DBMS >

ER Model-3

Enhanced ER diagram

EER diagram stands for **Enhanced Entity Relationship Diagram** which includes all the concept of ER model. In addition, it includes the concepts of subclass, and superclass and the related concepts of **specialization** and **generalization**. Another concept included in the **EER model** is that of a category or **union** type, which is used to represent a collection of objects that is the union of objects of different entity types. Associated with these concepts is the important mechanism of attribute and relationship inheritance. We also describe a diagrammatic technique for displaying these concepts when they arise in an EER schema. We call the resulting schema diagrams **enhanced ER or EER diagrams**.

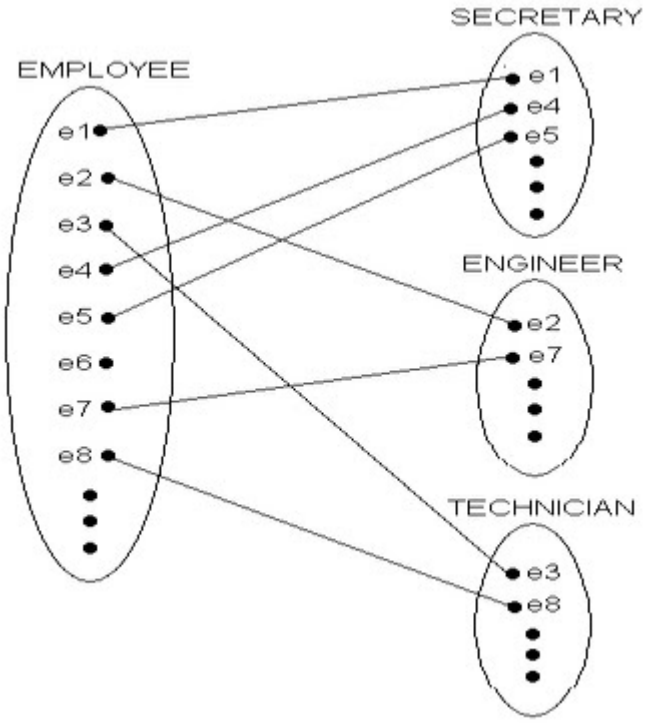
Subclass and Super Class

We have discussed that en entity type is used to represent both a type of entity and the entity set or collection of entities of that type that exist in the database. In many cases an entity type has numerous sub groupings of its entities that are meaningful and need to be represented explicitly because of their significance to the database application. For example: - the entities that are members of the EMPLOYEE entity type may be grouped further into SECRETARY, ENGINEER, MANAGER, TECHNICIAN, SLARIED_EMPLOYEE, HOURLY_EMPLOYEE, and so on. The set of entities in each of the latter groupings is a subset of the entities that belongs to the EMPLOYEE entity set, meaning that every entity that is a member of one of these sub groupings is also an employee. We call each of these sub groupings a subclass of the EMPLOYEE entity type, and the EMPLOYEE entity type is called the super class for each of these subclasses.

Specialization

Specialization is the process of defining a set of subclasses of an entity type. This entity type is called the super class of the specialization. The set of subclasses that form a specialization is defined on the basis of some distinguishing characteristic of the entities in the super class. For example the set of subclasses {SECRETARY, ENGINEER, TECHNICIAN} is a specialization of the super class EMPLOYEE that distinguishes among employee entities based on the job type of each employee entity. We have several specializations of the same entity type based on different distinguishing characteristic.

A crucial property of higher and lower level entities created by specialization a generalization is attribute inheritance. The attributes of higher level entity set are said to be inherited by the lower level entity set



Generalization

We can think over the reverse process of abstraction I which we suppress the difference among several entities types, and identify their common features and generalize them into a single super class of which the original entity types are special subclasses.In the above example there are two entity types CAR and TRUCK and figure C show the generalization of both entity types into the super class VEHICLE

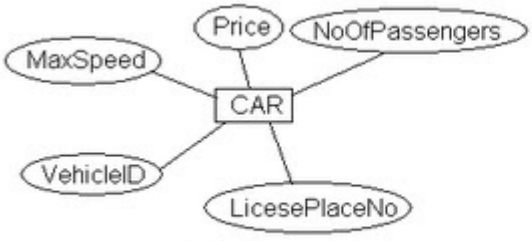


figure A

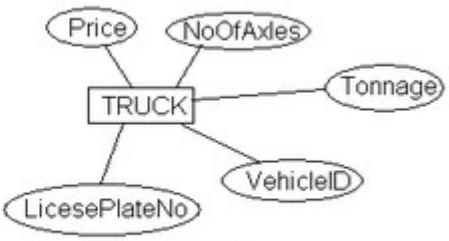


figure B

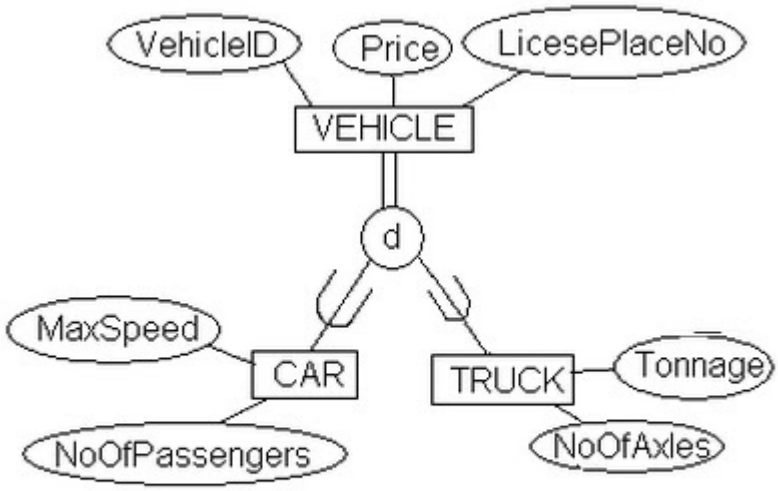


figure C

Constrains on specialization and generalization

Condition – defined / predicate defined specialization

In some specialization we can determine exactly the entities that will become member of each subclass by placing a condition on the value of some attribute of the super class. Such subclasses are called **predicate defined (or condition defined) subclasses**. For example, if the EMPLOYEE entity type has an attribute job type, as shown in the following figure, we can specify the condition of membership in the SECRETARY subclass by the condition (job type ='secretary'), which we call the **defining predicate** of the subclass. This condition is a constraint specifying that exactly those entities of the EMPLOYEE entity type whose attribute value for the job type is 'secretary' belong to the subclass. We display a predicate – defined subclass by writing the predicate condition next to the line that connects the subclass to the specialization circle.

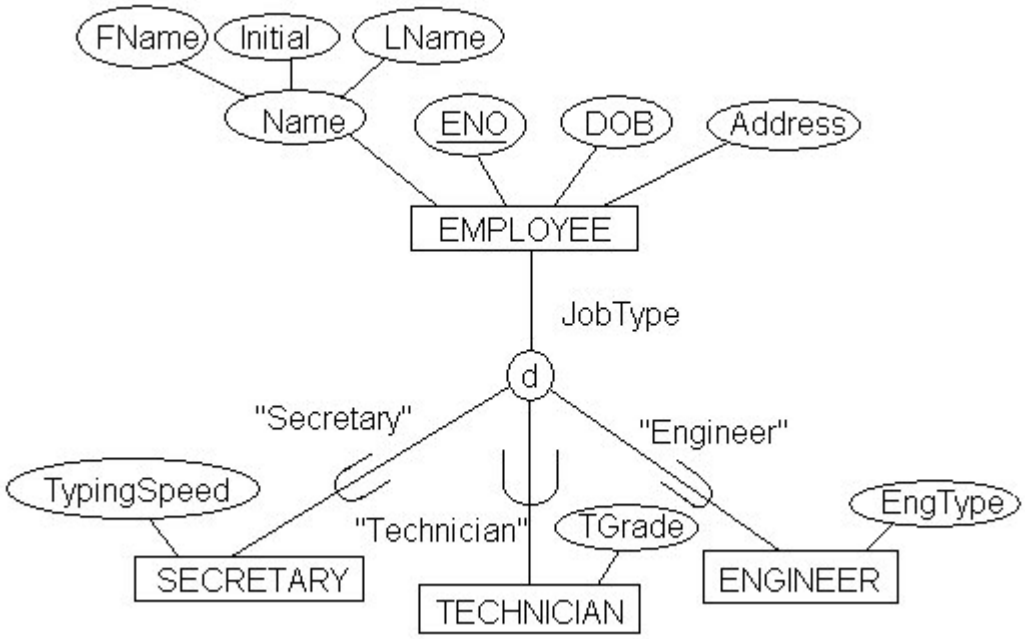
Attribute defined specialization

If all the subclasses in a specialization have their membership condition on the same attribute of the super class, the specialization itself is called an **attribute defined specialization**, and the attribute is called defining attribute of the specialization. We display an attribute defined specialization by placing the defining attribute name next to the arc from the circle of the super class as shown in the figure.

User defined specialization

When we do not have a condition for determining membership in a subclass, the subclass is called **user defined specialization**. Membership in such a subclass is determined by the database user when they apply the operation to add an entity to the

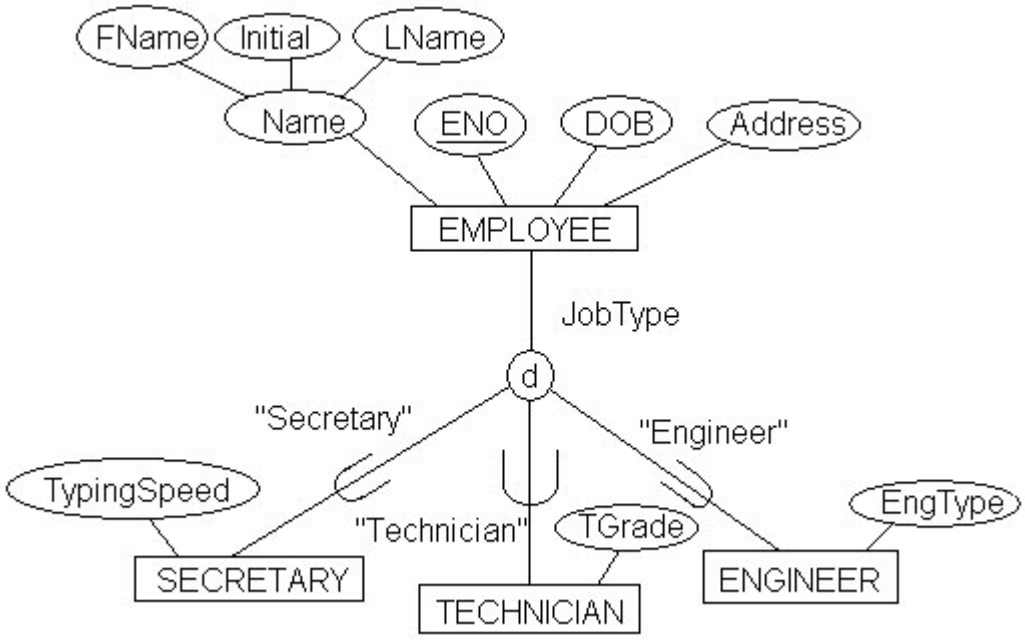
subclass. Membership is specified individual for each entity by the user, not by the condition that may be evaluated automatically.



Two other constraints are also there which may apply to a specialization. They are as follows

Disjoint constraint

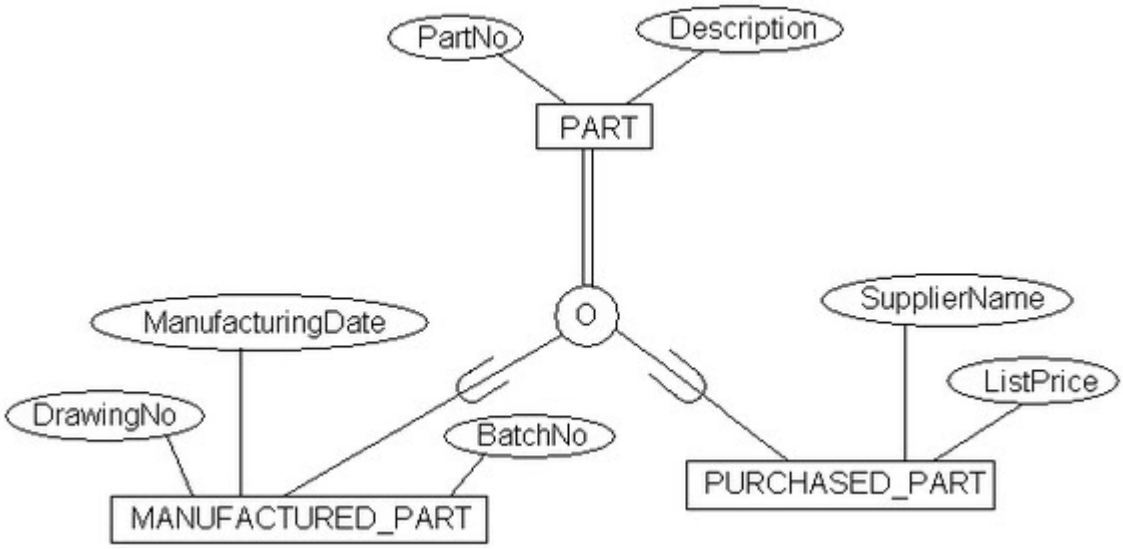
This constraint specifies that the subclasses of the specialization must be disjoint. This means that an entity can be a member of at most one of the subclasses of the specialization. A specialization that is attribute-defined implies the disjoint ness constraint if the attribute used to define the membership predicate is single valued. It is displayed by placing **d** in the circle



Overlap constraint

This constraint specifies that the subclasses of the specialization are not constrained to

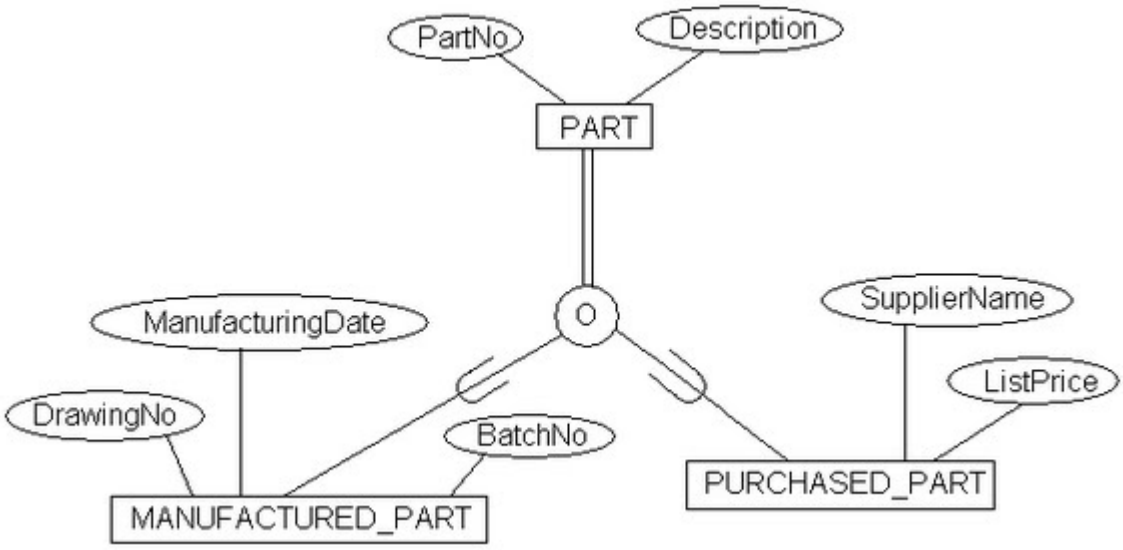
be disjoint. This means that an entity can be a member of more then one subclasses of the specialization. This case, which is the default, is displayed by placing **O** in the circle.



The second constraints on the specialization are called **completeness constraint**, which may be **total or partial**.

Total specialization

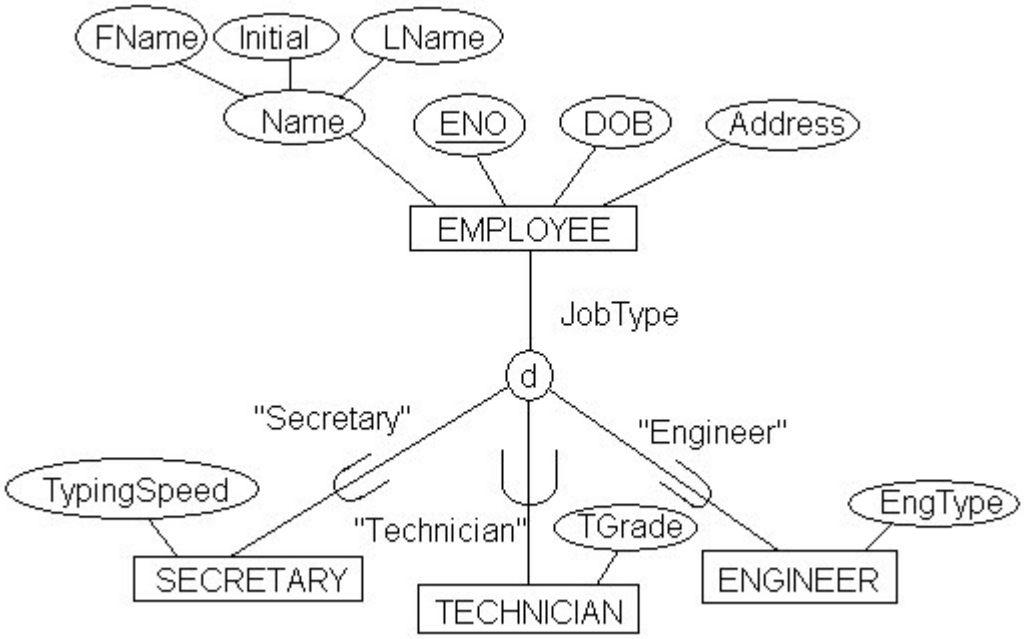
A total specialization constraint specifies that every entity in the super class must be the member of at least one subclass in the specialization. For example if every employee must be either an HOURLY_EMPLOYEE or SALARIED_EMPLOYEE, then the specialization {HOURLY_EMPLOYEE, SALARIED_EMPLOYEE} is a total specialization of EMPLOYEE. This is shown in the EER diagram by using a double line to connect the subclass to the circle.



Partial specialization

If an EMPLOYEE entity does not belong to any of subclasses, the specialization is called partial specialization which is shown by single line. For example some EMPLOYEE entities do not belong to any of the subclasses {SECRETARY, ENGINEER, TECHNICIAN}

then that specialization is partial.



Notice: that disjoint ness and completeness constraints are independent. Hence we have the following four possible constraints on specialization

1. Disjoint, total
2. Disjoint partial
3. Overlapping, total
4. Overlapping, partial

In general a super class that was identified through the generalization process usually is total, because the super class is derived from the subclasses and hence contains only the entities that are in the subclass.

Certain insertion and deletion rules apply to specialization and generalization as a consequence of the constraints specified earlier. Some of these rules are as follows

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