The disjoint and participation constraints of specialization and generalization are distinct, giving rise to four categories: "mandatory and disjoint," "optional and disjoint," "mandatory and nondisjoint," and "optional and nondisjoint."

13.1.7 Worked Example of using Specialization/Generalization to Model the Branch View of the *DreamHome* Case Study

The database design methodology described in this book includes the use of specialization/generalization as an optional step (Step 1.6) in building an EER model. The choice to use this step is dependent on the complexity of the enterprise (or part of the enterprise) being modeled and whether using the additional concepts of the EER model will help the process of database design.

In Chapter 12 we described the basic concepts necessary to build an ER model to represent the Branch user views of the *DreamHome* case study. This model was shown as an ER diagram in Figure 12.1. In this section, we show how specialization/generalization may be used to convert the ER model of the Branch user views into an EER model.

As a starting point, we first consider the entities shown in Figure 12.1. We examine the attributes and relationships associated with each entity to identify any similarities or differences between the entities. In the Branch user views' requirements specification there are several instances where there is the potential to use specialization/generalization as discussed shortly.

(a) For example, consider the Staff entity in Figure 12.1, which represents all members of staff. However, in the data requirements specification for the Branch user views of the *DreamHome* case study given in Appendix A, there are two key job roles mentioned, namely Manager and Supervisor. We have three options as to how we may best model members of staff. The first option is to represent all members of staff as a generalized Staff entity (as in Figure 12.1), the second option is to create three distinct entities Staff, Manager, and Supervisor, and the third option is to represent the Manager and Supervisor entities as subclasses of a Staff superclass. The option we select is based on the commonality of attributes and relationships associated with each entity. For example, all attributes of the Staff entity are represented in the Manager and Supervisor entities, including the same primary key, namely staffNo. Furthermore, the Supervisor entity does not have any additional attributes representing this job role. On the other hand, the Manager entity has two additional attributes: mgrStartDate and bonus. In addition, both the Manager and Supervisor entities are associated with distinct relationships, namely Manager *Manages* Branch and Supervisor *Supervises* Staff. Based on this information, we select the third option and create Manager and Supervisor subclasses of the Staff superclass, as shown in Figure 13.5. Note that in this EER diagram, the subclasses are shown above the superclass. The relative positioning of the subclasses and superclass is not significant, however; what is important is that the specialization/generalization triangle points toward the superclass.

The specialization/generalization of the Staff entity is optional and disjoint (shown as {Optional, Or}), as not all members of staff are Managers or Supervisors, and in addition a single member of staff cannot be both a Manager

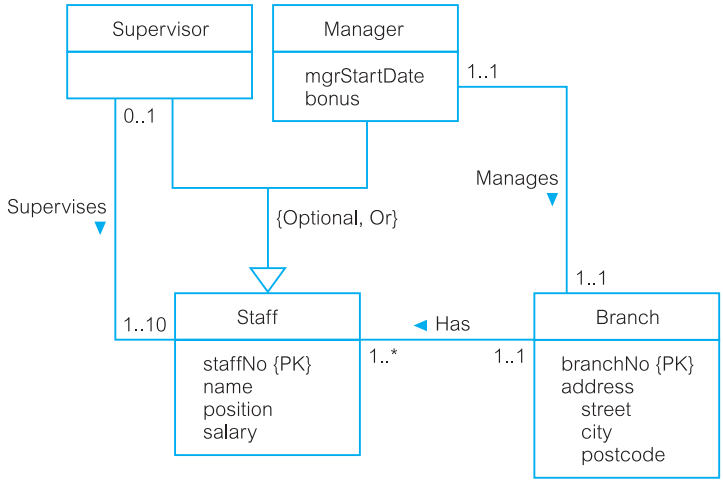

Figure 13.5
Staff superclass with Supervisor and Manager subclasses.

and a Supervisor. This representation is particularly useful for displaying the shared attributes associated with these subclasses and the Staff superclass and also the distinct relationships associated with each subclass, namely Manager *Manages* Branch and Supervisor *Supervises* Staff.

(b) Consider for specialization/generalization the relationship between owners of property. The data requirements specification for the Branch user views describes two types of owner, namely PrivateOwner and BusinessOwner as shown in Figure 12.1. Again, we have three options as to how we may best model owners of property. The first option is to leave PrivateOwner and BusinessOwner as two distinct entities (as shown in Figure 12.1), the second option is to represent both types of owner as a generalized Owner entity, and the third option is to represent the PrivateOwner and BusinessOwner entities as subclasses of an Owner superclass. Before we are able to reach a decision we first examine the attributes and relationships associated with these entities. PrivateOwner and BusinessOwner entities share common attributes, namely address and telNo and have a similar relationship with property for rent (namely PrivateOwner *POwns* PropertyForRent and BusinessOwner *BOwns* PropertyForRent). However, both types of owner also have different attributes; for example, PrivateOwner has distinct attributes ownerNo and name, and BusinessOwner has distinct attributes bName, bType, and contactName. In this case, we create a superclass called Owner, with PrivateOwner and BusinessOwner as subclasses, as shown in Figure 13.6.

The specialization/generalization of the Owner entity is mandatory and disjoint (shown as {Mandatory, Or}), as an owner must be either a private owner *or* a business owner, but cannot be both. Note that we choose to relate the Owner superclass to the PropertyForRent entity using the relationship called Owns.

The examples of specialization/generalization described previously are relatively straightforward. However, the specialization/generalization process can be taken further as illustrated in the following example.

(c) There are several persons with common characteristics described in the data requirements specification for the Branch user views of the *DreamHome* case

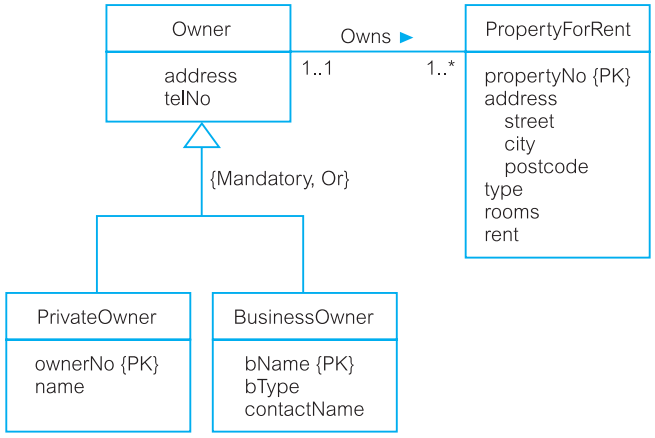

Figure 13.6
Owner superclass with PrivateOwner and BusinessOwner subclasses.

study. For example, members of staff, private property owners, and clients all have number and name attributes. We could create a Person superclass with Staff (including Manager and Supervisor subclasses), PrivateOwner, and Client as subclasses, as shown in Figure 13.7.

We now consider to what extent we wish to use specialization/generalization to represent the Branch user views of the *DreamHome* case study. We decide to use the specialization/generalization examples described in (a) and (b) above but not (c), as shown in Figure 13.8. To simplify the EER diagram only attributes

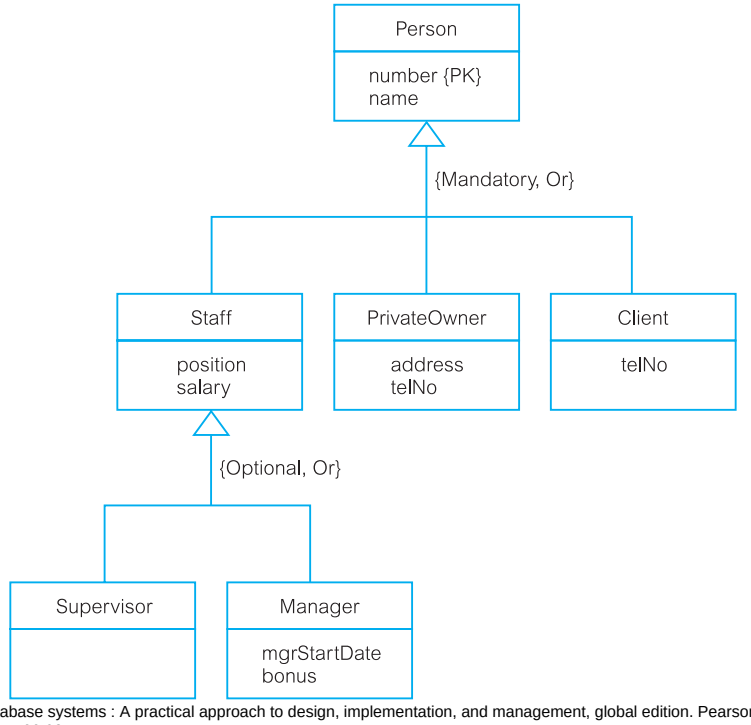

Figure 13.7
Person superclass with Staff (including Supervisor and Manager subclasses), PrivateOwner, and Client subclasses.

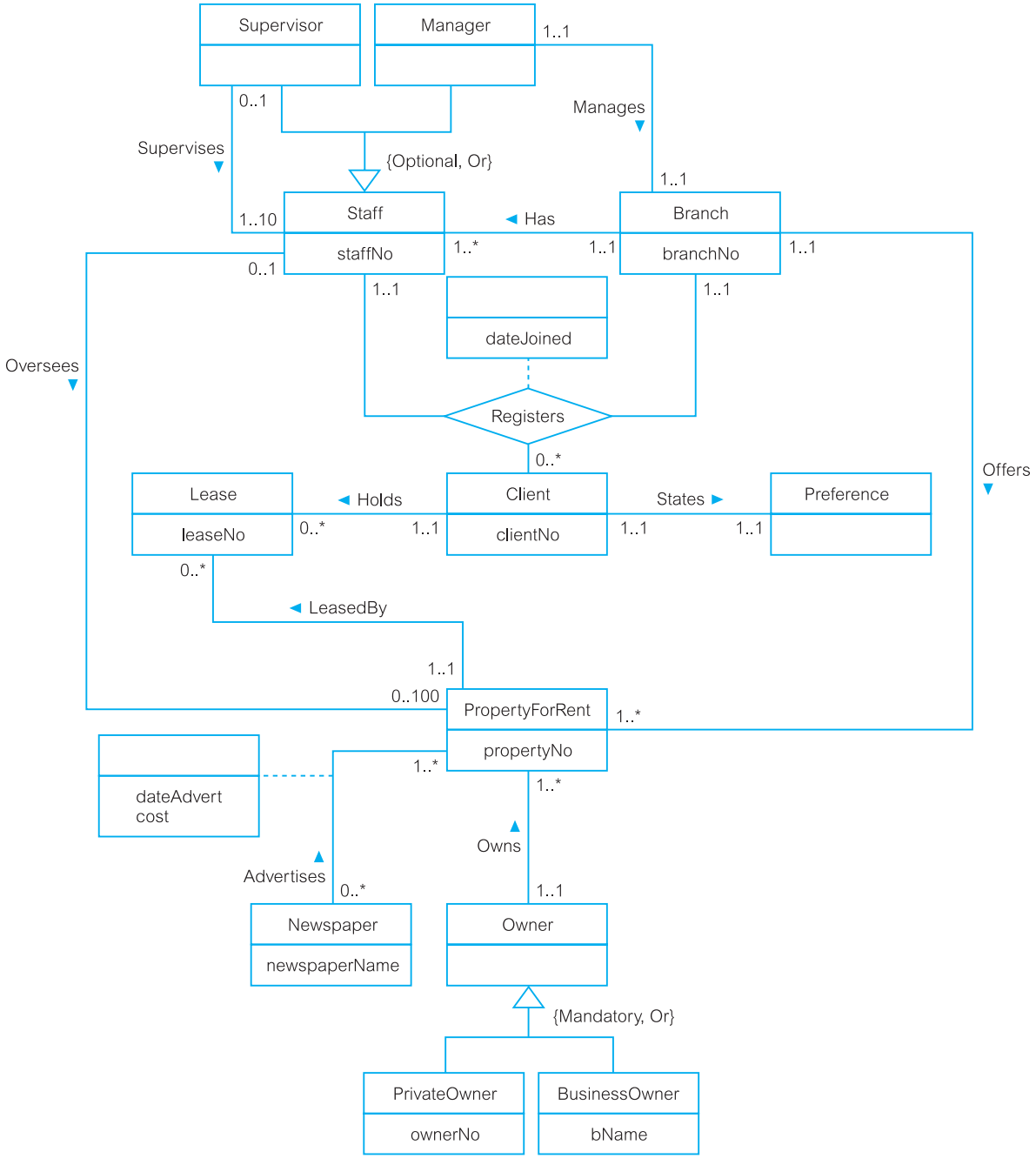

Figure 13.8 An EER model of the Branch user views of *DreamHome* with specialization/generalization.