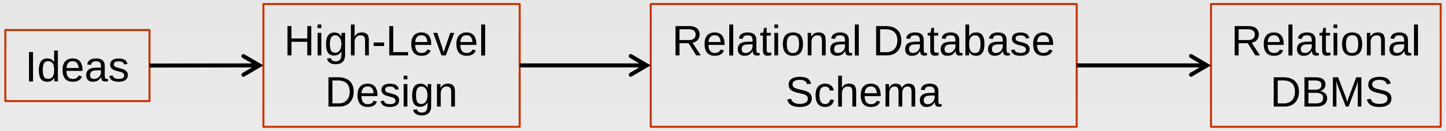
High-Level Database Models

Spring 2011

Instructor: Hassan Khosravi

Database Modeling and implemnation process

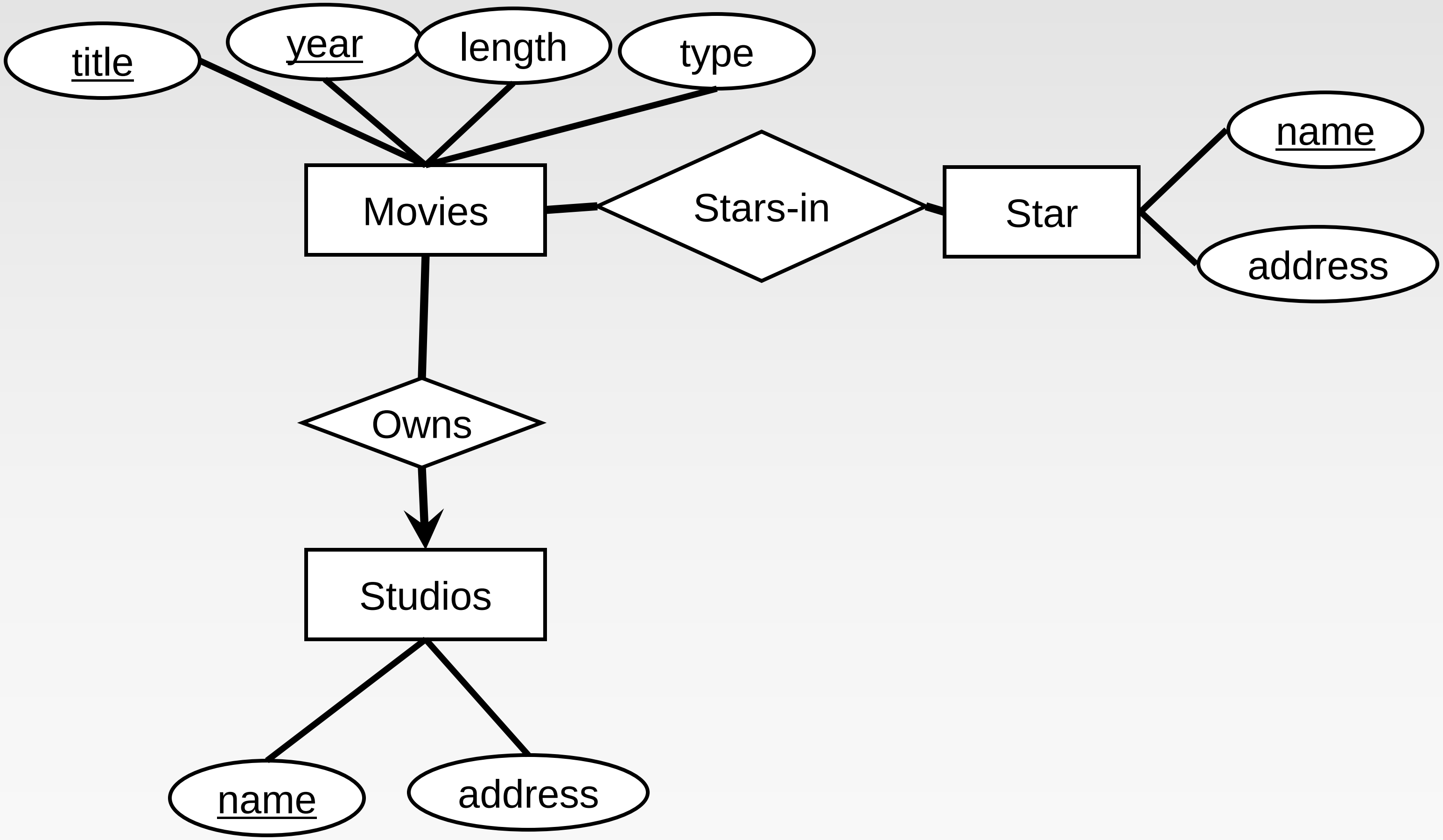


The Entity/Relationship Model

- The structure of data is represented graphically using
  - Entity sets
    - An abstract object of some sort
  - Attributes
    - properties of the entities
    - Primitive type : String, integer, real
  - Relationships
  - Connections among entities.

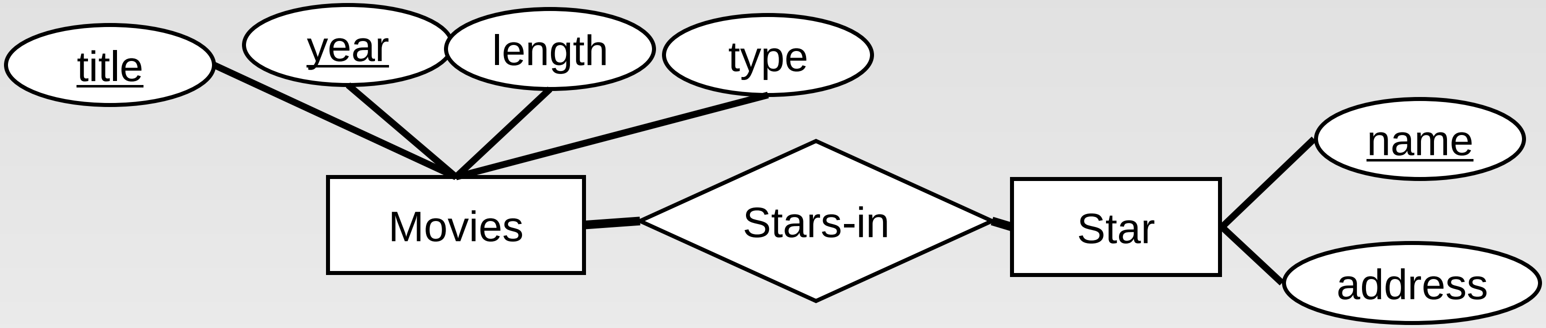
Entity/Relationship Diagram

- Entity sets are represented by rectangles
- Attributes are represented by ovals
- Relationships are represented by diamonds

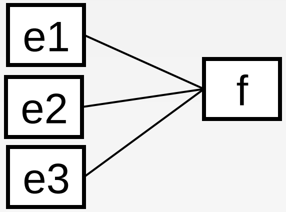


Multiplicity of Binary E/R Relationships

- 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.

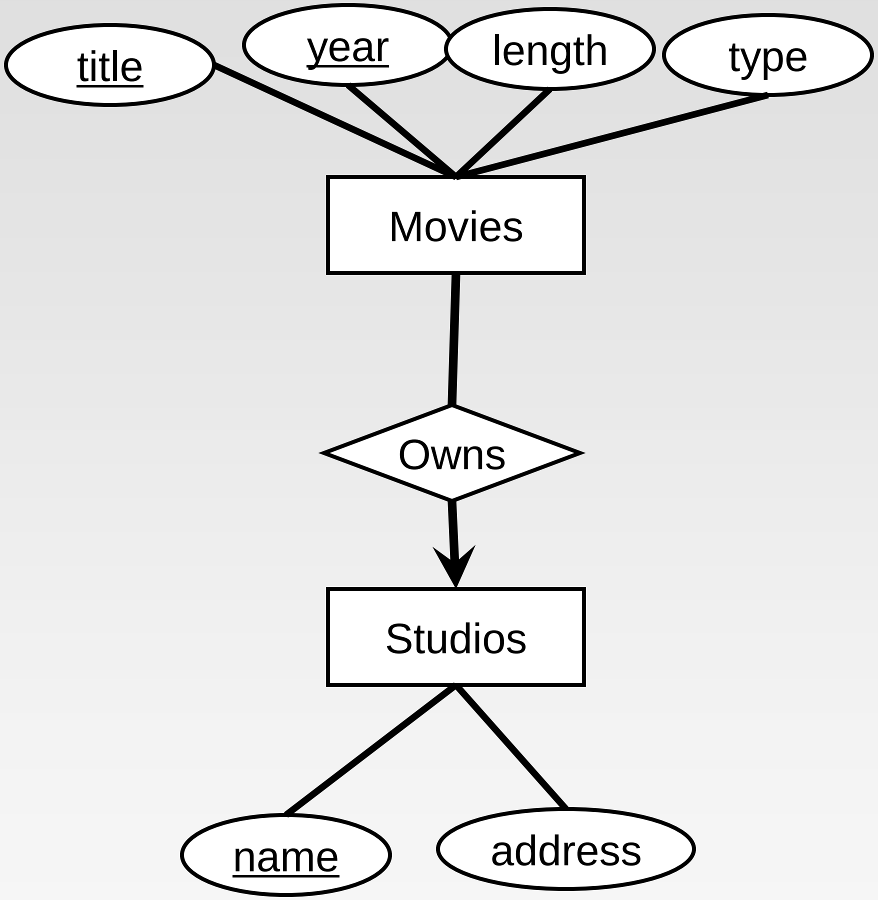


- Suppose R is a relation connecting entity sets E and F
  - If each member of E can be connected by R to **at most** one member of F, then we say R is **many-one** from **E to F**

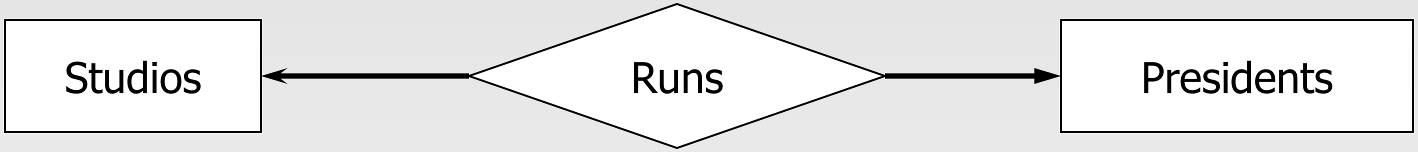


  - If each member of F can be connected by R to **at most** one member of E, then we say R is **many-one** from **F to E, or one-many** from **E to F**

- Example of a many-one relationship from Movie to studio

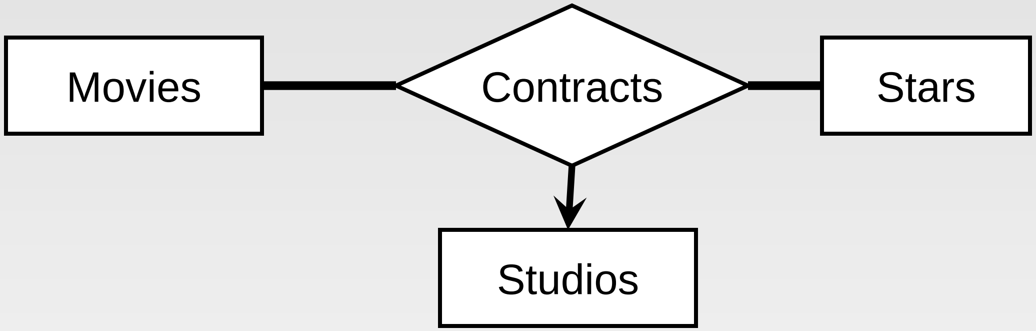


- If R is both many-one from E to F and many-one from F to E then we say that R is **one to one**



Multiway Relationships

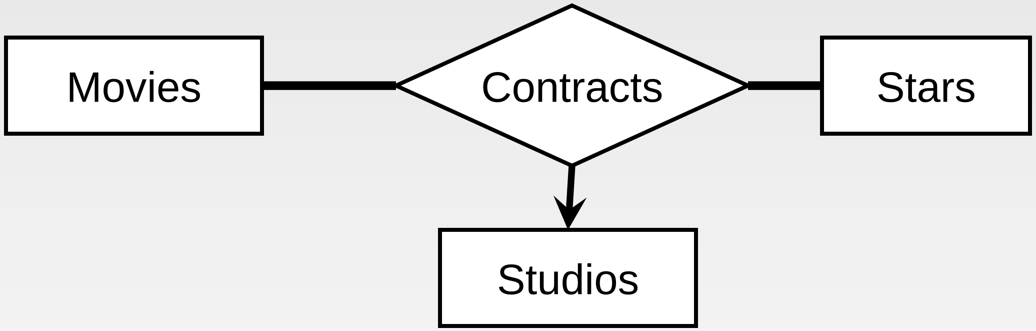
- E/R model makes it convenient to define relationships involving more than two entity sets.



- An arrow pointing to an entity E means that if we select one entity from each of the other entity sets, those entities are related to at most one entity in E.

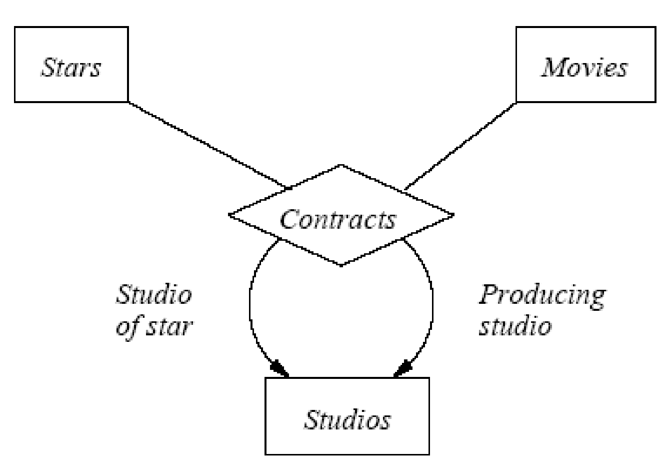
Limitations on Arrow notation

- Not enough choice of arrow to determine every situation
  - Movie determines studio?
  - stars determine studio?
  - Movie + star determine studio?



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.

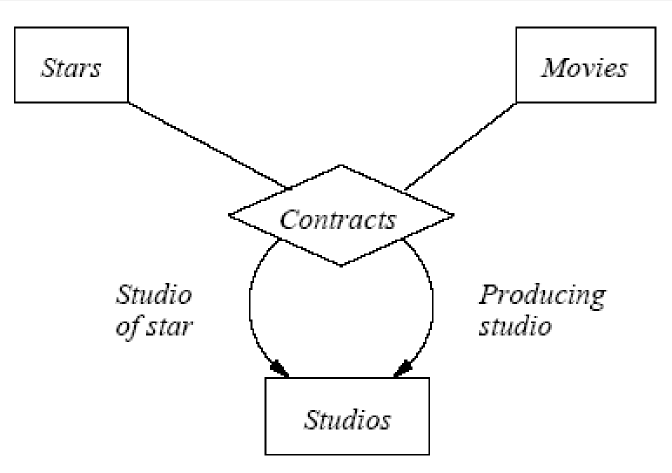


Contracts(starname, title, year, studioOfstar, producingStudio)

- One studio having a certain star under contract (in general) , one for a specific film.

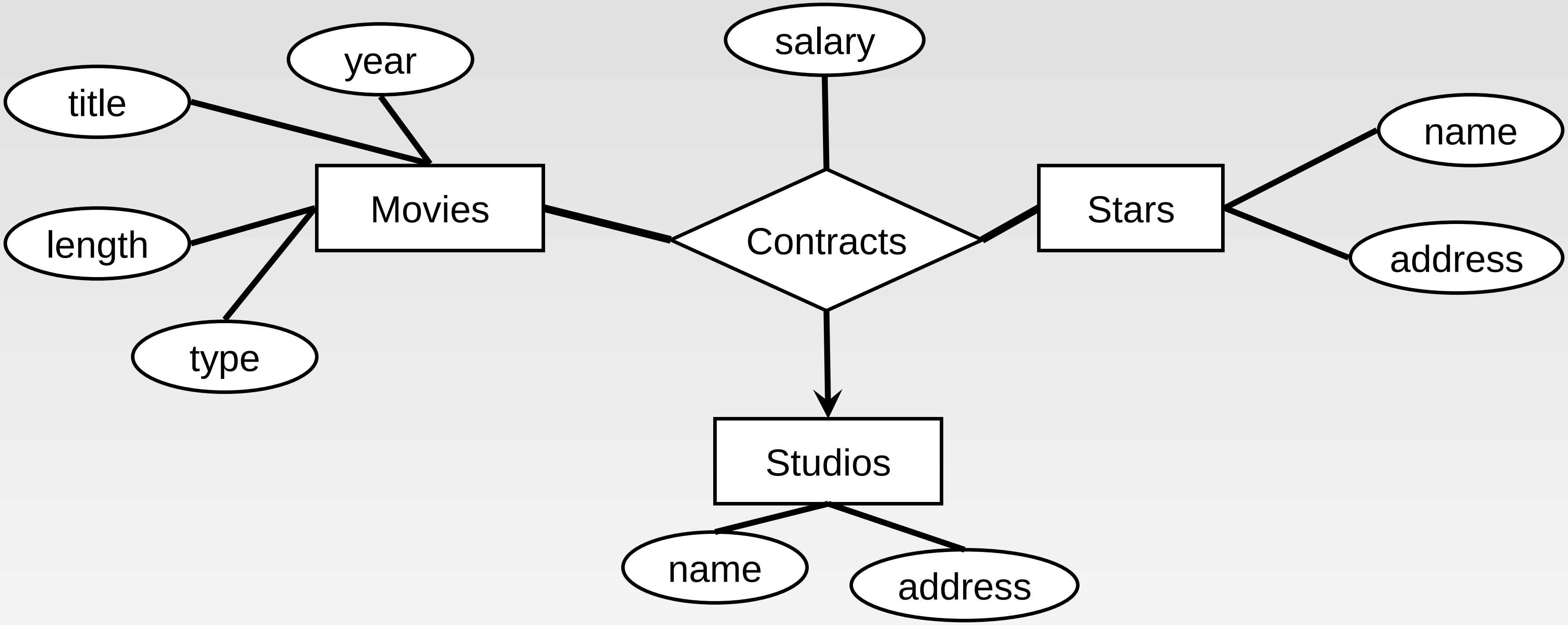
Roles in Relationships

- What do the arrows mean?
  - Given a star, a movie, and a producing studio, the studio of the star is unique
  - Given a star, a movie, and a studio for star, the producing studio is unique



Attributes on relationships

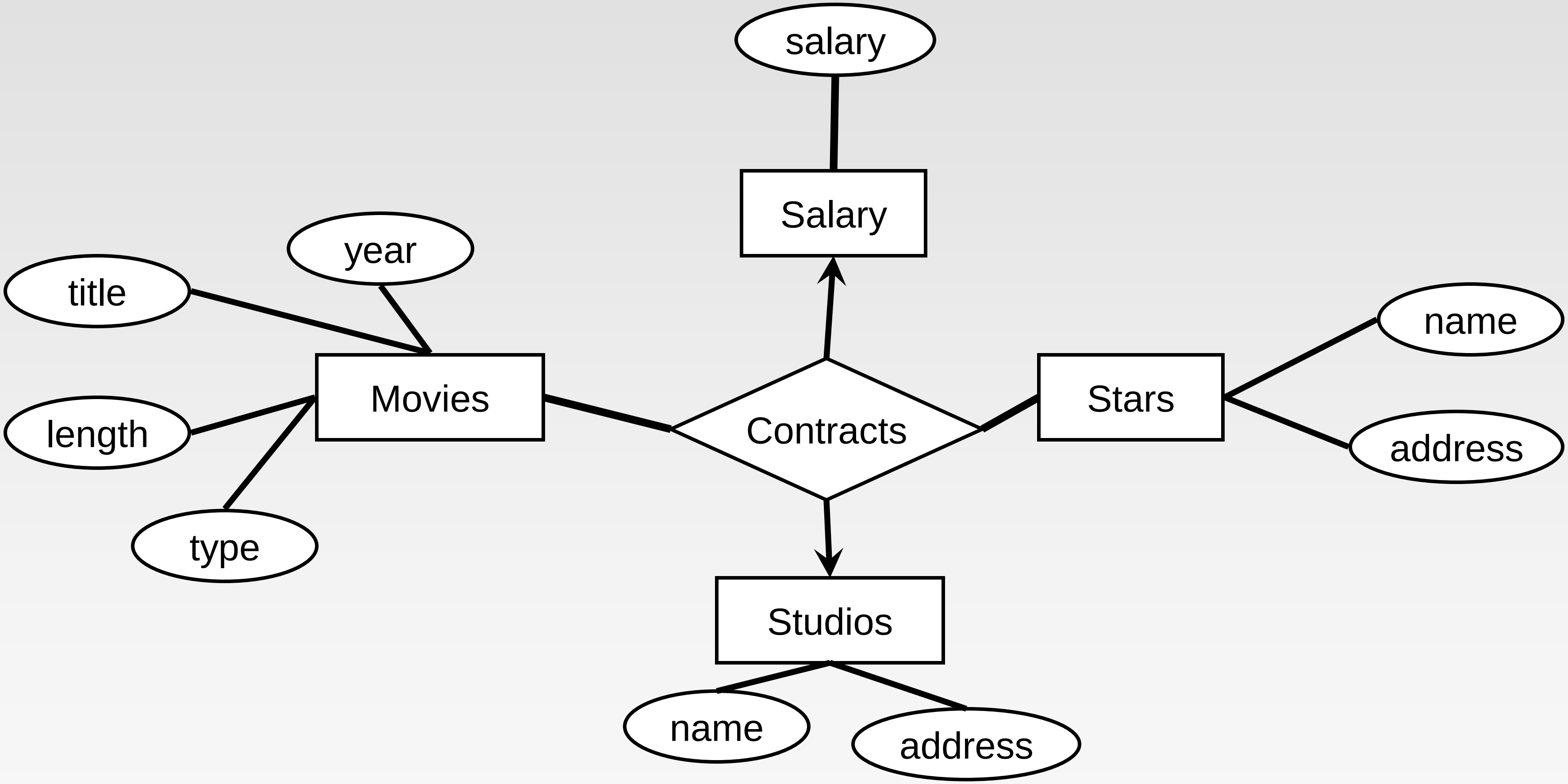
- Sometimes it is convenient or even essential to associate attributes with a relationship.



- Salary can not be part of stars table as they might get different salary for different movies.
- Salary cannot be part of Movies as different stars getting different salaries.

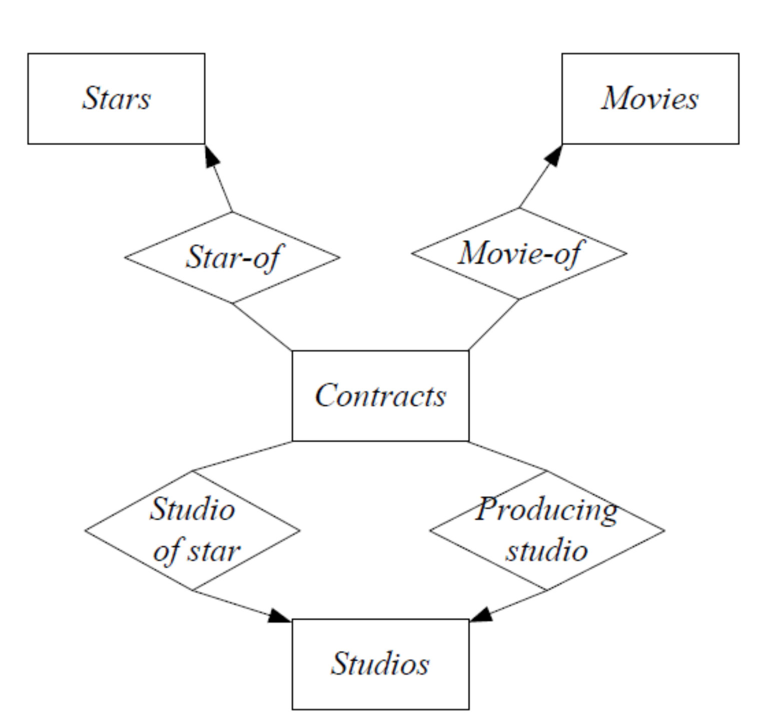
Attributes on relationships

- It is never necessary to place attributes on relationships. We can instead invent a new entity set



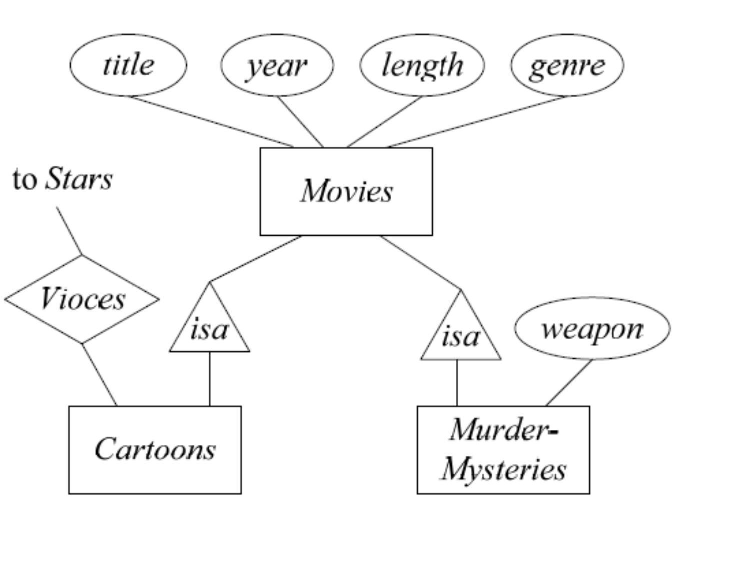
Converting Multiway relationships to Binary

- E/R model does not require binary relationships, but other models do
  - UML(4.7) and ODL(4.9) limit relationships to be binary
- It is generally useful to observe that any relationship connecting more than two entity sets can be converted to a collection of binary relations.



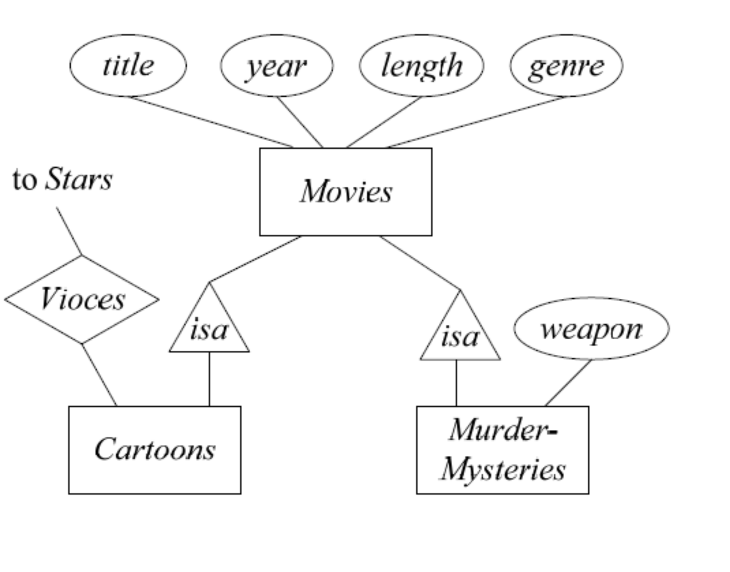
Subclasses in the E/R Model

- An entity set may contain certain entities that have special properties not associated with all members of the set.
  - We can use a “isa” relationship which is presented by a triangle
    - Cartoons have voice of stars
    - Murder mysteries have weapon
  - In general entity sets connected by “isa” relationship could have any structure. We shall limit it to trees



Subclasses in the E/R Model

- Typical movies being neither will have 4 attributes
- A cartoon movie would have 4 attributes and voice relationship
- A murder mystery would have 5 attributes
- A movie like Roger Rabbit which is both a cartoon and a murder mystery will have 5 attributes and voice relationship



Design Principles

- Faithfulness
- Avoiding redundancy
- Simplicity
- Right relationships
- Right elements

Faithfulness

- The design must be faithful to the specification of the application. It should reflect reality.
  - The stars-in relation between stars and movies must be many to many as observed in real world
  - Sometimes it is less obvious
    - Instructors, courses and a relation teaches between them. Is the relation many-many? Many-one?
      - The answer relies on the schools policy that a few instructors could teach the same course or not.

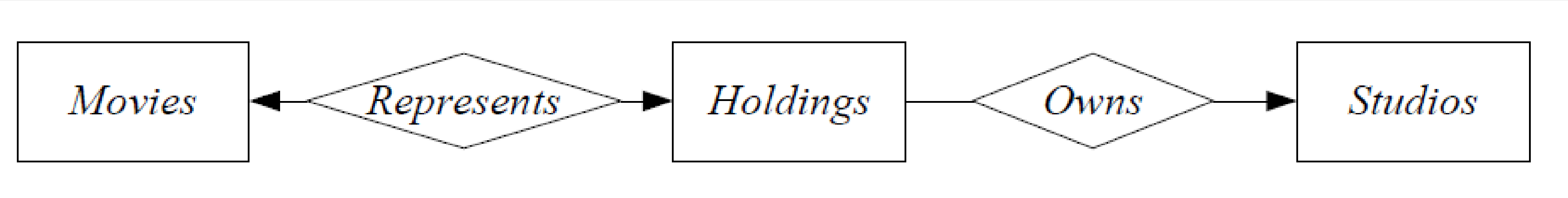
Avoiding Redundancy

- We should be careful to say everything once.
  - Redundancy: Unnecessarily repeated info in several tuples
    - Star Wars, 1977, 124, SciFi, and Fox is repeated.
  - Update Anomaly: Changing information in one tuple but leaving the same info unchanged in another
    - If you find out that Star Wars is 125 minute and you don't update all of them, you will lose the integrity.
  - Deletion Anomaly: Deleting some info and losing other info as a side effect

Title	Year	Length	Genre	StudioName	StarName
Star Wars	1977	124	SciFi	Fox	Carrie Fisher
Star Wars	1977	124	SciFi	Fox	Mark Hamill
Star Wars	1977	124	SciFi	Fox	Harrison Ford
Gone with the wind	1939	231	Drama	MGM	Vivien Leigh
Wayne's World	1992	95	Comedy	Paramount	Dana Carvey
Wayne's World	1992	95	Comedy	Paramount	Mike Meyers

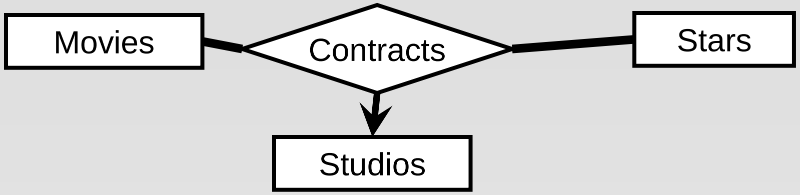
Simplicity

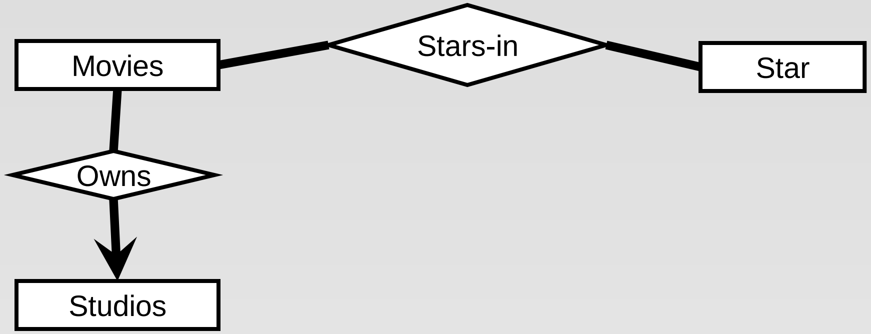
- Avoid introducing more elements into your design than is absolutely necessary. We need to make the data as abstract as possible
  - Existence of movie-holdings which shows the ownership of a single movie.



  - This structure is closer to reality, however it holds no useful info

Right Relationships

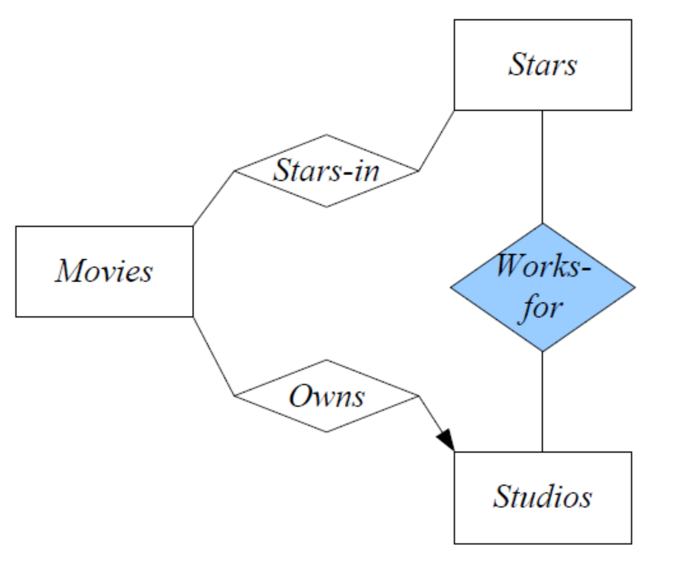




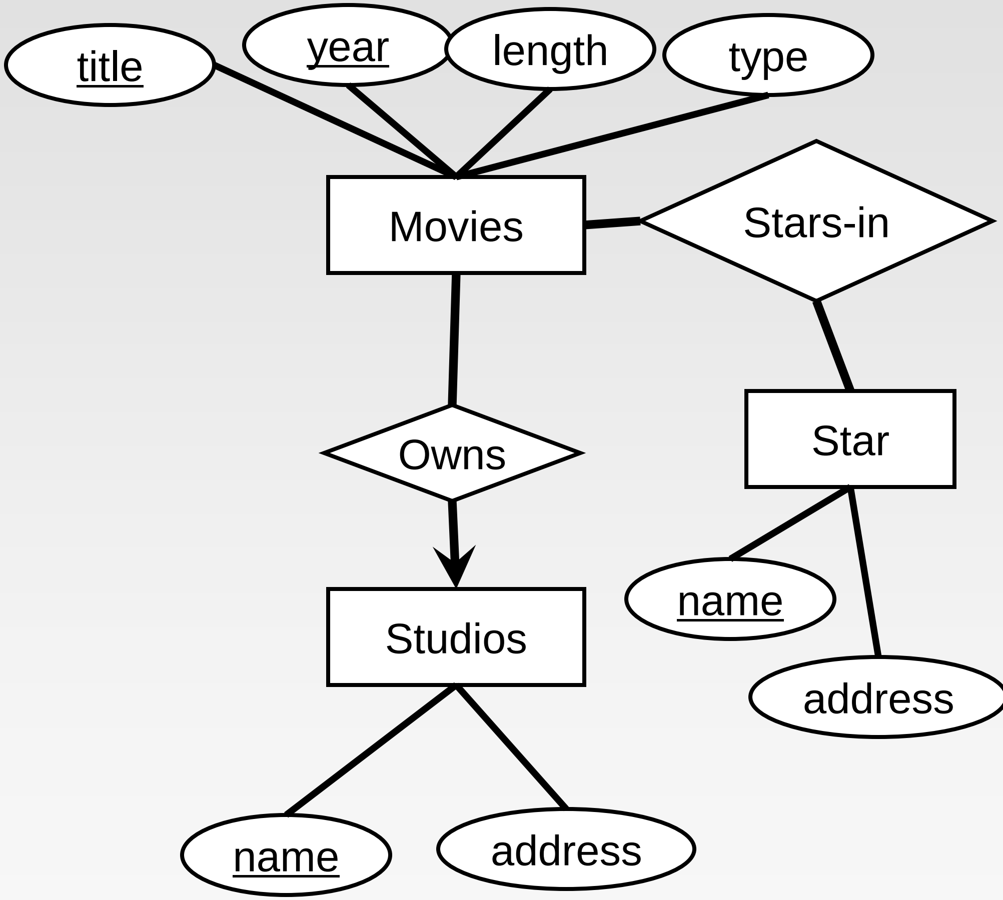
- We omitted the owns and the stars-in relationships when we introduced contract was that a right decision?
  - We don’t know. It depends on our assumptions
    - It might be possible to deduce the relationship stars-in from contract. If a star can appear in a movie only with a contract.
      - However there may be no contract
      - They may be no recorded contract
    - If for every movie there is at least one contract involving the movie, the studio and some stars then we can eliminate owns
    - If a studio can own a movie and yet there are still no stars then we can not eliminate owns

Right Relationships

- We can use the two relationships stars-in and owns to conclude that a star could work for a studio.
  - Is it rational to add such a relationship?
    - Depends, if it doesn’t add any new info basically means that star working for a movie owned by the studio then no
    - If its possible to work for a studio without being on the movie then yes



Right Elements



- were we wise to make studio an entity instead of adding it to the movie table
  - Redundancy in address
- What if there was no address for studio?
  - Then it would have been reasonable.

Right Elements

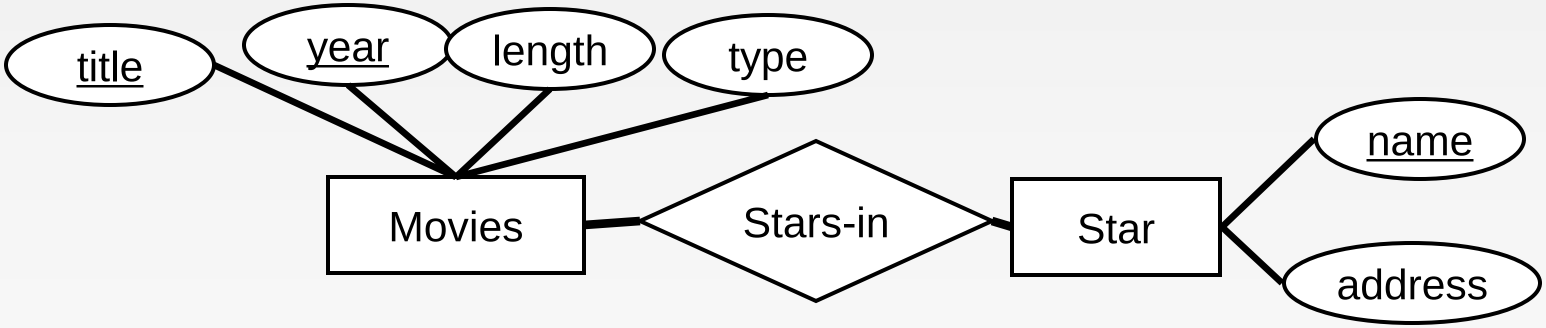
- Conditions under which we prefer to use an attribute instead of an entity set
- Suppose *E is an entity set*
  - *E must be the “one” in many-one relationships*
    - *If the movie can have more than studio it wouldn’t make sense to have an attribute for it*
  - The only key for *E is all its attributes*
    - *Address was dependent on name and that was stopping us from using studio as a attribute*
  - No relationship involves *E more than once*

Constraints in the E/R Model

- Keys in the E/R model
- Referential integrity
- Degree Constraints

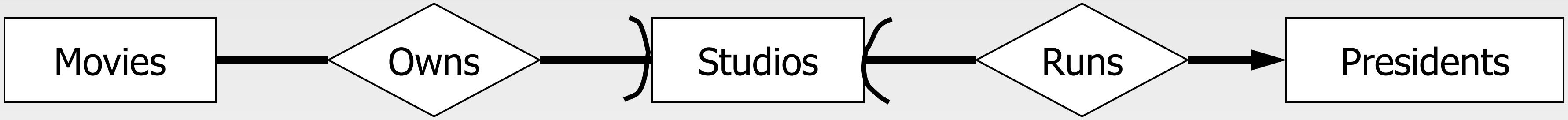
Keys in the E/R Model

- Every entity set must have a key
  - In some cases isa and weak entity sets have keys that belong to other tables
- There can be more than one key, we pick one to be the primary key.
- In isa relationships we require the root to have all the attributes needed for a key.
- We underline the attributes belonging to a key for an entity set.



Referential Integrity

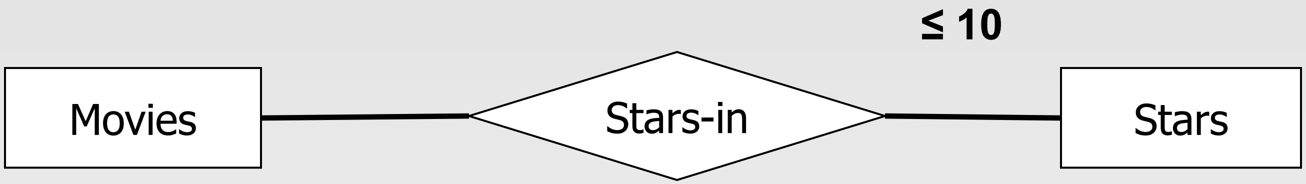
- Many-one requirements simply says that no movie can be owned by two studios. It doesn't say that a movie must be owned by a studio.
- The owns relationship has a referential integrity constraint
  - There must be one owning studio.
  - The studio must be listed in the studio tables.



- Suppose *R is a relationship from E to F*
  - A rounded arrow-head pointing to *F indicates* not only that the relationship is many-one from *E to F,* but that the entity of set *F related to a given entity of set E is required to exist*

Degree Constraint

- We can attach a bounding number
- A movie entity cannot be connected by relationship Stars-in to more than 10 star entities



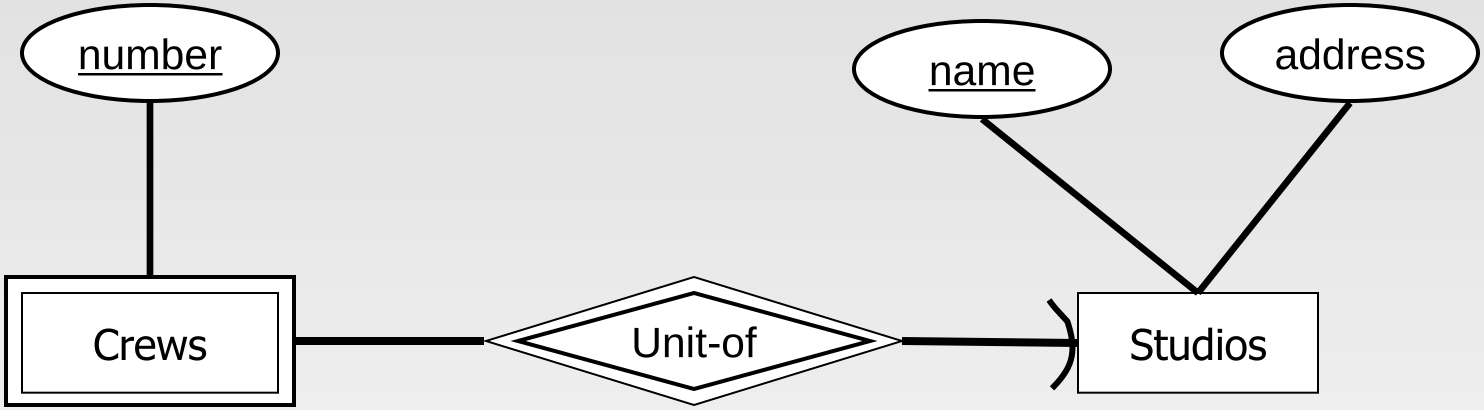
- The constraint <=1 shows many-one relationship
- The constraint =1 shows referential integrity

Weak Entity Sets

- Causes of weak entity sets
- Requirements for weak entity sets
- Weak entity set notations

Causes of Weak Entity Sets

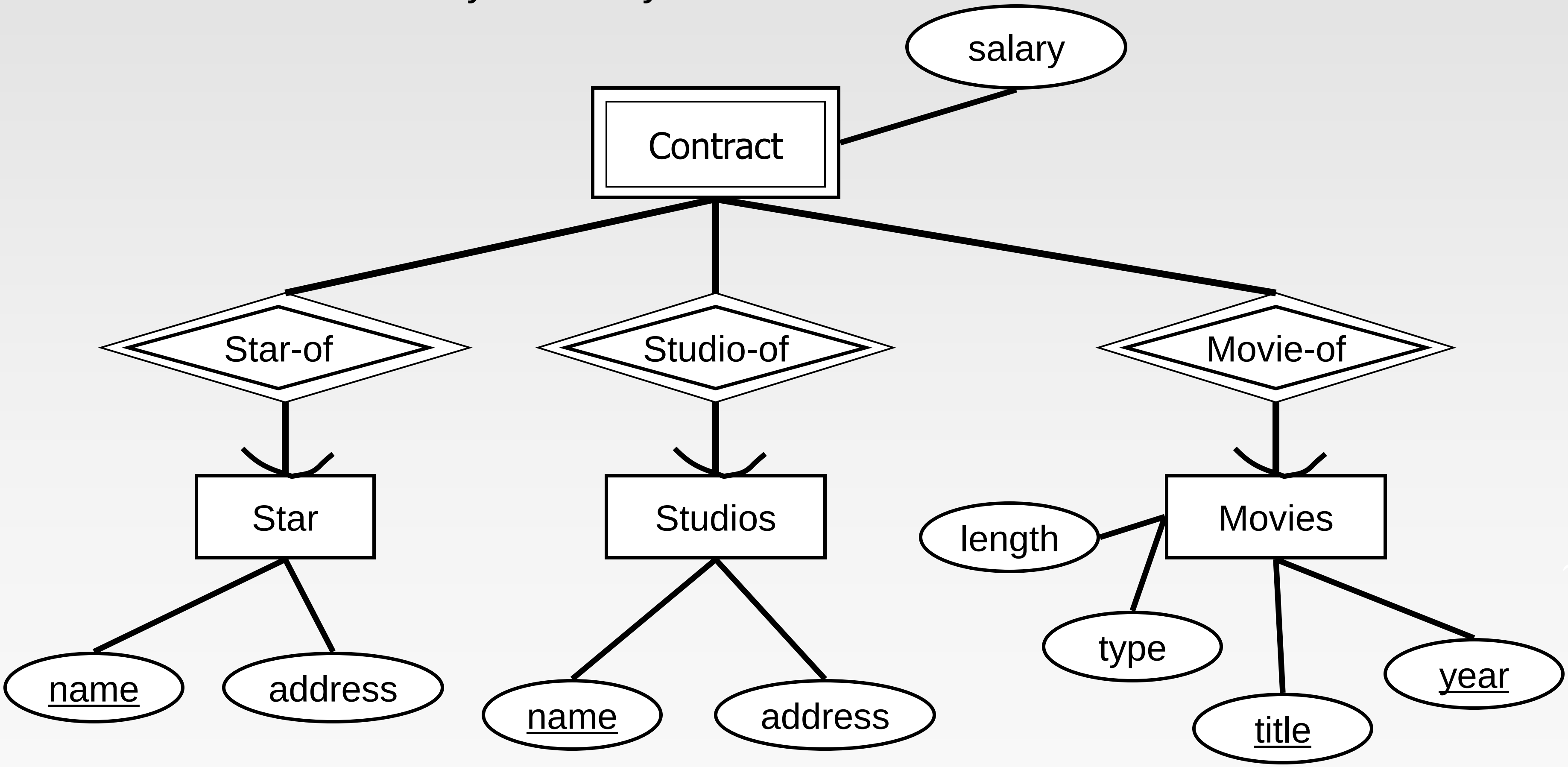
- 1. if entities of set *E* are subunits of entities in set *F*, then it is possible that the names of *E* entities are not unique until we take into account the name of the *F* entity to which the *E* entity is subordinate.



- If an entity set is weak, it will be shown as a rectangle with a double border.
- Its supporting many-one relationships will be shown as diamonds with a double border.
- If an entity set supplies any attributes for its own key, then those attributes will be underlined.

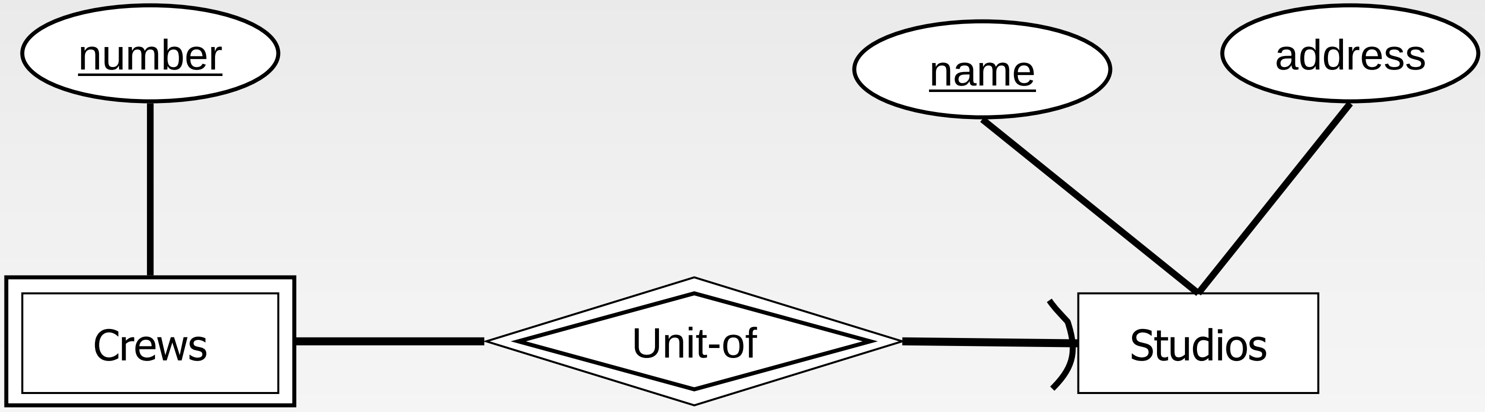
Causes of Weak Entity Sets

- 2.connecting entity sets to eliminate a multi-way relationship
  - These entity sets often have no attributes of their own. Their key is formed from the attributes that are the key attributes for the entity sets they connect.



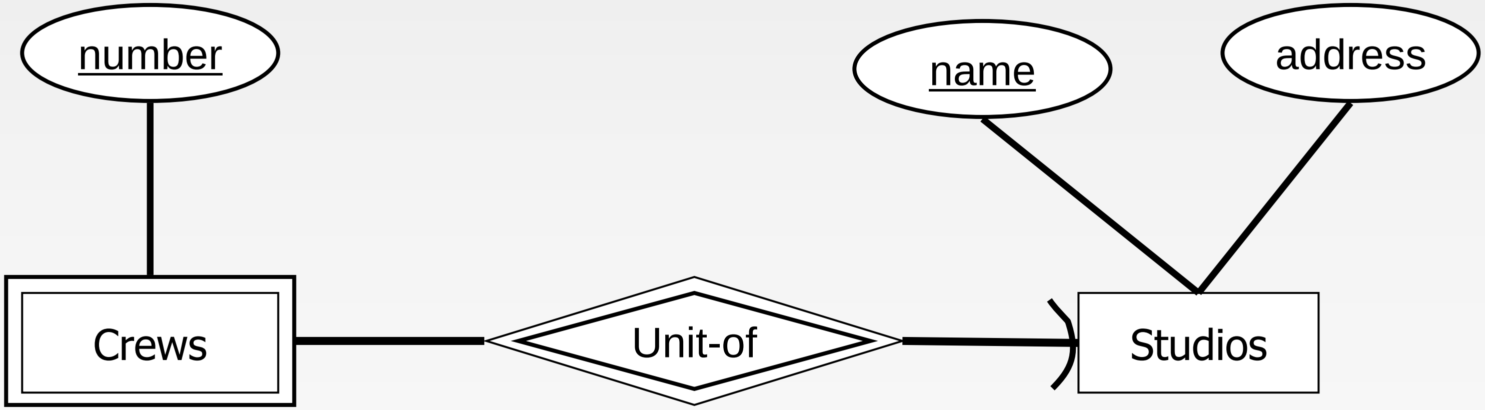
Requirements for Weak Entity Sets

- if E is a weak entity set, then its key consists of:
  - Zero or more of its own attributes, and
  - Key attributes from entity sets that are reached by certain many-one relationships from *E* to other entity sets. These many-one relationships are called supporting relationships for *E*.



Requirements for Weak Entity Sets

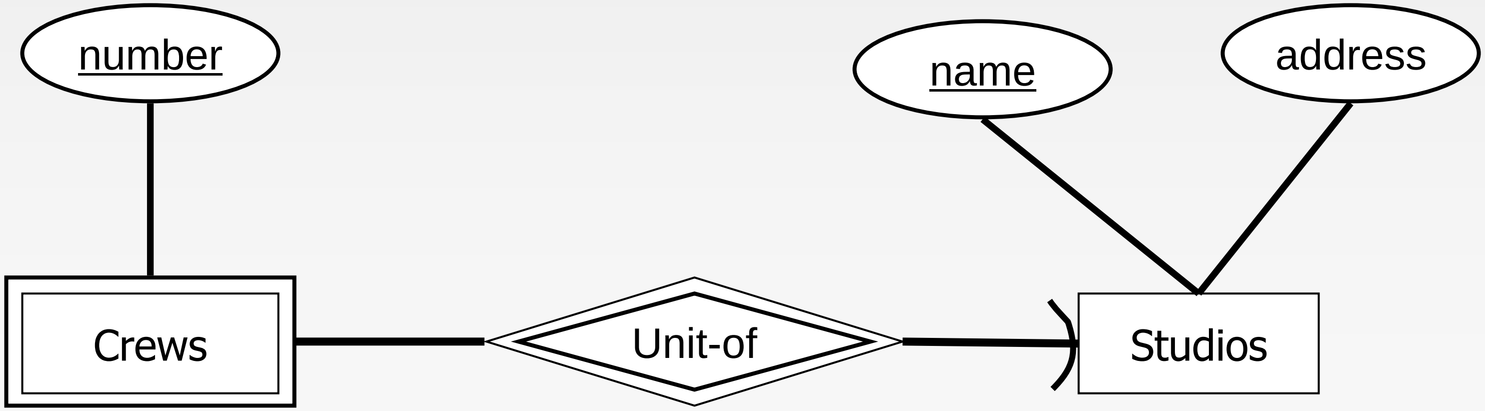
- In order for *R*, a many-one relationship from *E* to some entity set *F*, to be a supporting relationship for *E*, the following conditions must be obeyed:
  - *R* must be a binary, many-one relationship from *E* to *F*.
  - *R* must have referential integrity from *E* to *F*.
  - The attributes that *F* supplies for the key of *E* must be key attributes of *F*.
  - It is recursive if *F* itself is weak.
- Multiple supporting relationships are possible



Weak Entity Sets Notation

1. If an entity set is weak, it will be shown as a rectangle with a double border
2. Its supporting many-one relationship will be shown as diamonds with a double border
3. If an entity set supplies any attributes for its own key, then those attributes will be underlined

- Whenever we use an entity set *E with a double border,* it is weak. The key for *E is whatever attributes of E* are underlined plus the key attributes of those entity sets to which *E is connected by many-one* relationships with a double border.



From E/R Diagrams to Relational Designs

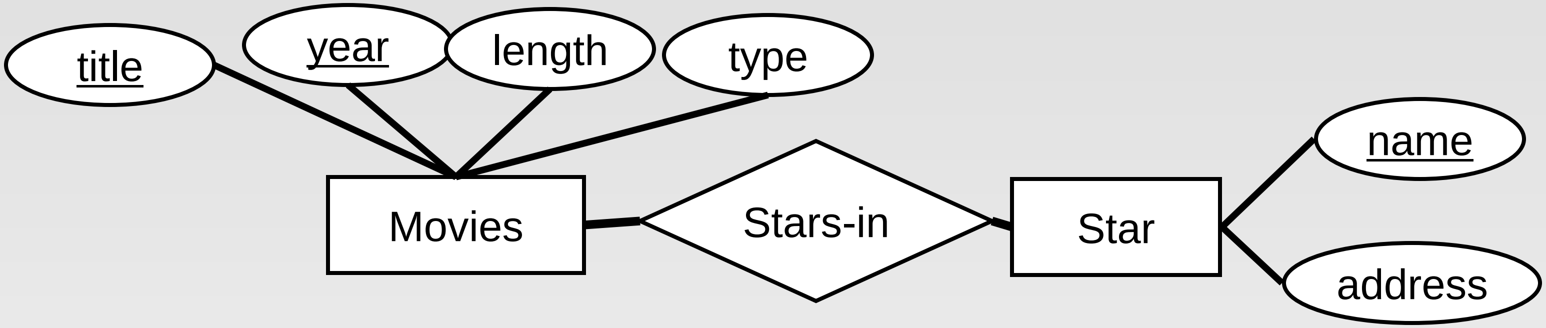
- From Entity Sets to Relations
- From E/R Relationships to Relations
- Combining Relations
- Handling Weak Entity Sets

General algorithm

- Turn each entity set into a relation with the same set of attributes
- Replace a relationship by a relation whose attributes are the keys for the connected entity sets.
- Special situations
  - Weak entity sets cannot be translated straightforwardly to relations
  - “Isa” relationships and subclasses require careful treatment
  - Sometimes, we do well to combine two relations, especially the relation for an entity set *E and the relation* that comes from a many-one relationship from *E to some* other entity set

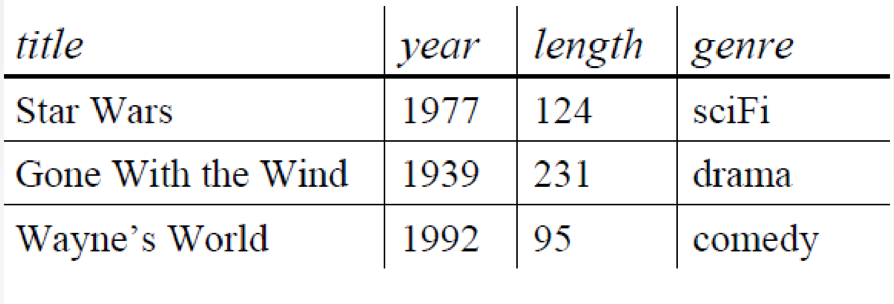
From Entity Sets to Relations

- For each non-weak entity set

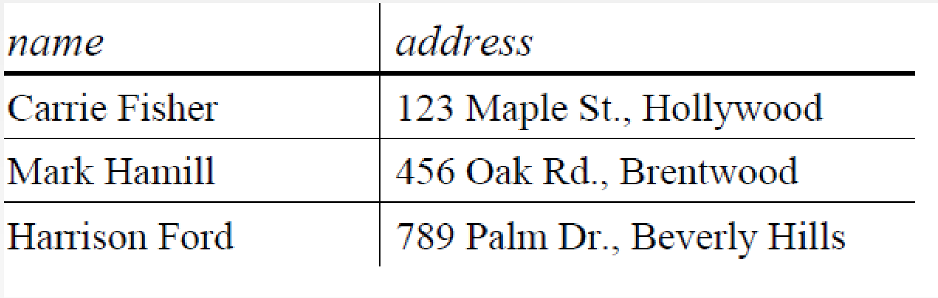


Movies (title, year, length, genre)

Stars (name, address)



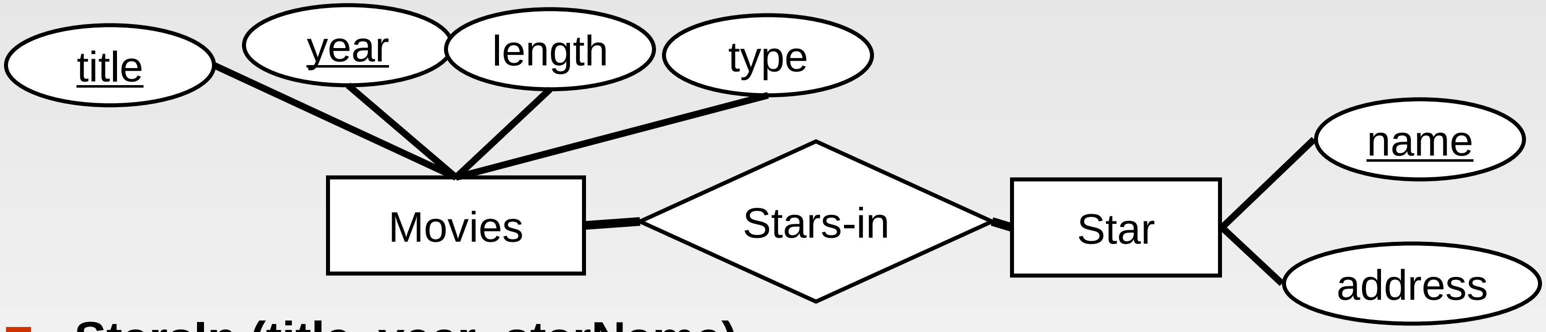
title	*year*	*length*	*genre*
Star Wars	1977	124	sciFi
Gone With the Wind	1939	231	drama
Wayne's World	1992	95	comedy



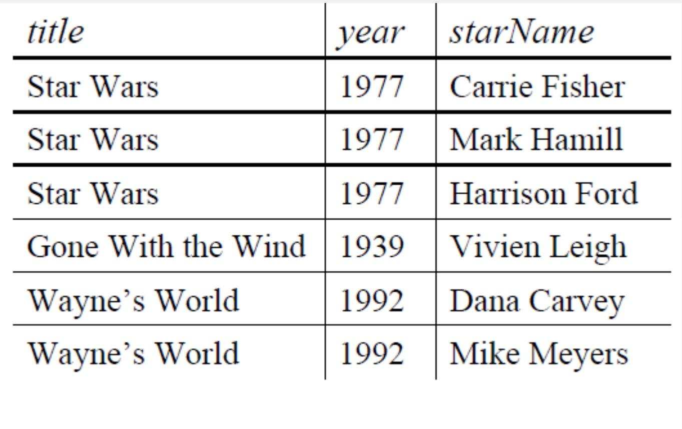
name	*address*
Carrie Fisher	123 Maple St., Hollywood
Mark Hamill	456 Oak Rd., Brentwood
Harrison Ford	789 Palm Dr., Beverly Hills

From E/R Relationships to Relations

- Relationships →Relations
  - For each entity set involved in relationship R, we take its key attribute and key attributes of its entities as part of the schema of the relation for R
  - If the relationship has attributes, then these are also attributes of relation for *R*



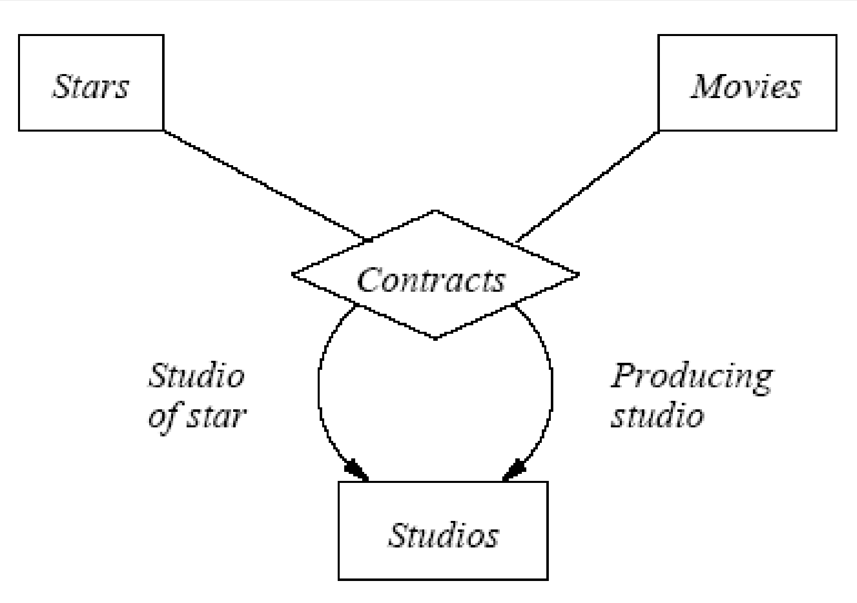
- **StarsIn (title, year, starName)**



title	*year*	*starName*
Star Wars	1977	Carrie Fisher
Star Wars	1977	Mark Hamill
Star Wars	1977	Harrison Ford
Gone With the Wind	1939	Vivien Leigh
Wayne's World	1992	Dana Carvey
Wayne's World	1992	Mike Meyers

From E/R Relationships to Relations

- Multiway relations are also easy to convert to relations.



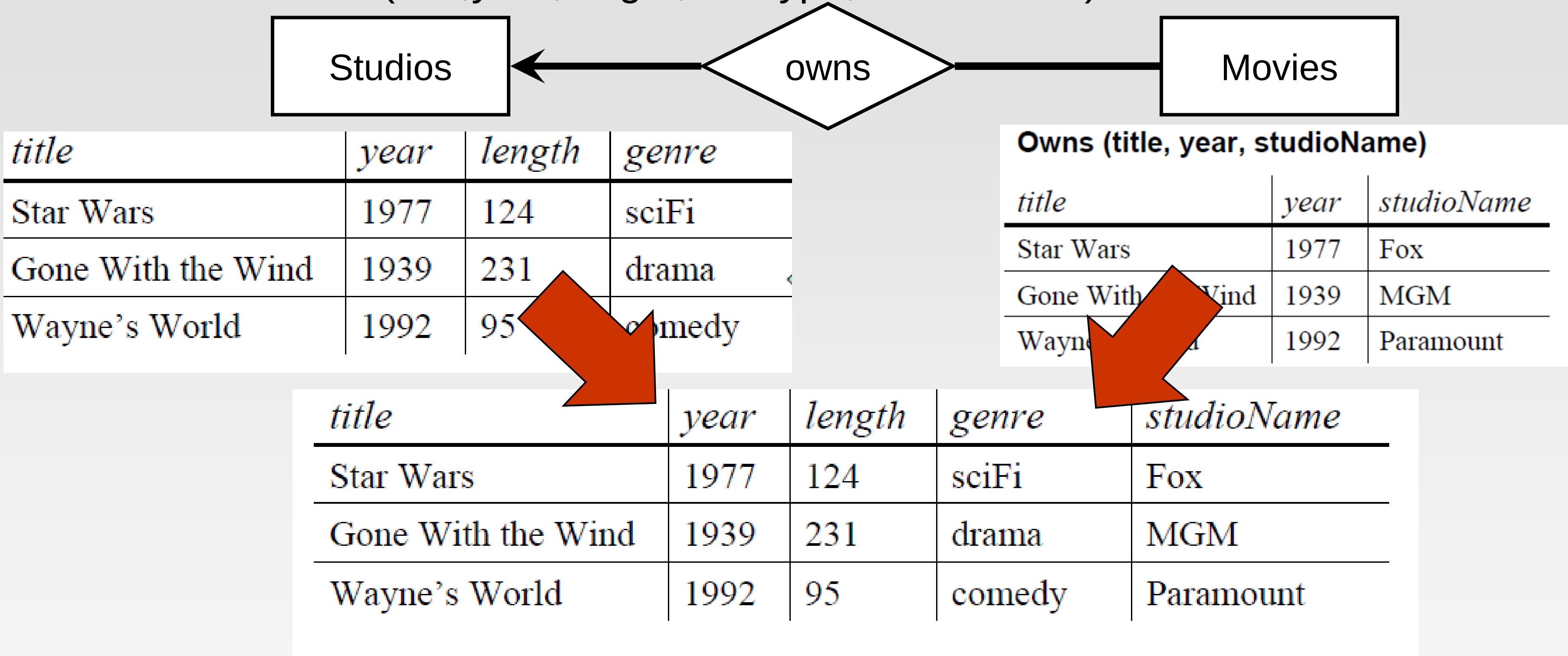
Contracts(starname, title, year, studioOfstar, producingStudio)

Combining Relations

- Combine relations for an entity set *E* and a relationship *R* (from *E* to *F*).
- Requirements:
  - R is a many-to-one relationship
  - Both relations *E* and *R* contain the key attribute(s) of *E*
- Then we can combine *E* and *R* with a new schema:
  - All attributes of E
  - The key attribute of F
  - Any attributes belonging to relationship R

Combining Relations

- Movie(title,year,length,filmType) and owns can be combined into one relation
  - Movie1(title,year,length,filmType, studioname)



title	*year*	*length*	*genre*
Star Wars	1977	124	sciFi
Gone With the Wind	1939	231	drama
Wayne's World	1992	95	comedy

Owns (title, year, studioName)

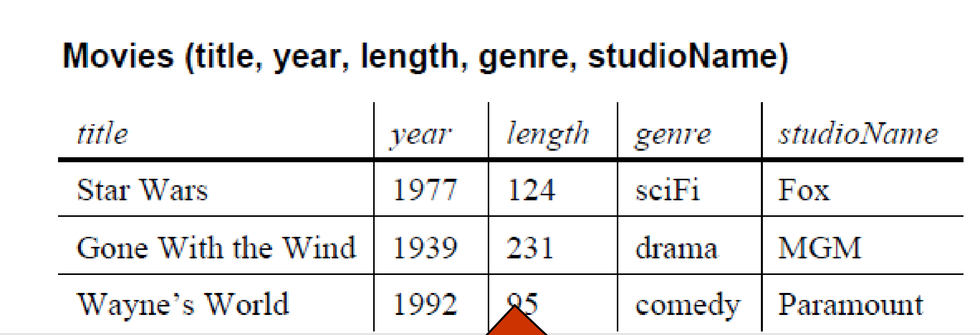
title	*year*	*studioName*
Star Wars	1977	Fox
Gone With the Wind	1939	MGM
Wayne's World	1992	Paramount

title	*year*	*length*	*genre*	*studioName*
Star Wars	1977	124	sciFi	Fox
Gone With the Wind	1939	231	drama	MGM
Wayne's World	1992	95	comedy	Paramount

How about an entity *e* in E is not related to any entity in F?

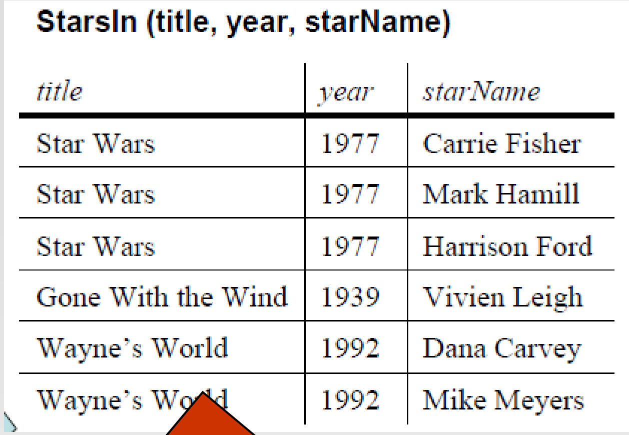
- "Null" value is introduced (it is not a formal part in relational model, but it is available in SQL).

Combining Relations



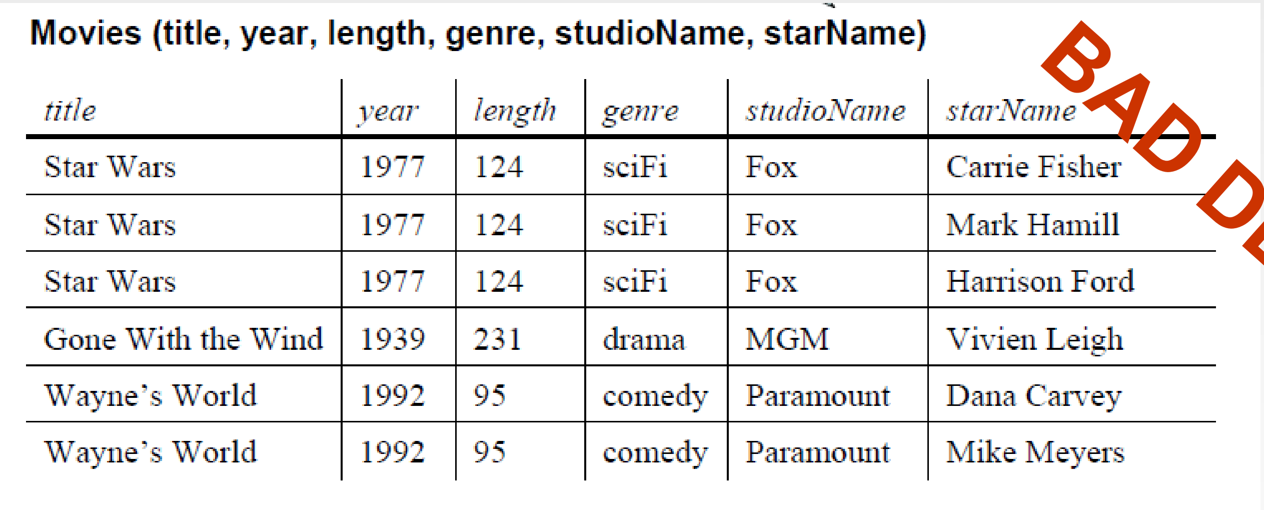
Movies (title, year, length, genre, studioName)

title	*year*	*length*	*genre*	*studioName*
Star Wars	1977	124	sciFi	Fox
Gone With the Wind	1939	231	drama	MGM
Wayne's World	1992	95	comedy	Paramount



StarsIn (title, year, starName)

title	*year*	*starName*
Star Wars	1977	Carrie Fisher
Star Wars	1977	Mark Hamill
Star Wars	1977	Harrison Ford
Gone With the Wind	1939	Vivien Leigh
Wayne's World	1992	Dana Carvey
Wayne's World	1992	Mike Meyers



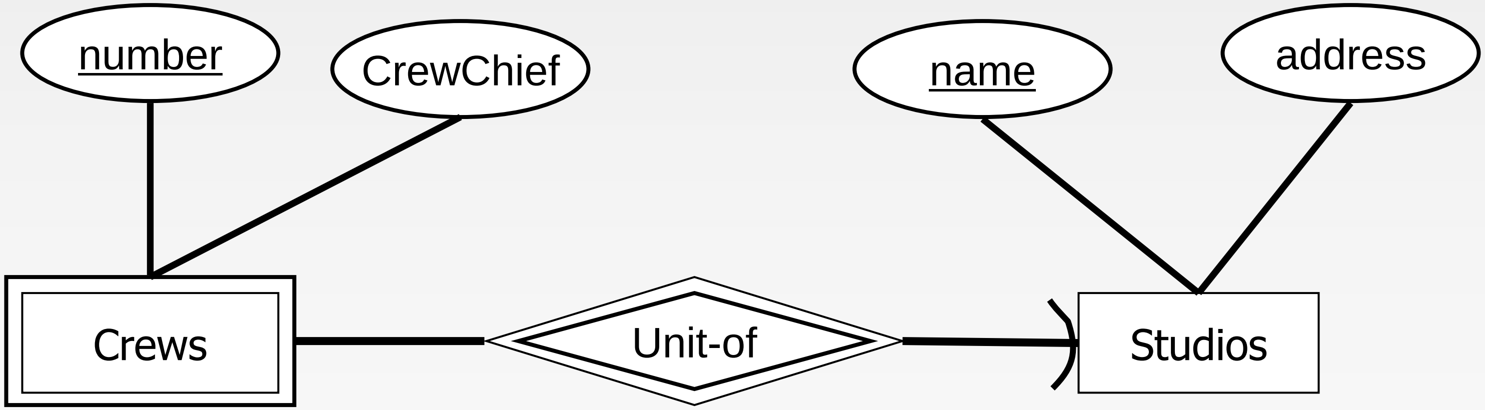
Movies (title, year, length, genre, studioName, starName)

title	*year*	*length*	*genre*	*studioName*	*starName*
Star Wars	1977	124	sciFi	Fox	Carrie Fisher
Star Wars	1977	124	sciFi	Fox	Mark Hamill
Star Wars	1977	124	sciFi	Fox	Harrison Ford
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramount	Dana Carvey
Wayne's World	1992	95	comedy	Paramount	Mike Meyers

BAD DESIGN

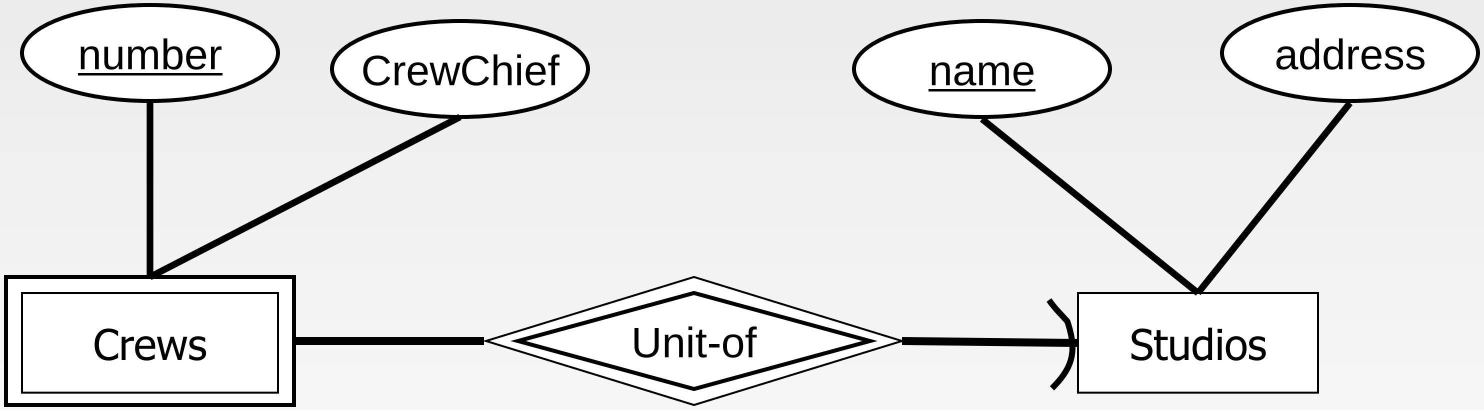
Handling Weak Entity Sets

- When weak entity sets appear
  - The relation for the weak entity set *W itself must include* not only the attributes of *W but also the key attributes of* the supporting entity sets.
  - The relation for any relationship in which the weak entity set *W appears must use as a key for W all of its key* attributes, including those of other entity sets that contribute to *W's key.*
  - However, a supporting relationship *R, from the weak entity* set *W to a supporting entity set, need not to be converted to a* relation at all.



Handling Weak Entity Sets

- Studio (name, addr)
- Crews (number, studioName, crewChief)
- ~~Unit-of (number, studioName, name)~~
  - **A supporting relationship needs no relation**



Handling Weak Entity Sets

- Modified rules
  - If W is a weak entity set, construct for *W a relation* whose schema consists of:

1. All attributes of *W*
2. All attributes of supporting relationships for *W*
3. For each supporting relationships for *W, say a many-one* relationship from *W to entity set E, all the key* attributes of *E*
4. Rename attributes, if necessary, to avoid name conflicts

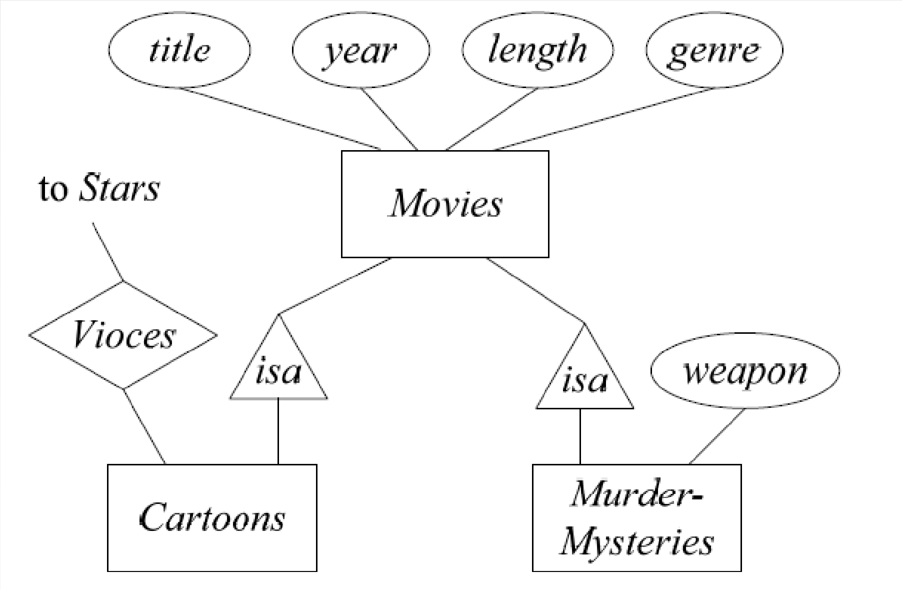
- –Do not construct a relation for any supporting relationship for *W*

Converting Subclass Structures to Relations

- The principal conversion strategies
  - Follow the E/R viewpoint
  - Treat entities as objects belonging to a single class
  - Use null values

E/R-Style Conversion

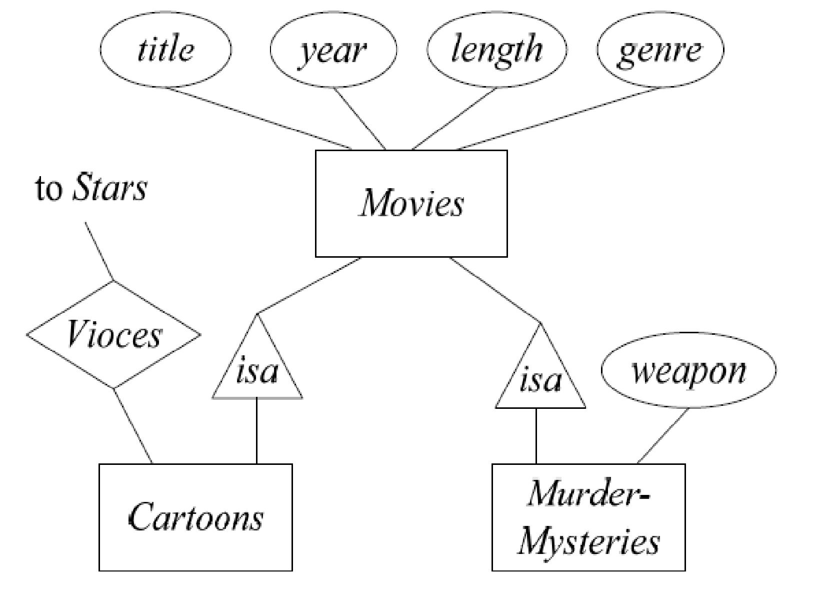
- The approach
  - Create a relation for each entity set, as usual.
  - If the entity set *E is not the root of the hierarchy, then the* relation for *E will include the key attributes at the root, to* identify the entity represented by each tuple, plus all the attributes of *E.*
- *Example*
  - Movies (title, year, length, genre)
  - MurderMysteries (title, year, weapon)
  - Cartoon (title, year)
  - Voice(title, year, starName)



An Object-Oriented Approach

- The approach
  - Enumerate all the possible subtrees that includes the root.
  - For each, create one relation that represents entities having components in exactly that subtree.
  - The schema for this relation has all the attributes of any entity set in the subtree. The assumption that entities are “objects” that belong to one and only one class.

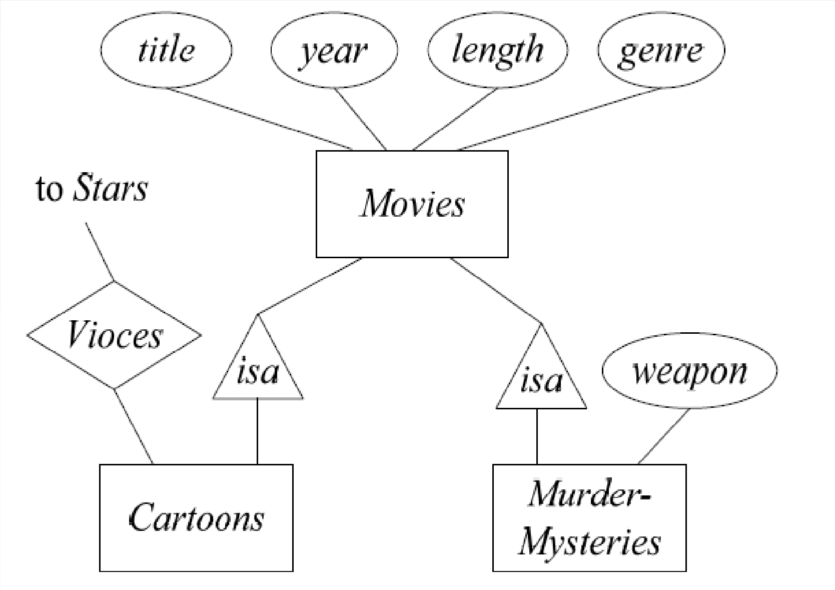
- Movies (title, year, length, genre)
- MoviesC (title, year, length, genre)
- MoviesMM (title, year, length, genre, weapon)
- MoviesCMM (title, year, length, genre, weapon)
- Voice(title, year, starName)



Using Null Values to Combine Relations

- The Approach
  - Create one relation with all the attributes of all the entity sets in the hierarch.
  - Each entity is represented by one tuple, and that tuple has a null value for whatever attributes the entity does not have.

Movies (title, year, length, genre, weapon)



Comparison of Approaches

1. For answering query the null method is faster because doesn’t need to join the tables.
    - What films of 2008 were longer than 150 minutes?
        - In E/R model it can be directly answered from the movie table but in the object oriented approach we need to look at all tables
    - What weapons were used in cartoons over 150 minutes
        - Is more difficult in the E/R model
        - In the object oriented method we need to only look at the MoviesCMM table

Comparison of Approaches

1. Not to use too many relations
    - The null method shines
    - The E/R approach uses one relation per entity set
    - Object oriented approach could have as many as 2 ^{n} relations where n is the number of entities.
2. Minimize space and avoid redundancy
    - Object oriented approach takes minimum space, nothing is repeated
    - The null method has a long tuple per each entity which may have many nulls. Potentially, with many entity sets in the hierarchy, a lot of nulls may happen
    - E/R method several tuples for each entity and the keys are repeated could take more or less space than null method.

Unified modeling Language

- Lecture given by Dr. Widom on Unified modeling Language