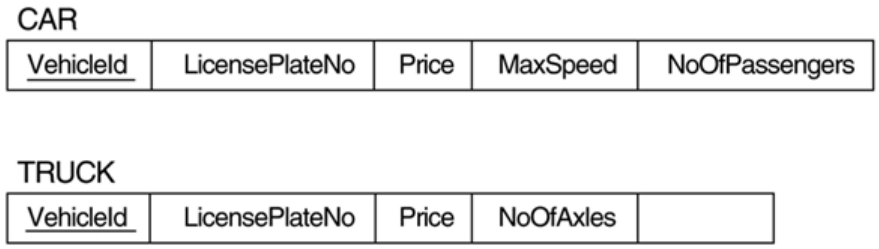
Generalization/Specialization

For sub-class relationships we use of one of following strategies, that are primarily derived from strategies discussed in text of elamsri/navathe.

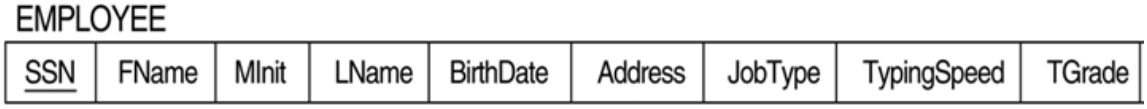
Option1: Supper we have relation for parent entity and for all sub-classes. Each sub-class has FK referring to parent entity relation. Example. Below is Relations derived from example …



- Option 2: We have relations only for sub-classes, and super-class attributes are repeated in all sub-class relations. Note that is only suitable when we have "total" and "disjoint" sub-classing, otherwise there is no room for entities that do not belong to any of sub-class.



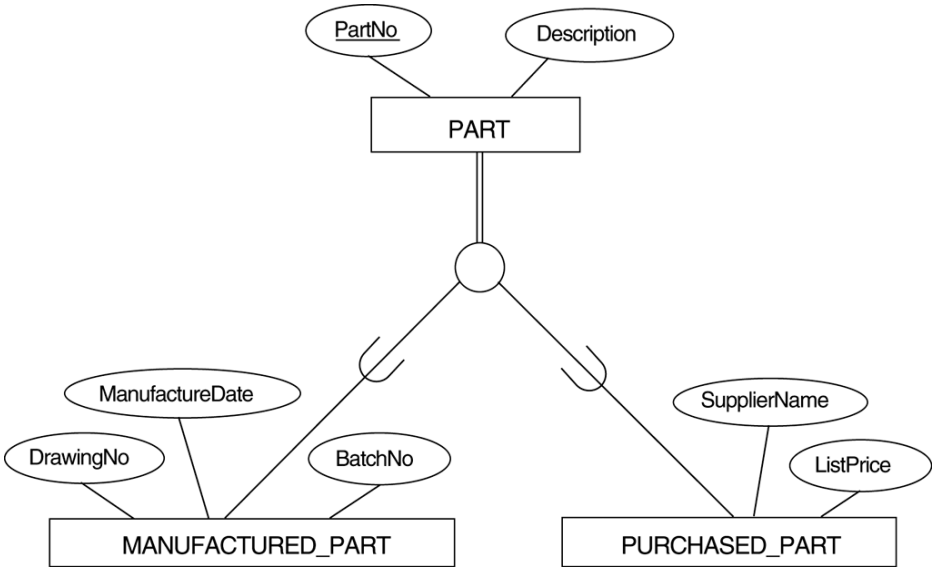
- Option3: All super and subclasses are merged in one, and have class-name as an additional attribute. Example-



This option is good for specializations whose subclasses are disjoint. However, this option has potential for generating a large number of null values, if many attributes appear in sub-classes (tuple can belong to only one sub-class and attributes of others will be null).

This option has benefit of avoiding JOINs.

- Option4: Again have single relation for all. This option is used to handle overlapping subclasses by including m Boolean type fields, one for each subclass.

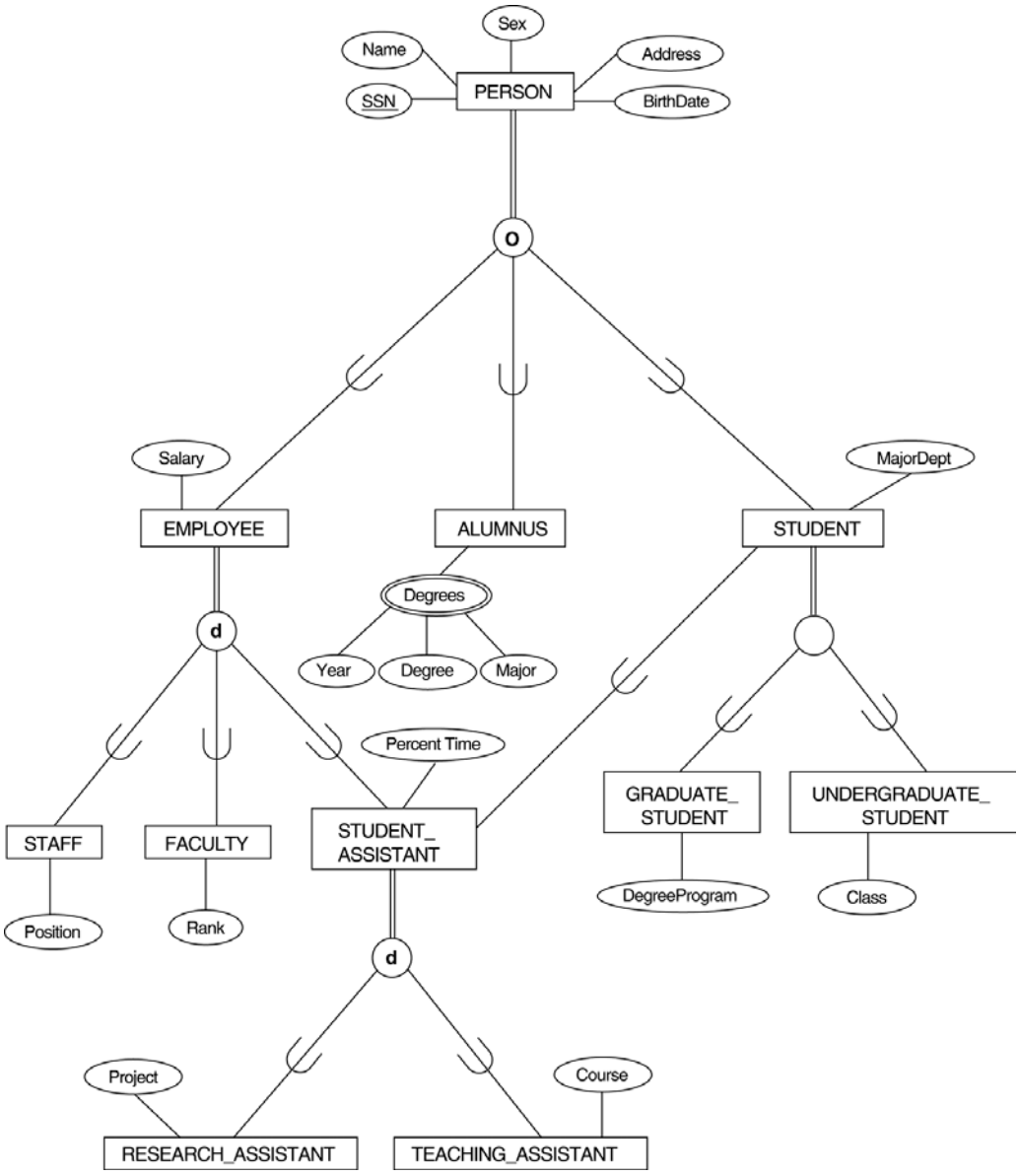


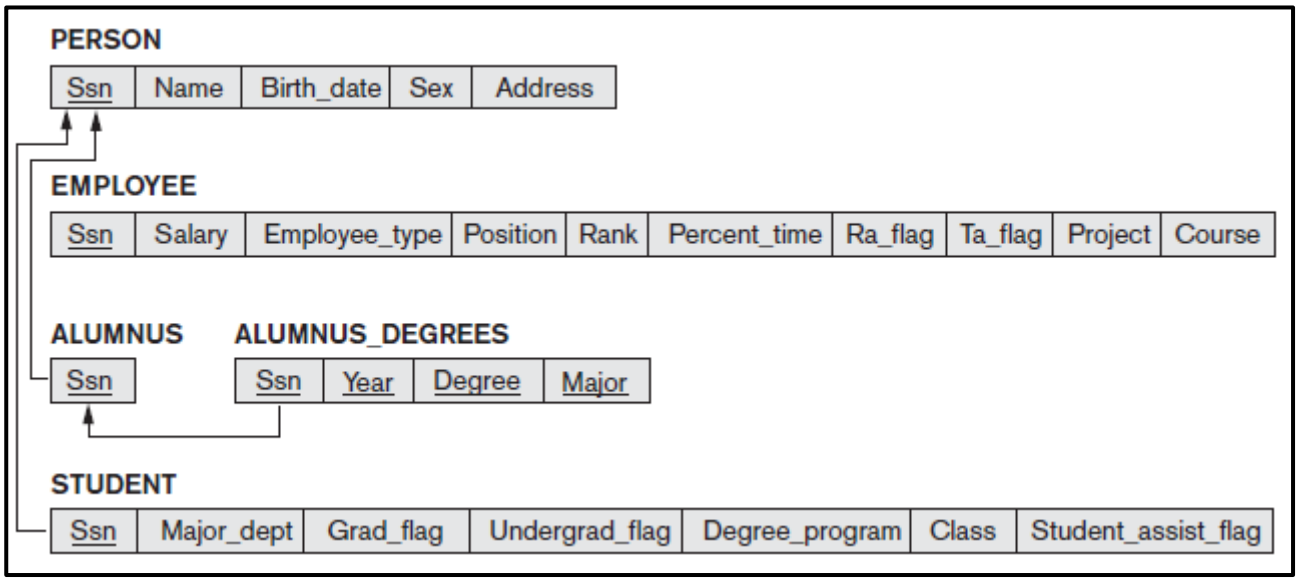
PART

PartNo	Description	MFlag	DrawingNo	ManufactureDate	BatchNo	PFlag	SupplierName	ListPrice

<---------Manufactured Part --------> <---- Purchased Part --->

- For Multiple-Inheritance. Any of the combination can be used; for employee let us say option 3, and option 4 for student and sub-classes.





- For UNION:
  - In the example of Library member, we can have a LibMemID as FK referencing to LibMember in all aggregates.
  - Another example is below for other situation-