Chapter III

Using Object-Oriented Features

In Chapter II, we discussed the different features available in Oracle™ that can be used to implement an object-oriented model. We will use those features in this chapter. The discussion in this chapter will be categorized based on the relationship types.

There are three distinct relationship types that we have to consider in object-oriented modeling for implementation in object-relational databases: inheritance, association, and aggregation. Some manipulations will be needed in order to accommodate the features of these relationships.

Using Inheritance Relationships

The concept of inheritance, where an object or a relation inherits the attribute (and methods) of another object, is not supported in the older versions of Oracle™ (prior to Oracle™ 9). The implementation of an inheritance relationship is established using primary-key and foreign-key relationships (shared ID) in order to simulate the relationship between a superclass and its subclasses.
**Union Inheritance Implementation**

Figure 3.1 shows an inheritance relationship of union type. It declares that the union of a group of subclasses constitutes the entire membership of the superclass. In a union inheritance, we know that every object in the superclass is an object of at least one of the subclasses. In the example (see Figure 3.1), the union type does not preclude a member of a subclass from being a member of another subclass. For example, a person who is a staff member may also be a student at that university.

In order to simulate the union inheritance, Student and Staff will carry the primary key of the superclass, Person, in their relational tables. The primary key of the superclass becomes a foreign key in the subclasses. The foreign keys in the subclasses are also their primary keys. It becomes the main difference between the primary-key and foreign-key relationships in association and in inheritance. Thus, in Figure 3.1 it is noted that the primary key of Person is also the primary key of both Student and Staff. At the same time, the constraint of the primary-key and foreign-key relationship between the ID attributes in Student and Staff and the ID in Person is maintained in order to make sure that each student and staff is also a person. Thus, we have to specify the referential integrity constraint every time we want to manipulate the target of a foreign-key reference.

If we use the newer Oracle™ version, which supports inheritance using the “under” keyword, we can create Student and Staff subclasses under the superclass Person. The implementation is shown in Figure 3.3. Note that for union inheritance, we need to create one table each for the superclass and all the subclasses. As can be seen in the later sections, this union inheritance has a different way of implementation compared with other inheritance types. Using

---

**Figure 3.1. Union inheritance**

![Diagram](image-url)