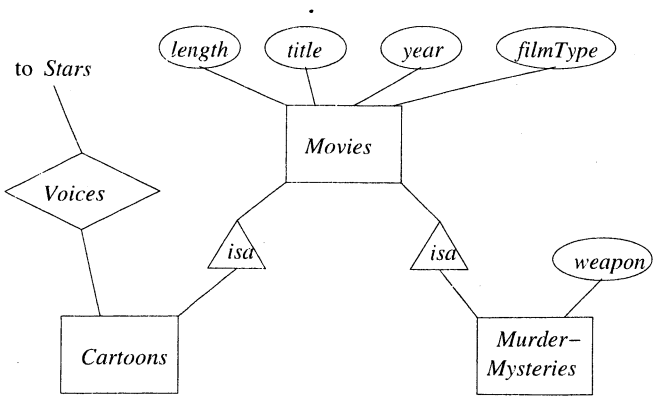

Figure 2.10: Isa relationships in an E/R diagram

2.1.12 Exercises for Section 2.1

* **Exercise 2.1.1:** Let us design a database for a bank, including information about customers and their accounts. Information about a customer includes their name, address, phone, and Social Security number. Accounts have numbers, types (e.g., savings, checking) and balances. We also need to record the customer(s) who own an account. Draw the E/R diagram for this database. Be sure to include arrows where appropriate, to indicate the multiplicity of a relationship.

Exercise 2.1.2: Modify your solution to Exercise 2.1.1 as follows:

a) Change your diagram so an account can have only one customer.

b) Further change your diagram so a customer can have only one account.

! c) Change your original diagram of Exercise 2.1.1 so that a customer can have a set of addresses (which are street-city-state triples) and a set of phones. Remember that we do not allow attributes to have nonatomic types, such as sets, in the E/R model.

! d) Further modify your diagram so that customers can have a set of addresses, and at each address there is a set of phones.

Exercise 2.1.3: Give an E/R diagram for a database recording information about teams, players, and their fans, including:

1. For each team, its name, its players, its team captain (one of its players), and the colors of its uniform.
2. For each player, his/her name.
3. For each fan, his/her name, favorite teams, favorite players, and favorite color.

Subclasses in Object-Oriented Systems

There is a significant resemblance between "isa" in the E/R model and subclasses in object-oriented languages. In a sense, "isa" relates a subclass to its superclass. However, there is also a fundamental difference between the conventional E/R view and the object-oriented approach: entities are allowed to have representatives in a tree of entity sets, while objects are assumed to exist in exactly one class or subclass.

The difference becomes apparent when we consider how the movie *Roger Rabbit* was handled in Example 2.11. In an object-oriented approach, we would need for this movie a fourth entity set, "cartoon-murder-mystery," which inherited all the attributes and relationships of *Movies*, *Cartoons*, and *Murder-Mysteries*. However, in the E/R model, the effect of this fourth subclass is obtained by putting components of the movie *Roger Rabbit* in both the *Cartoons* and *Murder-Mysteries* entity sets.

Remember that a set of colors is not a suitable attribute type for teams. How can you get around this restriction?

Exercise 2.1.4: Suppose we wish to add to the schema of Exercise 2.1.3 a relationship *Led-by* among two players and a team. The intention is that this relationship set consists of triples

(player1, player2, team)

such that player 1 played on the team at a time when some other player 2 was the team captain.

a) Draw the modification to the E/R diagram.

b) Replace your ternary relationship with a new entity set and binary relationships.

! c) Are your new binary relationships the same as any of the previously existing relationships? Note that we assume the two players are different, i.e., the team captain is not self-led.

Exercise 2.1.5: Modify Exercise 2.1.3 to record for each player the history of teams on which they have played, including the start date and ending date (if they were traded) for each such team.

! **Exercise 2.1.6:** Suppose we wish to keep a genealogy. We shall have one entity set, *Person*. The information we wish to record about persons includes their name (an attribute) and the following relationships: mother, father, and children. Give an E/R diagram involving the *Person* entity set and all the

relationships in which it is involved. Include relationships for mother, father, and children. Do not forget to indicate roles when an entity set is used more than once in a relationship.

! **Exercise 2.1.7:** Modify your "people" database design of Exercise 2.1.6 to include the following special types of people:

1. Females.
2. Males.
3. People who are parents.

You may wish to distinguish certain other kinds of people as well, so relationships connect appropriate subclasses of people.

Exercise 2.1.8: An alternative way to represent the information of Exercise 2.1.6 is to have a ternary relationship *Family* with the intent that a triple in the relationship set for *Family*

(person, mother, father)

is a person, their mother, and their father; all three are in the *People* entity set, of course.

* a) Draw this diagram, placing arrows on edges where appropriate.

b) Replace the ternary relationship *Family* by an entity set and binary relationships. Again place arrows to indicate the multiplicity of relationships.

Exercise 2.1.9: Design a database suitable for a university registrar. This database should include information about students, departments, professors, courses, which students are enrolled in which courses, which professors are teaching which courses, student grades, TA's for a course (TA's are students), which courses a department offers, and any other information you deem appropriate. Note that this question is more free-form than the questions above, and you need to make some decisions about multiplicities of relationships, appropriate types, and even what information needs to be represented.

! **Exercise 2.1.10:** Informally, we can say that two E/R diagrams "have the same information" if, given a real-world situation, the instances of these two diagrams that reflect this situation can be computed from one another. Consider the E/R diagram of Fig. 2.6. This four-way relationship can be decomposed into a three-way relationship and a binary relationship by taking advantage of the fact that for each movie, there is a unique studio that produces that movie. Give an E/R diagram without a four-way relationship that has the same information as Fig. 2.6.

and their presidents. If a studio ceases to exist, its president can no longer be called a (studio) president, so we would expect the president of the studio to be deleted from the entity set *Presidents*. Hence there is a rounded arrow to *Studios*. On the other hand, if a president were deleted from the database, the studio would continue to exist. Thus, we place an ordinary, pointed arrow to *Presidents*, indicating that each studio has at most one president, but might have no president at some time. □

2.3.7 Other Kinds of Constraints

As mentioned at the beginning of this section, there are other kinds of constraints one could wish to enforce in a database. We shall only touch briefly on these here, with the meat of the subject appearing in Chapter 7.

Domain constraints restrict the value of an attribute to be in a limited set. A simple example would be declaring the type of an attribute. A stronger domain constraint would be to declare an enumerated type for an attribute or a range of values, e.g., the *length* attribute for a movie must be an integer in the range 0 to 240. There is no specific notation for domain constraints in the E/R model, but you may place a notation stating a desired constraint next to the attribute, if you wish.

There are also more general kinds of constraints that do not fall into any of the categories mentioned in this section. For example, we could choose to place a constraint on the degree of a relationship, such as that a movie entity cannot be connected by relationship **stars** to more than 10 star entities. In the E/R model, we can attach a bounding number to the edges that connect a relationship to an entity set, indicating limits on the number of entities that can be connected to any one entity of the related entity set.

Figure 2.19: Representing a constraint on the number of stars per movie

Example 2.22: Figure 2.19 shows how we can represent the constraint that no movie has more than 10 stars in the E/R model. As another example, we can think of the arrow as a synonym for the constraint "≤ 1," and we can think of the rounded arrow of Fig. 2.18 as standing for the constraint "$= 1$." □

2.3.8 Exercises for Section 2.3

Exercise 2.3.1: For your E/R diagrams of:

* a) Exercise 2.1.1.

b) Exercise 2.1.3.

c) Exercise 2.1.6.

(i) Select and specify keys, and (ii) Indicate appropriate referential integrity constraints.

! **Exercise 2.3.2:** We may think of relationships in the E/R model as having keys, just as entity sets do. Let R be a relationship among the entity sets $E_1, E_2, \ldots, E_n$. Then a *key* for R is a set K of attributes chosen from the attributes of $E_1, E_2, \ldots, E_n$ such that if $(e_1, e_2, \ldots, e_n)$ and $(f_1, f_2, \ldots, f_n)$ are two different tuples in the relationship set for R, then it is not possible that these tuples agree in all the attributes of K. Now, suppose $n = 2$; that is, R is a binary relationship. Also, for each i, let K_i be a set of attributes that is a key for entity set E_i. In terms of E_1 and E_2, give a smallest possible key for R under the assumption that:

a) R is many-many.

* b) R is many-one from E_1 to E_2.

c) R is many-one from E_2 to E_1.

d) R is one-one.

!! **Exercise 2.3.3:** Consider again the problem of Exercise 2.3.2, but with n allowed to be any number, not just 2. Using only the information about which arcs from R to the E_i's have arrows, show how to find a smallest possible key K for R in terms of the K_i's.

! **Exercise 2.3.4:** Give examples (other than those of Example 2.20) from real life of attributes created for the primary purpose of being keys.

2.4 Weak Entity Sets

There is an occasional condition in which an entity set's key is composed of attributes some or all of which belong to another entity set. Such an entity set is called a *weak entity set.*

2.4.1 Causes of Weak Entity Sets

There are two principal sources of weak entity sets. First, sometimes entity sets fall into a hierarchy based on classifications unrelated to the "isa hierarchy" of Section 2.1.11. If entities of set E are subunits of entities in set F, then it is possible that the names of E entities are not unique until we take into account the name of the F entity to which the E entity is subordinate. Several examples will illustrate the problem.

3. If an entity set supplies any attributes for its own key, then those attributes will be underlined. An example is in Fig. 2.20, where the number of a crew participates in its own key, although it is not the complete key for *Crews*.

We can summarize these conventions with the following rule:

- Whenever we use an entity set E with a double border, it is weak. E's attributes that are underlined, if any, plus the key attributes of those entity sets to which E is connected by many-one relationships with a double border, must be unique for the entities of E.

We should remember that the double-diamond is used only for supporting relationships. It is possible for there to be many-one relationships from a weak entity set that are not supporting relationships, and therefore do not get a double diamond.

Example 2.26: In Fig. 2.22, the relationship *Studio-of* need not be a supporting relationship for *Contracts*. The reason is that each movie has a unique owning studio, determined by the (not shown) many-one relationship from *Movies* to *Studios*. Thus, if we are told the name of a star and a movie, there is at most one contract with any studio for the work of that star in that movie. In terms of our notation, it would be appropriate to use an ordinary single diamond, rather than the double diamond, for *Studio-of* in Fig. 2.22. □

2.4.4 Exercises for Section 2.4

* **Exercise 2.4.1:** One way to represent students and the grades they get in courses is to use entity sets corresponding to students, to courses, and to "enrollments." Enrollment entities form a "connecting" entity set between students and courses and can be used to represent not only the fact that a student is taking a certain course, but the grade of the student in the course. Draw an E/R diagram for this situation, indicating weak entity sets and the keys for the entity sets. Is the grade part of the key for enrollments?

Exercise 2.4.2: Modify your solution to Exercise 2.4.1 so that we can record grades of the student for each of several assignments within a course. Again, indicate weak entity sets and keys.

Exercise 2.4.3: For your E/R diagrams of Exercise 2.2.6(a)–(c), indicate weak entity sets, supporting relationships, and keys.

Exercise 2.4.4: Draw E/R diagrams for the following situations involving weak entity sets. In each case indicate keys for entity sets.

a) Entity sets *Courses* and *Departments*. A course is given by a unique department, but its only attribute is its number. Different departments can offer courses with the same number. Each department has a unique name.

*! b) Entity sets *Leagues*, *Teams*, and *Players*. League names are unique. No league has two teams with the same name. No team has two players with the same number. However, there can be players with the same number on different teams, and there can be teams with the same name in different leagues.

2.5 Summary of Chapter 2

- *The Entity/Relationship Model*: In the E/R model we describe entity sets, relationships among entity sets, and attributes of entity sets and relationships. Members of entity sets are called entities.

- *Entity/Relationship Diagrams*: We use rectangles, diamonds, and ovals to draw entity sets, relationships, and attributes, respectively.

- *Multiplicity of Relationships*: Binary relationships can be one-one, many-one, or many-many. In a one-one relationship, an entity of either set can be associated with at most one entity of the other set. In a many-one relationship, each entity of the "many" side is associated with at most one entity of the other side. Many-many relationships place no restriction on multiplicity.

- *Keys*: A set of attributes that uniquely determines an entity in a given entity set is a key for that entity set.

- *Good Design*: Designing databases effectively requires that we represent the real world faithfully, that we select appropriate elements (e.g., relationships, attributes), and that we avoid redundancy — saying the same thing twice or saying something in an indirect or overly complex manner.

- *Referential Integrity*: A requirement that an entity be connected, through a given relationship, to an entity of some other entity set, and that the latter entity exists in the database, is called a referential integrity constraint.

- *Subclasses*: The E/R model uses a special relationship *isa* to represent the fact that one entity set is a special case of another. Entity sets may be connected in a hierarchy with each child node a special case of its parent. Entities may have components belonging to any subtree of the hierarchy, as long as the subtree includes the root.

- *Weak Entity Sets*: An occasional complication that arises in the E/R model is a weak entity set that requires attributes of some related entity set(s) to identify its own entities. A special notation involving diamonds and rectangles with double borders is used to distinguish weak entity sets.