MODELING RELATIONSHIPS

Relationships are the glue that holds together the various components of an E-R model. Intuitively, a *relationship* is an association representing an interaction among the instances of one or more entity types that is of interest to the organization. Thus, **a relationship has a verb phrase name**. Relationships and their characteristics (degree and cardinality) represent business rules.

To understand relationships more clearly, we must distinguish between relationship types and relationship instances. To illustrate, consider the entity types EMPLOYEE and COURSE, where COURSE represents training courses that may be taken by employees. To track courses that have been completed by particular employees, we define a relationship called Completes between the two entity types (see Figure 2-10a). This is a many-to-many relationship, because each employee may complete any number of courses (zero, one, or many courses), whereas a given course may be completed by any number of employees (nobody, one employee, many employees). For example, in Figure 2-10b, the employee Melton has completed three courses (C++, COBOL, and Perl). The SQL course has been completed by two employees (Celko and Gosling), and the Visual Basic course has not been completed by anyone. In this example, there are two entity types (EMPLOYEE and COURSE) that participate in the relationship named Completes.

Basic Concepts and Definitions in Relationships

A **relationship type** is a meaningful association between (or among) entity types. A relationship type is denoted by a line labeled with the name of the relationship, as in the example shown in Figure 2-10a, or with two names.

A **relationship instance** is an association between (or among) entity instances. For example, in Figure 2-10b, each of the 10 lines in the figure represents a

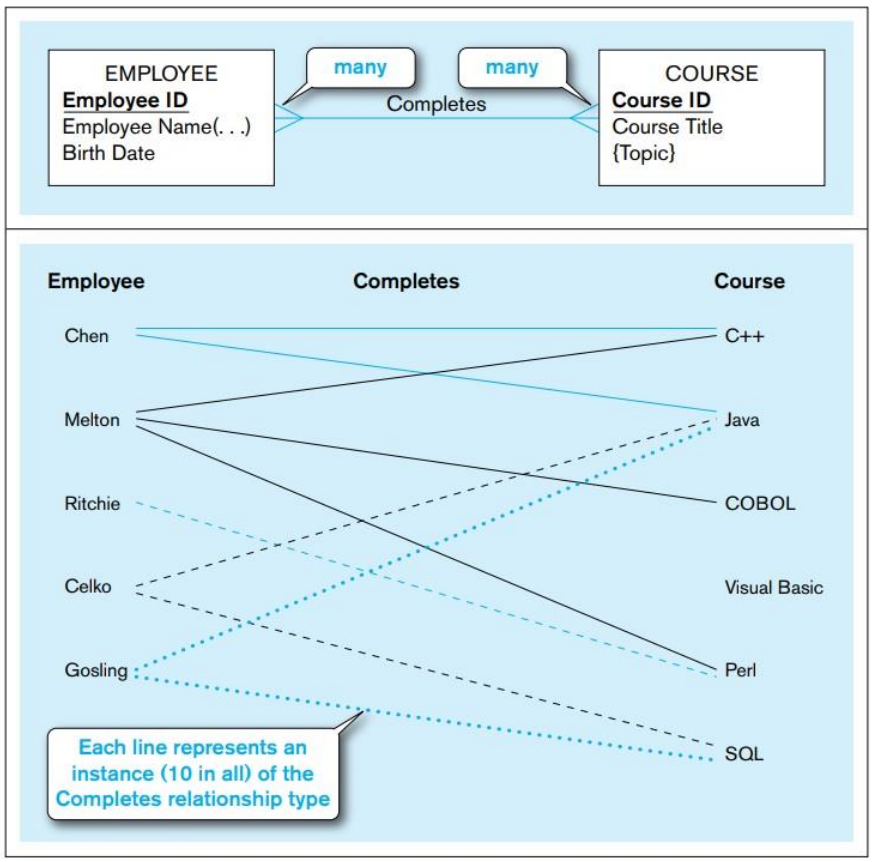

FIGURE 2-10 Relationship type and instances
(a) Relationship type (Complete)

(b) Relationship instances

relationship instance between one employee and one course, indicating that the employee has completed that course. For example, the line between Employee Ritchie and Course Perl is one relationship instance.

Attributes on Relationships

It is probably obvious to you that entities have attributes, **but attributes may be associated with a many-to-many (or one-to-one) relationship**, too. For example, suppose the organization wishes to record the date (month and year) when an employee completes each course. This attribute is named Date Completed. For some sample data, see Table 2-2. Where should the attribute Date Completed be placed on the E-R diagram? Referring to Figure 2-10a, you will notice that Date Completed has not been associated with either the EMPLOYEE or COURSE entity. That is because Date Completed is a property of the relationship Completes, rather than a property of either entity. In other

words, for each instance of the relationship Completes, there is a value for Date Completed. One such instance (for example) shows that the employee named Melton completed the course titled C++ in 06/2014. A revised version of the ERD for this example is shown in Figure 2-11a. In this diagram, the attribute Date Completed is in a rectangle connected to the Completes relationship line. Other attributes might be added to this relationship if appropriate, such as Course Grade, Instructor, and Room Location. **It is interesting to note that an attribute cannot be associated with a one-to-many relationship.**

TABLE 2-2 Instances Showing Date Completed

Employee Name	Course Title	Date Completed
Chen	C++	06/2014
Chen	Java	09/2014
Melton	C++	06/2014
Melton	COBOL	02/2015
Melton	SQL	03/2014
Ritchie	Perl	11/2014
Celko	Java	03/2014
Celko	SQL	03/2015
Gosling	Java	09/2014
Gosling	Perl	06/2014

Associative Entities

The presence of one or more attributes on a relationship suggests to the designer that the relationship should perhaps instead be represented as an entity type.

An **associative entity** is an entity type that associates the instances of one or more entity types and contains attributes that are peculiar to the relationship between those entity instances.

The associative entity CERTIFICATE is represented with the rectangle with rounded corners, as shown in Figure 2-11b. Notice that the name of associative entity is converted to noun.

FIGURE 2-11 An associative entity

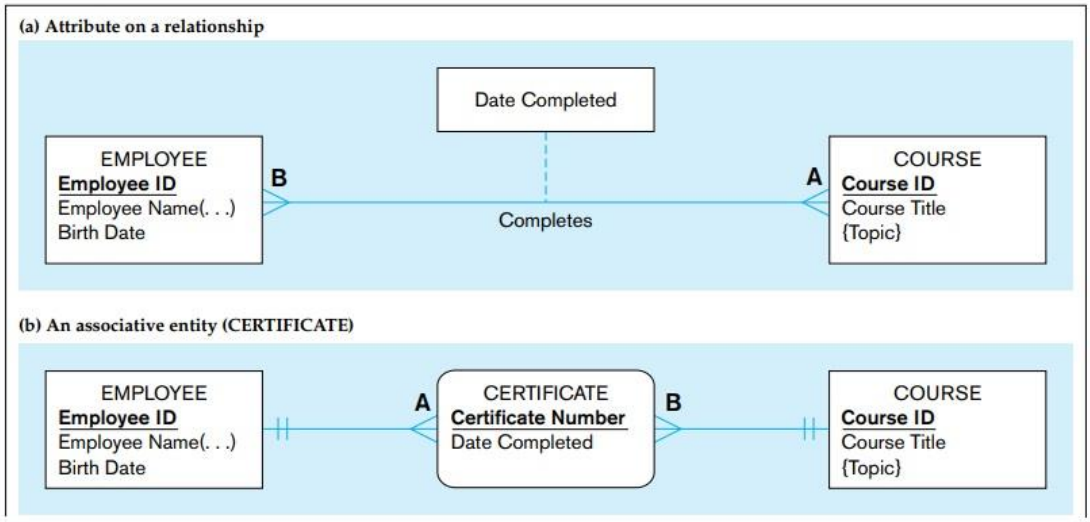

Figure 2-11b shows the relationship Completes converted to an associative entity type. In this case, the training department for the company has decided to award a certificate to each employee who completes a course. Thus, the entity is named CERTIFICATE. Also, each certificate has a number (Certificate Number) that serves as the identifier. The attribute Date Completed is also included. Note also in Figure 2-11b that both EMPLOYEE and COURSE are mandatory participants in the two relationships with CERTIFICATE. This is exactly what occurs when you have to represent a many-to-many relationship (in Figure 2-11a) as two one-to-many relationships.

Notice that converting a relationship to an associative entity has caused the relationship notation to move. That is, the "many" cardinality now terminates at the associative entity, rather than at each participating entity type. In Figure 2-11, this shows that an employee, who may complete one or more courses (notation A in Figure 2-11a), may be awarded more than one certificate (notation A in Figure 2-11b); and that a course, which may have one or more employees complete it (notation B in Figure 2-11a), may have many certificates awarded (notation B in Figure 2-11b).