CS 4440 A

# Emerging Database Technologies

Lecture 2
01/10/24

# Announcements

- OH starts next week
  - Instructor: Thursday 3:30-4:30PM, KACB 3322
  - Catherine: Monday 2:00-3:00PM, outside KACB 3319
  - Hantian: Friday 1:00-2:00PM, outside KACB 3319
- Assignment 1 (due Jan 29, individual assignment, 10% grade)
  - Survey 3 technologies of your interest (must cover at least two categories)
    - Business Application Related
    - Core DBMS
    - Emerging Technology
  - Find and reference research papers properly (guidelines)
  - Fill out the Technology Survey

# Agenda

Data Model

E/R diagram

- Entity, entity set, attribute, relationship, relationship set
- Multiway relationship
- Subclass

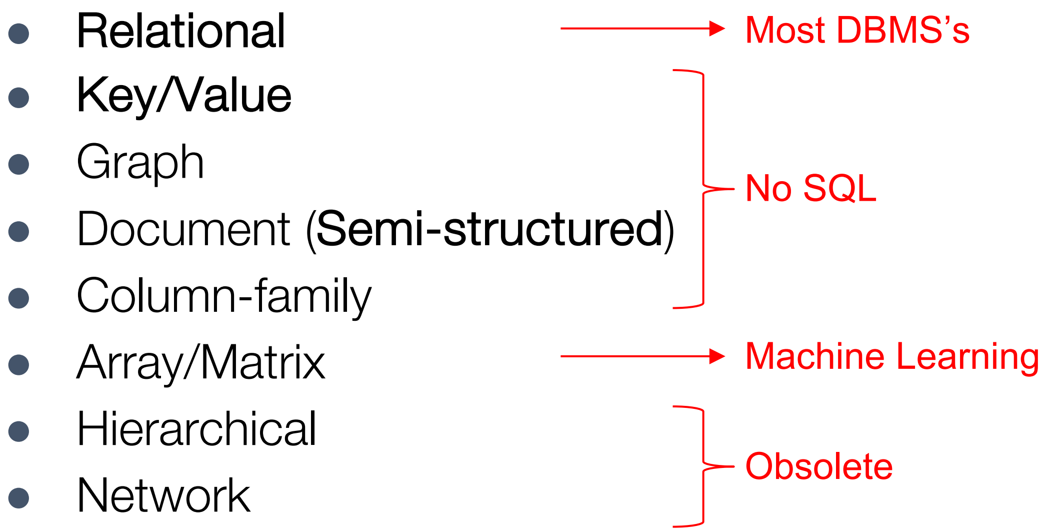
Design principles for ER Diagram

From E/R diagram to relational design

# Data model

- A notation for describing data or information
- Consists of:
  - Structure of the data
  - Operations on the data
  - Constraints on the data

# Data models



- **Relational**
- **Key/Value**
- Graph
- Document (**Semi-structured**)
- Column-family
- Array/Matrix
- Hierarchical
- Network

Most DBMS's

No SQL

Machine Learning

Obsolete

# The relational model

- Structure
  - Based on tables (relations)
  - Looks like an array of structs in C, but this is just one possible implementation
  - In database systems, tables are not stored as main-memory structures and must take into account the need to access relations on disk

| *title* | *year* | *length* | *genre* |
|---|---|---|---|
| Oldboy | 2003 | 120 | mystery |
| Ponyo | 2008 | 103 | anime |
| Frozen | 2013 | 102 | anime |

# The relational model

- Operations
  - Relational algebra
  - E.g., all the rows where genre is “anime”
- Constraints
  - E.g., Genre must be one of a fixed list of values, no two movies can have the same title

| *title* | *year* | *length* | *genre* |
|---|---|---|---|
| Oldboy | 2003 | 120 | mystery |
| Ponyo | 2008 | 103 | anime |
| Frozen | 2013 | 102 | anime |

# The semi-structured model

- Structure
  - Resembles trees or graphs, rather than tables or arrays
  - Represent data by hierarchically nested tagged elements
- Operations
  - Involve following path from element to subelements
- Constraints
  - Involve types of values associated with tags
  - E.g., <Length> tag values are integers,
    each <Movie> element must have a <Length>

```
<Movies>
  <Movie title=”Oldboy”>
    <Year>2003</Year>
    <Length>120</Length>
    <Genre>mystery</Genre>
  </Movie>
  <Movie title=”Ponyo”>
    <Year>2008</Year>
    …
</Movies>
```

# The key-value model

- Structure
  - (key, value) pairs
  - Key is a string or integer
  - Value can be any blob of data
- Operations
  - get (key), put(key, value)
  - Operations on values not supported
- Constraints
  - E.g., key is unique, value is not NULL

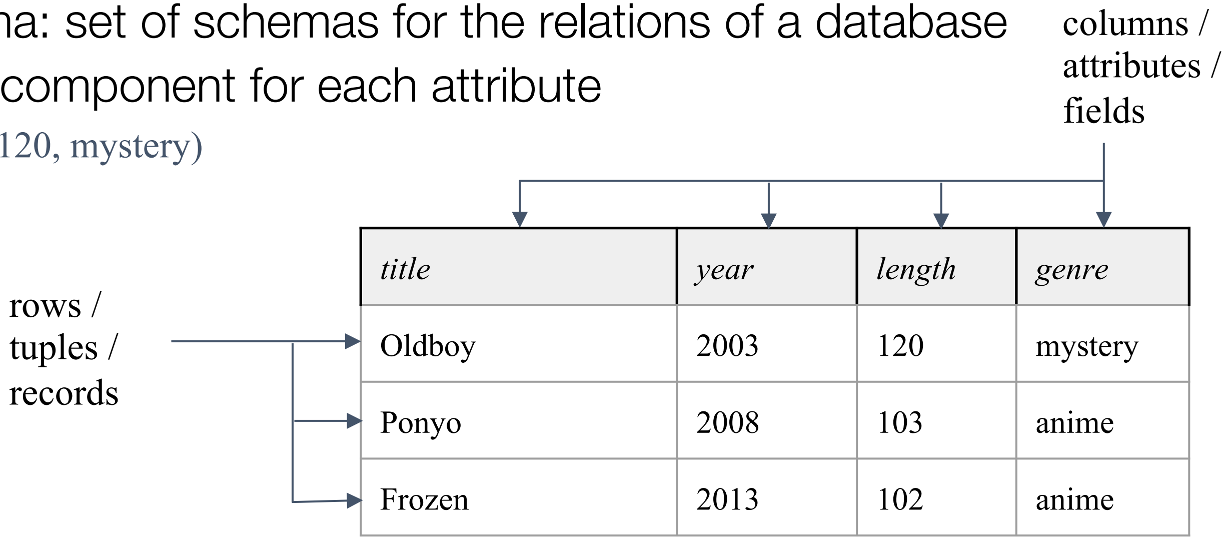
| *key* | *value* |
|---|---|
| 1000 | (oldboy, 2003) |
| 1001 | (ponyo, 2008) |
| 1002 | (frozen, 2013) |

# Comparison of modeling approaches

- Relational model
  - Simple and limited, but reasonably versatile
  - Limited, but useful operations
  - Efficient access to large data
  - A few lines of SQL can do the work of 1000’s of lines of C code
  - Preferred in DBMS’s
- Semi-structured model
  - More flexible, but slower to query
- Key-value model
  - Even more flexible, but cannot query

# Basics of the relational model

- Relation: two-dimensional table containing data
- Schema: relation name and set of attributes
  - Movies(title, year, length, genre)
- Database schema: set of schemas for the relations of a database
- A tuple has one component for each attribute
  - (Oldboy, 2003, 120, mystery)



columns / attributes / fields

rows / tuples / records

| *title* | *year* | *length* | *genre* |
|---|---|---|---|
| Oldboy | 2003 | 120 | mystery |
| Ponyo | 2008 | 103 | anime |
| Frozen | 2013 | 102 | anime |

# Equivalent representations of a relation

- A relation is a set of tuples (not a list)
- A schema is a set of attributes (not a list)
- Hence, the order of tuples or attributes of a relation is immaterial

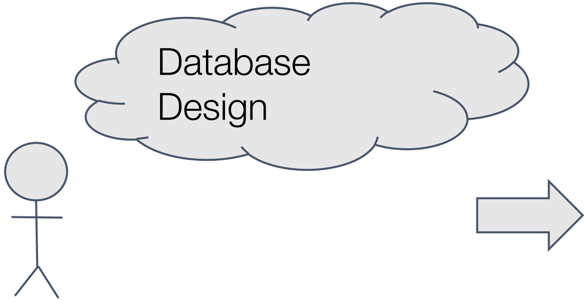
| *title* | *year* | *length* | *genre* |
|---|---|---|---|
| Oldboy | 2003 | 120 | mystery |
| Ponyo | 2008 | 103 | anime |
| Frozen | 2013 | 102 | anime |

=

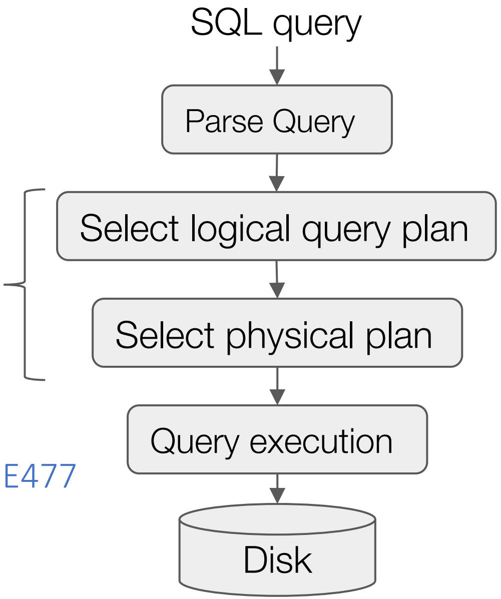
| *year* | *genre* | *title* | *length* |
|---|---|---|---|
| 2013 | anime | Frozen | 102 |
| 2003 | mystery | Oldboy | 120 |
| 2008 | anime | Ponyo | 103 |

# Skipping ahead

- In CS4400, we learned about SQL and relational algebra (for logical query plans)
- Before going into the details of query processing, let's discuss database design



Query optimization



Acknowledgement: The following slides have been adapted from EE477 (Database and Big Data Systems) taught by Steven Whang.

# Reading Materials

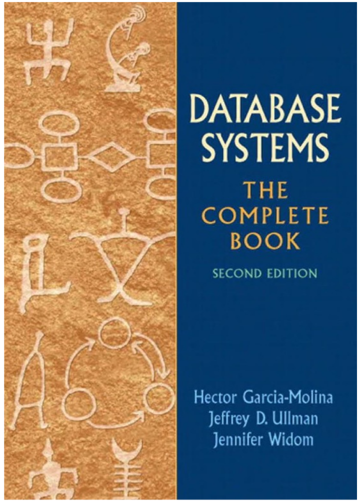
Database Systems: The Complete Book (2nd edition)

- Chapter 4: High-Level Database Models (4.1- 4.6)

Supplementary materials

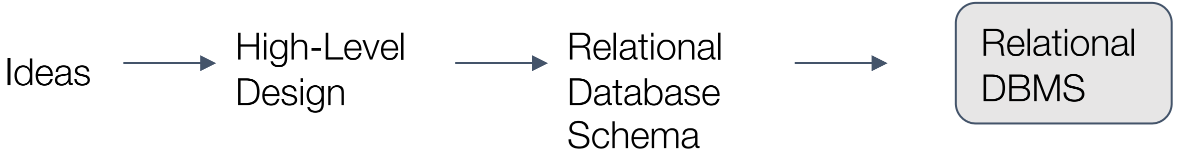
Fundamental of Database Systems (7th Edition)

- Chapter 3 - Data Modeling Using the Entity–Relationship (ER) Model



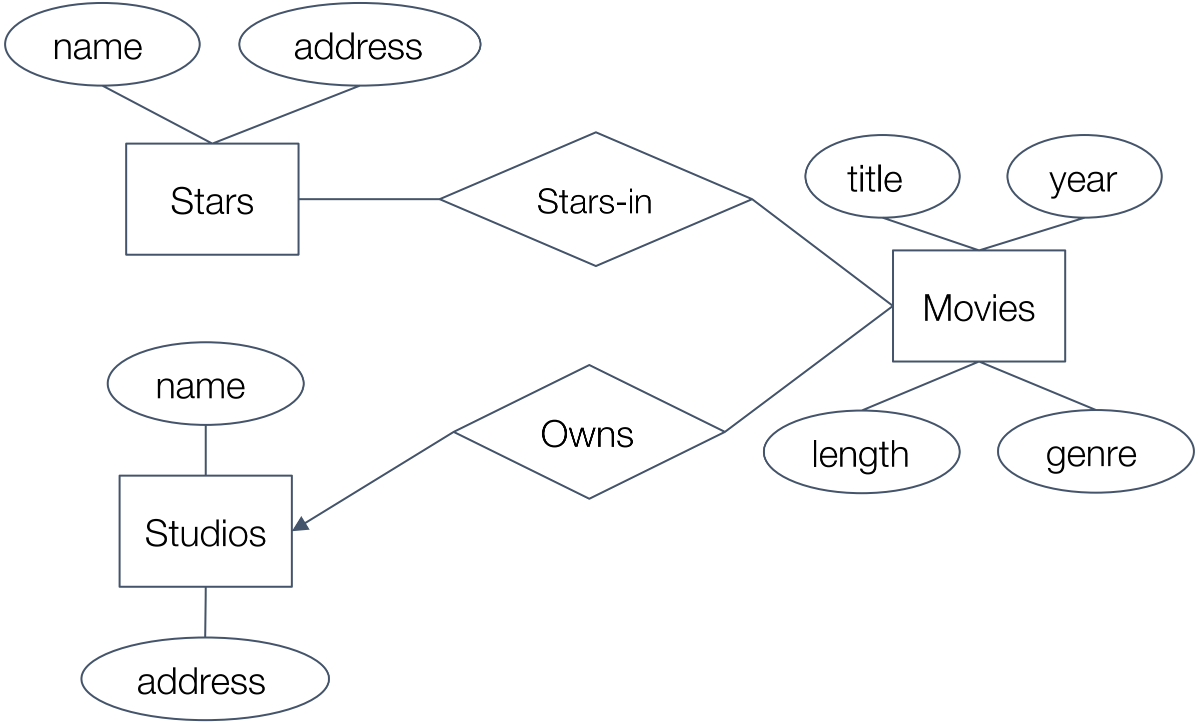
# Designing a database

- It is easier to start with a higher-level model and convert to a relational model
  - The relational model has only one concept (the relation)
  - This is good for efficient implementation, but not for designing
- Several possible notations
  - We will use the classic entity-relationship (E/R) diagram
    - Proposed in 1976 by Peter Chen to model databases
    - Most suitable for describing relations
  - Unified modeling language (UML)
    - Subsumes E/R diagrams, but also very general (used to visualize the design of a system)

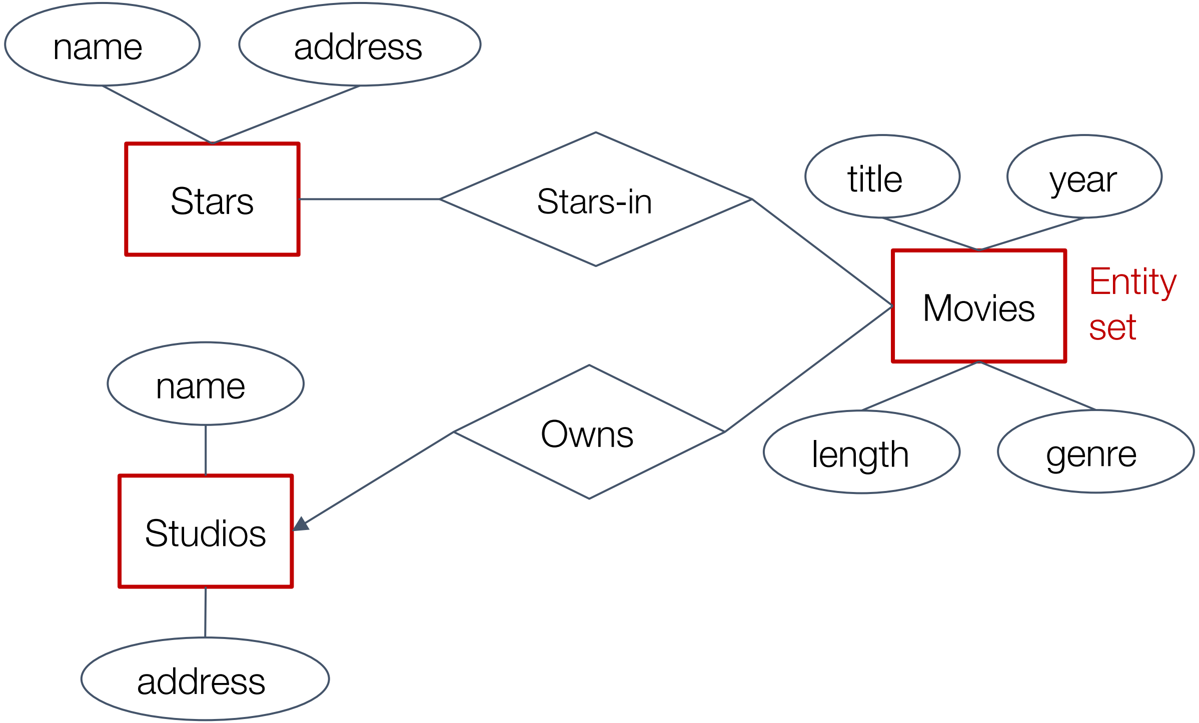


Ideas → High-Level Design → Relational Database Schema → Relational DBMS

# Entity-relationship (E/R) diagram for movie DB



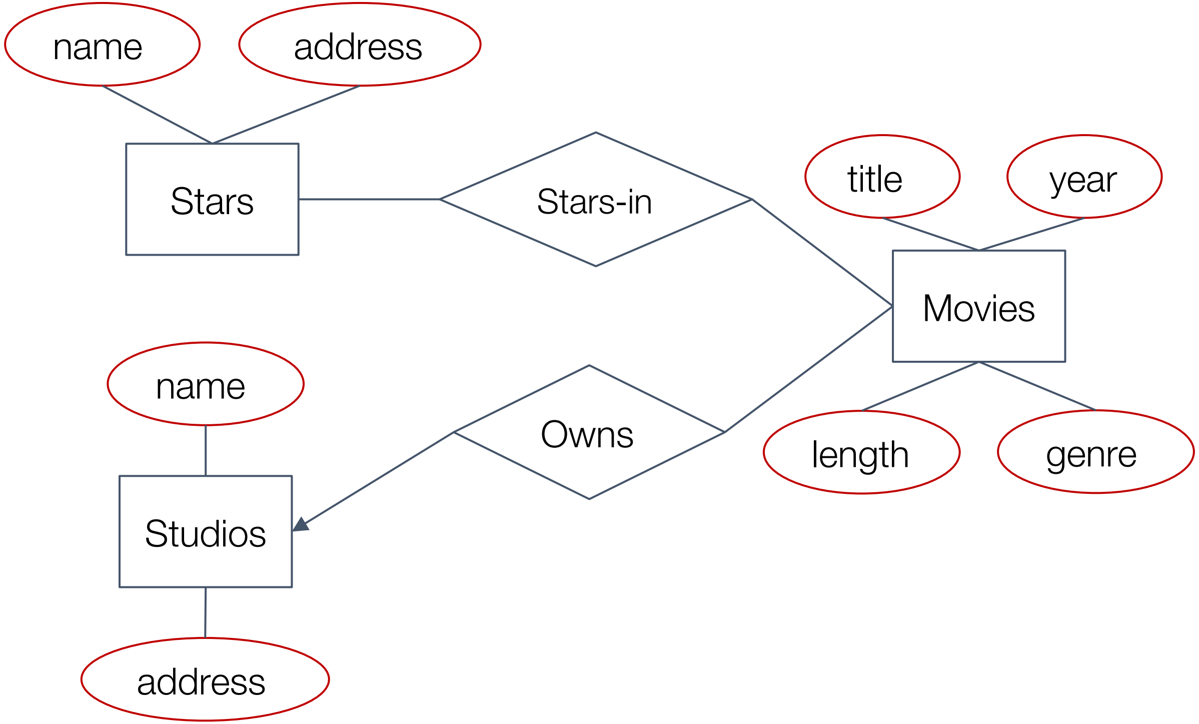
# Entity-relationship (E/R) diagram for movie DB



Entity: an object (no associated methods)

Entity set: a collection of similar entities

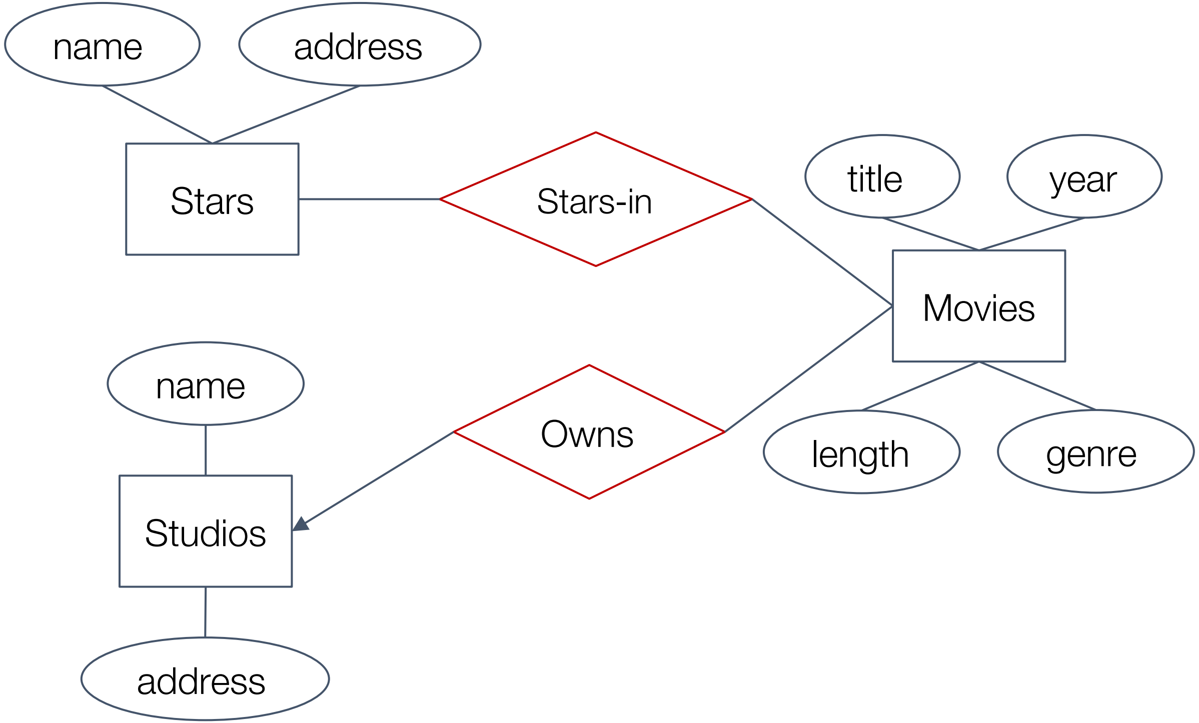
# Entity-relationship (E/R) diagram for movie DB



Attribute: a property of an entity in an entity set.

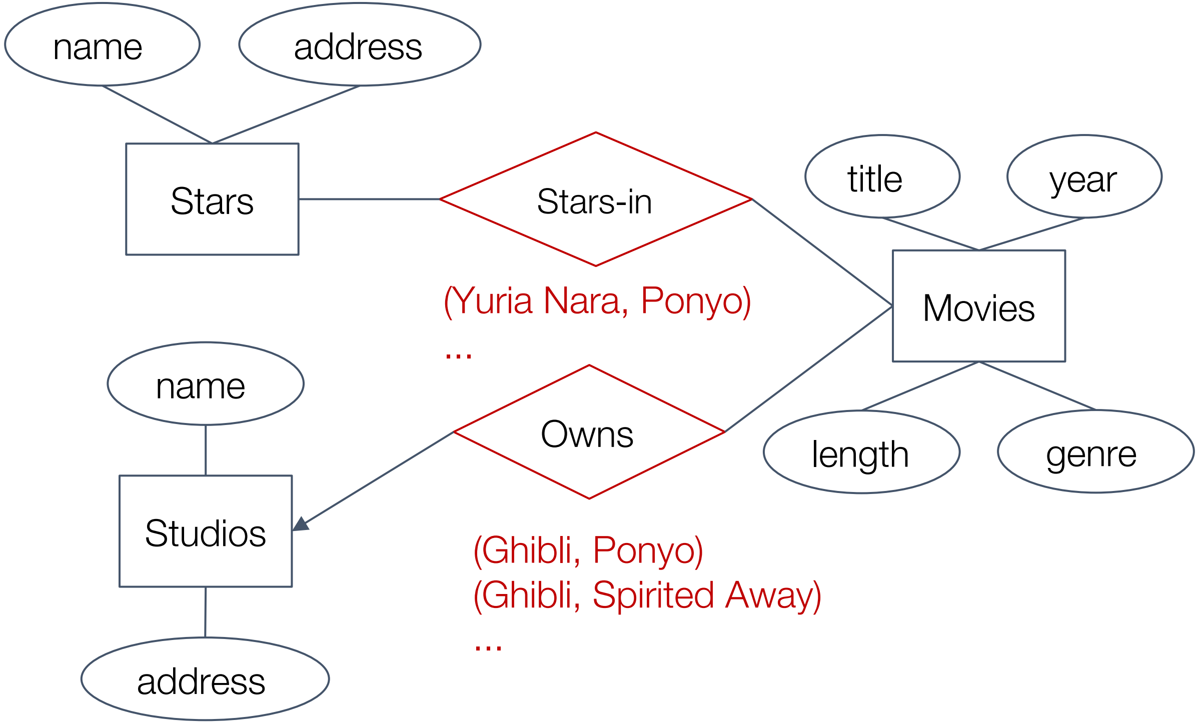
Types can be a primitive type (strings, integers, reals), struct, or set of primitives/structs.

# Entity-relationship (E/R) diagram for movie DB



Relationship: connections among two or more entity sets.

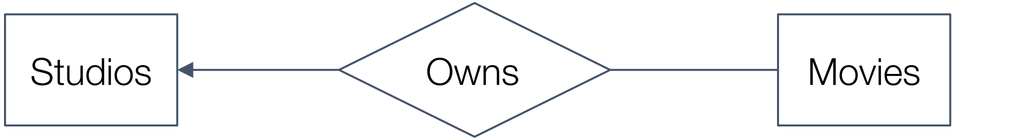
# Entity-relationship (E/R) diagram for movie DB



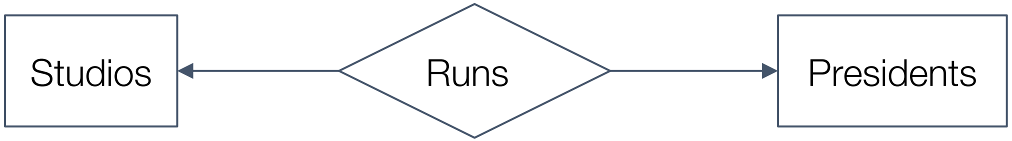
Relationship: connections among two or more entity sets.

# Multiplicity of binary relationships

- Relationships can be one-one, one-many, or many-many
- An arrow indicates “related to at most one entity”
  - Different than “exactly one”



*A movie has at most one studio*



*A president has at most one studio, and a studio has at most one president*

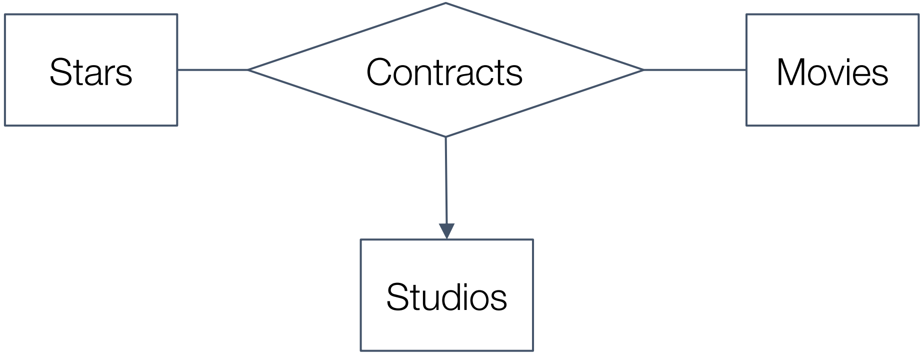
Q: What type of relationship does the bottom diagram indicate between Presidents and Studios?

# Exercise #1

- Draw an E/R diagram of Courses, Instructors, and Teaches

# Multiway relationships

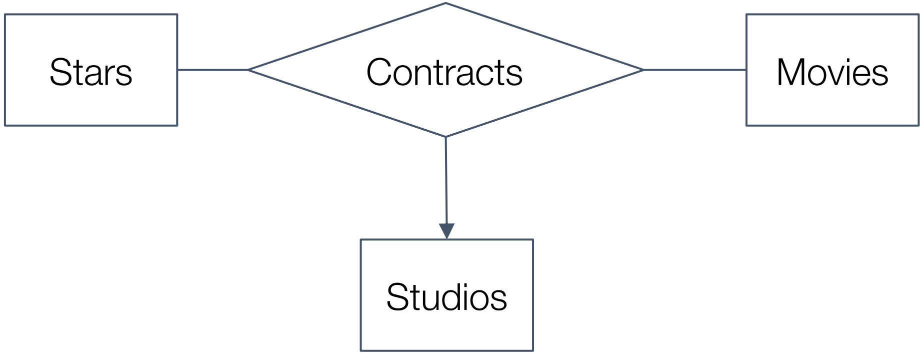
- Relationships may involve more than two entity sets (rare, but sometimes necessary)
- Below is a ternary or three-way relationship



*For each (star, movie), there is at most one studio with which the star has contracted for that movie*

# Multiway relationships

- Relationships may involve more than two entity sets (rare, but sometimes necessary)
- Below is a ternary or three-way relationship

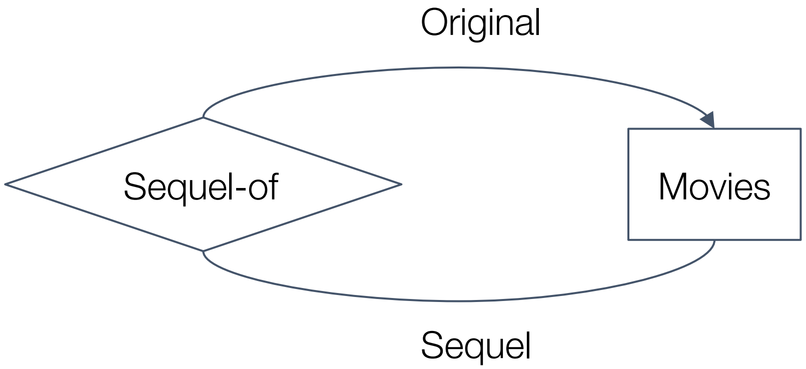


*For each (star, movie), there is at most one studio with which the star has contracted for that movie*

Q1: Can a studio contract with two stars for a movie?
Q2: Can a star contract with one studio for two movies?

# Roles in relationships

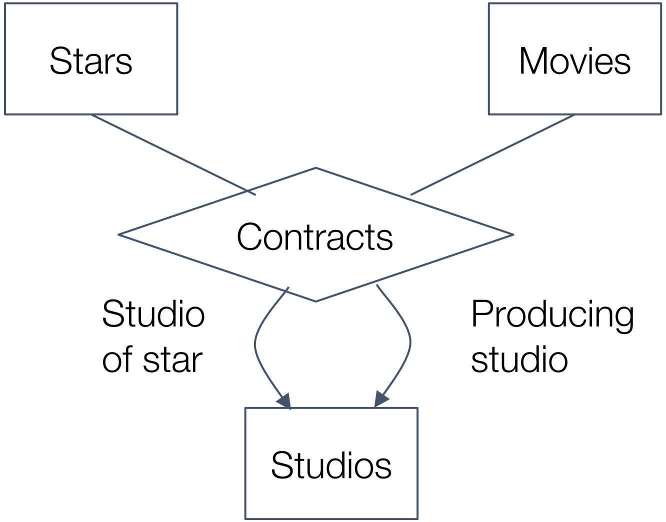
- One entity set can appear two or more times in a single relationship
- Label each edge with a role



A movie may have many sequels, but each sequel has at most one original movie

# Multiway relationship and multiple roles

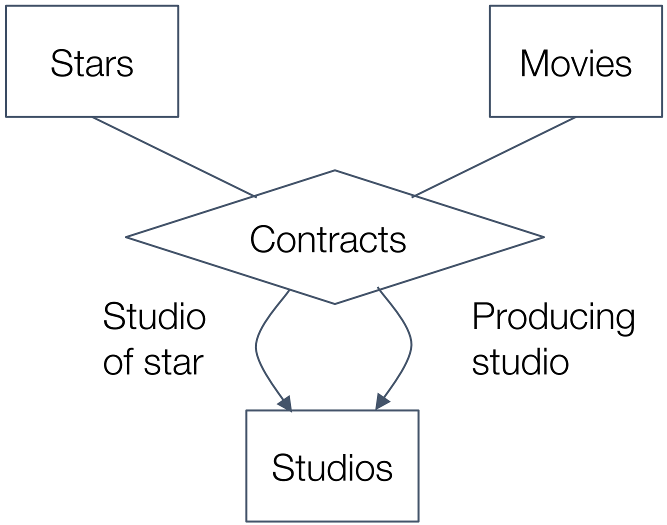
- A 4-way relationship



Q: Why no arrows to Stars or Movies?

# Interpretation of arrow in multiway relationships

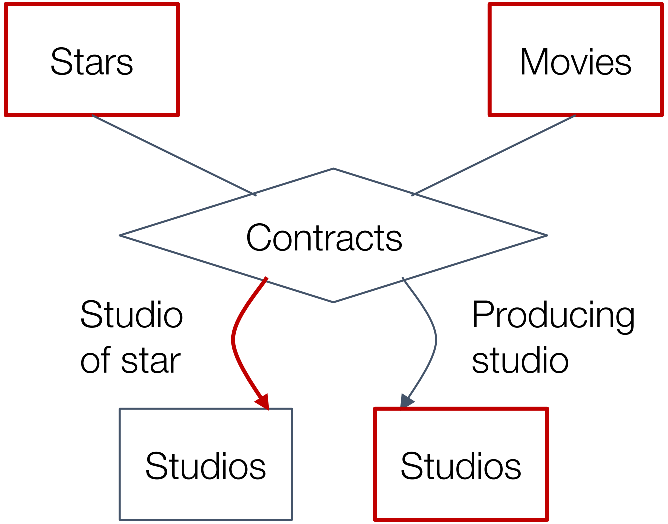
- A 4-way relationship



Arrow: if we select one entity from **each of the other entity sets in the relationship**, those entities are related to at most one entity in E.

# Interpretation of arrow in multiway relationships

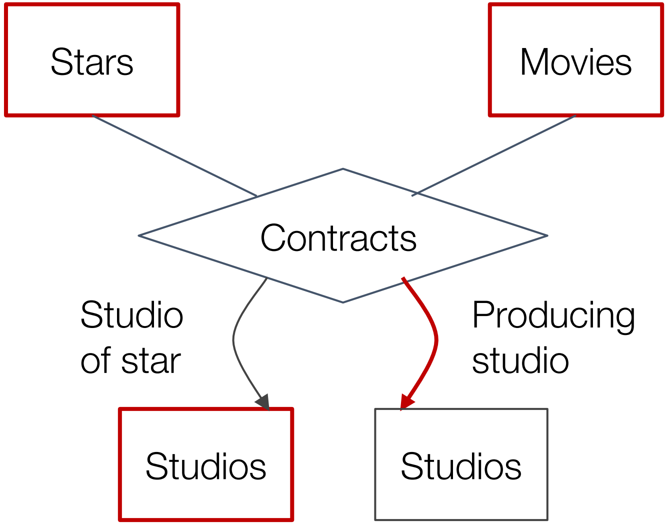
- A 4-way relationship



Arrow: if we select one entity from each of the other entity sets in the relationship, those entities are related to at most one entity in E.

# Interpretation of arrow in multiway relationships

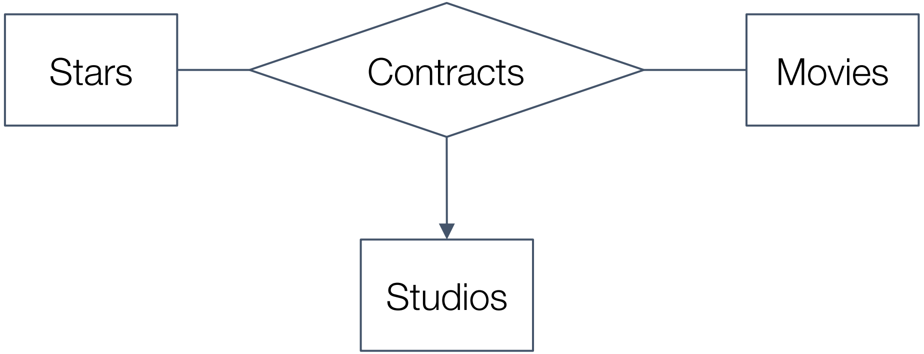
- A 4-way relationship



Arrow: if we select one entity from each of the other entity sets in the relationship, those entities are related to at most one entity in E.

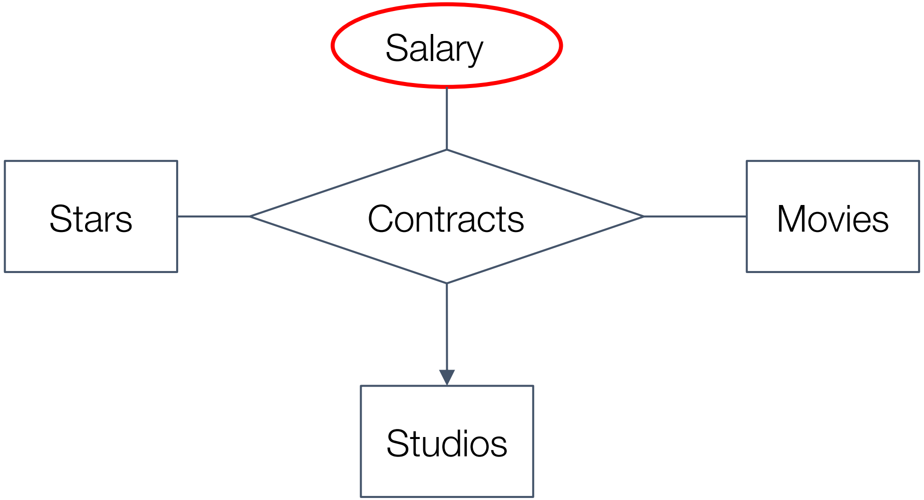
# Limits on arrow notation in multiway relationship

- The below notation says studio is a function of star and movie
- In reality, studio may be a function of movie alone
- Solution: also use functional dependencies (covered later)



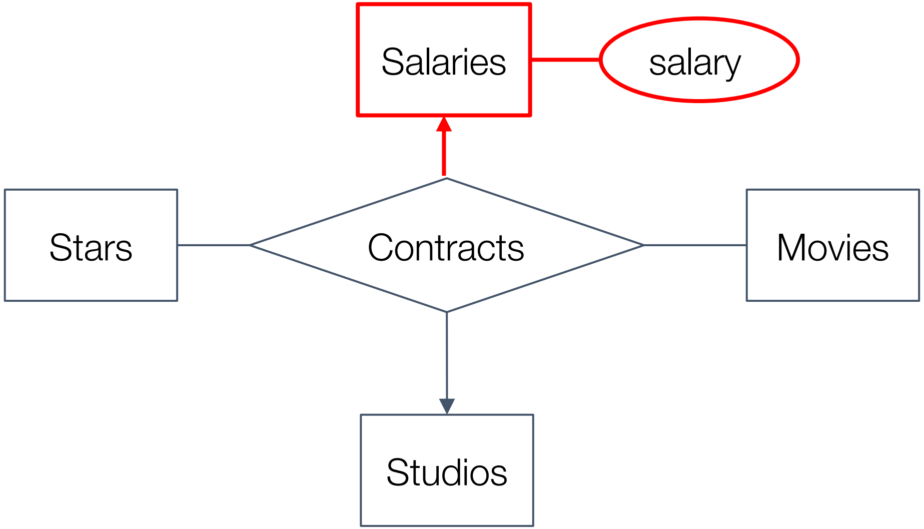
# Attributes on relationships

- Attributes can be associated with relationships instead of entity sets
- The attribute value is determined by the entire tuple in the relationship set



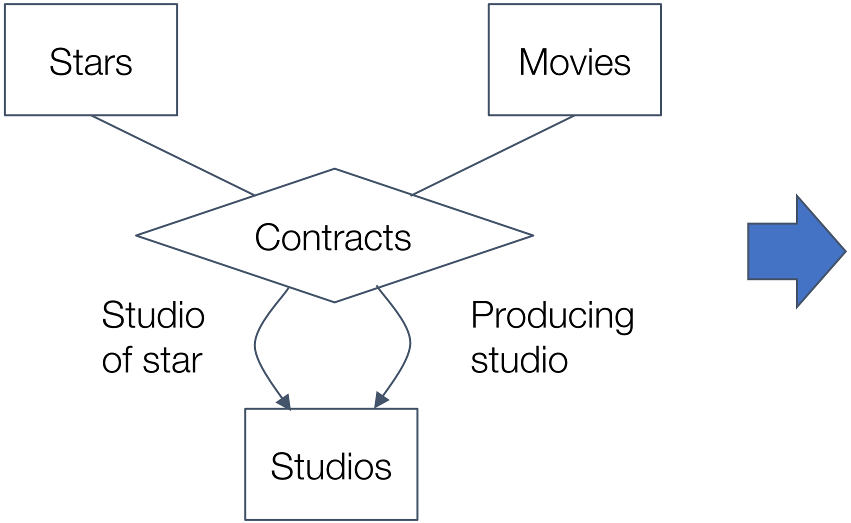
# Attributes on relationships

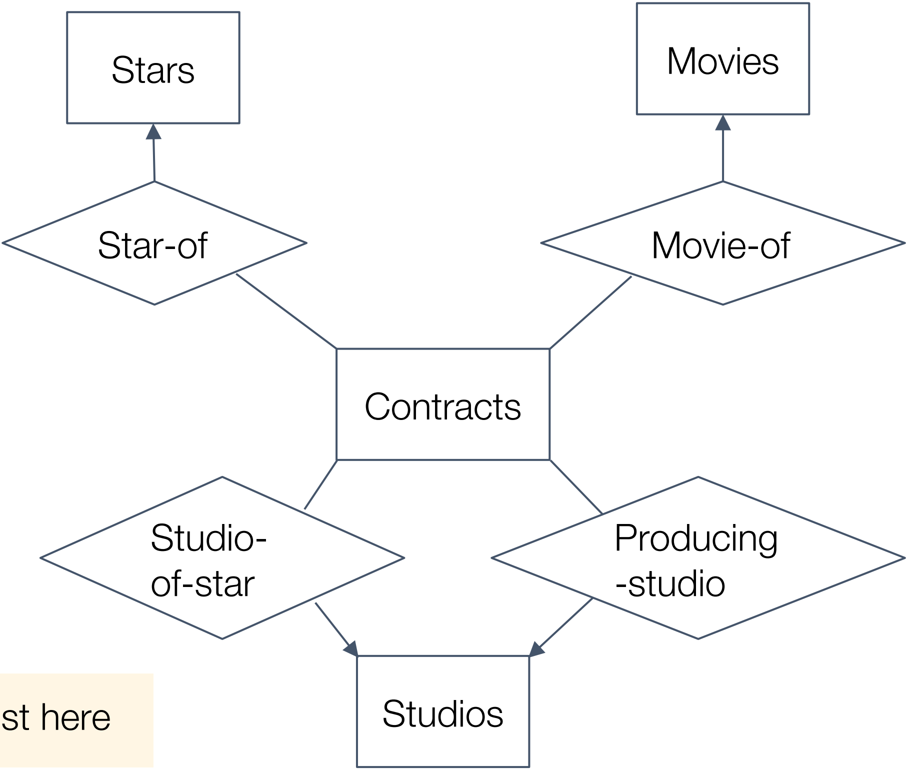
- Attributes can be associated with relationships instead of entity sets
- The attribute value is determined by the entire tuple in the relationship set
- Relationship attributes are not necessary and can be replaced with a new entity set



# Converting multiway relationships to binary

- Create a connecting entity set
- Introduce many-one relationships

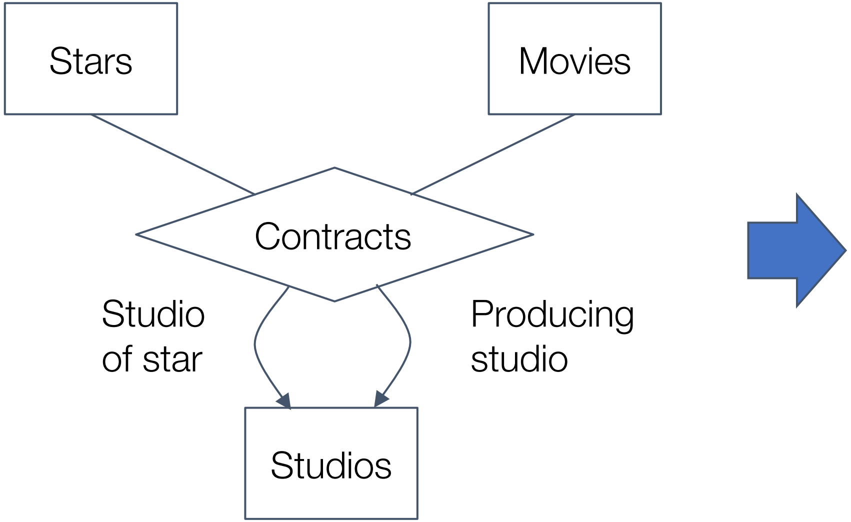


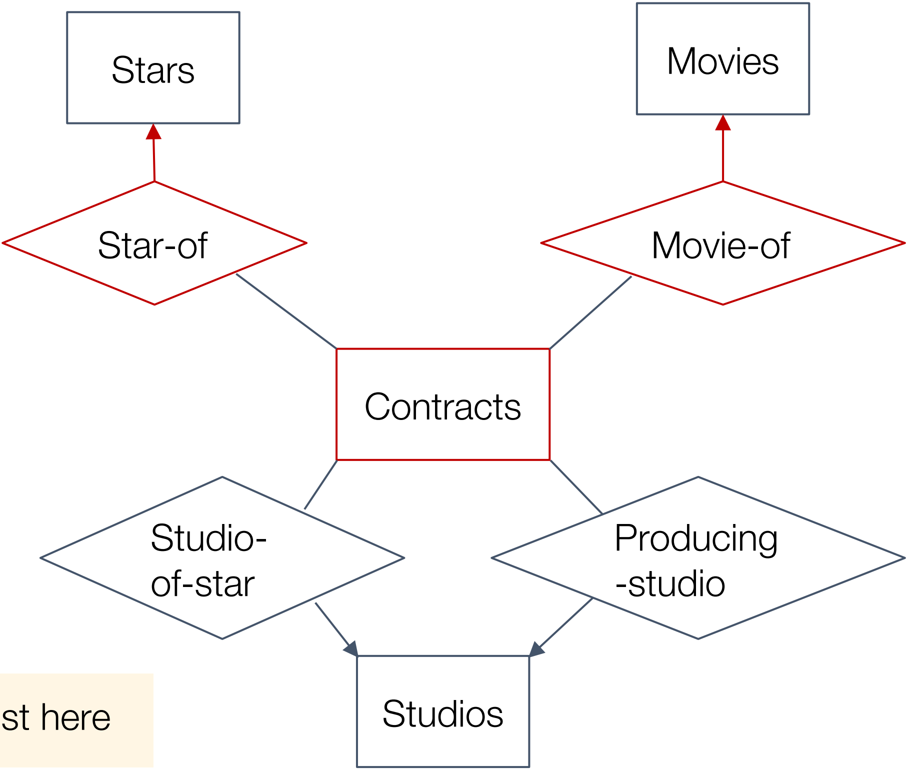


Note: some information is lost here

# Converting multiway relationships to binary

- Create a connecting entity set
- Introduce many-one relationships

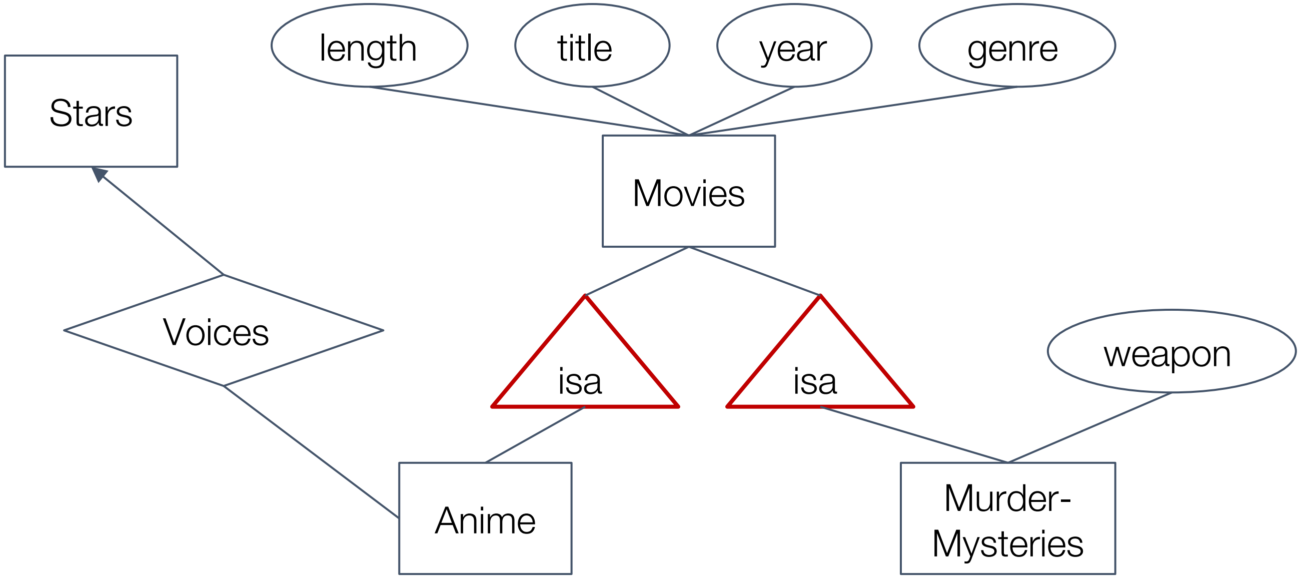




Note: some information is lost here

# Subclass

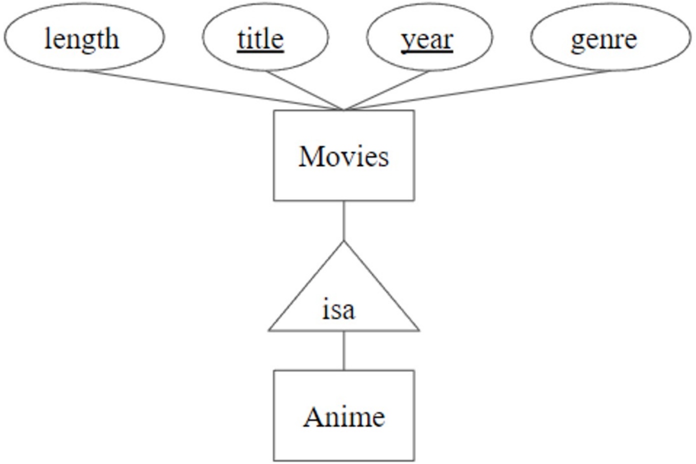
- An entity set may have a subclass of entities that have special properties not associated with other entities of the set
- Use isa relationships (triangle notation), which are always one-one and form a tree

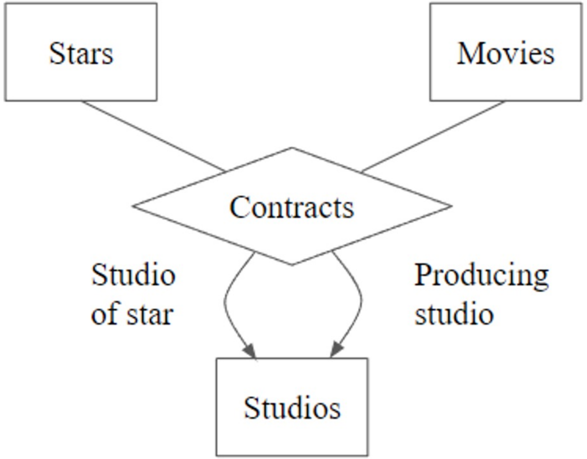


# Recap so far

## E/R diagram

- Entity, entity set, attribute, relationship, relationship set
- Multiway relationship
- Subclass

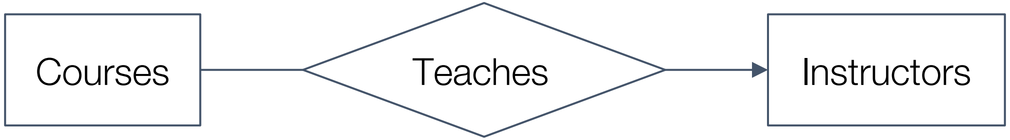




# Design principles for ER Diagrams

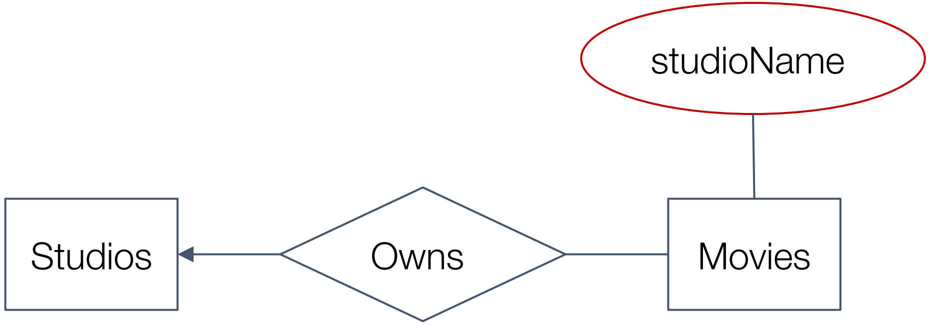
- #1: Faithfulness to reality

Doesn’t apply to GT



# Design principles for ER Diagrams

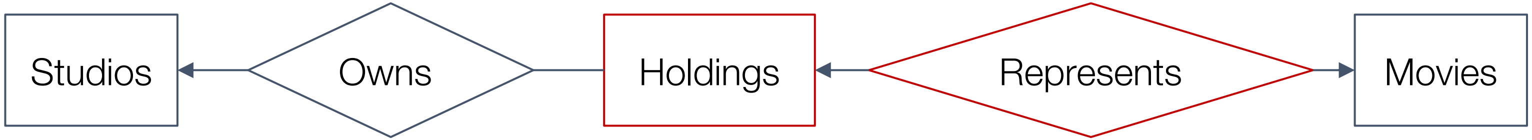
- #2: Avoiding redundancy



Redundant info

# Design principles for ER Diagrams

- #3: Simplicity counts

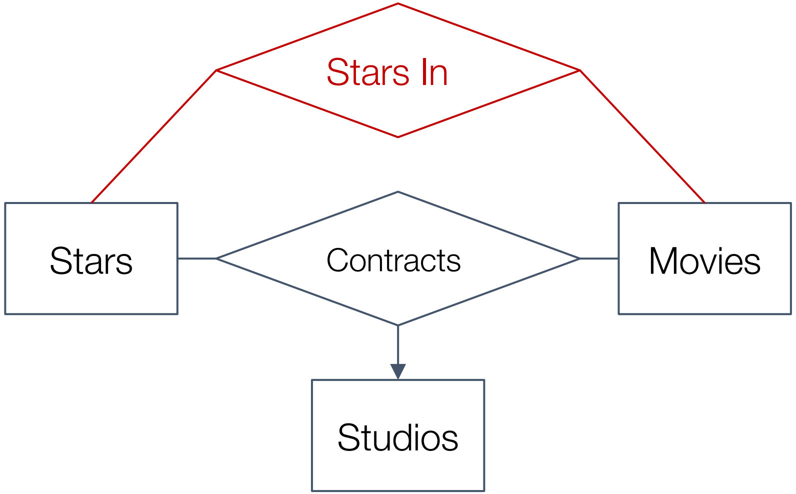


Unnecessary entity set and relationship

# Design principles for ER Diagrams

- #4: Choose the right relationships

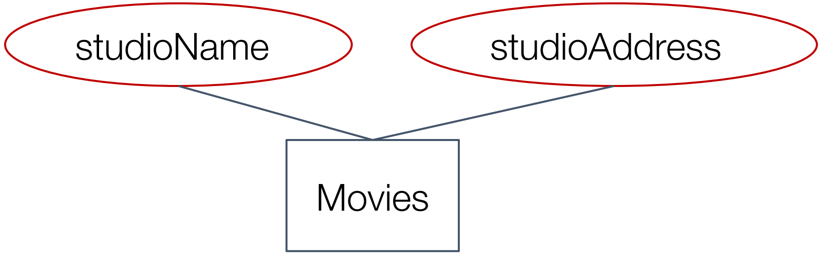
Unnecessary if contract implies stars in



# Design principles for ER Diagrams

- #5 Pick the right kind of element
  - Using attributes vs entity set/relationship combinations

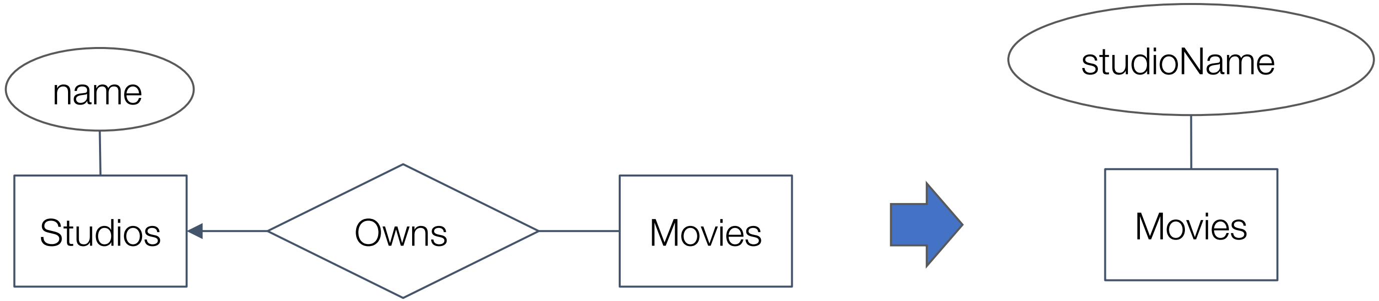
Problem when replacing studios entity set to movie attributes:



Need to repeat the address of the same studio for each movie, which is a cause for “update anomalies”

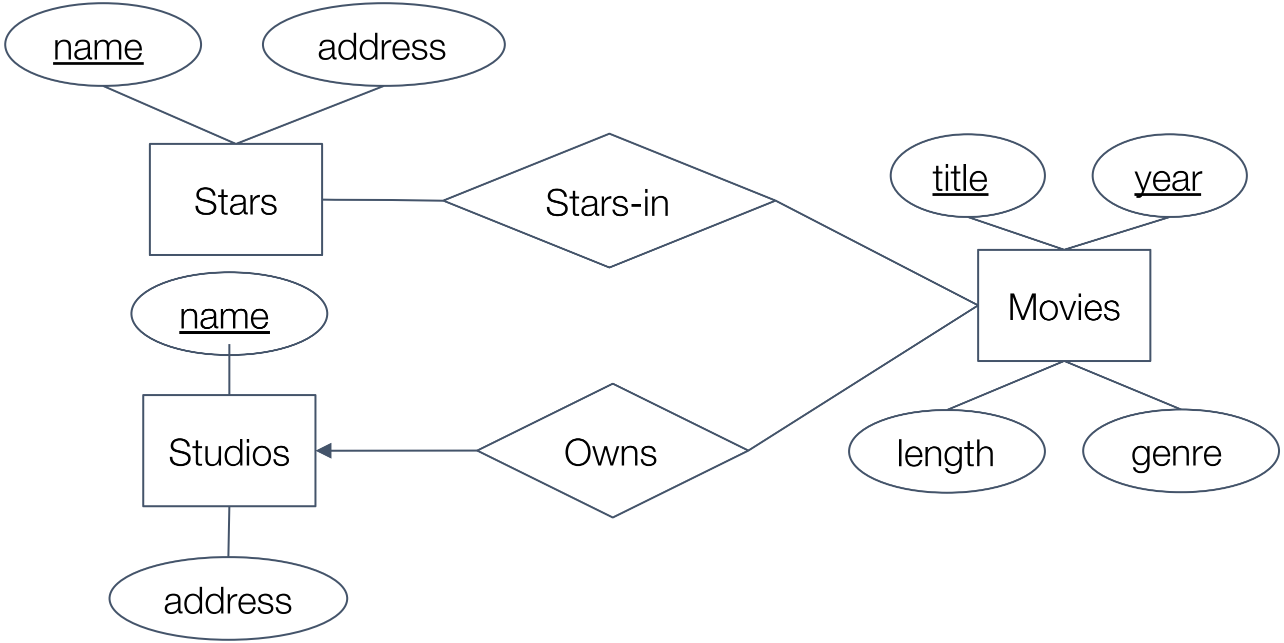
# Design principles for ER Diagrams

- #5 Pick the right kind of element
  - Using attributes vs entity set/relationship combinations
  - Replace entity set E with attribute(s) if:
    - All relationships have arrows to E
    - E's attributes do not depend on each other
    - No relationship involves E more than once



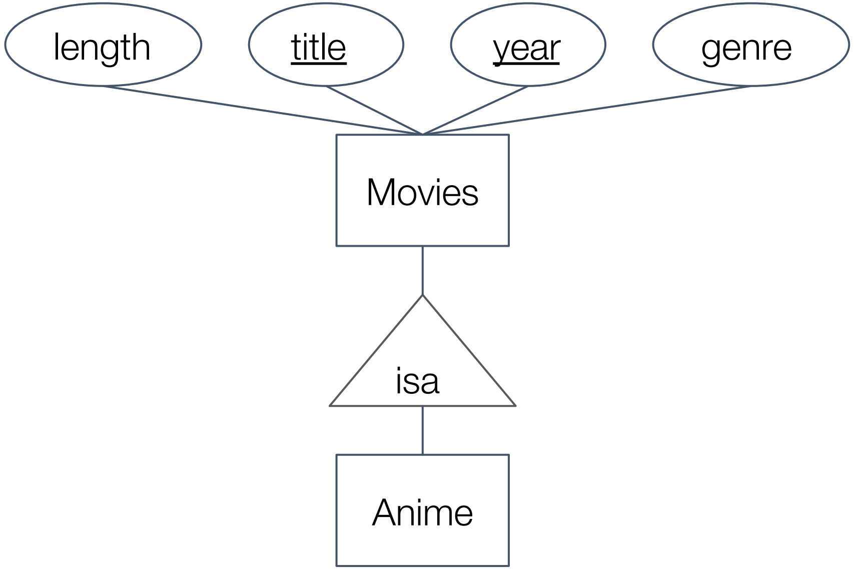
# Key constraints

- One or more attributes of an entity set can form a key
- Although there can be multiple keys, only the primary key attributes are underlined



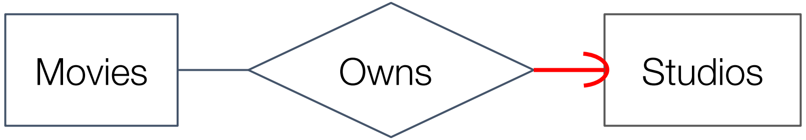
# Key constraints in the E/R Model

- If an entity set is involved in an isa-hierarchy, the root entity set must contain all the attributes needed for a key
- If an entity set is "weak", the key may belong to another entity set (explained later)



# Referential integrity constraints

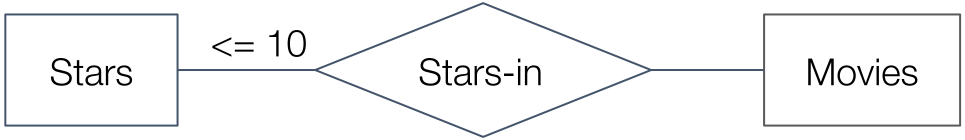
- Suppose R is a relationship from E to F
- A rounded arrow to F means
  - The relationship is many-one and
  - The entity of set F related to an entity of E must exist



*Every movie must be owned by one studio, and this studio is present in the Studios entity set*

# Degree constraints

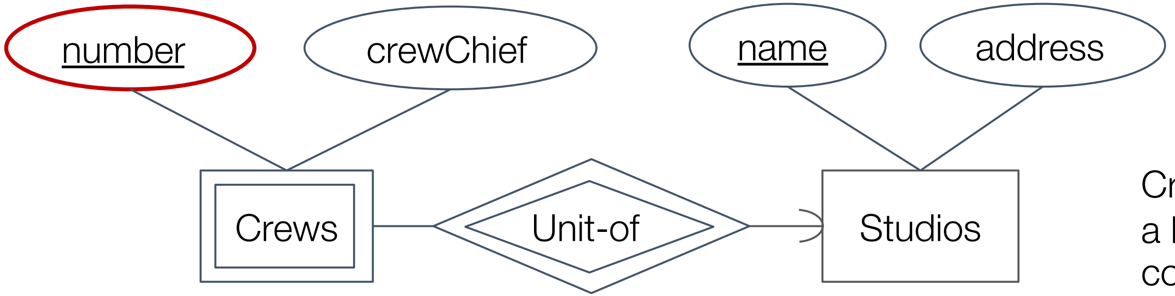
- Limit the number of entities connected to any one entity of the related entity set
  - Arrow is same as “<=1” constraint
  - Rounded arrow is same as “=1” constraint



*Every movie can be connected to at most 10 stars*

# Weak entity set

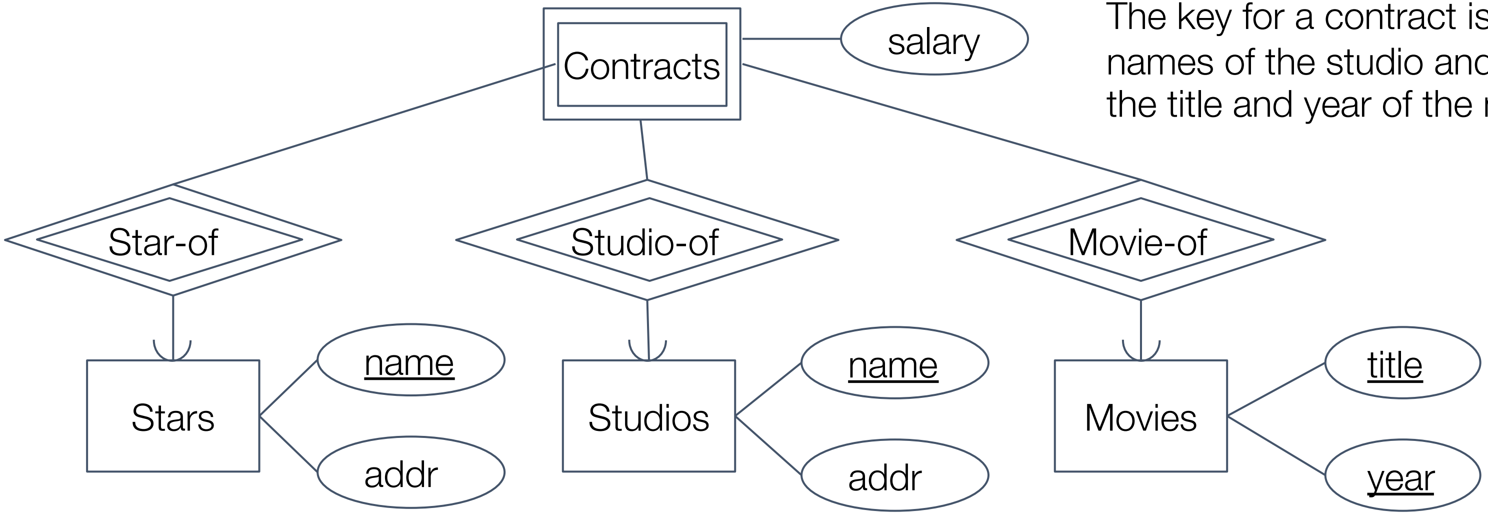
- An entity set where some of its attributes belong to another entity set
- A weak entity set may occur if
  - The entities of set E are subunits of entities in set F (≠ isa relation)



Crew number itself may not be a key and needs to be combined with the studio name

# Weak entity set

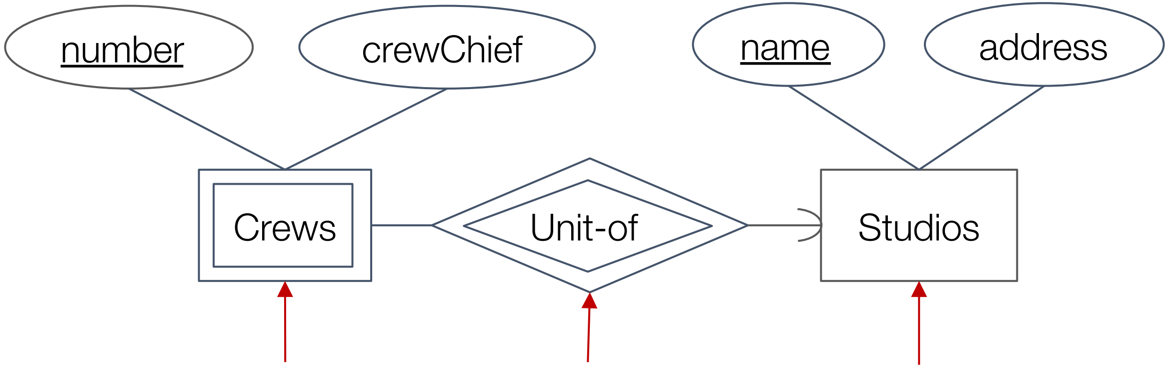
- An entity set where some of its attributes belong to another entity set
- A weak entity set may occur if
  - The entities of set E are subunits of entities in set F (≠ isa relation)
  - A connecting entity is introduced to eliminate a multiway relationship



The key for a contract is the names of the studio and star and the title and year of the movie

# Weak entity set notation

- A weak entity set is shown as a rectangle with a double border
- Supporting many-one relationships are diamonds with a double border
- Any attributes that form a key are underlined



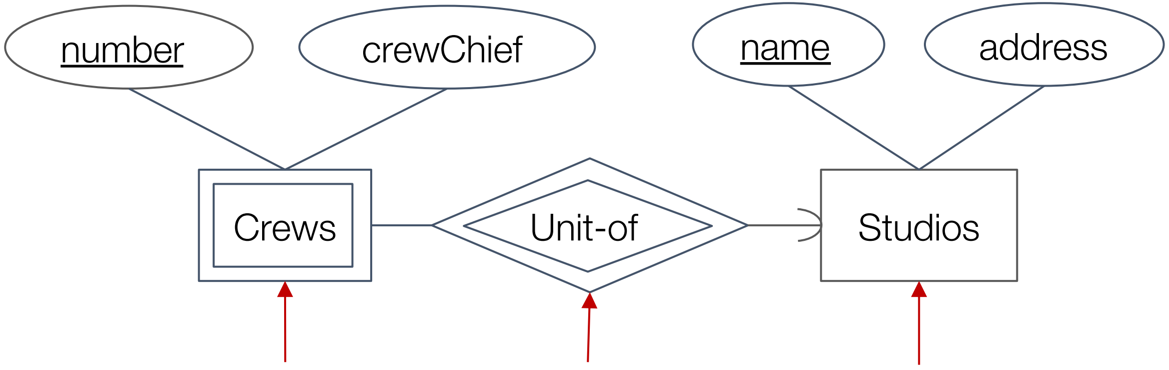
Weak entity set

Supporting relationship

Supporting entity set

# Requirements for weak entity set

- Key: zero or more of its own attributes + key attributes of supporting entity sets through supporting relationships
- Supporting relationship must be binary, be many-one, and have referential integrity



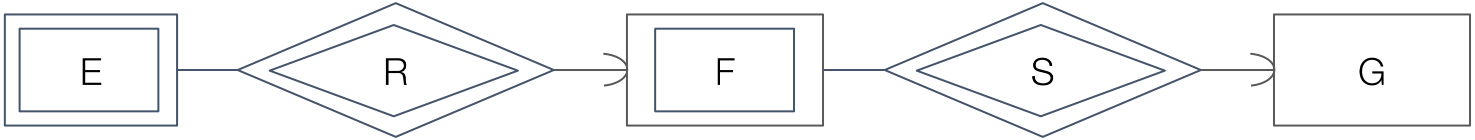
Weak entity set

Supporting relationship

Supporting entity set

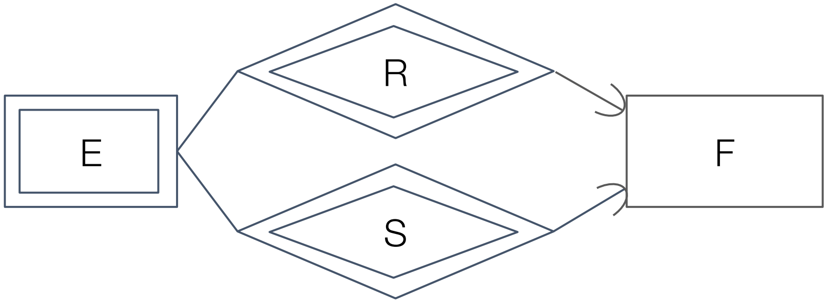
# Requirements for weak entity set

- The supporting entity set may itself be weak where some of its key attributes are supplied by other supporting entity sets, possibly recursively



# Requirements for weak entity set

- There may be several different supporting relationships to the same entity set



# Exercise #2

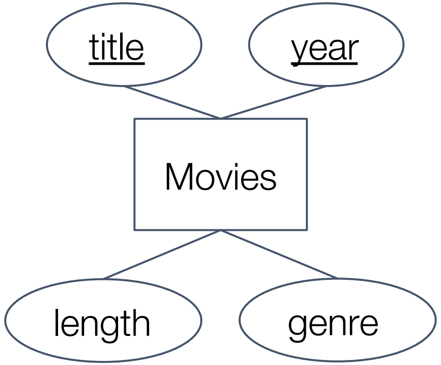
- Represent students and grades they get in courses using the entity sets: Students, Courses, and Enrollment (connects students and courses and represents grades)

# From E/R diagram to relational design

- Once we have an E/R design, we can convert it to a relational database schema
- Common conversions
  - Entity set -> relation with same set of attributes
  - Relationship -> relation whose attributes are from connected entity sets
- Special cases
  - Weak entity sets
  - Isa relationships
  - Sometimes better to combine relations

# Entity set to relation

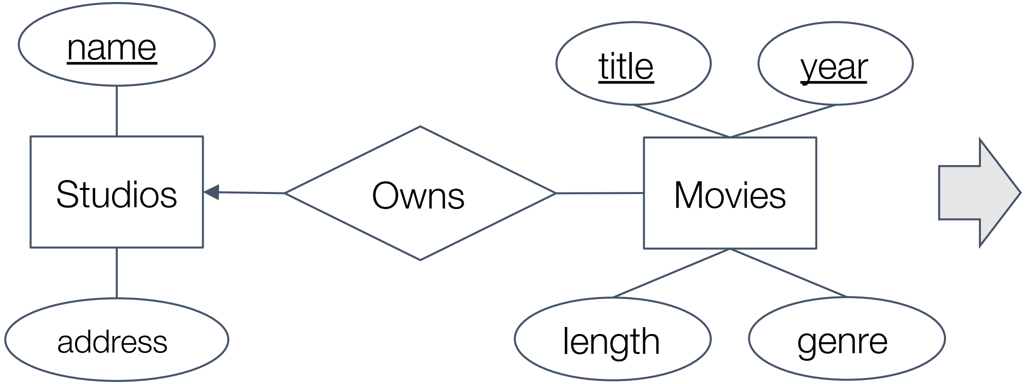
- For each non-weak entity set, create a relation with the same name and attributes



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

# Relationship to relation

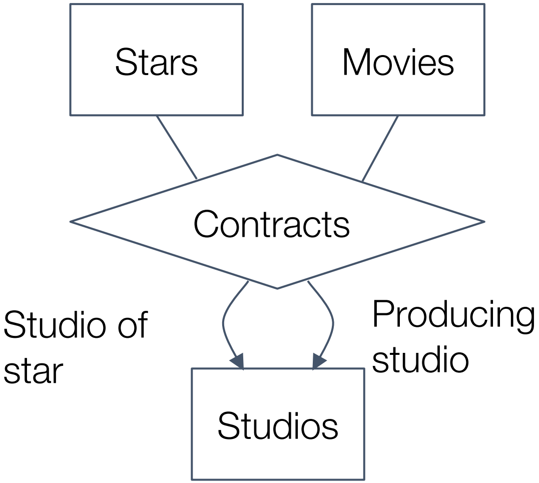
- For each relationship, create a relation with the following attributes
  - For each entity set involved, take its key attribute(s)
  - The relationship's attributes, if there are any
  - Avoid duplicate attribute names using renaming



```
Owns(title, year, studioName)
```

# Relationship to relation

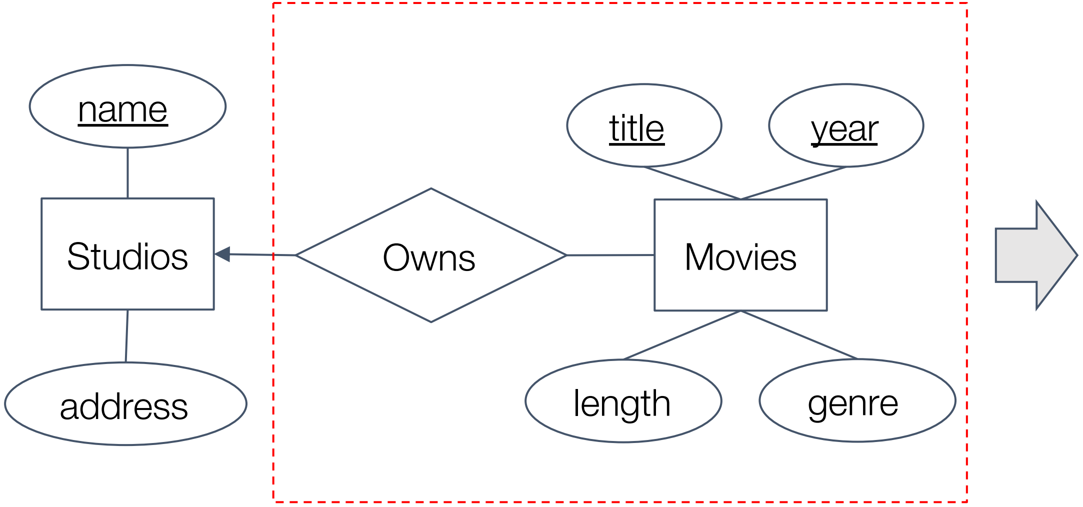
- For each relationship, create a relation with the following attributes
  - For each entity set involved, take its key attribute(s)
  - The relationship’s attributes, if there are any
  - Avoid duplicate attribute names using renaming



```
Contracts(sName, title, year, sStudio, pStudio)
```

# Combining relations

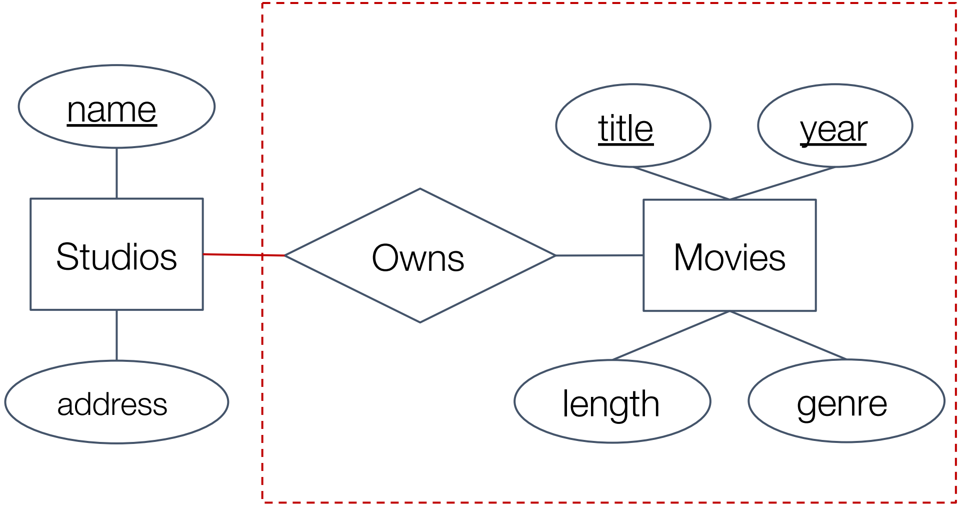
- If E is connected to F through a many-one relationship R, combine E and R
  - Attributes of E and R, and the key attributes of F
- Advantage: querying one relation is faster than querying several relations



```
Owns(title, year, length,
    genre, studioName)
```

# Combining relations

- Why only consider many-one relationships?
  - Otherwise, the combined relation is not good design and may contain anomalies
  - This topic will be covered more formally in the Design Theory lectures



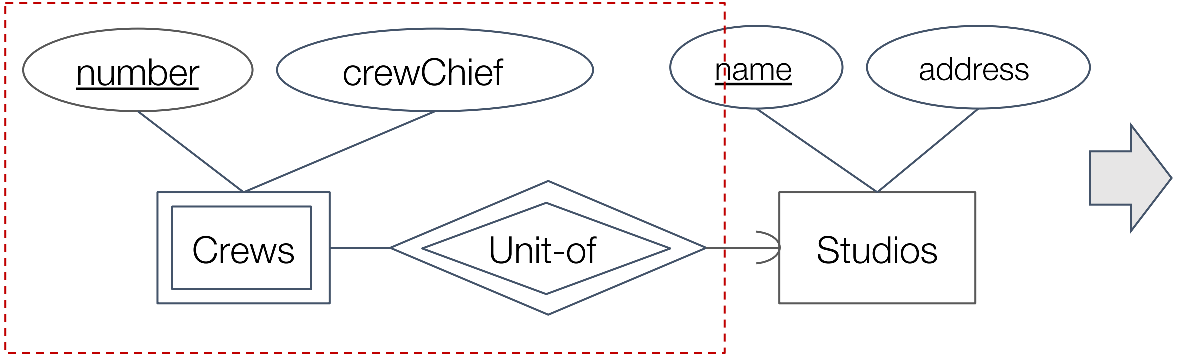
This information is redundant, and updating one tuple may leave the other one incorrect (update anomaly)

Owns

| title | year | length | genre | studioName |
|---|---|---|---|---|
| t1 | y1 | l1 | g1 | n1 |
| t1 | y1 | l1 | g1 | n2 |
| t2 | y2 | l2 | g2 | n2 |

# Weak entity set to relation

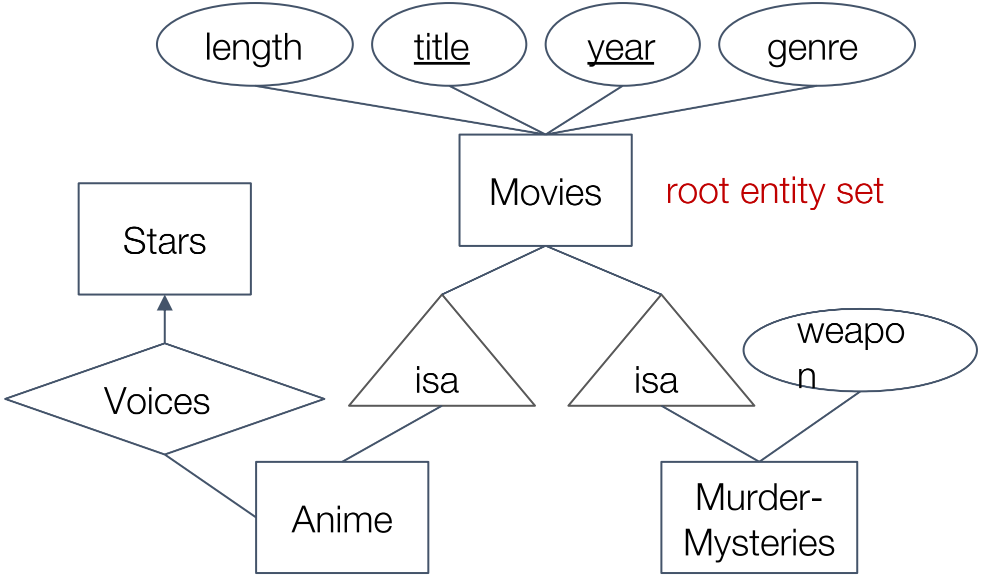
- Construct a relation whose schema has
  - All attributes of the weak entity set
  - All attributes of the supporting relationships
  - The key attributes of the supporting entity sets
- Rename attributes to avoid name conflicts
- No relations for supporting relationships



```
Crews(number, crewChief,
      studioName)
```

# Converting subclass structures to relations

- Isa-hierarchy assumptions
  - There is a root entity set
  - This entity set has a key that identifies every entity represented by the hierarchy
  - A given entity may have components of entity sets of any subtree of the hierarchy



root entity set

# Option 1: E/R style conversion

- For each entity set E, create a relation that includes the key attributes from the root and any attributes of E
- Since an isa relationship connects components of a single entity instead of distinct entities, we do not create a relation for it

```
Movies(title, year, length, genre)
Anime(title, year)
MurderMysteries(title, year, weapon)
Voices(title, year, starName)
```

# Option 2: Object-oriented approach

- For each possible subtrees of the hierarchy, create a corresponding relation
- Rationale: entities are objects that belong to one and only one class

```
Movies(title, year, length, genre)
MoviesAnime(title, year, length, genre)
MoviesMystery(title, year, length, genre, weapon)
MoviesAnimeMystery(title, year, length, genre, weapon)
Voices(title, year, starName)
```

Unless Anime has a separate attribute, the relations with the same attributes can be combined

# Option 3: Use NULL values to combine relations

- Create a single relation with all the attributes in the hierarchy
- A tuple has a NULL value for each attribute not defined for the corresponding entity

```
Movies(title, year, length, genre, weapon)
Voices(title, year, starName)
```

# Comparison of approaches

- NULLs
    - One relation (answering query does not involve multiple relations)
    - However, tuples may be long with many NULL values
- Object oriented
    - One tuple per entity (space efficient)
    - However, there may be an exponential number of relations
- E/R style
    - One relation per entity set in the hierarchy
    - Several tuples per entity, but only the key attributes are repeated

# Exercise #3

- Between object oriented and E/R style, which is better for answering following queries?
  - Q1: which movies are longer than 100 minutes?
  - Q2: which anime movies are longer than 100 minutes and have a weapon?