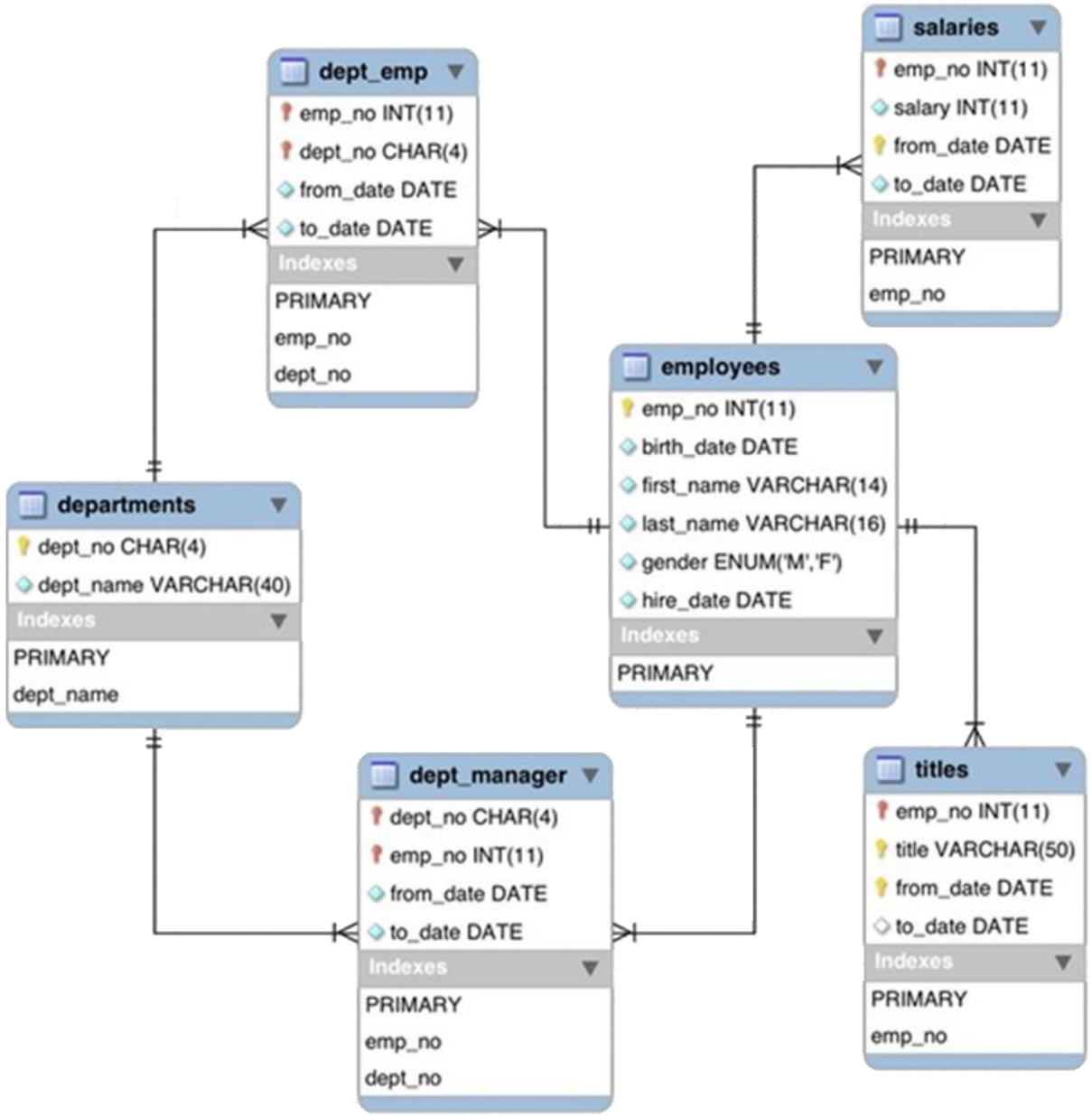


INTRODUCTION TO DATABASES USING ORACLE

420-983-VA

DATABASE DESIGN I

CONCEPTUAL MODELS

Consider the process whereby a new database, such as our movie database, is created.

We begin with a design phase, in which we address and answer questions about what information will be stored, how information elements will be related to one another, what constraints such as keys or referential integrity may be assumed, and so on. This phase may last for a long time, while options are evaluated and opinions are reconciled.

CONCEPTUAL MODELS

Since the great majority of commercial database systems use the relational model, we might suppose that the design phase should use this model too. However, in practice it is often easier to start with a higher-level model and then convert the design to the relational model. The primary reason for doing so is that the relational model has only one concept — the relation — rather than several complementary concepts that more closely model real-world situations.

CONCEPTUAL MODELS

There are several options for the notation in which the design is expressed. The first, and oldest, method is the E/R "entity-relationship model" . A more recent trend is the use of UML "Unified Modeling Language", a notation that was originally designed for describing object-oriented software projects, but which has been adapted to describe database schemas as well. Finally, another model considered is ODL "Object Description Language", which was created to describe databases as collections of classes and their objects.

CONCEPTUAL MODELS

The Entity/Relationship Model

In the E/R model, the structure of data is represented graphically, as an “entity-relationship diagram,” using three principal element types:

1. Entity sets.
2. Attributes.
3. Relationships.

CONCEPTUAL MODELS

The Entity/Relationship Model

1. Entity Sets.

An entity is an abstract object of some sort, and a collection of similar entities forms an entity set. An entity in some ways resembles an "object" in the sense of object-oriented programming. Likewise, an entity set bears some resemblance to a class of objects.

Example: Each movie is an entity, and the set of all movies constitutes an entity set. Likewise, the stars are entities, and the set of stars is an entity set. A studio is another kind of entity, and the set of studios is a third entity set that will appear in our examples.

Movies relation

Movies(title:string, year:integer, length:integer, genre:string, studioName:string, producerC#:integer)

CONCEPTUAL MODELS

The Entity/Relationship Model

2. Attributes.

Entity sets have associated attributes, which are properties of the entities in that set. For instance, the entity set Movies might be given attributes such as title and length. It is common for entity sets to be implemented as relations, although not every relation in the final relational design will come from an entity set.

We assume that attributes are of primitive types, such as strings, integers, or reals. There are other variations of this model in which attributes can have some limited structure such as struct (similar to C) or set values.

CONCEPTUAL MODELS

The Entity/Relationship Model

2. Attributes.

Example: The attributes for the entity set Movies resemble the attributes of the relation Movies in our example.

Movies relation

Movies(title:string, year:integer, length:integer, genre:string, studioName:string, producerC#:integer)

CONCEPTUAL MODELS

The Entity/Relationship Model

2. Multivalued Attributes.

Attributes which values are composed of a set of primitive values are called Multivalued Attributes.

Example: Continuing with the Movies relation we may want to consider the set of distribution years, when the movies was distributes in theaters.

Movies relation

Movies(title, yea, length, genre, studioName, producerC#, years)

We will see later, during the normalization phase, that relations with multivalued attributes will be decomposed in relations with atomic types.

CONCEPTUAL MODELS

The Entity/Relationship Model

3. Relationships.

Relationships are connections among two or more entity sets.

Example: If Movies and MovieStars are two entity sets, we could have a relationship Stars-in that connects movies and stars. The intent is that a movie entity m is related to a star entity s by the relationship Stars-in if s appears in movie m. While binary relationships, those between two entity sets, are by far the most common type of relationship, the E/R model allows relationships to involve any number of entity sets.

Movies relation

```
Movies(title:string,   year:integer,   length:integer,   genre:string,   studioName:string,
producerC#:integer )
```

MovieStar relation

```
MovieStar( name:string, address:string, gender:char, birthdate:date )
```

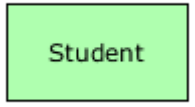
CONCEPTUAL MODELS

The Entity/Relationship Model

Entity-Relationship Diagrams.

An E/R diagram is a graph representing entity sets, attributes, and relationships. Elements of each of these kinds are represented by nodes of the graph, and we use a special shape of node to indicate the kind, as follows:

- Entity sets are represented by rectangles.



- Attributes are represented by ovals.

- Multivalued Attributes are represented using double bordered ovals.

- Relationships are represented by diamonds.

Edges connect an entity set to its attributes and also connect a relationship to its entity sets.

CONCEPTUAL MODELS

The Entity/Relationship Model

Entity-Relationship Diagrams.

Example: consider Movie database relations.

Movies(title:string, year:integer, length:integer, genre:string, studioName:string, producerC#:integer)

MovieStar(name:string, address:string, gender:char, birthdate:date)

StarsIn(movieTitle:string, movieYear:integer, starName:string)

MovieExec(name:string, address:string, cert#:integer, netWorth:integer)

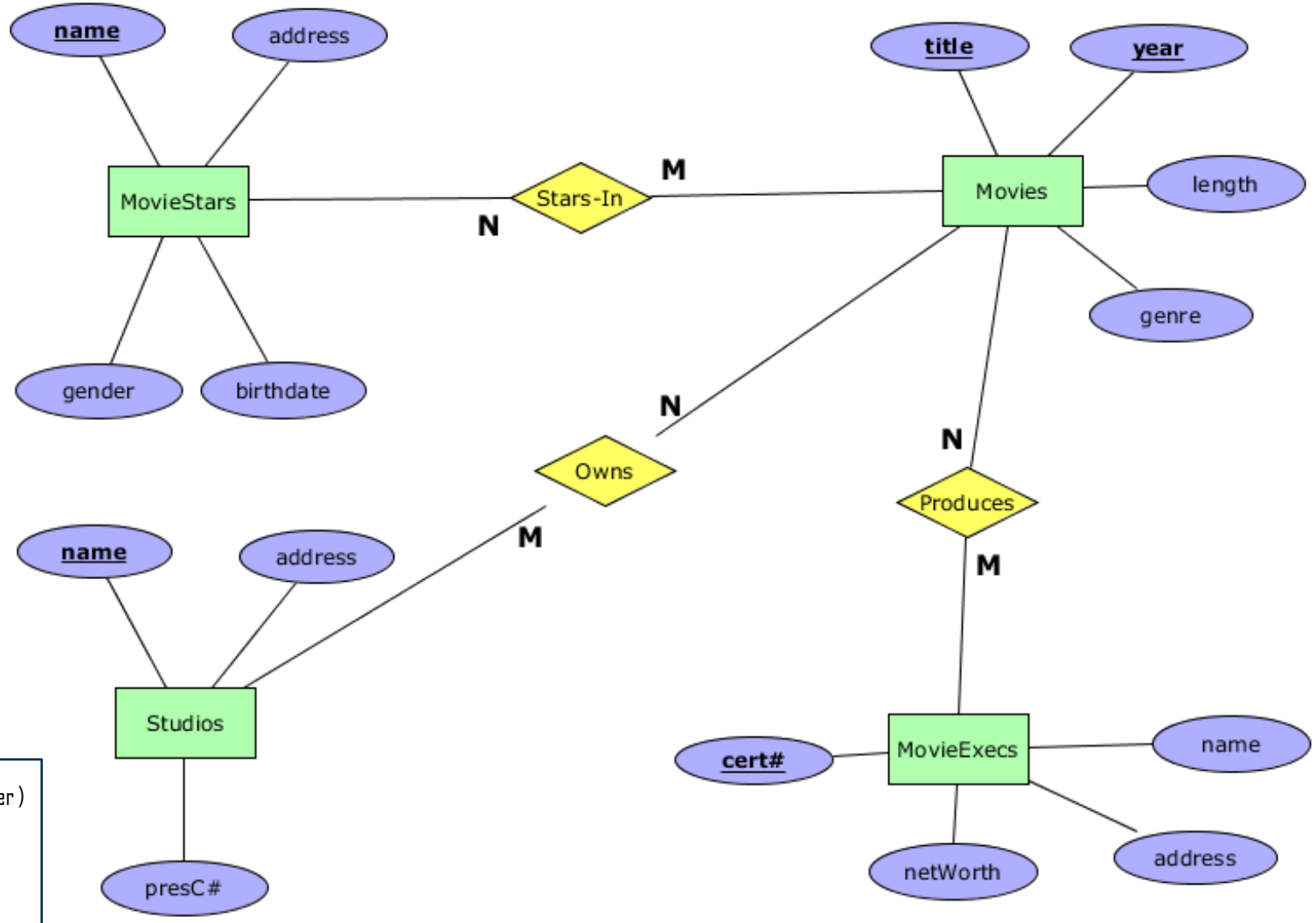
Studio(name:string, address:string, presC#:integer)

CONCEPTUAL MODELS

The Entity/Relationship Model

Entity-Relationship Diagrams.

Example: Movie E/R Diagram.



Movies database relations

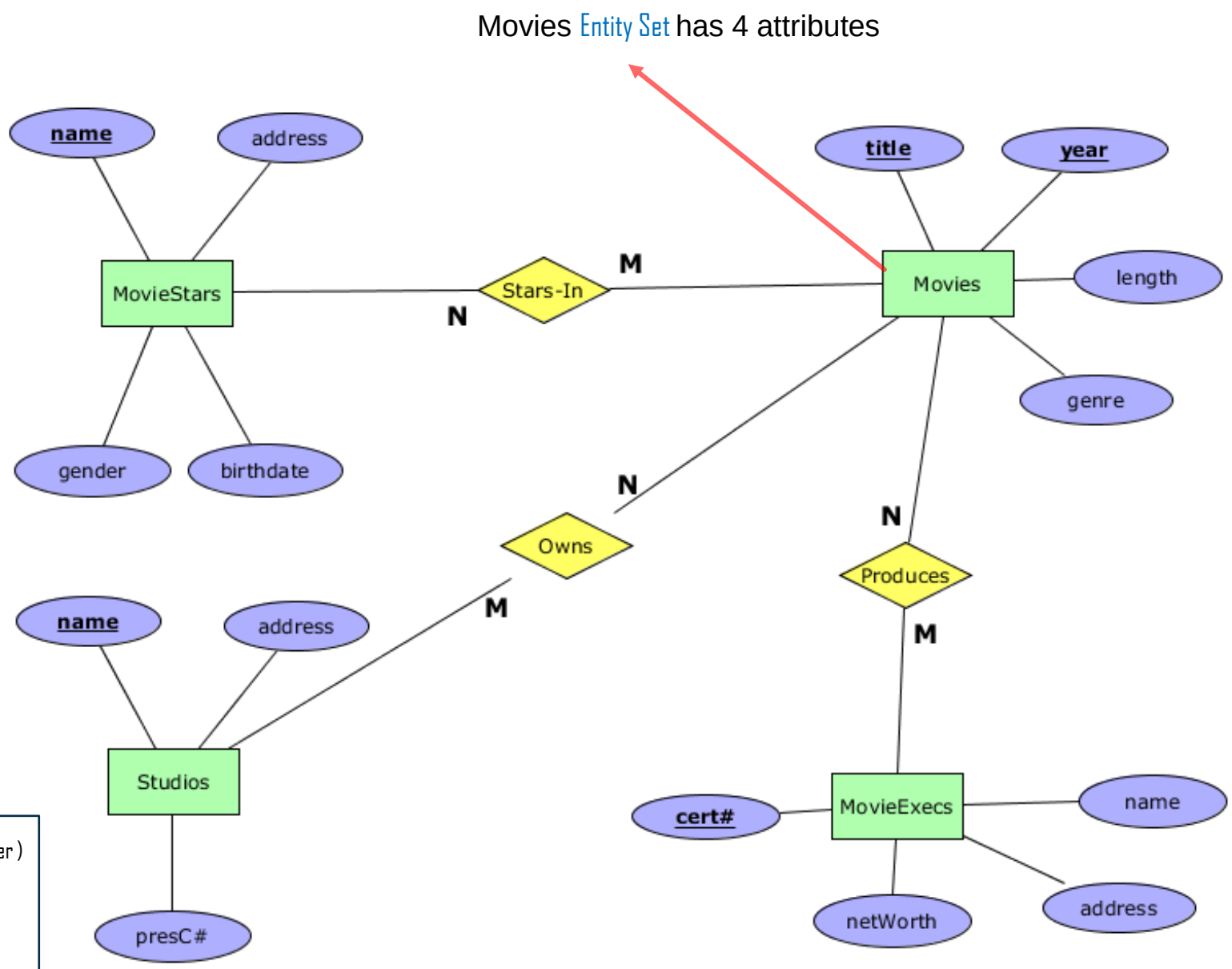
Movies(title:string, year:integer, length:integer, genre:string, studioName:string, producerC#:integer)
MovieStar(name:string, address:string, gender:char, birthdate:date)
StarsIn(movieTitle:string, movieYear:integer, starName:string)
MovieExec(name:string, address:string, cert#:integer, netWorth:integer)
Studio(name:string, address:string, presC#:integer)

CONCEPTUAL MODELS

The Entity/Relationship Model

Entity-Relationship Diagrams.

Example: Movie E/R Diagram.



Movies database relations

Movies(title:string, year:integer, length:integer, genre:string, studioName:string, producerC#:integer)
MovieStar(name:string, address:string, gender:char, birthdate:date)
StarsIn(movieTitle:string, movieYear:integer, starName:string)
MovieExec(name:string, address:string, cert#:integer, netWorth:integer)
Studio(name:string, address:string, presC#:integer)

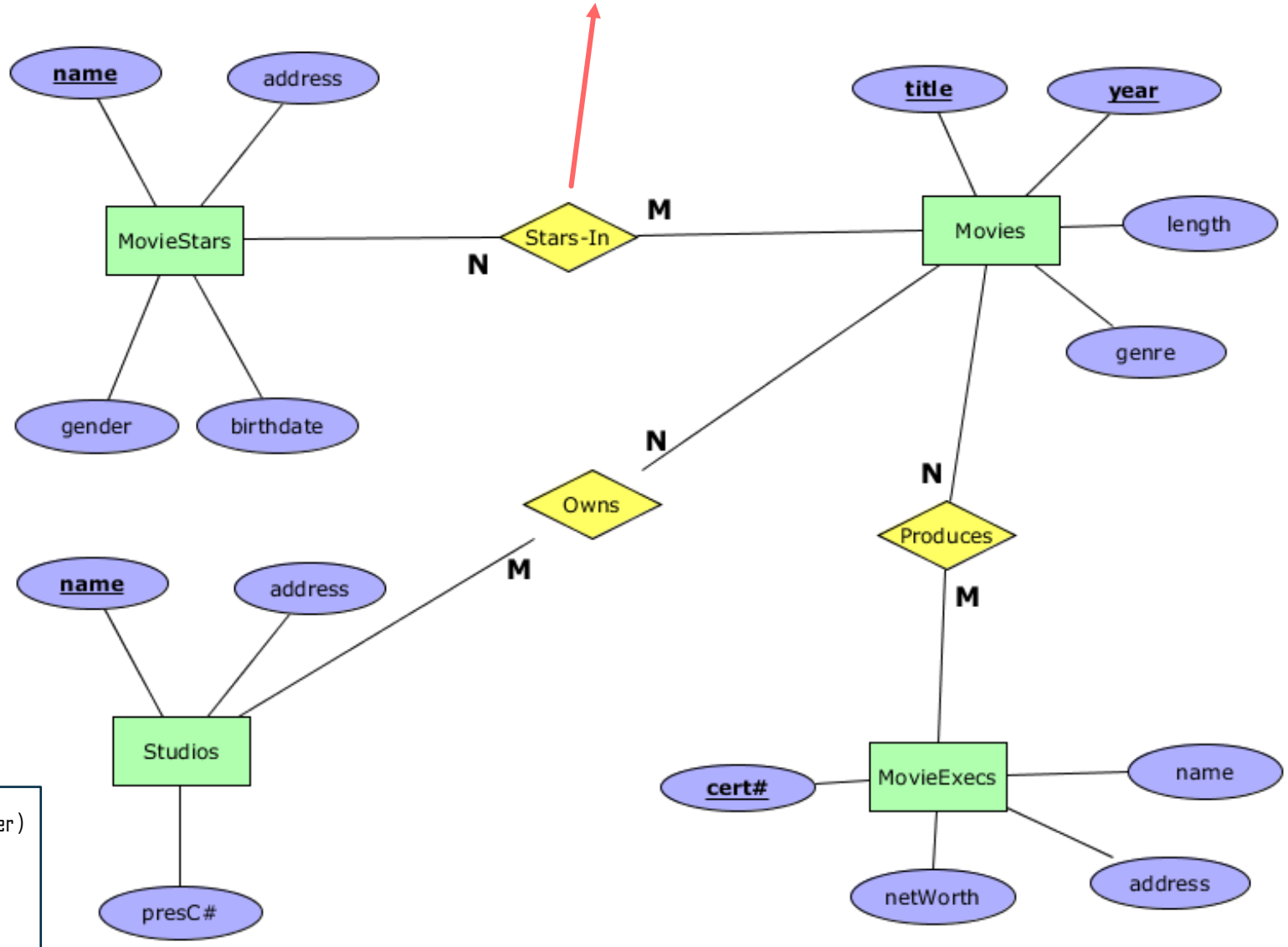
CONCEPTUAL MODELS

The Entity/Relationship Model

Entity-Relationship Diagrams.

Example: Movie E/R Diagram.

Stars-In is a relationship connecting each movie to the stars of that movie.



Movies database relations

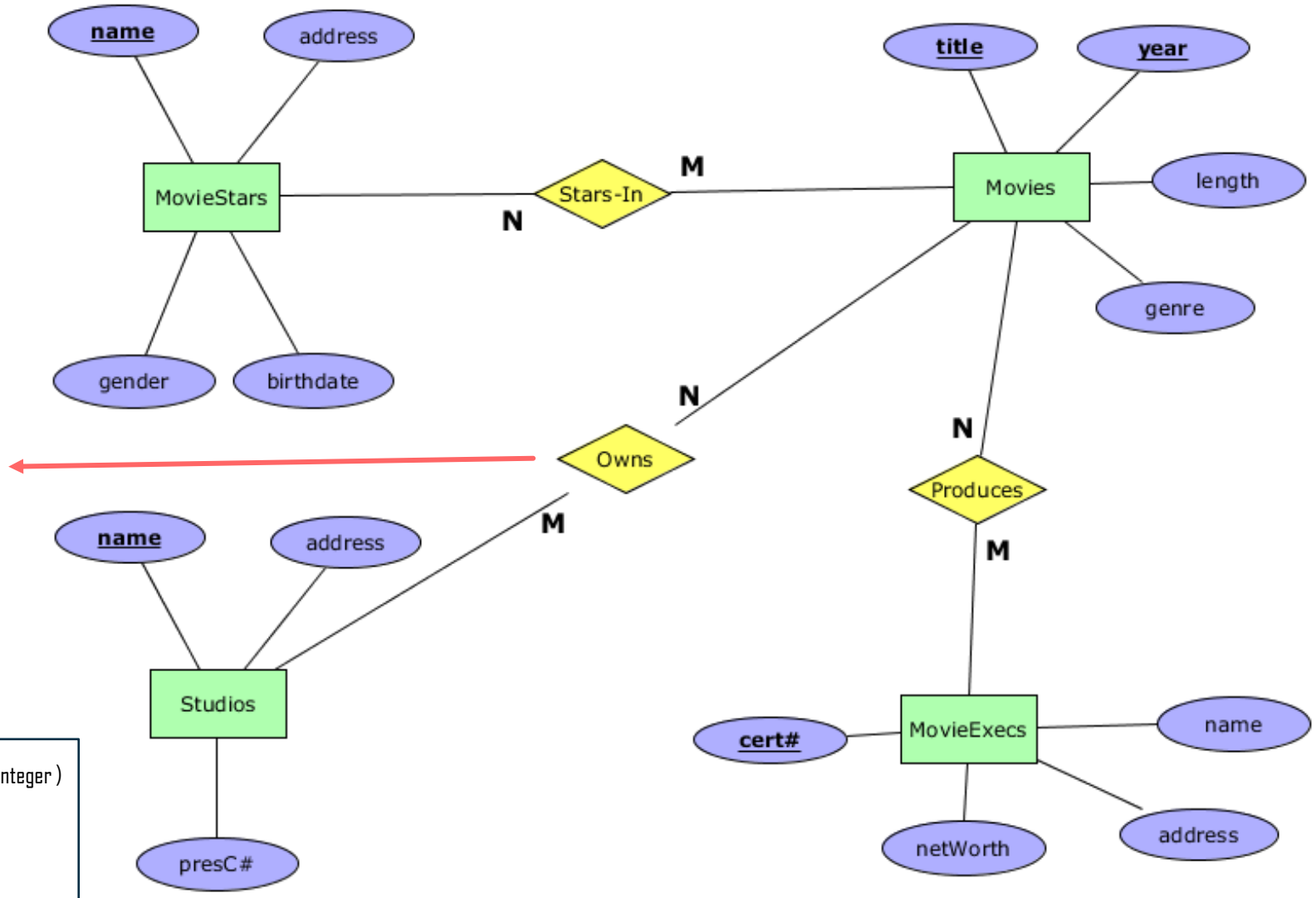
Movies(title:string, year:integer, length:integer, genre:string, studioName:string, producerC#:integer)
MovieStar(name:string, address:string, gender:char, birthdate:date)
StarsIn(movieTitle:string, movieYear:integer, starName:string)
MovieExec(name:string, address:string, cert#:integer, netWorth:integer)
Studio(name:string, address:string, presC#:integer)

CONCEPTUAL MODELS

The Entity/Relationship Model

Entity-Relationship Diagrams.

Example: Movie E/R Diagram.



Owns connects each movie to the studio that owns the movie.

Movies database relations

Movies(title:string, year:integer, length:integer, genre:string, studioName:string, producerC#:integer)
MovieStar(name:string, address:string, gender:char, birthdate:date)
StarsIn(movieTitle:string, movieYear:integer, starName:string)
MovieExec(name:string, address:string, cert#:integer, netWorth:integer)
Studio(name:string, address:string, presC#:integer)

CONCEPTUAL MODELS

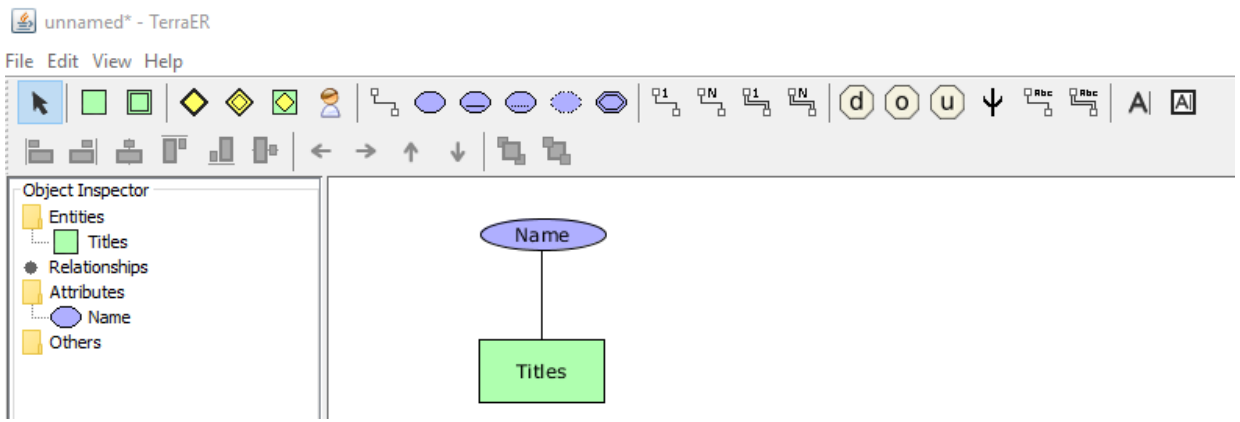
Tools to build Entity-Relationship Diagrams

Free tools to build E/R diagrams:

1. TerraER: http://www.terraer.com.br

Stand alone Java application

- Intuitive interface
- Easy to learn
- Save in XML
- Clipboard Transfer
- Printing




```
▼<drawing>
  ▼<figures>
    ▼<ent id="0">
      ▼<children>
        ▼<r id="1" x="96" y="103" w="80" h="40">
          ▼<a>
            ▼<fillColor>
                <color rgba="#ffebffe8"/>
              </fillColor>
            </a>
          </r>
        ▼<t id="2" x="119.9453125" y="115.7080078125">
          ▼<a>
            ▼<text>
                <string>Titles</string>
              </text>
            </a>
          </t>
        </children>
      </ent>
    ▼<atr id="3" nullable="false" attributeType="VARCHAR2(128)">
      ▼<children>
        ▼<e id="4" x="97" y="26" w="80" h="20">
          ▼<a>
            ▼<fillColor>
                <color rgba="#ffffebeb"/>
              </fillColor>
            </a>
          </e>
        ▼<t id="5" x="119.498046875" y="28.7080078125">
          ▼<a>
            ▼<text>
                <string>Name</string>
              </text>
            </a>
          </t>
        </children>
      </atr>
    ▼<lcaf id="6">
      ▼<points>
          <p colinear="true" x="136.88505747126436" y="46" c1x="0" c1y="0" c2x="0" c2y="0"/>
          <p colinear="true" x="136.22988505747128" y="103" c1x="0" c1y="0" c2x="0" c2y="0"/>
        </points>
      ▼<startConnector>
        ▼<rConnector id="7">
          ▼<Owner>
              <atr ref="3"/>
            </Owner>
          </rConnector>
        </startConnector>
      ▼<endConnector>
        ▼<rConnector id="8">
          ▼<Owner>
              <ent ref="0"/>
            </Owner>
          </rConnector>
        </endConnector>
      </lcaf>
    </figures>
</drawing>
```

CONCEPTUAL MODELS

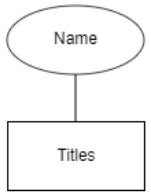
Tools to build Entity-Relationship Diagrams

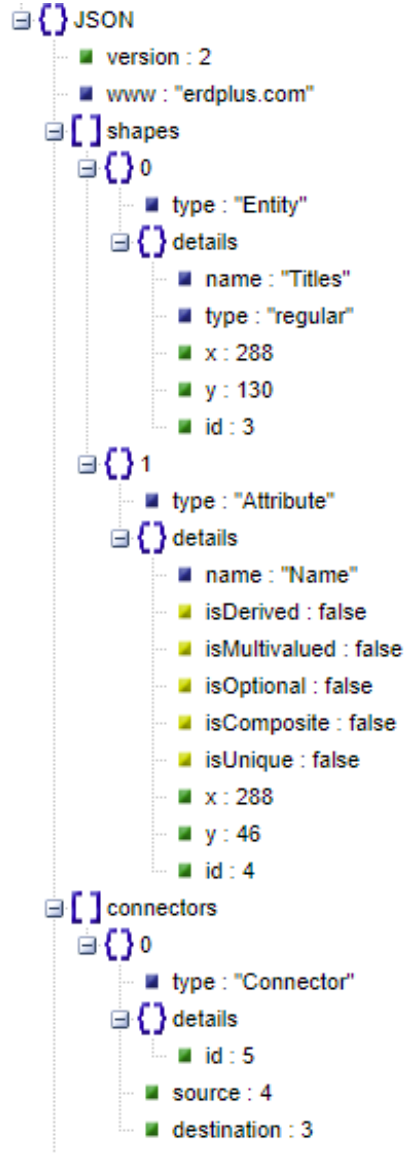
Free tools to build E/R diagrams:

2. ERDPlus: https://erdplus.com/standalone

Web based application

- Intuitive interface
- No need to install locally
- Creates also Relational Model
- Save in JSON



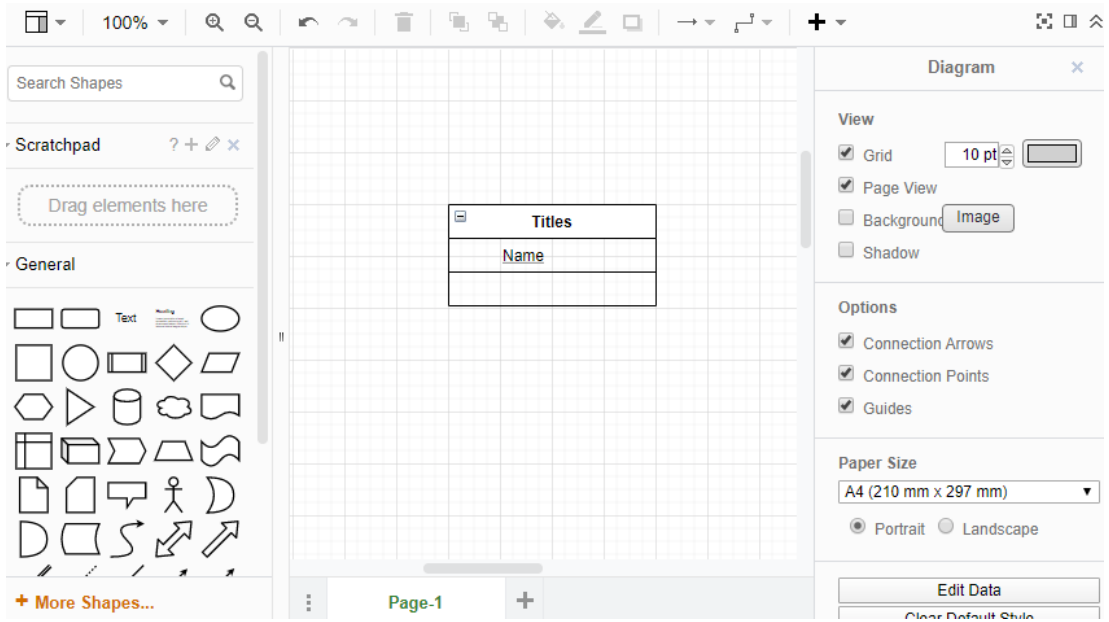


CONCEPTUAL MODELS

Tools to build Entity-Relationship Diagrams

Free tools to build E/R diagrams:

3. Diagrams.net: https://app.diagrams.net

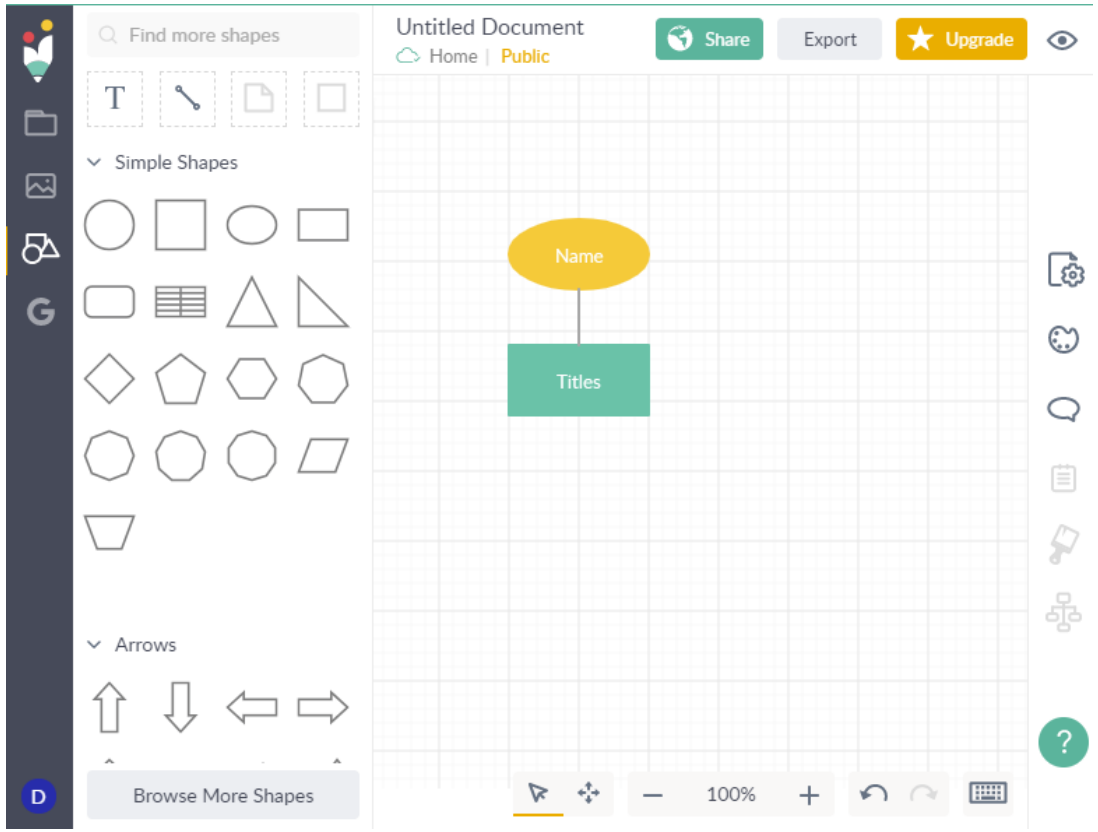


CONCEPTUAL MODELS

Tools to build Entity-Relationship Diagrams

Free tools to build E/R diagrams:

4. Creately: https://creately.com



CONCEPTUAL MODELS

Tools to build Entity-Relationship Diagrams

Free tools to build E/R diagrams:

5. SmartDraw: https://www.smartdraw.com/entity-relationship-diagram/er-diagram-tool.htm
6. Edraw Max: https://www.edrawsoft.com/er-diagram-software
7. Lucichart: https://www.lucidchart.com/pages/er-diagrams

CONCEPTUAL MODELS

Building Relational Model from E/R Diagram

E/R diagrams are a notation for describing schemas of databases. We may imagine that a database described by an E/R diagram contains particular data, an "instance" of the database. Since the database is not implemented in the E/R model, only designed, the instance never exists in the sense that a relation's instances exist in a DBMS. However, it is often useful to visualize the database being designed as if it existed.

For each entity set, the database instance will have a particular finite set of entities. Each of these entities has particular values for each attribute.

A relationship R that connects n entity sets $E_1,E_2,\ldots,E_n$ may be imagined to have an "instance" that consists of a finite set of tuples $(e_1,e_2,\ldots,e_n)$, where each ei is chosen from the entities that are in the current instance of entity set E_i. This set of tuples is called the relationship set for R.

CONCEPTUAL MODELS

Building Relational Model from E/R Diagram

Example: An instance of the Stars-In relationship could be visualized as a table with pairs such as:

Movies	Stars
Basic Instinct	Sharon Stone
Total Recall	Arnold Schwarzenegger
Total Recall	Sharon Stone

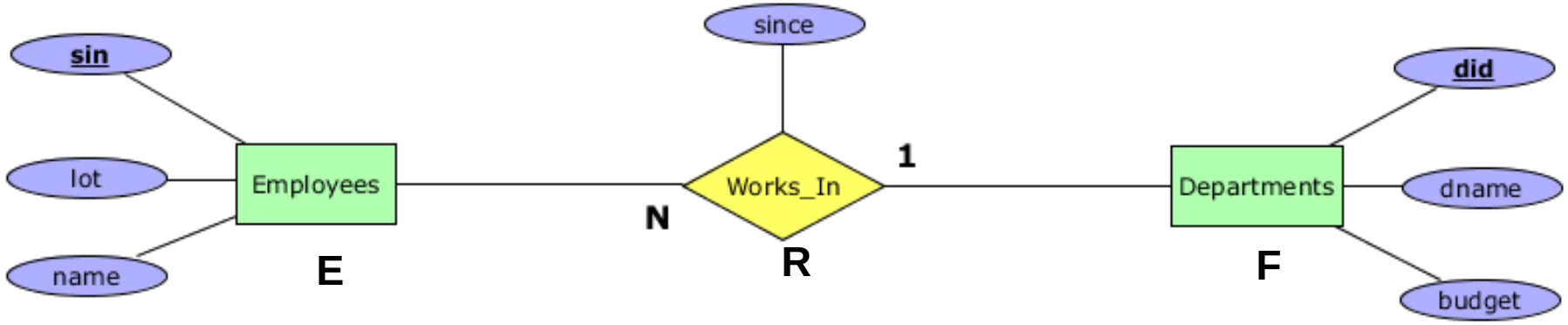
The members of the relationship set are the rows of the table. For instance, (Basic Instinct, Sharon Stone) is a tuple in the relationship set for the current instance of relationship Stars-in.

CONCEPTUAL MODELS

Multiplicity of Binary E/R Relationships

In general, a binary relationship can connect any member of one of its entity sets to any number of members of the other entity set. However, it is common for there to be a restriction on the "multiplicity" of a relationship. Suppose **R** is a relationship connecting entity sets **E** and **F**. Then:

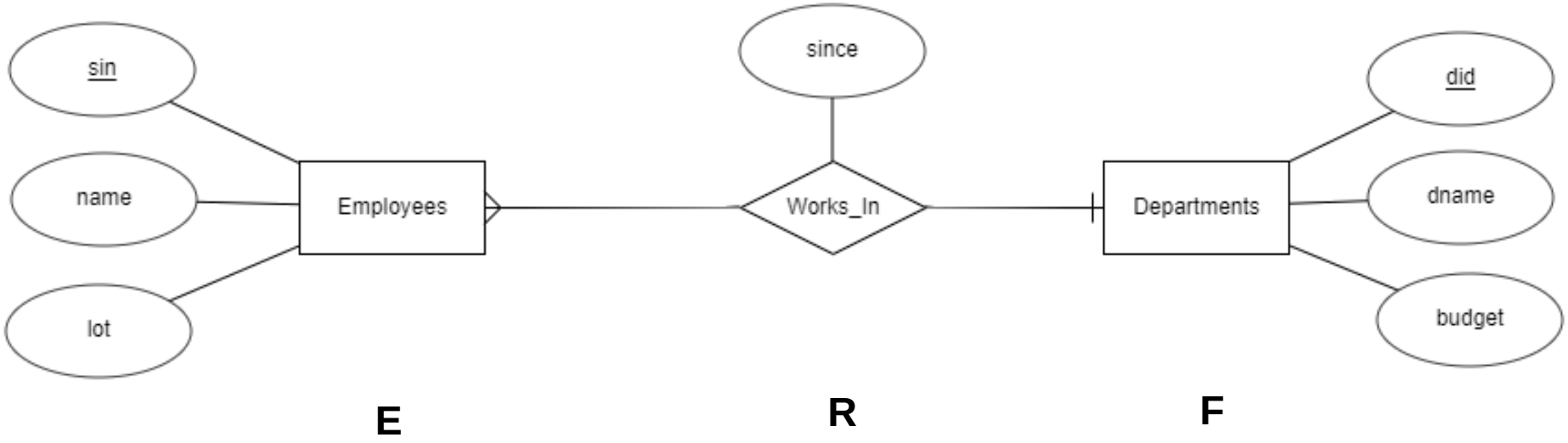
- If each member of **E** can be connected by **R** to at most one member of **F**, then we say that **R** is many-to-one relationship (or simply many-one) from **E** to **F**. Note that in a many-to-one relationship from **E** to **F**, each entity in **F** can be connected to many members of **E**. Similarly, if instead a member of **F** can be connected by **R** to at most one member of **E**, then we say **R** is one-many from **E** to **F**.



CONCEPTUAL MODELS

Multiplicity of Binary E/R Relationships

Same relationship represented using ERDPlus

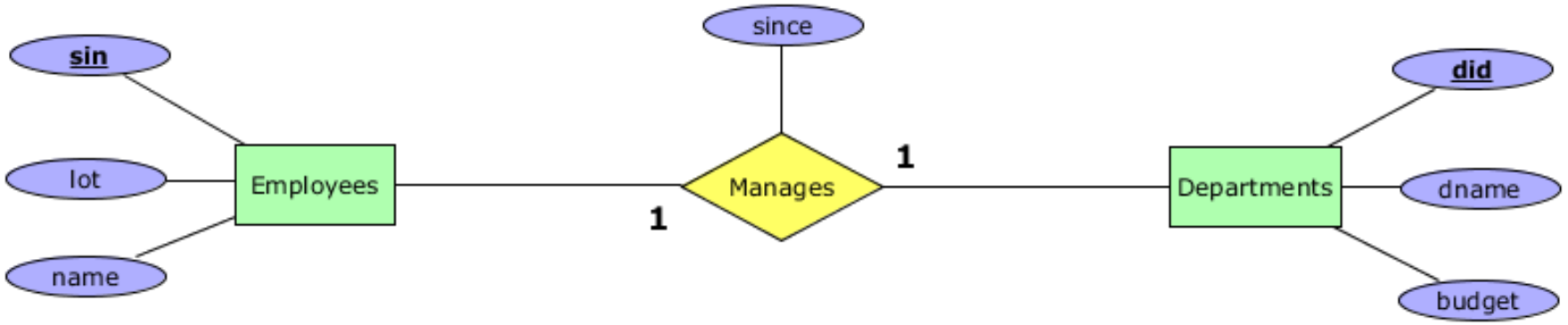


CONCEPTUAL MODELS

Multiplicity of Binary E/R Relationships

- If **R** is both many-one from **E** to **F** and many-one from **F** to **E**, then we say that **R** is one-to-one relationship (or simply one-one). In a one-one relationship an entity of either entity set can be connected to at most one entity of the other set.

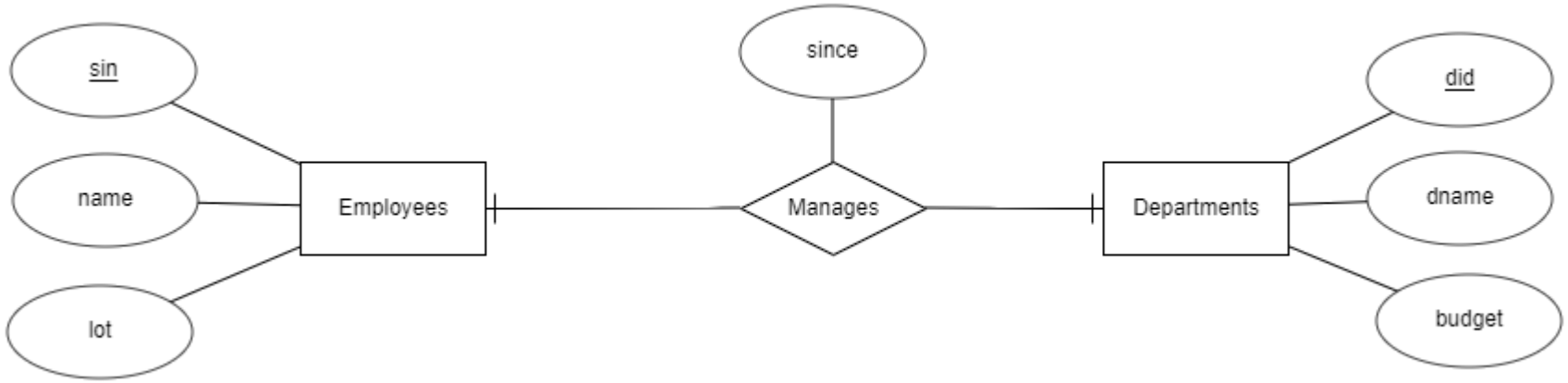
Example: We use Manages relation to show that employees may manage at most one Department and each Department is managed by at most one employee.



CONCEPTUAL MODELS

Multiplicity of Binary E/R Relationships

Same relationship represented using ERDPlus

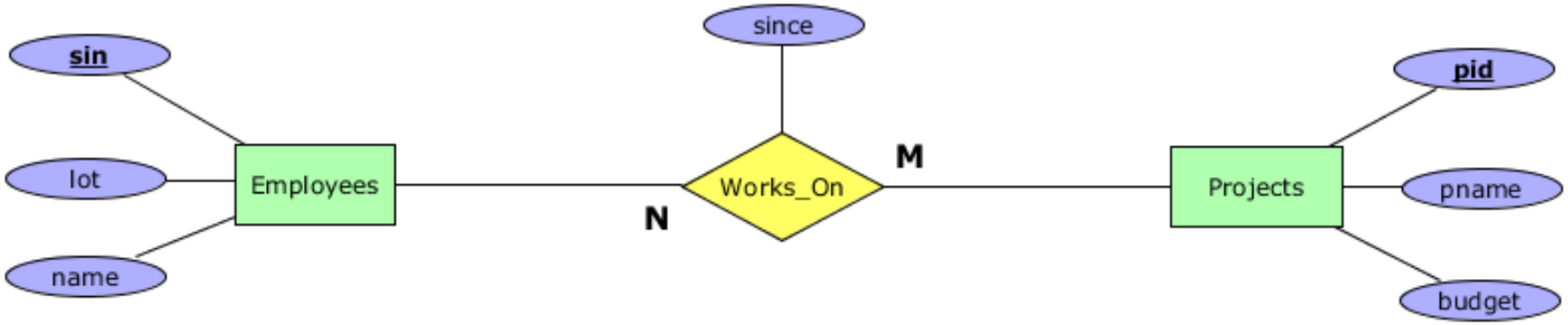


CONCEPTUAL MODELS

Multiplicity of Binary E/R Relationships

- If R is neither many-one from E to F or from F to E, then we say R is many-to-many relationship (or simply many-many).

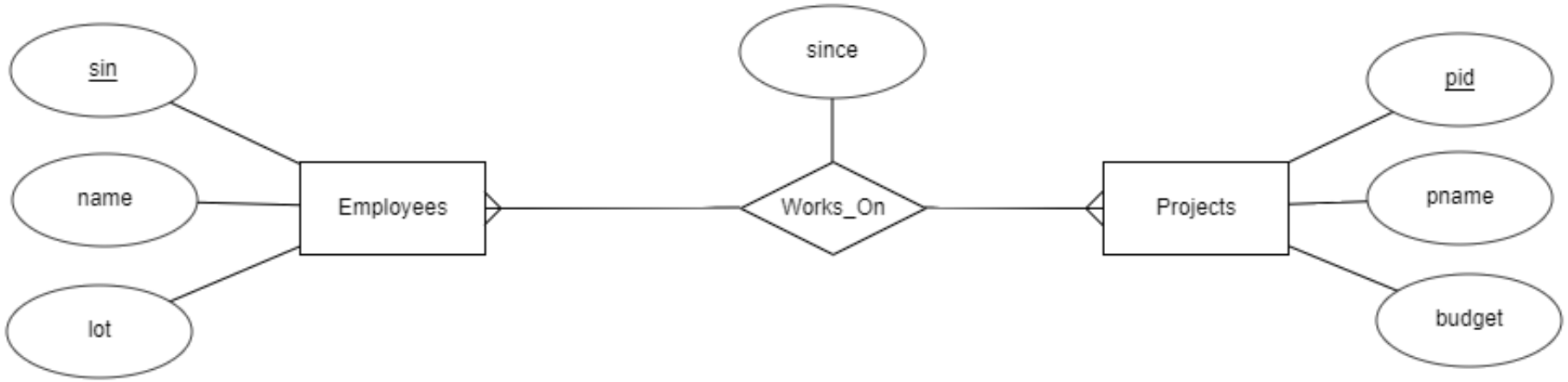
Example: The Works_On relates that each employee may work on multiple projects and on each project there may be multiple employee working on it.



CONCEPTUAL MODELS

Multiplicity of Binary E/R Relationships

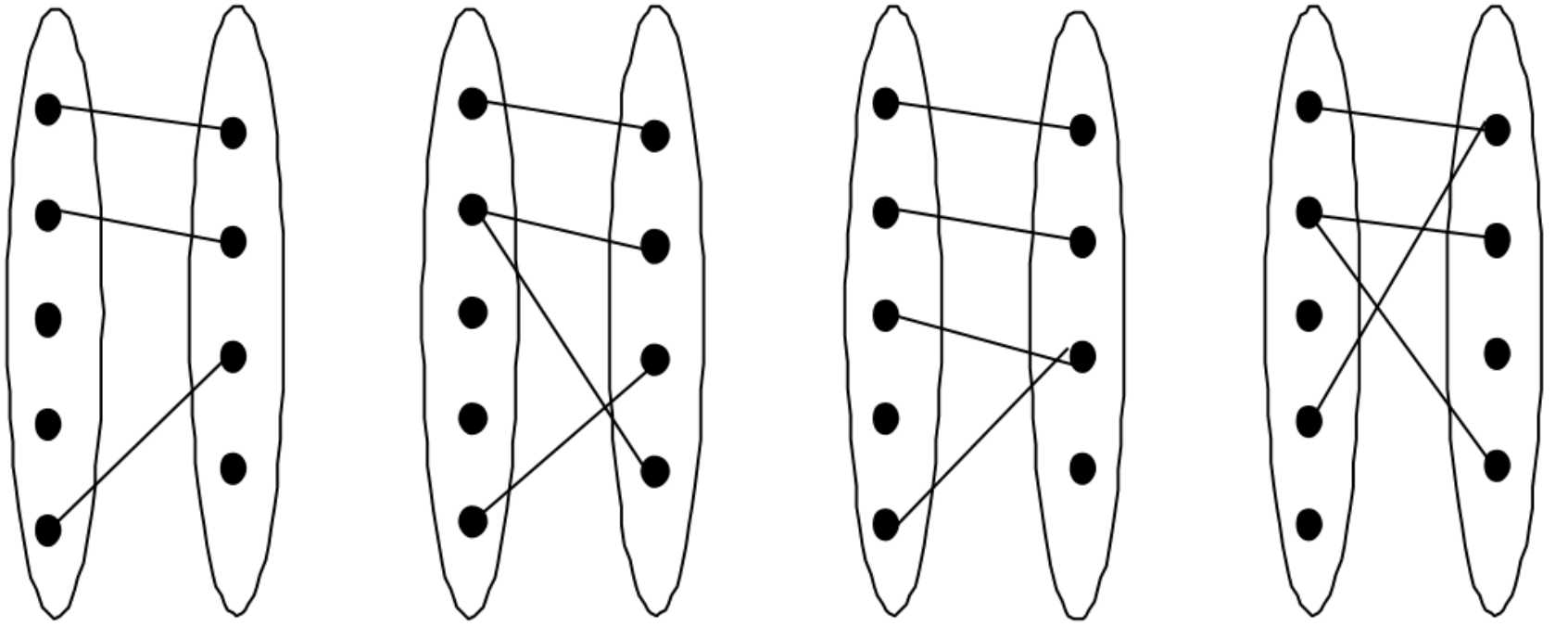
Same relationship represented using ERDPlus



E **R** **F**

CONCEPTUAL MODELS

Multiplicity of Binary E/R Relationships



1-to-1 | 1-to-many | many-to-1 | many-to-many

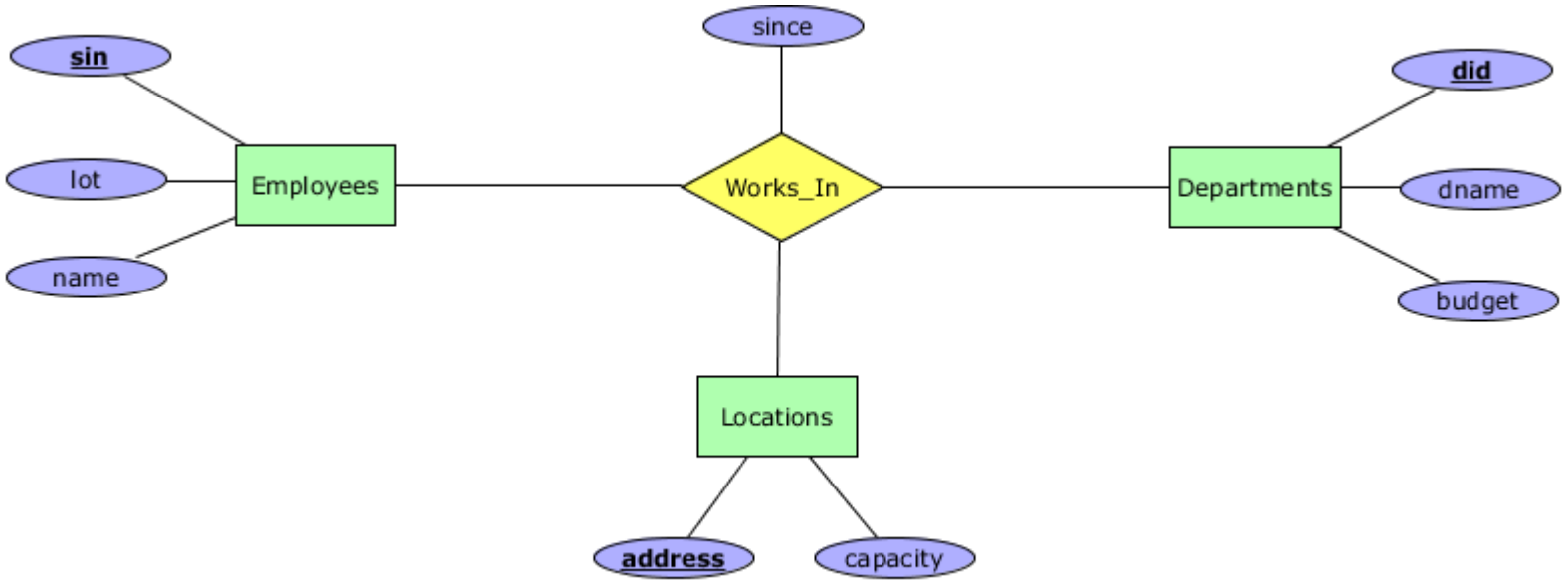
E.g. Employees-manages-Departments | Clients-orders-Invoices | Employees-worksIn-Departments | Employees-travels-Cities

CONCEPTUAL MODELS

Multiway Relationships

The E/R model makes it convenient to define relationships involving more than two entity sets. In practice, ternary (three-way) or higher-degree relationships are rare.

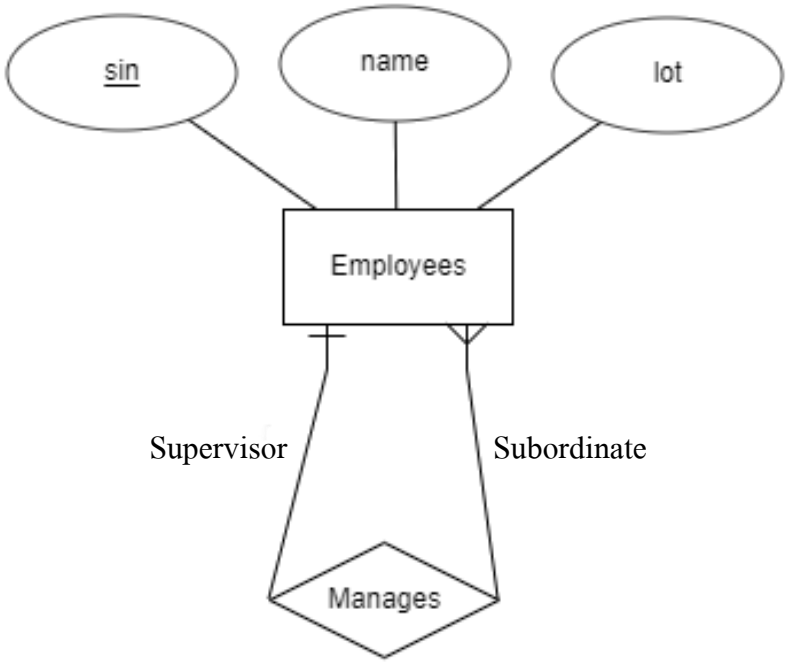
Example: We can extend the Work_In relation to include the location. This each employee works in at most one department and at one location.



CONCEPTUAL MODELS

Roles in Relationships

It is possible that one entity set appears two or more times in a single relationship. If so, we draw as many lines from the relationship to the entity set as the entity set appears in the relationship. Each line to the entity set represents a different role that the entity set plays in the relationship. We therefore label the edges between the entity set and relationship by names, which we call "roles."



Example: Each Employee may supervise multiple employees shown below using the Manages relationship.

CONCEPTUAL MODELS

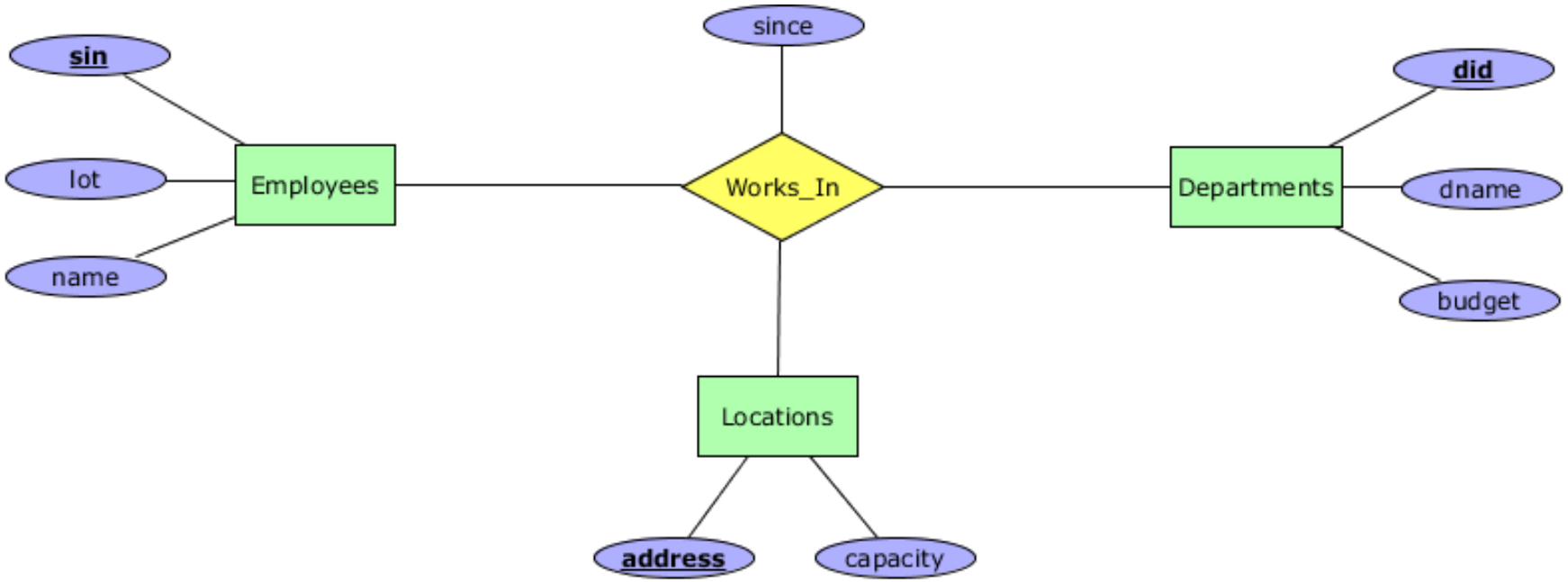
Converting Multiway Relationships to Binary

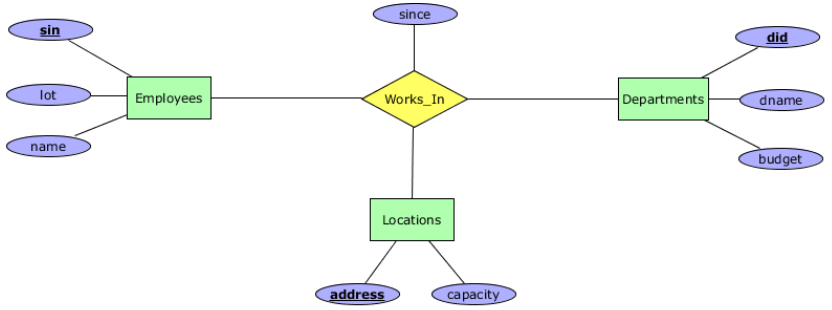
There are some data models, such as UML and ODL, that limit relationships to be binary. Thus, while the E/R model does not require binary relationships, it is useful to observe that any relationship connecting more than two entity sets can be converted to a collection of binary, many-to-one relationships. To do so, introduce a new entity set, called connecting entity set, whose entities we may think of as tuples of the relationship set for the multiway relationship. We then introduce many-one relationships from the connecting entity set to each of the entity sets that provide components of tuples in the original, multiway relationship.

CONCEPTUAL MODELS

Converting Multiway Relationships to Binary

Example: Consider ternary Works_In relationship.

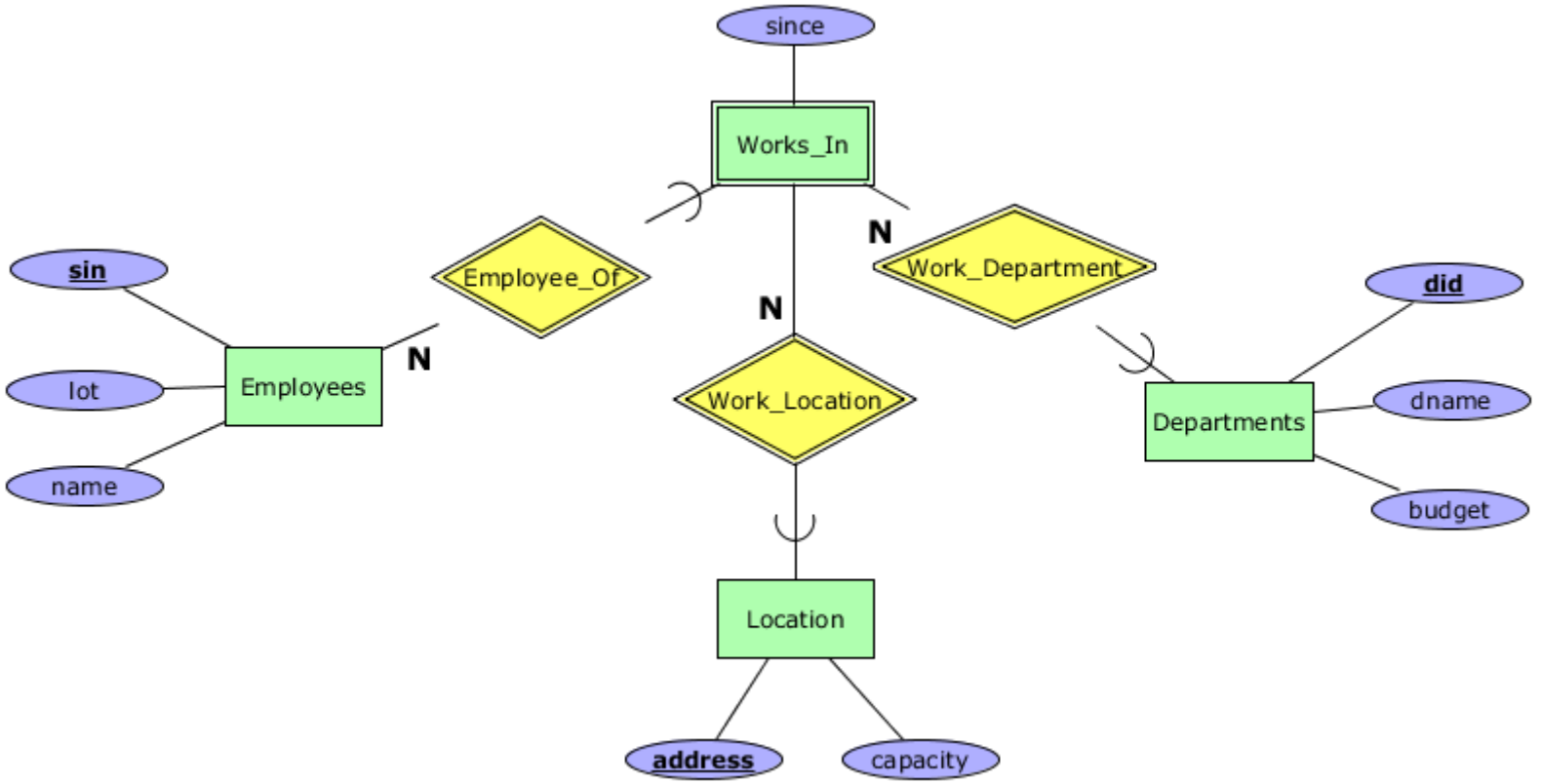




CONCEPTUAL MODELS

Converting Multiway Relationships to Binary

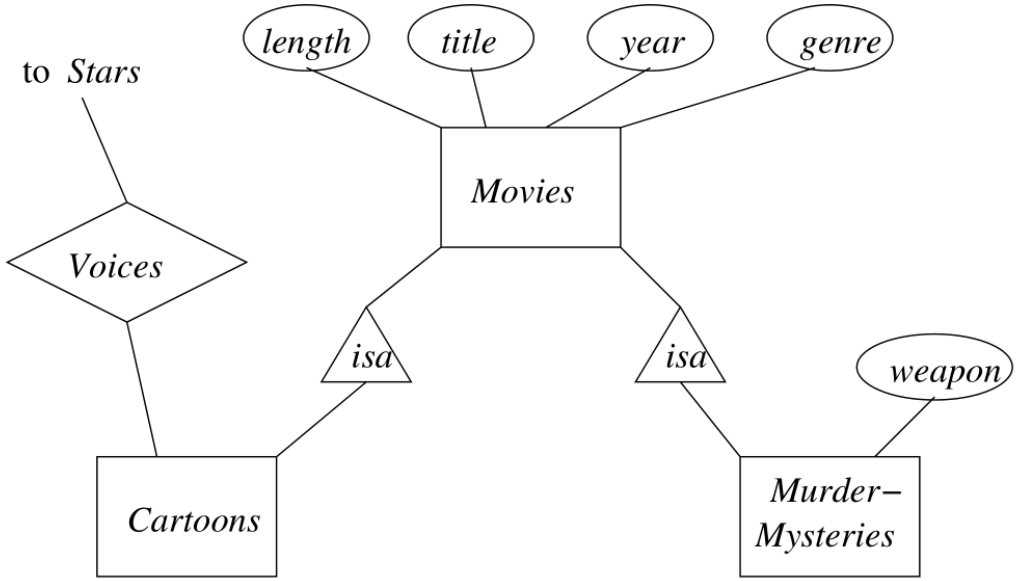
Example: Will be replaced with binary relationships (we will get back to this example when we will talk about weak entity sets)



CONCEPTUAL MODELS

Subclasses in the E/R Model

Often, an entity set contains certain entities that have special properties not associated with all members of the set. If so, we find it useful to define certain special-case entity sets, or subclasses, each with its own special attributes and/or relationships. We connect an entity set to its subclasses using a relationship called isa (i.e., "an A is a B" expresses an "isa" relationship from entity set **A** to entity set **B**). An isa relationship is a special kind of relationship, and to emphasize that it is unlike other relationships, we use a special notation: a triangle.



CONCEPTUAL MODELS

Design Principles

Faithfulness

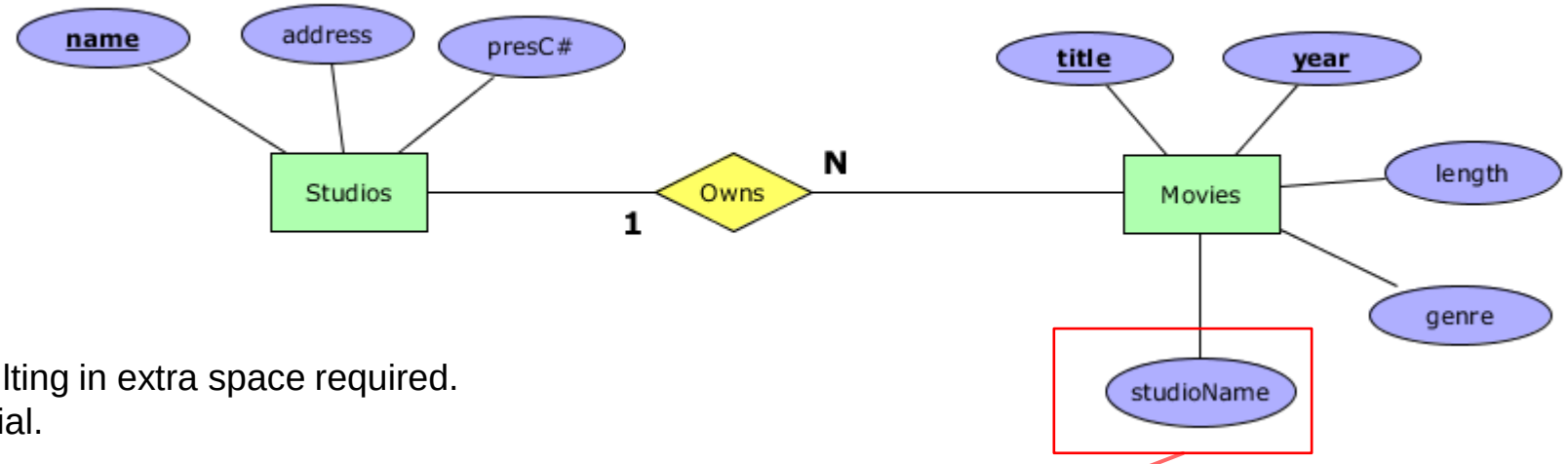
First and foremost, the design should be faithful to the specifications of the application. That is, entity sets and their attributes should reflect reality. You can't attach an attribute number-of-cylinders to Stars, although that attribute would make sense for an entity set Automobiles. Whatever relationships are asserted should make sense given what we know about the part of the real world being modeled.

CONCEPTUAL MODELS

Design Principles

Avoiding redundancy

Problems regarding redundancy and anomalies are typical of problems that can arise in E/R designs. However, in the E/R model, there are several new mechanisms whereby redundancy and other anomalies can arise.



Problems:

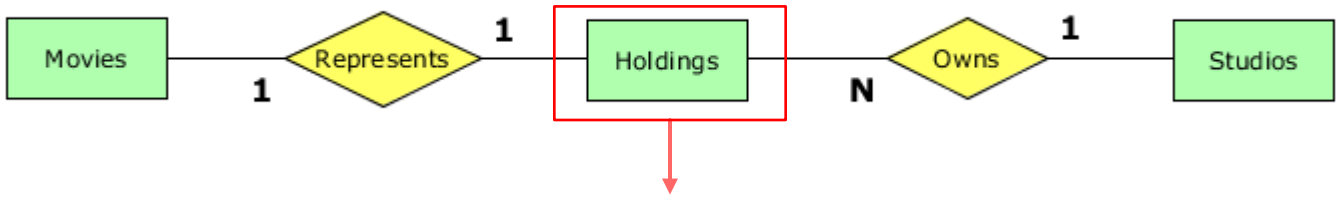
- Repetition of a fact resulting in extra space required.
- Update-anomaly potential.

CONCEPTUAL MODELS

Design Principles

Simplicity

Avoid introducing more elements into your design than is absolutely necessary.



Not illegal but not necessary

CONCEPTUAL MODELS

Design Principles

Choosing the Right Relationship

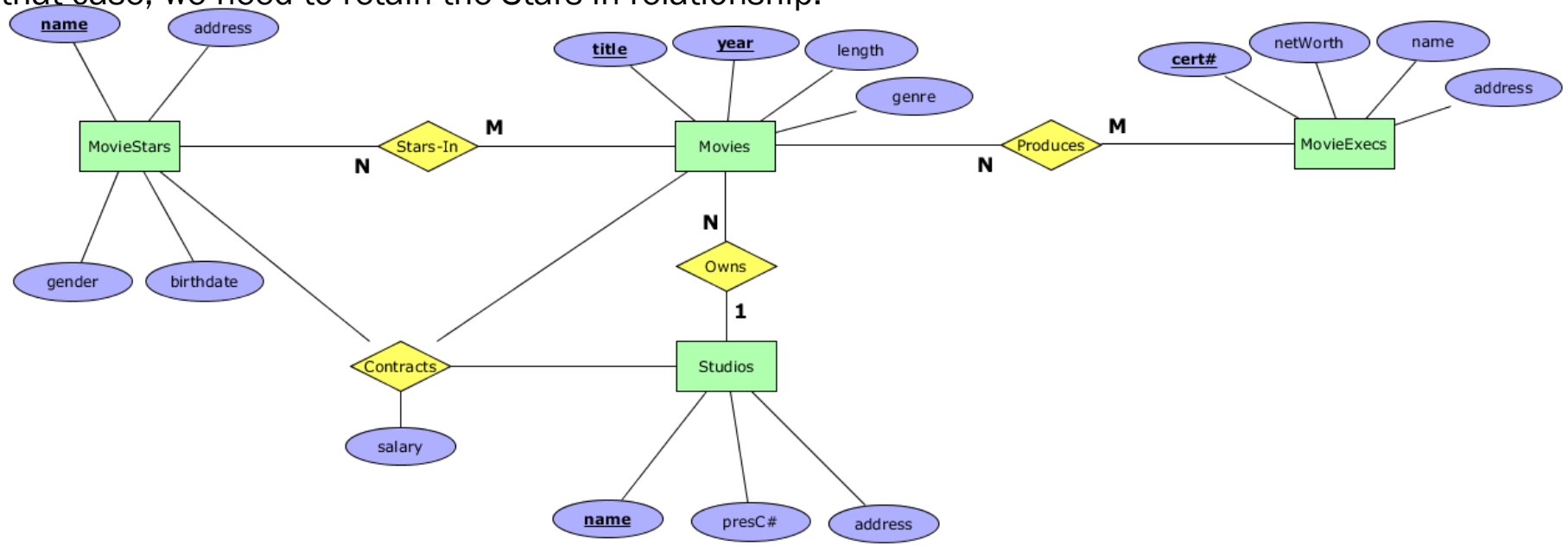
Entity sets can be connected in various ways by relationships. However, adding to our design every possible relationship is not often a good idea. Doing so can lead to redundancy, update anomalies, and deletion anomalies, where the connected pairs or sets of entities for one relationship can be deduced from one or more other relationships.

CONCEPTUAL MODELS

Design Principles

Choosing the Right Relationship

Example: Is Stars-In needed in below example? If a star can appear in a movie only if there is a contract involving that star, that movie, and the owning studio for the movie, then there truly is no need for relationship Stars-in. However, if a star can work on a movie without there being a contract then there could be star-movie pairs in Stars-in that are not part of star-movie-studio triples in Contracts. In that case, we need to retain the Stars-in relationship.

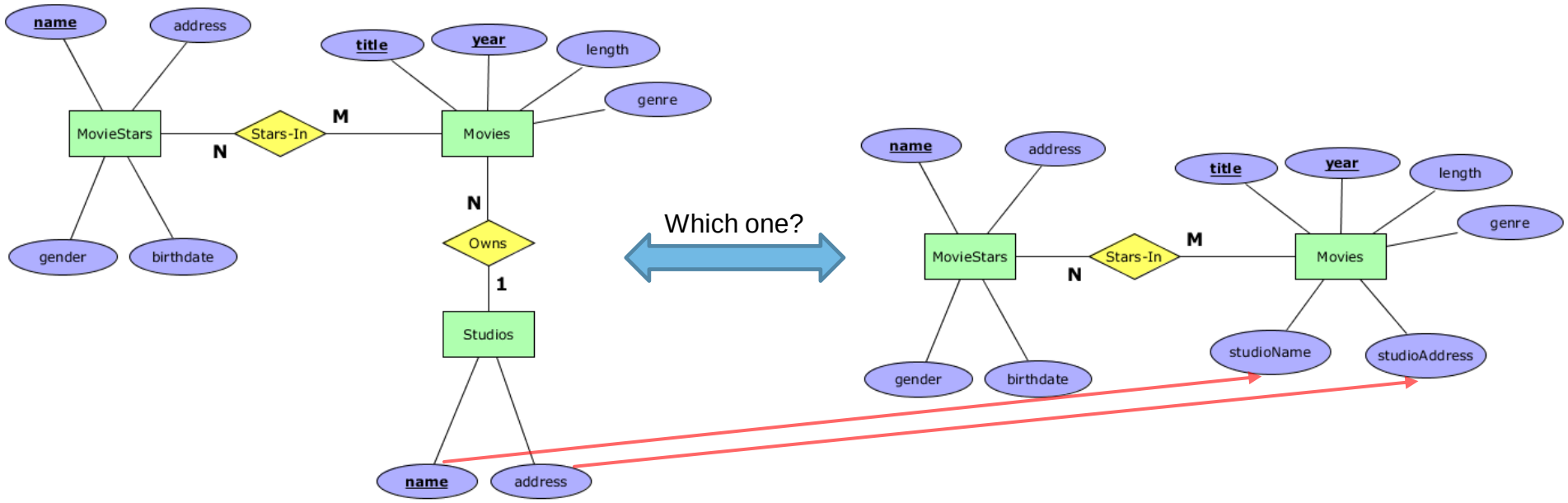


CONCEPTUAL MODELS

Design Principles

Selecting the Right Kind of Element

Sometimes we have options regarding the type of design element used to represent a real-world concept. Many of these choices are between using attributes and using entity set/relationship combinations. In general, an attribute is simpler to implement than either an entity set or a relationship. However, making everything an attribute will usually get us into trouble.



CONCEPTUAL MODELS

Design Principles

Selecting the Right Kind of Element

If **E** is an entity set, here are conditions that **E** must obey in order for us to replace **E** by an attribute or attributes:

1. All relationships in which **E** is involved must have arrows entering **E**. That is, **E** must be the “one” in many-to-one relationships.

2. If **E** has more than one attribute, then no attribute depends on the other attributes, the way address depends on name for Studios. That is, the only key for **E** is all its attributes.

3. No relationship involves **E** more than once.