Design of Databases; ER Model; Relational Model

(5 Jan 2018. Aula prática 1)

Introductory note

Taking into account that this course was designed for 15 hours, given the vastness of the subject, the course will be theoretical-practical and intensive, covering the various phases and tools related to the design and exploitation of Databases, with special focus on rigor of Data Models. For a supported study we suggest the following publication: Databses Systems, Database System Concepts - 5th Edition, Silberschatz, Korth and Sudarsham..

Basic Concepts

Entities and Attributes

In the context of Databases, term **Entity** means an object that is characterized by a set of attributes. In turn, term **Attribute** can be viewed simply as an object not normally characterized by other attributes, or elsewhere, that describe it. For example, *sócio* (*member* in Portuguese) entity (the member of a video club), is characterized by the following attributes: *num_sócio*, which means the number that is assigned to the member by the video club; *BI_sócio*, whose meaning is, of course, the BI number of the partner; *nome_sócio*, which means the name that is in the member's BI; *data_nsc_sócio*, which is the birthdate of the member; *morada_sócio*, (address) *tlf_sócio* (phone) and *sexo_sócio* (sex) are three other attributes whose meaning does not raise doubts.

Since *sócio* is a person, we could at first consider to include many other attributes: *num_contribuinte* (taxpayer number), *local_nascimento* (birth place), etc. However, it is not predictable any interest in these attributes for the possible applications in the context of a video club. The *sócio* entity is therefore characterized by these attributes which, in turn, are not characterized by more attributes, being completely described by their meaning. However, the meaning of each attribute must be precise, that is, it can not be vague. In this way it is guaranteed that the meaning of the entities will also be well defined.

A set of entities is often represented in a table, which is a structure that can be implemented through the tools that are part of a Database Management System (DBMS). According to the representation E-R (Entity-Relationship), the set of *sócio* entities can be represented as:

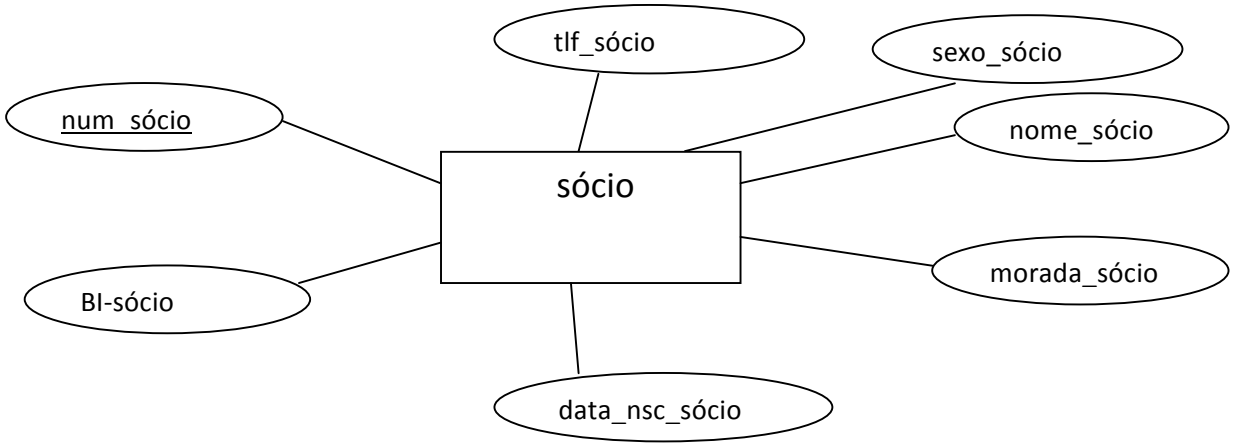

Figura 1

In E-R notation, **rectangles** represent sets of entities; **diamonds** represent sets of relations (we will address them later); **lines** link attributes to sets of entities and sets of associations/relationships; **ellipses** represent attributes; **underline** attributes represent attributes of the primary key (a concept focused below).

Functional Dependencies, Superkeys and Keys

When a set of attributes α uniquely determines a set of attributes β, considering a set of entities, it is said that the **functional dependence** α → β is true in this set of entities. For example, in the *sócio* entity set, *num_sócio* attribute uniquely determines *nome_sócio*, that is, there are no 2 different nome_sócio for the same *num_sócio*, whatever the entity set. This means that the functional dependency *num_sócio* → *nome_sócio* is true (or exists) in the *sócio* entity set. Similarly, a *num_sócio* → *tlf_sócio* is also true in *sócio*, as one understands. Thus, it is easy to conclude that a *num_sócio* → *tlf_sócio, nome_sócio* is also true in *sócio*. That is, if the *num_sócio* attribute determines several attributes separately, it also determines them simultaneously. It is also easy to see that *nome_sócio* →*tlf_sócio* is true, since each *nome_sócio* only matches one *tlf_sócio*. It is also understood that any attribute determines itself, that is, α → α, a functional dependence also known as **trivial**.

However, neither *tlf_sócio*→ *nome_sócio* nor *morada_sócio* →*BI_sócio* are true in *sócio*, among other functional dependences, since, considering the first case, there may be two entities with different values for the attribute *nome_sócio* but with the same value for the *tlf_sócio* attribute: two *sócio* living in the same house may have the same telephone number.

In the particular case where a set of attributes α determines the entire set of attributes constituting the entity (usually referred to as the R **schema**), that is, α → R, it is said that α is a superkey in the set of entities. In other words, when a set of attributes uniquely determines each entity within the entity set, it is said that this set of attributes is a **superkey** of the entity set. Often the set of attributes consists of

only one attribute, as is the case of *num_sócio* as it is easily perceived. Of course, if we add the *data_nsc_sócio* attribute to the *num_sócio* attribute, the set formed by these two attributes also uniquely determines each entity, thus being another superkey. There is, however, a particular case of superkeys: the minimal superkeys, that is, those that will cease to be superkeys, if any of the attributes are removed. Minimal superkeys are called **keys**.

Thus, it is easy to see that *num_sócio* is a key, since *num_sócio→num_sócio*, *nome_sócio, BI_sócio, tlf_sócio, morada_sócio, data_nsc_sócio*; or even in abbreviated form, *num_sócio→R* (as we have said, symbol *R* is reserved to indicate the whole set of attributes). For the same reasons, *BI_sócio* is also a key. The *num_sócio* and *BI_sócio* attributes are called **candidate keys**. Between the two attributes, the one that physically is shorter will be chosen and will be called **primary key**, sometimes also known by **designated key**.

The reader is invited to reflect on what other sets of attributes in the *sócio* schema could also be key. Can the set *nome_sócio, morada_sócio* be a key?

Design of a Data Model using E-R Diagram. Problem 1

Suppose we want to create a data model and then the corresponding DB for a video club.

Besides the *sócio*, which we have talked about, there are films to rent to the members (sócios). Films are characterized by the name that identifies them, the year of launch, the price per day of rental, the number of days of rental "without fine" and the price per day beyond the expected date of delivery. At least one actor is involved in the film, as well as at least one director. Let us assume that a film always belongs to one genre (comic, dramatic, etc.) and always belongs to a single publisher who is identified by its name. Films may have one or more copies. Copies belonging to a movie have copy consecutive numbers starting at 1. In other words, in a movie with 3 copies, copies will be numbered 1, 2, and 3. Copies of the movies are rented to members. A rental is always made on a date and is also characterized by a predictable date of return of the copy. The state of preservation of the copy is recorded in the return act; in this act there may be a fine for late delivery.

Design a Data Model for this video club. **Keep in mind that data must be organized independently of the possible applications it will work with.**

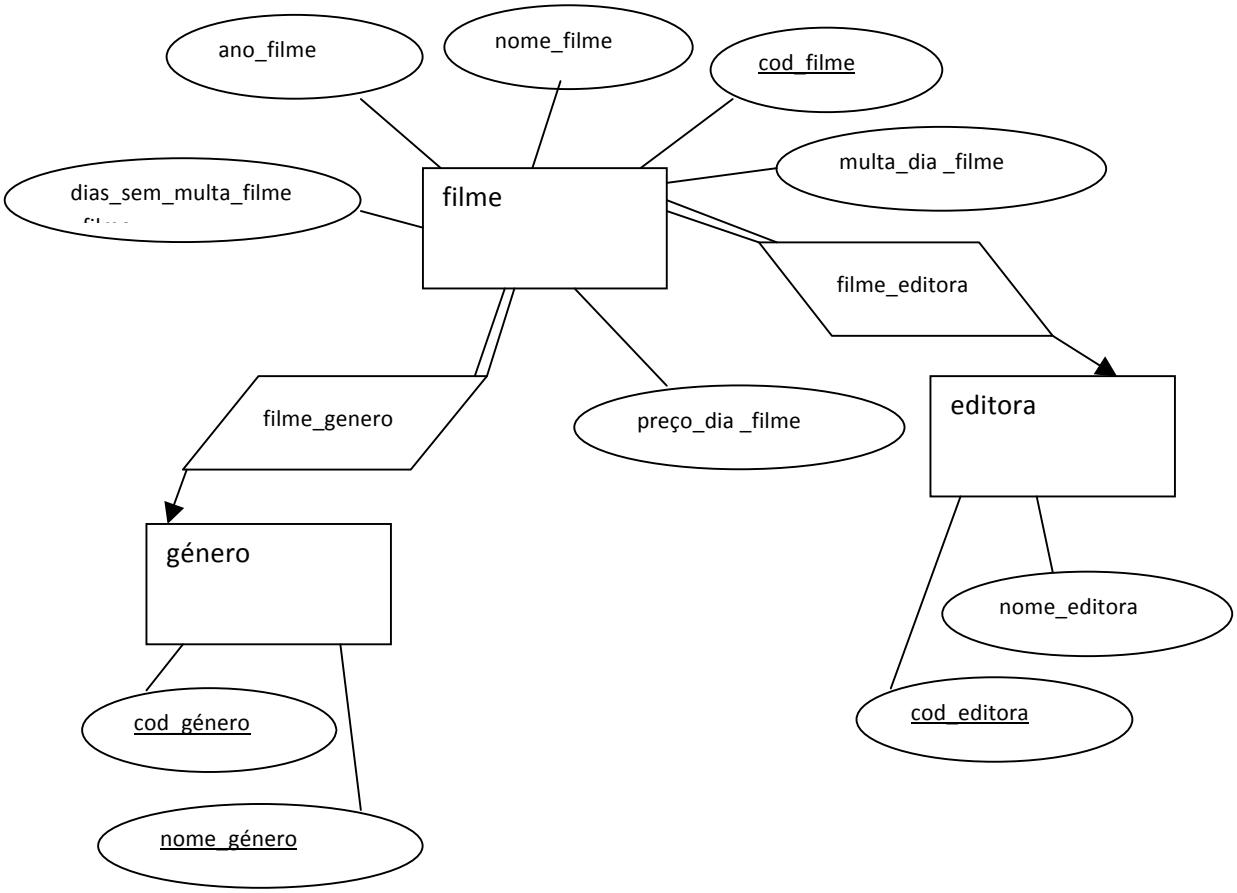

Figura 2

The *nome_filme* (movie name) attribute uniquely identifies the movie, that is, it is a candidate key in the entity / schema set, since we can assume that there are no two movies with the same name. However, *nome_filme* is an attribute that can occupy 30 or 40 characters and is therefore not advisable to be a designated key. In these cases it is customary to create a numerical and much shorter code (*cod_filme*), which also uniquely identifying each entity, being chosen for designated key. The same technique is applied in the sets of *editora* (publisher) and *género* (genre) entities, that is, the attributes *cod_editora* and *cod_género* have been created.

There is a **many-to-one relationship** from *filme* to *genre*; the diamond represents the relation; the arrow indicates the direction of the many-to-one association. On the other hand, the participation of *filme* in *filme_género* is total, that is, each particular entity in *filme* (each concrete film) always belongs to a specific genre; the double dash represents this participation of totality.

From the diagram, we see that, analogously, there is another many-to-one relationship from *filme* to *editora*.

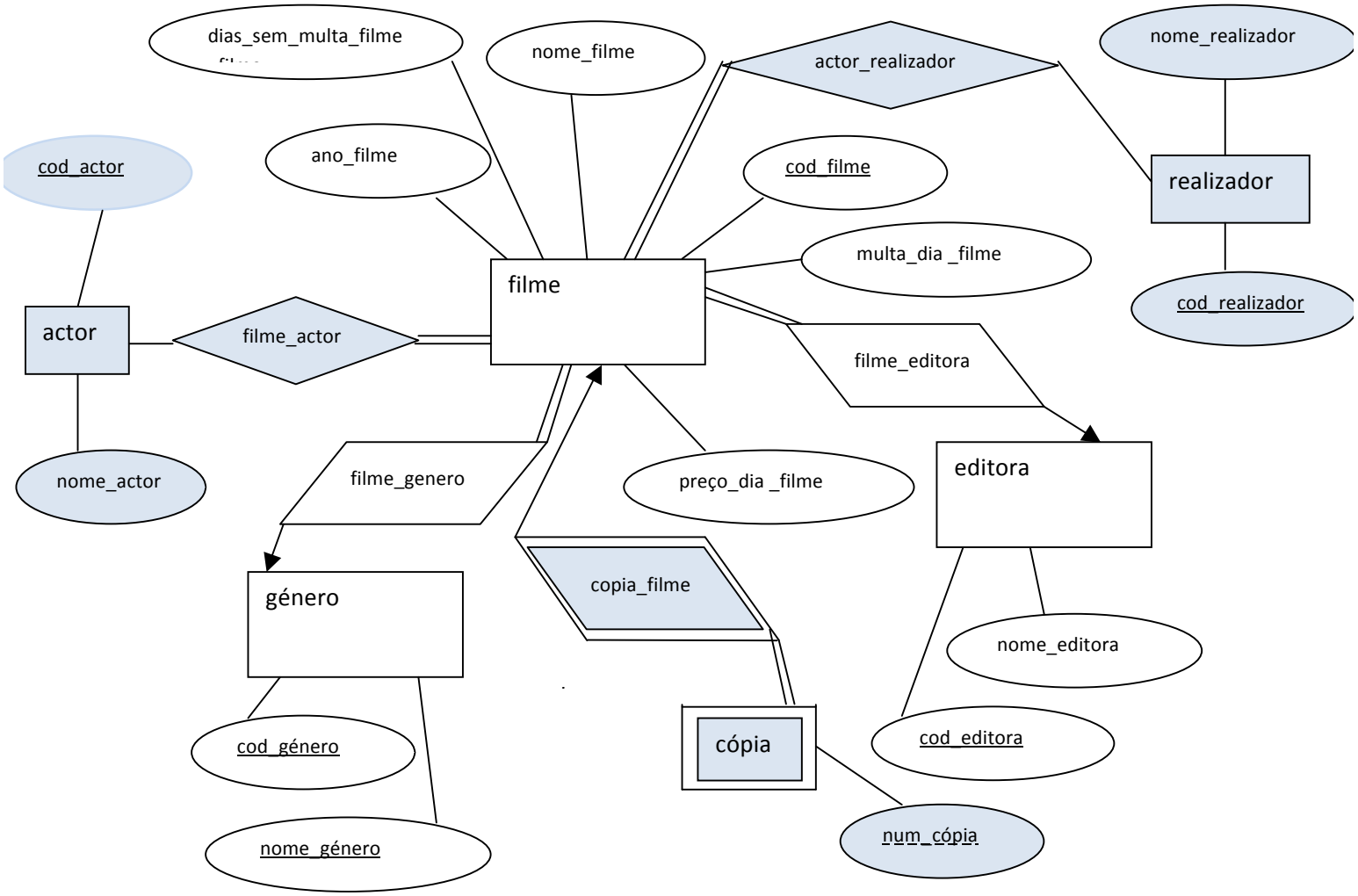

Figura 3

In the diagram, the shaded parts are not part of the DER notation, they serve only to highlight what has been added to the previous DER.

Thus, there is a relation / association between *filme* and *actor*. The absence of an arrow in both directions indicates that the association is **many-to-many**. However, the participation of *filme* in *filme_actor* is total, that is, each movie must always have at least one actor. The relation *filme_actor* will have as a candidate key (and designated) the combination of the attribute that is designated key in *film* and the attribute that is designated key in actor, that is, (*cod_filme, cod_actor*). The order by which the constituent attributes of the key are arranged can be any one at the beginning, as we shall see later, we can opt for the disposition that favors the most wanted searches.

Analogous to the relation / association *film_actor*, there is another between *filme* and *realizador* (director); the considerations to be taken into account in this case are identical. See diagram of figure 3.

According to the so-called functional specifications, that is, the problem statement, the copy numbers will be assigned from 1 for each film; this means that in the set of *cópia* entities there may be several entities with the same value for the *num_copy* attribute, which prevents this attribute of being alone a designated key. In these cases, when the set of attributes that arise, say, "naturally" in the set of entities, is not enough to form a key, then the set of entities is considered **weak** and the **double line** that delimits the *cópia* rectangle characterizes this. In these cases, in order to obtain a designated key, it is necessary to add other attributes that are part of other sets of entities connected to this set, to the so-called discriminant attribute --- in this case *num_cópia*. So, as we know, every copy belongs to one and only one movie --- see double line from *cópia* to *cópia_filme* and an arrow from *cópia_filme* to *filme* --- then the candidate (and designated) key includes the attribute that is key in *filme*, that is, *cod_filme*, in addition to the discriminant attribute, *num_copia*, as has already been said; this inclusion is represented by the "double diamond" of *cópia_filme* in the same figure 3.

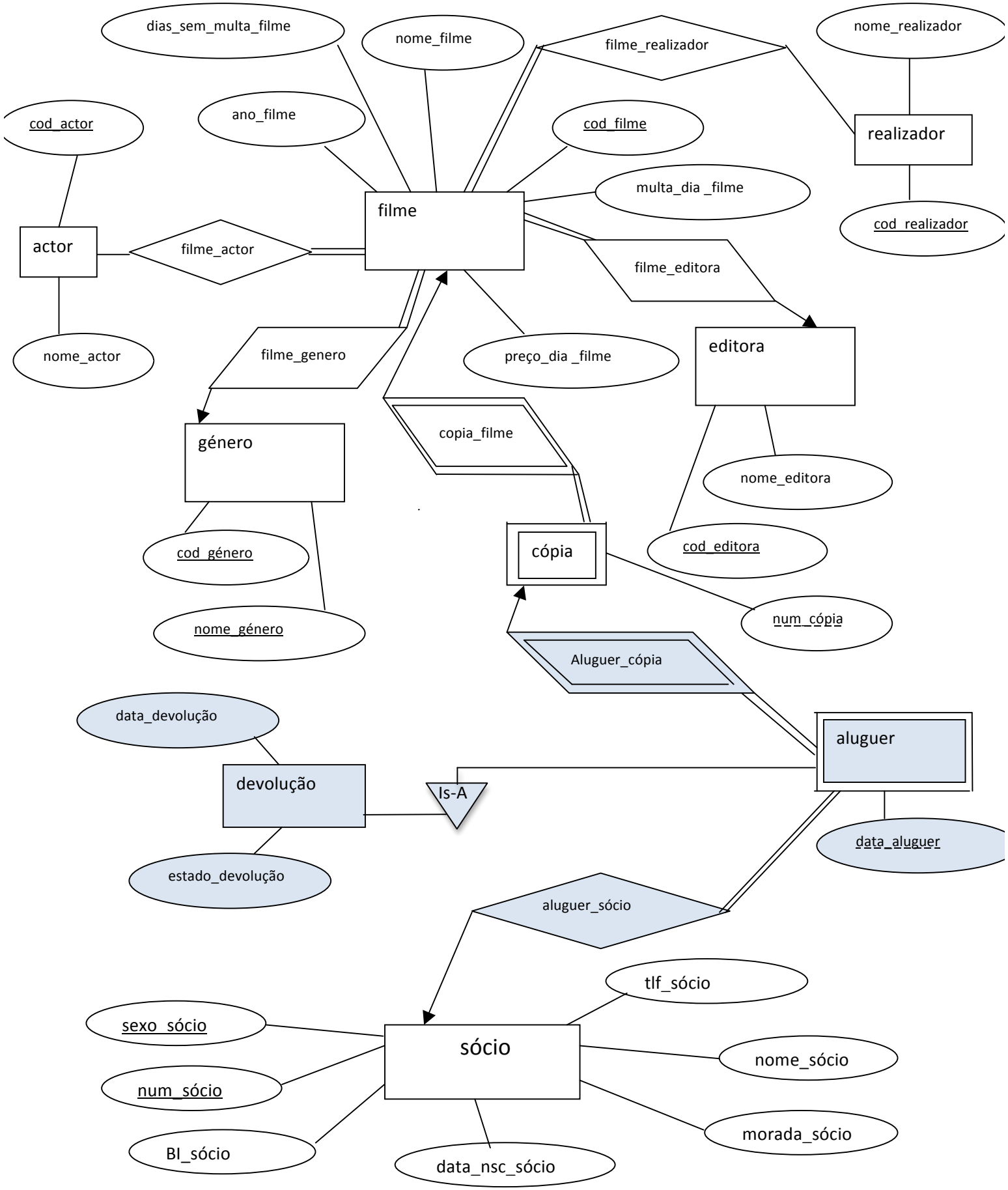
dias_sem_multa_filme
nome_filme
filme_realizador
nome_realizador
cod_actor
ano_filme
cod_filme
realizador
multa_dia _filme
filme
actor
filme_actor
cod_realizador
filme_editora
nome_actor
filme_genero
preço_dia _filme
editora
copia_filme
género
nome_editora
cod_género
cópia
cod_editora
nome_género
num_cópia
Aluguer_cópia
data_devolução
aluguer
devolução
Is-A
data_aluguer
estado_devolução
aluguer_sócio
tlf_sócio
sexo_sócio
sócio
nome_sócio
num_sócio
morada_sócio
BI_sócio
data_nsc_sócio

Figura 4

According to the statement, regarding the set of *aluguer* (rental) entities, of course, the data_aluguer (rental date) is an attribute that, alone, is not enough to constitute the key, since there will be several leases made on the same date. Thus, this set can also be considered as weak --- please note that the weak adjective is only due to the DER notation; it does not mean that it is a group of entities of less importance as by suggestion we can be led to think. Thus, it is easy to understand that we need attributes that, when combined, they uniquely identify any entity in *aluguer* (any rental). As a result, the *aluguer_cópia* relation is represented with a double diamond and the candidate key of *aluguer* will therefore consist of the combination of 3 attributes: (*cod_filme, num_copy*, *data_aluguer*); here, the order of the attributes is not important at this stage too.

Of course, any *aluguer* (rental) is made to a *sócio* (member), hence the many-to-one relationship from *aluguer* to *sócio*. See Figure 4.

The reader is asked to answer the following question: why not replace, in *aluguer,* the attributes that are key in *cópia* by the attribute that is key in *sócio*?

Considering the event of *devolução* (return) of the rental, which means the delivery of the physical copy on a given date, one might think of adding an attribute, *data_devolução* (date of delivery), to the set of *aluguer* attributes. But while this is an intuitive solution, it is not the most robust solution. In fact, if we opted for it, the *data_devolução* attribute would be empty most of the time, which must be avoided. In addition, the *devolução* event has other, say, natural attributes, among them the state of conservation of the copy at the time it was returned; this would imply having one more attribute, the estado_devolução (status of the delivered copy), which would also be empty most of the time; such as the *data_devolução* (date of delivery) attribute, the *estado_devolução* would be filled only when the copy was delivered --- we will return to this subject later ---. Having said this, the most robust solution, and therefore advisable, is to consider that devolução is a set of entities.

It turns out that between *aluguer* and *devolução* the cardinality is **one-to-one**, with the particularity that, to one *devolução* it always corresponds a *aluguer*. However, it is not true that every *aluguer* correponds to a *devolução*, since each *devolução* takes place only some days after the correponding *aluguer*. In these one-to-one cases the set of entities with the full participation inherits as its designated key, the designated key of the other set of entities. This circumstance configures a structure known as Is-A (is one of); more specifically, a *devolução* is a particular case of *aluguer*, that is, a type of rental. See Figure 4.

Translating the E-R data model to table schemas

The translation from ER data model to table schemas is done with simple rules. Roughly, each set of entities correponds to a table. Let's start by translating entity sets that do not dependen

on many-to-one relationships from them; for example the entity sets *sócio*, *genre*, *editora*, *actor* and *realizador.*

Sócio=(num_sócio, nome_sócio, bi_sócio, data_nsc_sócio, morada_sócio, tlf_sócio, sexo_sócio)
Género=(cod_género, nome_genero)
Editora=(cod_editora, nome_editora)
Actor=(cod_actor, nome_actor)
Realizador=(cod_realizador, nome_realizador)

In these cases, just include as fields / attributes of the table, the same attributes that are part of the set of entities in E-R model, ignoring the links to other relationships. The underlined attributes are the designated keys. Among the candidate keys in a *num_sócio* e *bi_sócio*, the former was chosen as the designated key (primary) because it was shorter.

Now let's look at the case of the *filme* table.

Filme=(cod_filme, nome_filme, ano_filme, preço_dia_filme, dias_sem_multa_filme, multa_dia_filme, cod_género, cod_editora)

In this case, in addition to the attributes that are represented in ellipses in the E-R model surrounding the *filme* rectangle (as in the previous case for *sócio*, *género, editora*, *actor* and *realizador*) we must include as attributes those (attributes) that fwork as designated keys in entity sets with which there is a many-to-one relationship from the *filme*. Hence the *filme* table also includes the attributes *cod_género* and *cod_editora*. In fact, each concrete *filme* (movie) has exactly one specific *editora* (publisher), so the attribute that is the designated key of the *editora* (the "pointer" for the publisher) can and should appear in the attributes of the *filme* table. An analogous reasoning applies to the designated key of the *género* table which is also 'imported' as an attribute for the *filme* table.

It should be noted that the relation *filme_género* does not correspond to any table. In fact, it could exist with the structure *filme_género* ({*cod_filme, cod_género*}), that is, it would have the same key as the f*ilme* table; and so in the so-called **simplified table model**, the non-key attributes of this table are included in the table table. The same rationale is applied with respect to the *filme-editora* relationship and filme_editora table.

However, there is a different cardinality relationship (many_to_many) from *actor* to *filme*. This type of relationship always gives an additional table that normally receives the same name as the relationship; in this case filme_actor. The attributes of this table are the attributes that make up the keys of each set of entities that are at the origin of this many-to-many relationship, and the union / combination of all these attributes is the key to this new table. That is:

Filme_actor=(cod_filme,cod_actor)

The same reasoning applies to the *filme_realizador* relationship; hence the table:

Filme_realizador=(cod_filme,cod_realizador)

Notice that we could admit an attribute that would portray in free text the role of the concrete actor in the concrete *filme* (role_in_filme_actor). This attribute would be appended to the filme_actor table but would not be part of the key.

For the cópia table, there are only the num_cópia attributes, which is the so-called discriminant attribute as already mentioned, and cod_filme (the designated key in the movie table; see the arrow direction to the *cópia_filme* relationship). This last attribute will complete the key assigned in this table. In fact, a concrete copy always belongs to a given film:

Cópia=(cod_filme,num_cópia)

Regarding the aluguer table, given that it derives from a set of weak entities, its key will be the discriminant attribute data_aluguer and the attributes that constitute the designated key of the ccópia table. In addition to these, there will also be the num_sócio attribute, not used in the key of aluguer, but being key in the sócio table; see the arrow and its meaning in the aluguer relationship in figure 4.

Aluguer=(cod_filme,num_cópia,data_aluguer,num_sócio)

Finally, the devolução table, which is derived from the set of entities with the same name, will have the same key as the aluguer table, since there is a one_to_one relation between the two sets of entities but with a full participation by the devolução. This table will also have attributes, data_devolução and estado_devolução; these attributes do not form the table key.

Devolução=(cod_filme, num_cópia, data_aluguer, data_devolução, estado_devolução)

Exercise 1

Create a Data Model for a physician's office. Consider appointments, patients, doctors and specialties (stomatology, pneumology, etc.). Every doctor can be a specialist in more than one

specialty. During the appointments, doctors prescribe drugs (one or more). There are also appointment bookings.

Exercício 2 (**parte do projecto para avaliação**)

Together with your work group, create a data model on a subject of your choice. The chosen subject must be approved by the teacher of this course.