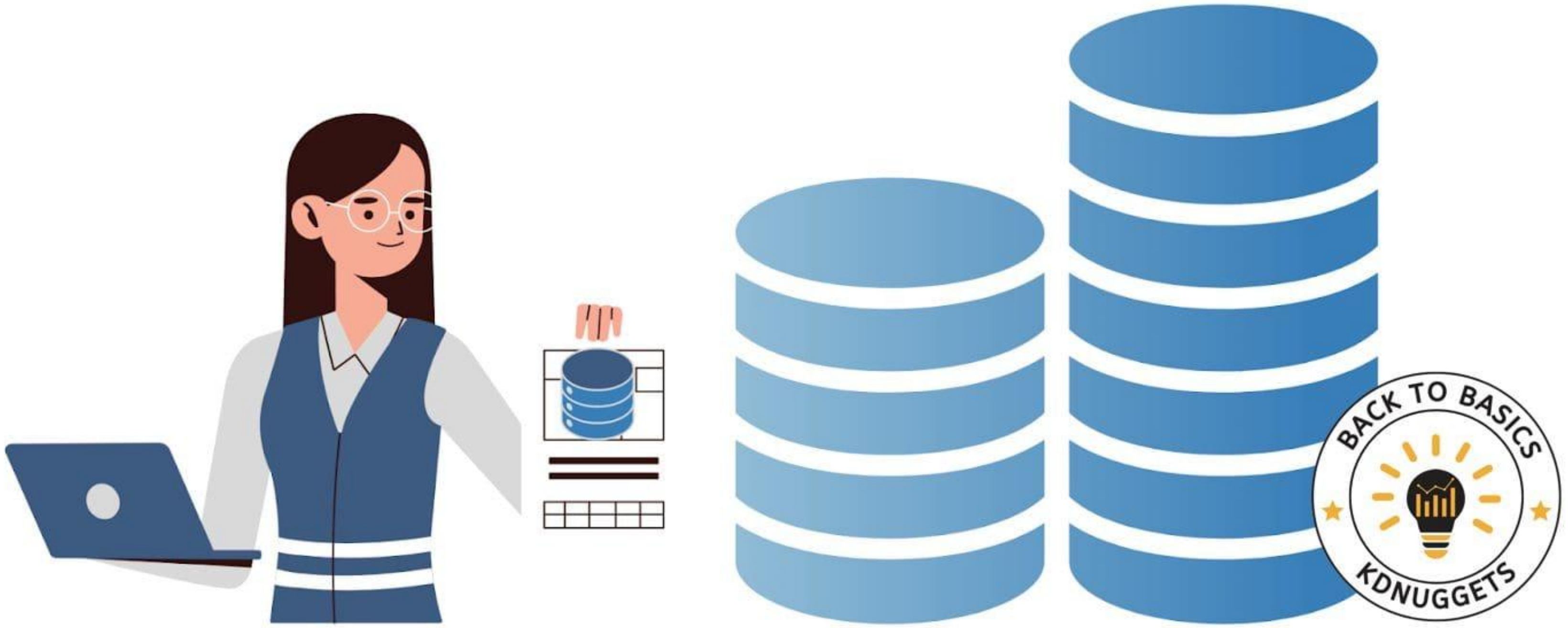
Intro to Databases



Entity Relationship Modelling (ERM) Part III

Lecture 7

02

Developing an ER Diagram

Case Study

- As you can see in Figure 4.23, a class can occur more than once in the ENROLL table. For example, CLASS_CODE = 10014 occurs twice. However, CLASS_CODE = 10014 occurs only once in the CLASS table to reflect that the relationship between CLASS and ENROLL is 1:M. Note that in Figure 4.25, the "M" is located on the ENROLL side, while the "1" is located on the CLASS side.

4-2 Developing an ER Diagram

The process of database design is iterative rather than a linear or sequential process. The verb *iterate* means "to do again or repeatedly." Thus, an **iterative process** is based on repetition of processes and procedures. Building an ERD usually involves the following activities:

- Create a detailed narrative of the organization's description of operations.
- Identify the business rules based on the description of operations.
- Identify the main entities and relationships from the business rules.
- Develop the initial ERD.
- Identify the attributes and primary keys that adequately describe the entities.
- Revise and review the ERD.

During the review process, additional objects, attributes, and relationships probably will be uncovered. Therefore, the basic ERM will be modified to incorporate the newly discovered ER components. Subsequently, another round of reviews might yield additional components or clarification of the existing diagram. The process is repeated until the end users and designers agree that the ERD is a fair representation of the organization's activities and functions.

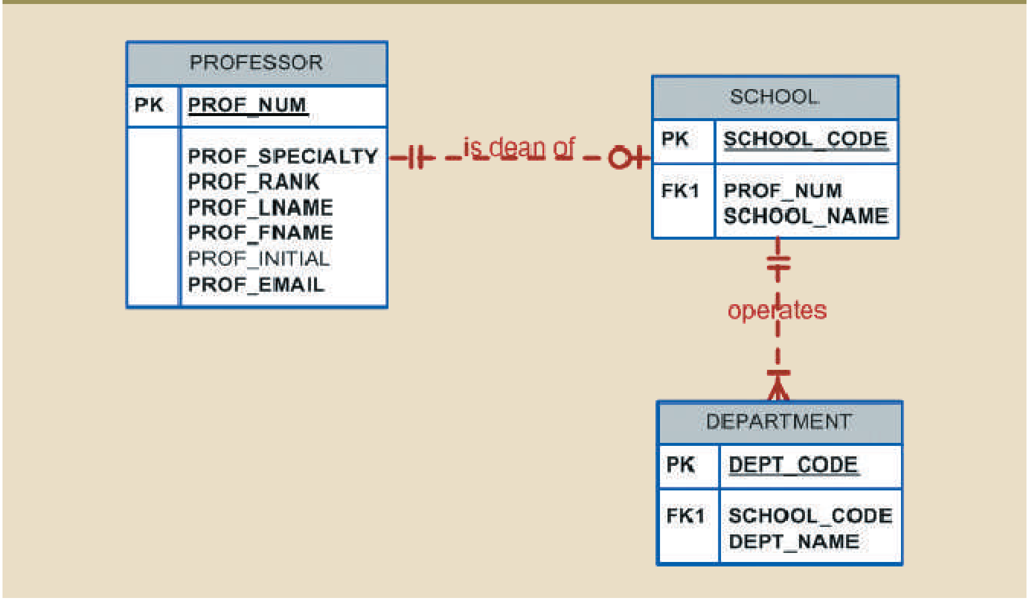
During the design process, the database designer does not depend simply on interviews to help define entities, attributes, and relationships. A surprising amount of information can be gathered by examining the business forms and reports that an organization uses in its daily operations.

To illustrate the use of the iterative process that ultimately yields a workable ERD, start with an initial interview with the Tiny College administrators. The interview process yields the following business rules:

1. Tiny College (TC) is divided into several schools: business, arts and sciences, education, and applied sciences. Each school is administered by a dean who is a professor. Each professor can be the dean of only one school, and a professor is not required to be the dean of any school. Therefore, a 1:1 relationship exists between PROFESSOR and SCHOOL. Note that the cardinality can be expressed by writing (1,1) next to the entity PROFESSOR and (0,1) next to the entity SCHOOL.
2. Each school comprises several departments. For example, the school of business has an accounting department, a management/marketing department, an economics/finance department, and a computer information systems department. Note again the cardinality rules: The smallest number of departments operated by a school is one, and the largest number of departments is indeterminate (N). On the other hand, each department belongs to only a single school; thus, the cardinality is expressed by (1,1). That is, the minimum number of schools to which a department belongs is one, as is the maximum number. Figure 4.26 illustrates these first two business rules.
3. Each department may offer courses. For example, the management/marketing department offers courses such as Introduction to Management, Principles of Marketing, and Production Management. The ERD segment for this condition is

iterative process
A process based on repetition of steps and procedures.

FIGURE 4.26 THE FIRST TINY COLLEGE ERD SEGMENT



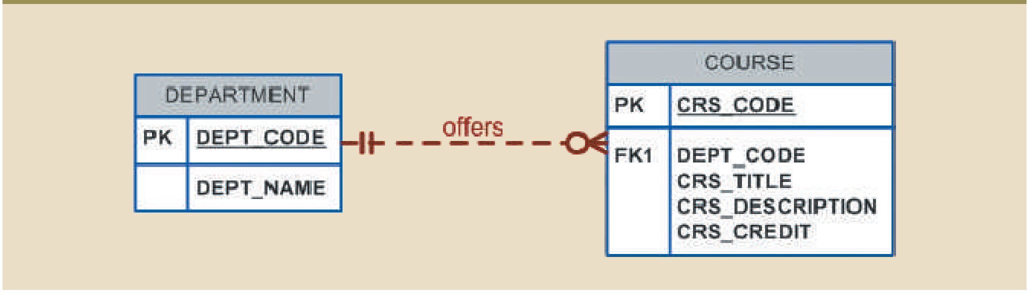
Note

It is again appropriate to evaluate the reason for maintaining the 1:1 relationship between PROFESSOR and SCHOOL in the "PROFESSOR is dean of SCHOOL" relationship. It is worth repeating that the existence of 1:1 relationships often indicates a misidentification of attributes as entities. In this case, the 1:1 relationship could easily be eliminated by storing the dean's attributes in the SCHOOL entity. This solution would also make it easier to answer the queries "Who is the dean?" and "What are the dean's credentials?" The downside of this solution is that it requires the duplication of data that is already stored in the PROFESSOR table, thus setting the stage for anomalies. However, because each school is run by a single dean, the problem of data duplication is rather minor. The selection of one approach over another often depends on information requirements, transaction speed, and the database designer's professional judgment. In short, do not use 1:1 relationships lightly, and make sure that each 1:1 relationship within the database design is defensible.

shown in Figure 4.27. Note that this relationship is based on the way Tiny College operates. For example, if Tiny College had some departments that were classified as "research only," they would not offer courses; therefore, the COURSE entity would be optional to the DEPARTMENT entity.

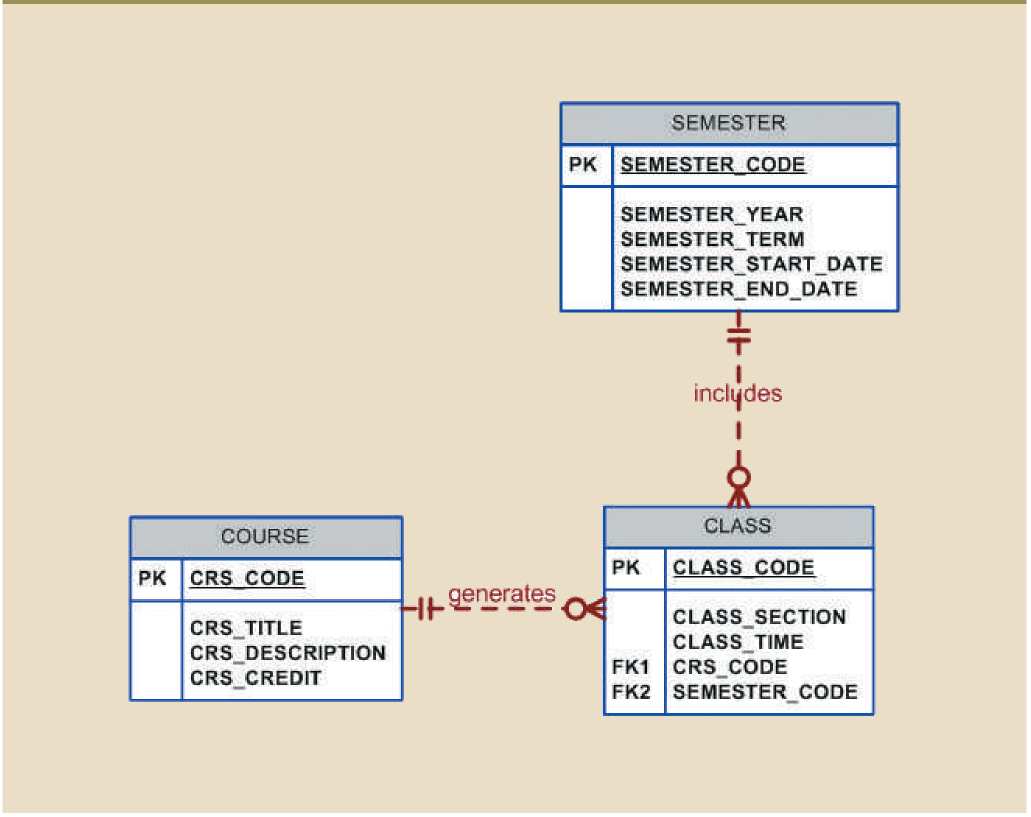
4. The relationship between COURSE and CLASS was illustrated in Figure 4.9. Nevertheless, it is worth repeating that a CLASS is a section of a COURSE. That is, a department may offer several sections (classes) of the same database course. Each of those classes is taught by a professor at a given time in a given place. In short, a 1:M relationship exists between COURSE and CLASS. Additionally, each class is offered during a given semester. SEMESTER defines the year and the term that the class will be offered. Note that this is different from the date when the student actually enrolls in a class. For example, students are able to enroll in summer and fall term classes near the end of the spring term. It is possible that the Tiny College calendar is set with semester beginning and ending dates prior to the creation of the semester class

FIGURE 4.27 THE SECOND TINY COLLEGE ERD SEGMENT



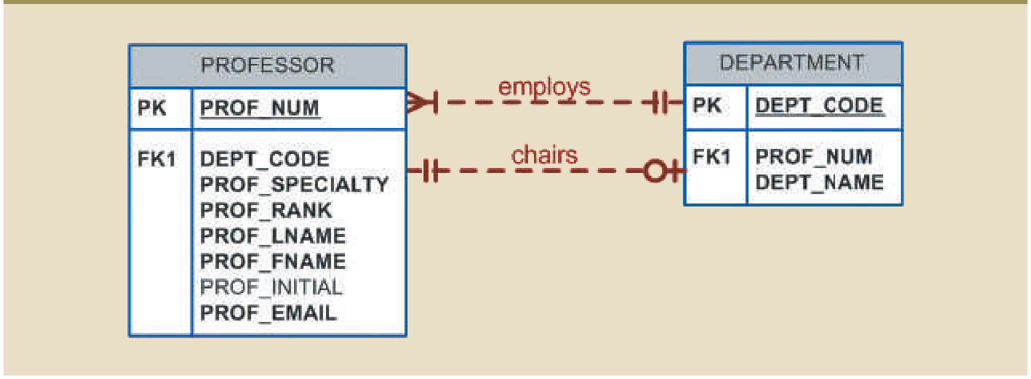
schedule so CLASS is optional to SEMESTER. This design will also help for reporting purposes, for example, you could answer questions such as: what classes were offered X semester? Or, what classes did student Y take on semester X? Because a course may exist in Tiny College's course catalog even when it is not offered as a class in a given semester, CLASS is optional to COURSE. Therefore, the relationships between SEMESTER, COURSE, and CLASS look like Figure 4.28.

FIGURE 4.28 THE THIRD TINY COLLEGE ERD SEGMENT



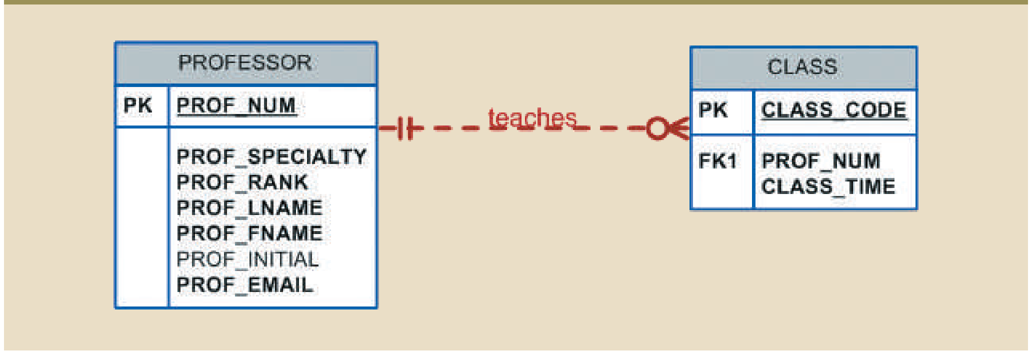
5. Each department should have one or more professors assigned to it. One and only one of those professors chairs the department, and no professor is required to accept the chair position. Therefore, DEPARTMENT is optional to PROFESSOR in the "chairs" relationship. Those relationships are summarized in the ER segment shown in Figure 4.29.

FIGURE 4.29 THE FOURTH TINY COLLEGE ERD SEGMENT



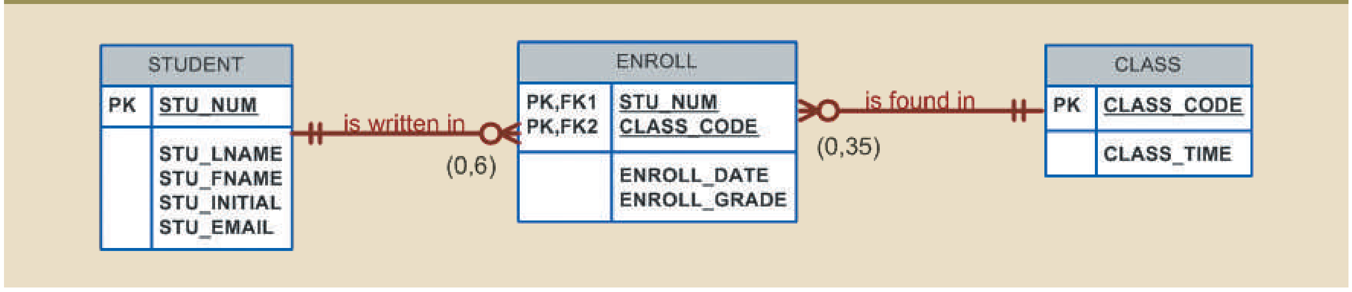
6. Each professor may teach up to four classes; each class is a section of a course. A professor may also be on a research contract and teach no classes at all. The ERD segment in Figure 4.30 depicts those conditions.

FIGURE 4.30 THE FIFTH TINY COLLEGE ERD SEGMENT



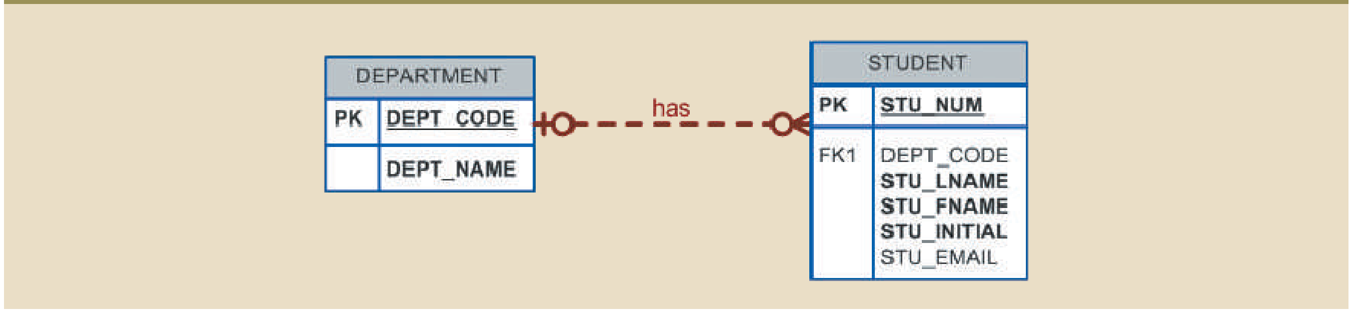
7. A student may enroll in several classes but take each class only once during any given enrollment period. For example, during the current enrollment period, a student may decide to take five classes—Statistics, Accounting, English, Database, and History—but that student would not be enrolled in the same Statistics class five times during the enrollment period! Each student may enroll in up to six classes, and each class may have up to 35 students, thus creating an M:N relationship between STUDENT and CLASS. Because a CLASS can initially exist at the start of the enrollment period even though no students have enrolled in it, STUDENT is optional to CLASS in the M:N relationship. This M:N relationship must be divided into two 1:M relationships through the use of the ENROLL entity, shown in the ERD segment in Figure 4.31. However, note that the optional symbol is shown next to ENROLL. If a class exists but has no students enrolled in it, that class does not occur in the ENROLL table. Note also that the ENROLL entity is weak: it is existence-dependent, and its (composite) PK is composed of the PKs of the STUDENT and CLASS entities. You can add the cardinalities (0,6) and (0,35) next to the ENROLL entity to reflect the business rule constraints, as shown in Figure 4.31. (Visio Professional does not automatically generate such cardinalities, but you can use a text box to accomplish that task.)

FIGURE 4.31 THE SIXTH TINY COLLEGE ERD SEGMENT



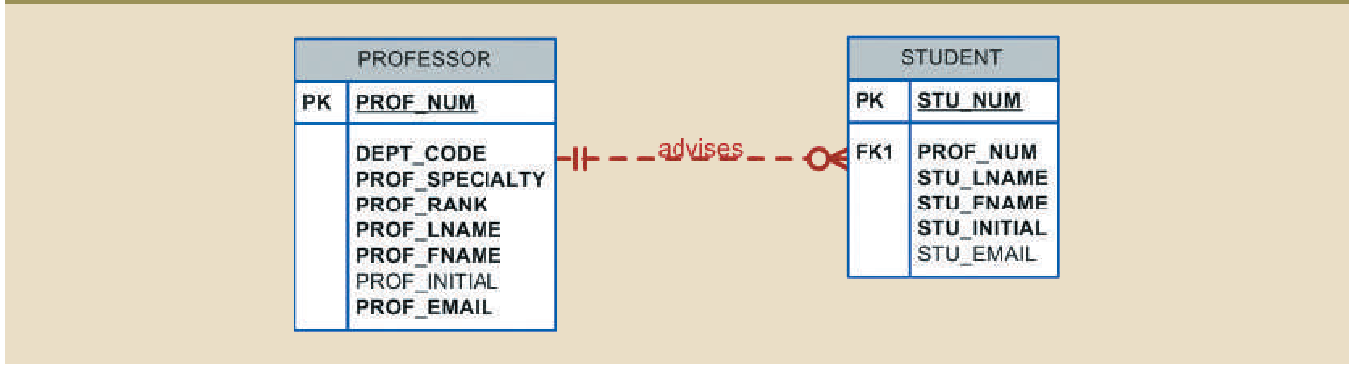
8. Each department has several (or many) students whose major is offered by that department. However, each student has only a single major and is therefore associated with a single department. (See Figure 4.32.) However, in the Tiny College environment, it is possible—at least for a while—for a student not to declare a major field of study. Such a student would not be associated with a department; therefore, DEPARTMENT is optional to STUDENT. It is worth repeating that the relationships between entities and the entities themselves reflect the organization's operating environment. That is, the business rules define the ERD components.

FIGURE 4.32 THE SEVENTH TINY COLLEGE ERD SEGMENT



9. Each student has an advisor in his or her department; each advisor counsels several students. An advisor is also a professor, but not all professors advise students. Therefore, STUDENT is optional to PROFESSOR in the "PROFESSOR advises STUDENT" relationship. (See Figure 4.33.)

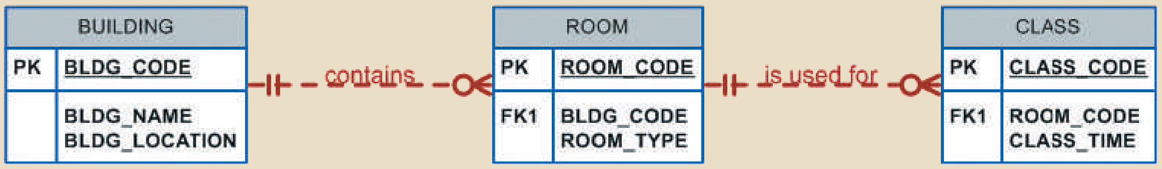
FIGURE 4.33 THE EIGHTH TINY COLLEGE ERD SEGMENT



10. As you can see in Figure 4.34, the CLASS entity contains a ROOM_CODE attribute. Given the naming conventions, it is clear that ROOM_CODE is an FK to another entity. Clearly, because a class is taught in a room, it is reasonable to assume that

the ROOM_CODE in CLASS is the FK to an entity named ROOM. In turn, each room is located in a building. So, the last Tiny College ERD is created by observing that a BUILDING can contain many ROOMs, but each ROOM is found in a single BUILDING. In this ERD segment, it is clear that some buildings do not contain (class) rooms. For example, a storage building might not contain any named rooms at all.

FIGURE 4.34 THE NINTH TINY COLLEGE ERD SEGMENT



Using the preceding summary, you can identify the following entities:

PROFESSOR	SCHOOL	DEPARTMENT
COURSE	CLASS	SEMESTER
STUDENT	BUILDING	ROOM
ENROLL (the associative entity between STUDENT and CLASS)		

Once you have discovered the relevant entities, you can define the initial set of relationships among them. Next, you describe the entity attributes. Identifying the attributes of the entities helps you to better understand the relationships among entities. Table 4.4 summarizes the ERM's components, and names the entities and their relations.

TABLE 4.4

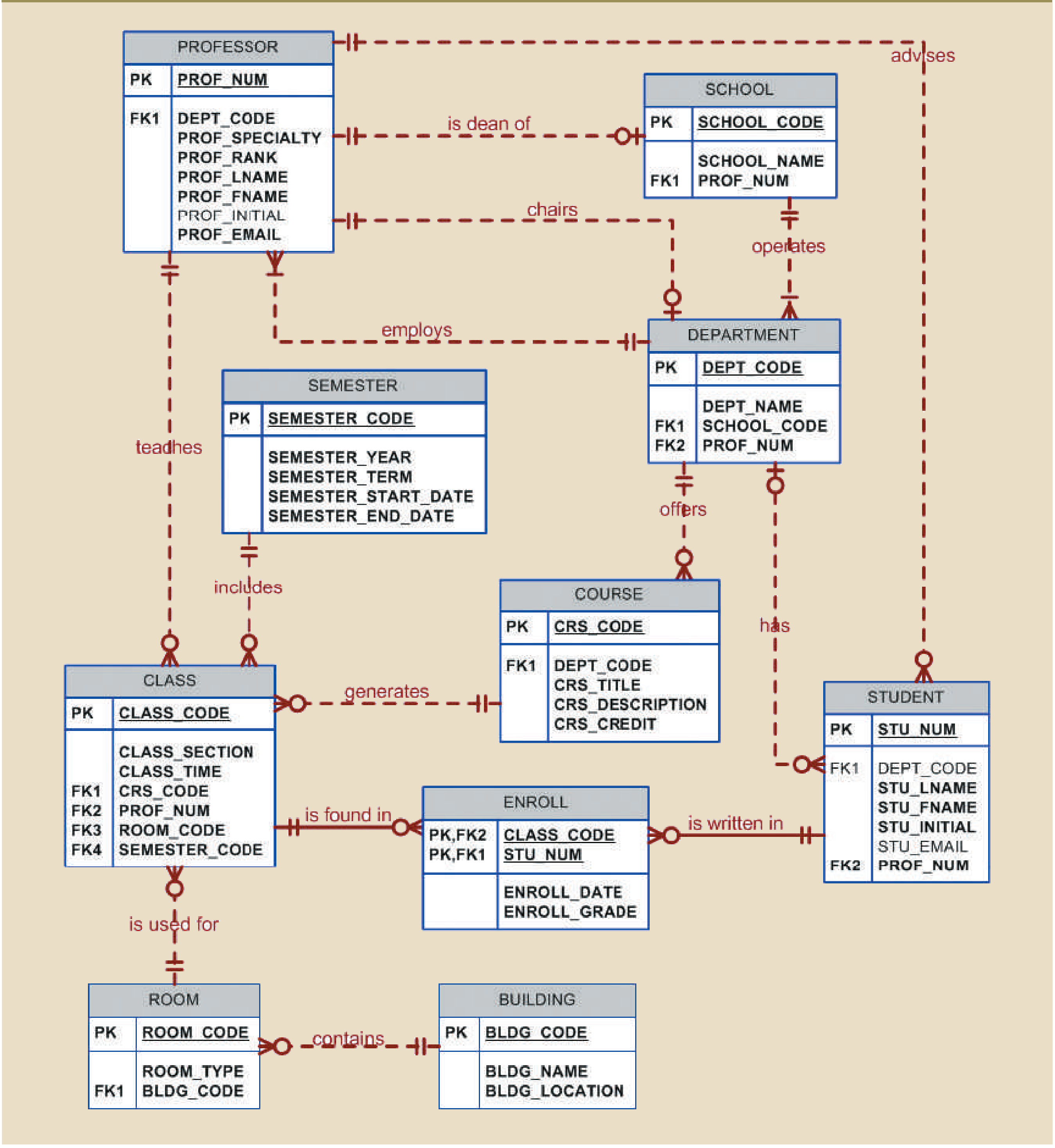
COMPONENTS OF THE ERM

ENTITY	RELATIONSHIP	CONNECTIVITY	ENTITY
SCHOOL	operates	1:M	DEPARTMENT
DEPARTMENT	has	1:M	STUDENT
DEPARTMENT	employs	1:M	PROFESSOR
DEPARTMENT	offers	1:M	COURSE
COURSE	generates	1:M	CLASS
SEMESTER	includes	1:M	CLASS
PROFESSOR	is dean of	1:1	SCHOOL
PROFESSOR	chairs	1:1	DEPARTMENT
PROFESSOR	teaches	1:M	CLASS
PROFESSOR	advises	1:M	STUDENT
STUDENT	enrolls in	M:N	CLASS
BUILDING	contains	1:M	ROOM
ROOM	is used for	1:M	CLASS

Note: ENROLL is the composite entity that implements the M:N relationship "STUDENT enrolls in CLASS."

You must also define the connectivity and cardinality for the just-discovered relations based on the business rules. However, to avoid crowding the diagram, the cardinalities are not shown. Figure 4.35 shows the Crow's Foot ERD for Tiny College. Note that this is an implementation-ready model, so it shows the ENROLL composite entity.

FIGURE 4.35 THE COMPLETED TINY COLLEGE ERD



Although we focus on Crow's Foot notation to develop our diagram, as mentioned at the beginning of this chapter, UML notation is also popular for conceptual and implementation modeling. Figure 4.36 shows the conceptual UML class diagram for Tiny College. Note that this class diagram depicts the M:N relationship between STUDENT and CLASS. Figure 4.37 shows the implementation-ready UML class diagram for Tiny College (note that the ENROLL composite entity is shown in this class diagram). If you are a good observer, you will also notice that the UML class diagrams in Figures 4.36 and 4.37 show the entity and attribute names but do not identify the primary key attributes. The reason goes back to UML's roots. UML class diagrams are an object-oriented modeling language, and therefore do not support the notion of "primary or foreign keys" found mainly in the relational world. Rather, in the object-oriented world, objects inherit a unique object identifier at creation time. For more information, see Appendix G, Object-Oriented Databases.

FIGURE 4.36 THE CONCEPTUAL UML CLASS DIAGRAM FOR TINY COLLEGE

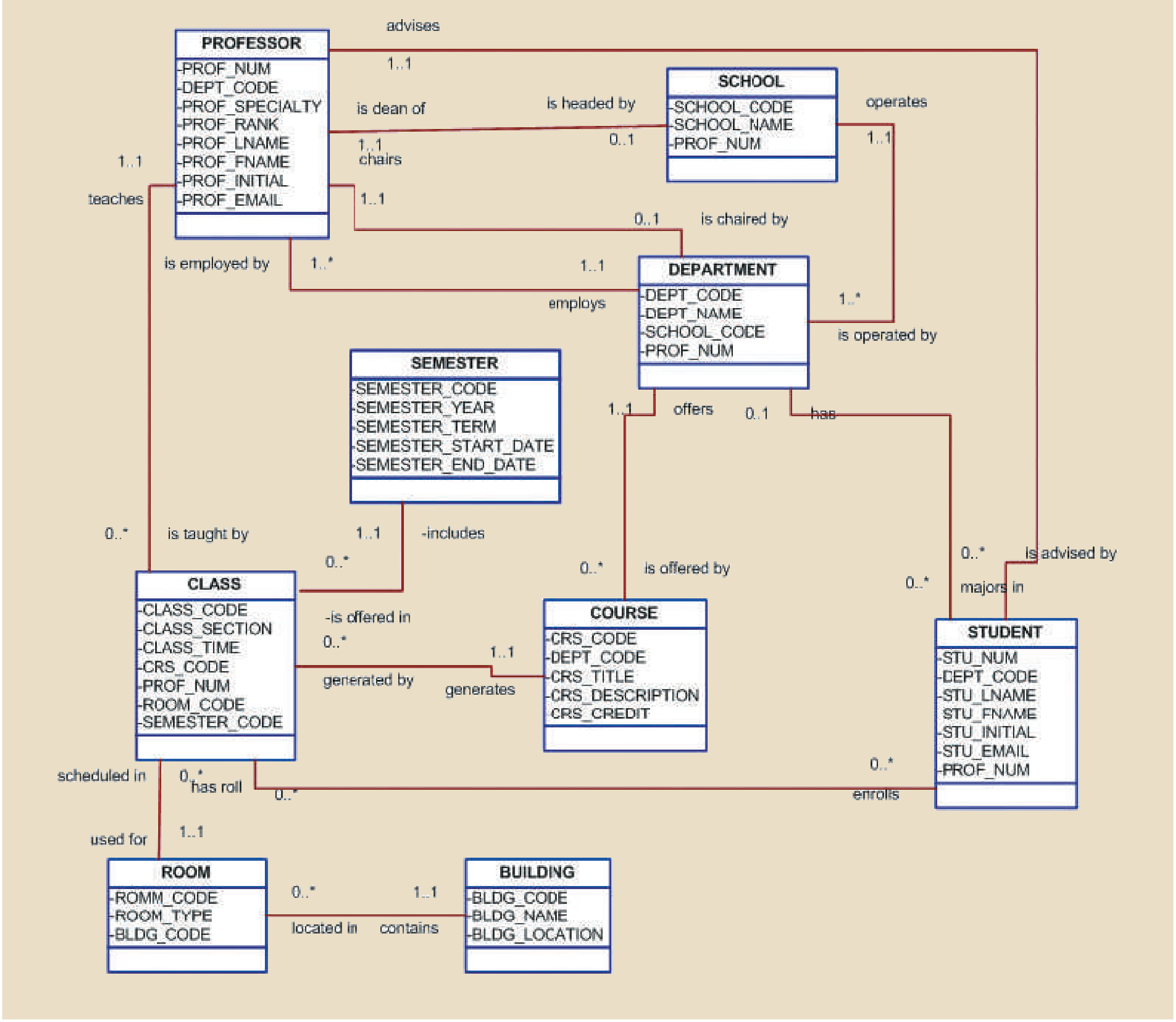
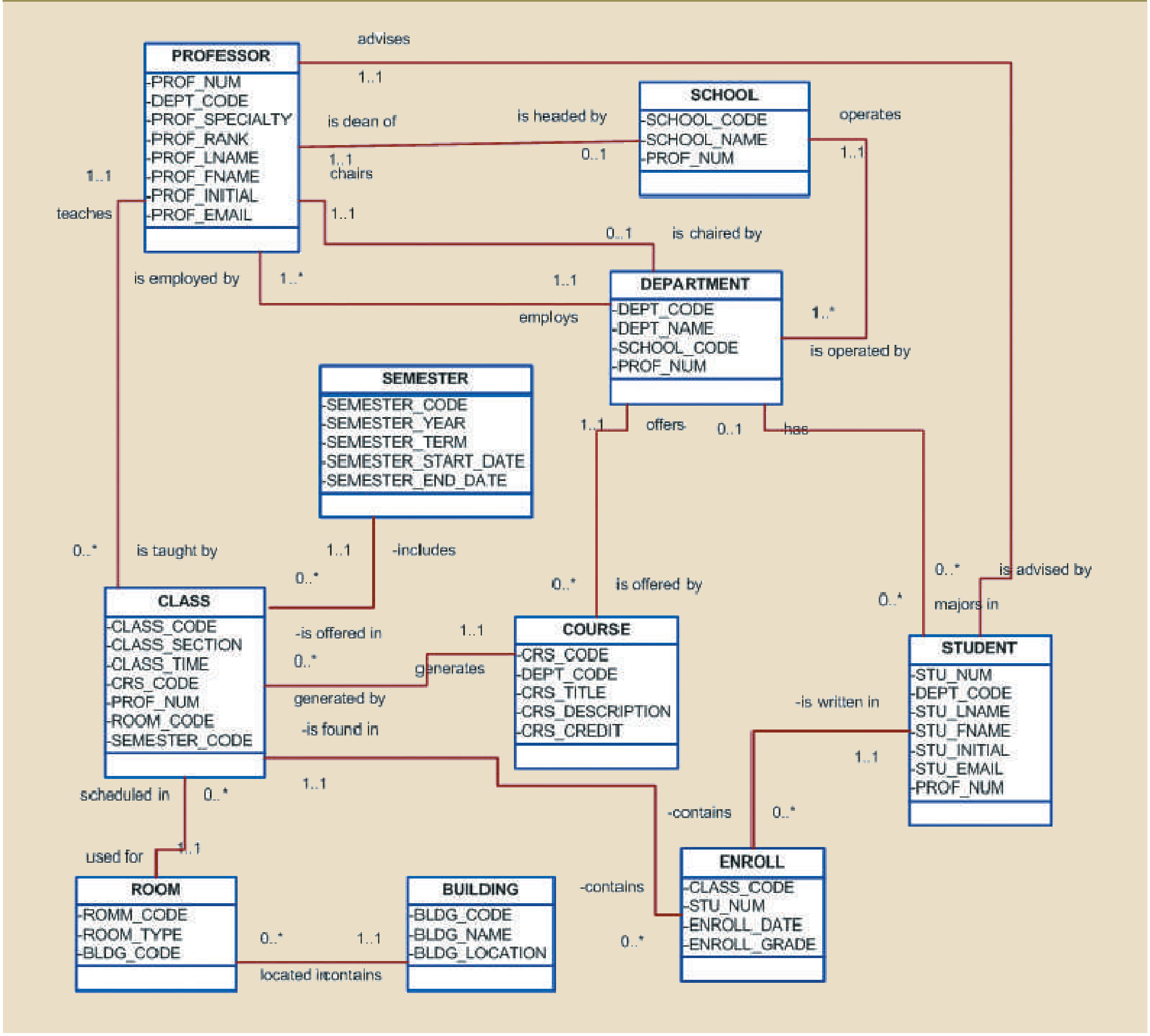


FIGURE 4.37 THE IMPLEMENTATION-READY UML CLASS DIAGRAM FOR TINY COLLEGE



4-3 Database Design Challenges: Conflicting Goals

Database designers must often make design compromises that are triggered by conflicting goals, such as adherence to design standards (design elegance), processing speed, and information requirements.

- *Design standards*. The database design must conform to design standards. Such standards guide you in developing logical structures that minimize data redundancies, thereby minimizing the likelihood that destructive data anomalies will occur. You have also learned how standards prescribe avoiding nulls to the greatest extent possible. In fact, you have learned that design standards govern the presentation of all