C H A P T E R 2

Entity Relationship Model

Exercises

2.2 Answer: See Figure 2.1

2.4 Answer: See Figure 2.2.
In the answer given here, the main entity sets are *student*, *course*, *course-offering*, and *instructor*. The entity set *course-offering* is a weak entity set dependent on *course*. The assumptions made are :

a. a class meets only at one particular place and time. This E-R diagram cannot model a class meeting at different places at different times.
b. There is no guarantee that the database does not have two classes meeting at the same place and time.

2.5 Answer:

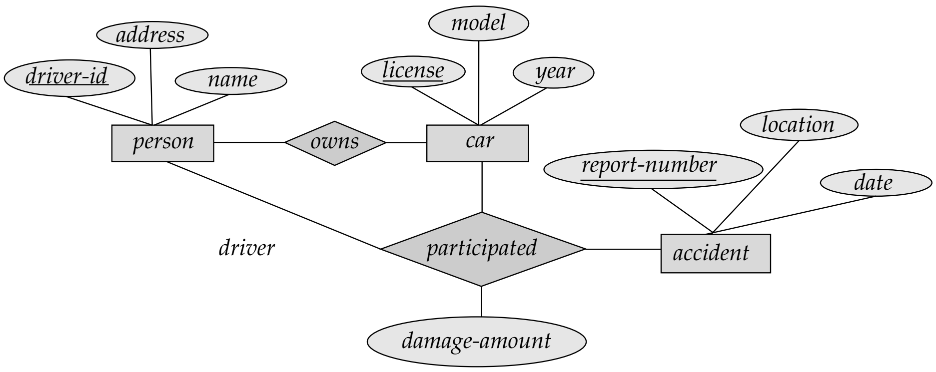

Figure 2.1 E-R diagram for a Car-insurance company.

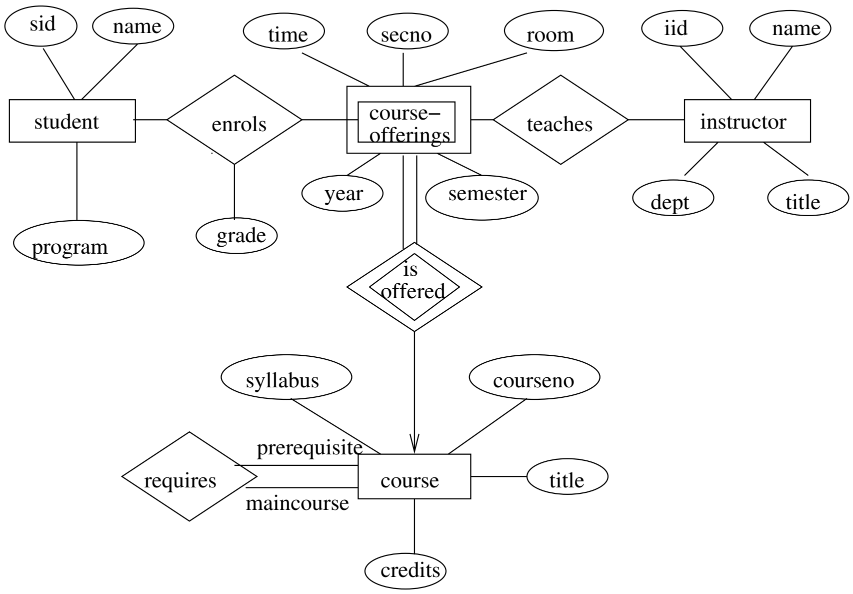

Figure 2.2 E-R diagram for a university.

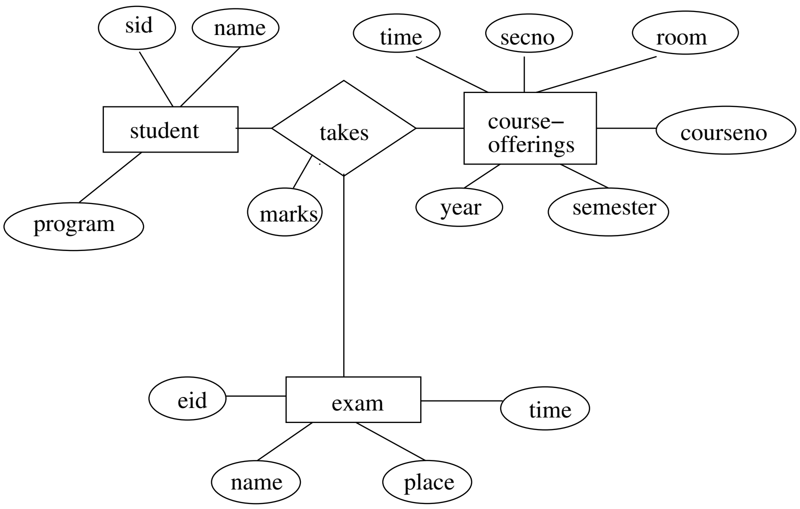

Figure 2.3 E-R diagram for marks database.

a. See Figure 2.3
b. See Figure 2.4

2.7 Answer: See Figure 2.5

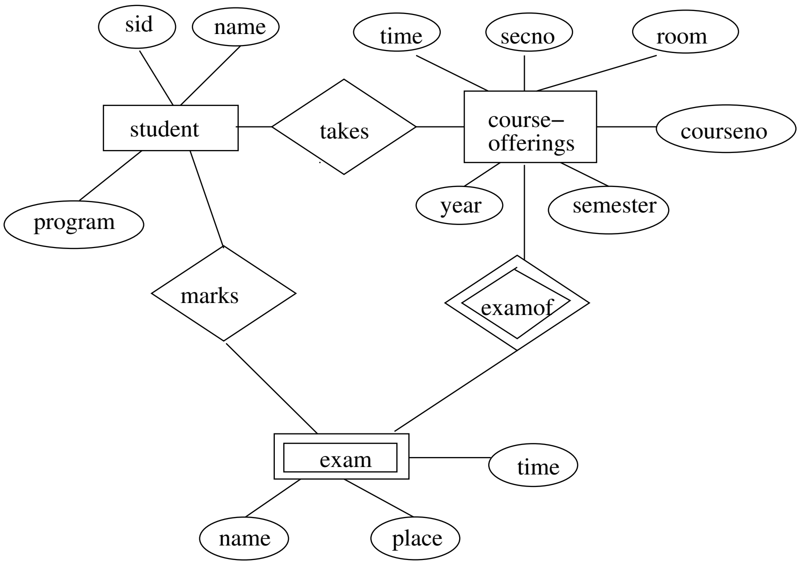

Figure 2.4 Another E-R diagram for marks database.

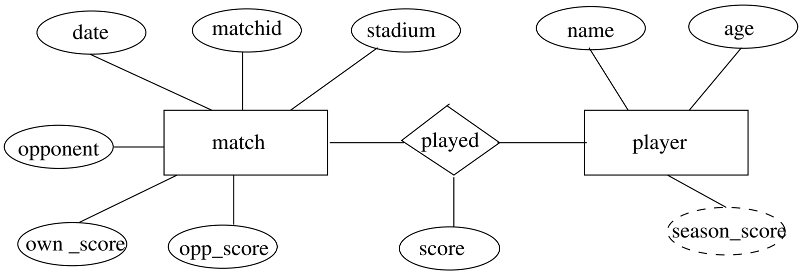

Figure 2.5 E-R diagram for favourite team statistics.

2.13 Answer: By using one entity set many times we are missing relationships in the model. For example, in the E-R diagram in Figure 2.6: the students taking classes are the same students who are athletes, but this model will not show that.

However, E-R diagrams are often split into parts so that each part will fit in a single page. In such a case, an entity set may appear on different pages, but to avoid repetition (and inconsistencies), its attributes should be specified at only one place. The absence of attributes at other places indicates that they are placeholders, and the full definition is present elsewhere.

2.14 Answer:

a. See Figure 2.7

b. The additional entity sets are useful if we wish to store their attributes as part of the database. For the *course* entity set, we have chosen to include

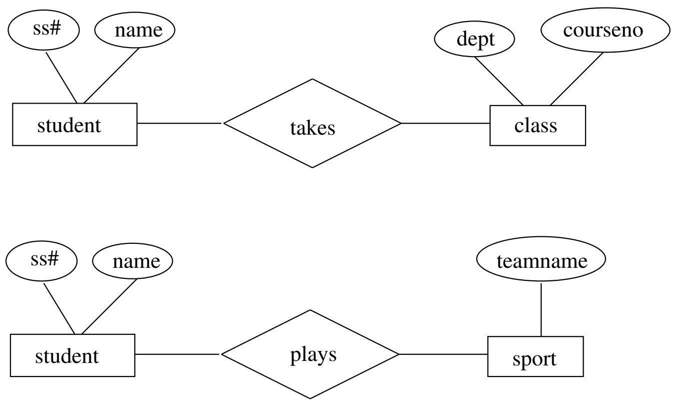

Figure 2.6 E-R diagram with entity duplication.

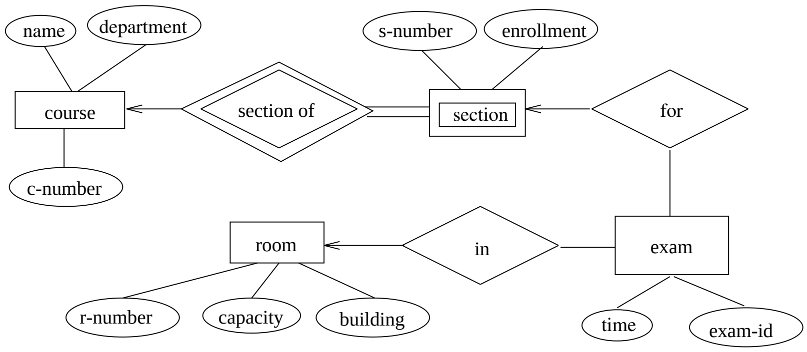

Figure 2.7 E-R diagram for exam scheduling.

three attributes. If only the primary key (*c-number*) were included, and if courses have only one section, then it would be appropriate to replace the *course* (and *section*) entity sets by an attribute (*c-number*) of *exam*. The reason it is undesirable to have multiple attributes of *course* as attributes of *exam* is that it would then be difficult to maintain data on the courses, particularly if a course has no exam or several exams. Similar remarks apply to the *room* entity set.

2.15 Answer:

a. The criteria to use are intuitive design, accurate expression of the real-world concept and efficiency. A model which clearly outlines the objects and relationships in an intuitive manner is better than one which does not, because it is easier to use and easier to change. Deciding between an attribute and an entity set to represent an object, and deciding between an entity set and relationship set, influence the accuracy with which the real-world concept is expressed. If the right design choice is not made, inconsistency and/or

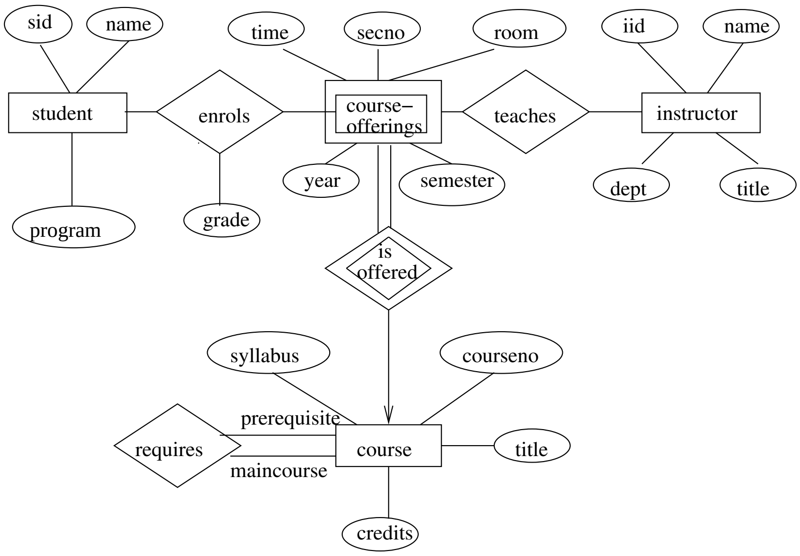

Figure 2.8 E-R diagram for University(a) .

loss of information will result. A model which can be implemented in an efficient manner is to be preferred for obvious reasons.

b. Consider three different alternatives for the problem in Exercise 2.4.
- See Figure 2.8
- See Figure 2.9
- See Figure 2.10

Each alternative has merits, depending on the intended use of the database. Scheme 2.8 has been seen earlier. Scheme 2.10 does not require a separate entity for *prerequisites*. However, it will be difficult to store all the prerequisites(being a multi-valued attribute). Scheme 2.9 treats prerequisites as well as classrooms as separate entities, making it useful for gathering data about prerequisites and room usage. Scheme 2.8 is in between the others, in that it treats prerequisites as separate entities but not classrooms. Since a registrar's office probably has to answer general questions about the number of classes a student is taking or what are all the prerequisites of a course, or where a specific class meets, scheme 2.9 is probably the best choice.

2.16 Answer:

a. If a pair of entity sets are connected by a path in an E-R diagram, the entity sets are related, though perhaps indirectly. A disconnected graph implies that there are pairs of entity sets that are unrelated to each other. If we split the graph into connected components, we have, in effect, a separate database corresponding to each connected component.

b. As indicated in the answer to the previous part, a path in the graph between a pair of entity sets indicates a (possibly indirect) relationship between the two entity sets. If there is a cycle in the graph then every pair of entity sets

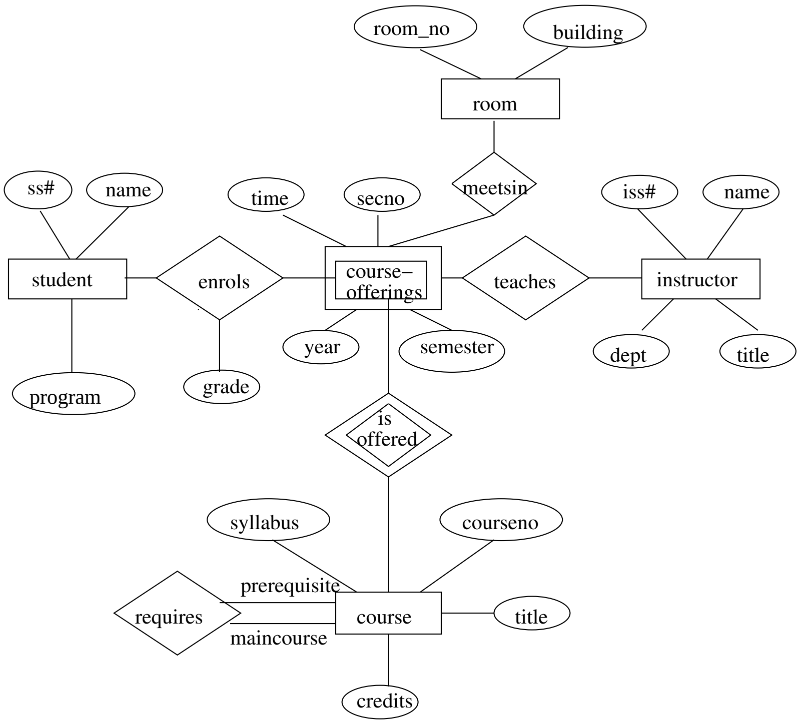

Figure 2.9 E-R diagram for University(b).

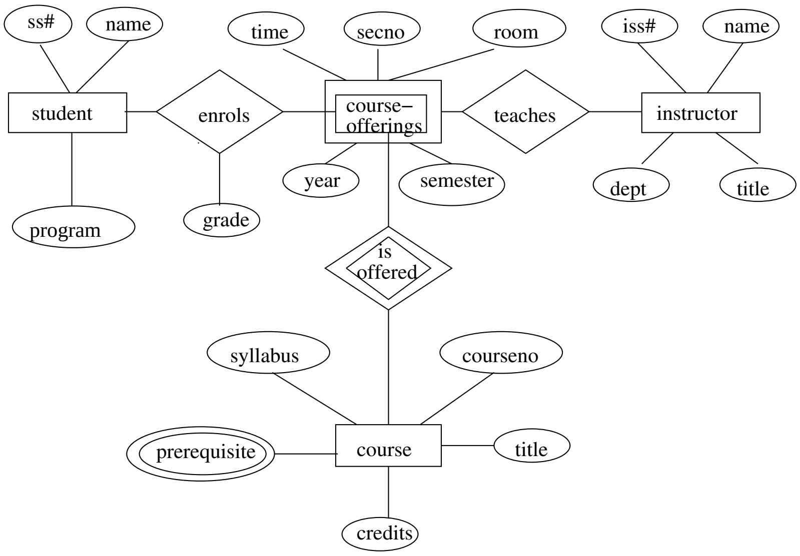

Figure 2.10 E-R diagram for University(c).

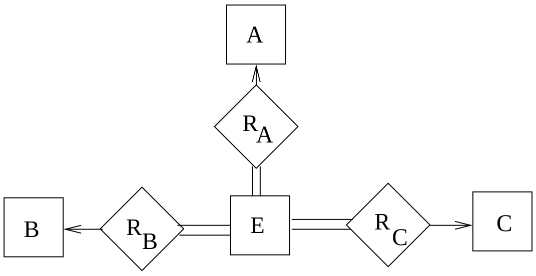

Figure 2.11 E-R diagram to Exercise 2.17b.

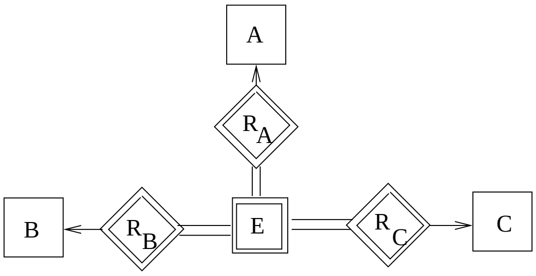

Figure 2.12 E-R diagram to Exercise 2.17d.

on the cycle are related to each other in at least two distinct ways. If the E-R diagram is acyclic then there is a unique path between every pair of entity sets and, thus, a unique relationship between every pair of entity sets.

2.18 Answer:

a. Let $E = \{e_1, e_2\}$, $A = \{a_1, a_2\}$, $B = \{b_1\}$, $C = \{c_1\}$, $R_A = \{(e_1, a_1), (e_2, a_2)\}$, $R_B = \{(e_1, b_1)\}$, and $R_C = \{(e_1, c_1)\}$. We see that because of the tuple (e_2, a_2), no instance of R exists which corresponds to E, R_A, R_B and R_C.

b. See Figure 2.11. The idea is to introduce total participation constraints between E and the relationships R_A, R_B, R_C so that every tuple in E has a relationship with A, B and C.

c. Suppose A totally participates in the relationhip R, then introduce a total participation constraint between A and R_A.

d. Consider E as a weak entity set and R_A, R_B and R_C as its identifying relationship sets. See Figure 2.12.

2.19 Answer: The primary key of a weak entity set can be inferred from its relationship with the strong entity set. If we add primary key attributes to the weak entity set, they will be present in both the entity set and the relationship set and they have to be the same. Hence there will be redundancy.

2.24 Answer: A inherits all the attributes of X plus it may define its own attributes. Similarly C inherits all the attributes of Y plus its own attributes. B inherits the attributes of both X and Y. If there is some attribute *name* which belongs to both X and Y, it may be referred to in B by the qualified name X.*name* or Y.*name*.

2.26 Answer: In this example, we assume that both banks have the shared identifiers for customers, such as the social security number. We see the general solution in the next exercise.

Each of the problems mentioned does have potential for difficulties.

a. *branch-name* is the primary-key of the *branch* entity set. Therefore while merging the two banks' entity sets, if both banks have a branch with the same name, one of them will be lost.

b. customers participate in the relationship sets *cust-banker*, *borrower* and *depositor*. While merging the two banks' *customer* entity sets, duplicate tuples of the same customer will be deleted. Therefore those relations in the three mentioned relationship sets which involved these deleted tuples will have to be updated. Note that if the tabular representation of a relationship set is obtained by taking a union of the primary keys of the participating entity sets, no modification to these relationship sets is required.

c. The problem caused by *loans* or *accounts* with the same number in both the banks is similar to the problem caused by branches in both the banks with the same *branch-name*.

To solve the problems caused by the merger, no schema changes are required. Merge the *customer* entity sets removing duplicate tuples with the same *social-security* field. Before merging the *branch* entity sets, prepend the old bank name to the *branch-name* attribute in each tuple. The *employee* entity sets can be merged directly, and so can the *payment* entity sets. No duplicate removal should be performed. Before merging the *loan* and *account* entity sets, whenever there is a number common in both the banks, the old number is replaced by a new unique number, in one of the banks.

Next the relationship sets can be merged. Any relation in any relationship set which involves a tuple which has been modified earlier due to the merger, is itself modified to retain the same meaning. For example let 1611 be a loan number common in both the banks prior to the merger, and let it be replaced by a new unique number 2611 in one of the banks, say bank 2. Now all the relations in *borrower*, *loan-branch* and *loan-payment* of bank 2 which refer to loan number 1611 will have to be modified to refer to 2611. Then the merger with bank 1's corresponding relationship sets can take place.

2.27 Answer: This is a case in which the schemas of the two banks differ, so the merger becomes more difficult. The identifying attribute for persons in the US is *social-security*, and in Canada it is *social-insurance*. Therefore the merged schema cannot use either of these. Instead we introduce a new attribute *person-id*, and use this uniformly for everybody in the merged schema. No other change to the schema is required. The values for the *person-id* attribute may be obtained by several ways. One way would be to prepend a country code to the old *social-security* or *social-insurance* values ("U" and "C" respectively, for instance), to get the corresponding *person-id* values. Another way would be to assign fresh numbers starting from 1 upwards, one number to each *social-security* and *social-insurance* value in the old databases.

Once this has been done, the actual merger can proceed as according to the answer to the previous question. If a particular relationship set, say *borrower*, involves only US customers, this can be expressed in the merged database by specializing the entity-set *customer* into *us-customer* and *canada-customer*, and making only *us-customer* participate in the merged *borrower*. Similarly *employee* can be specialized if needed.