These lecture notes include some material from Professors Bertossi, Ullman, Widom, Ramakrishnan, Gehrke

Entity Relationship Model

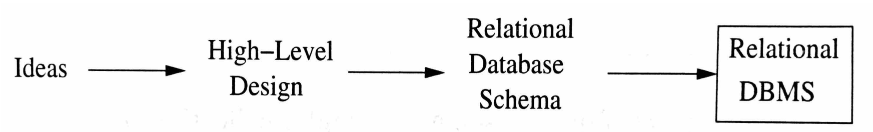
Dr Eugenia Ternovska
Associate Professor

Simon Fraser University

Purpose of ER Model (a Review)

Entity-relationship (ER) model is a notation for describing **schemas in databases**

- A high-level sketch
- Includes some constraints, but not operations
- Designs are pictures called entity-relationship diagrams
- Converted to relational DB schemas



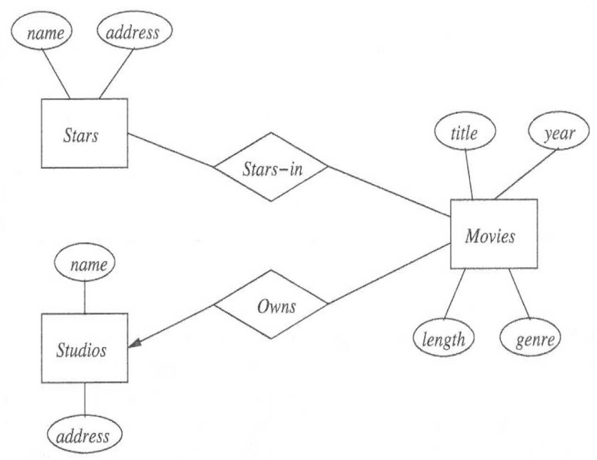
Ideas → High–Level Design → Relational Database Schema → Relational DBMS

Entity Relationship Model

1. The Entity Relationship Model
    - Database design
    - Entity, Entity Set, Attribute, Relationship
2. E/R Design Considerations
    - Constraints: Key, Referential, Degree
    - Relationship Conditions: Multiplicity of Relationships, Multiway Relationships
3. More E/R Concepts
    - Combining Relations, Constraints, Subclasses, Weak Entity Sets
4. Conversion to SQL

In this lecture, we cover Parts 3 and 4

ER Diagrams: Notations (a Review)



Entity set = **rectangle**

Attribute = **oval**, with a line to its entity set.

Relationship = **diamond**,
with lines to each of the two or more entity sets involved.

Arrow indicates that each movie is owned by **at most one** studio (uniqueness constraint)

Weak Entity Sets (1)

An entity set is weak when the key of an entity set is composed of attributes, some or all of which belong to another entity set

Causes:

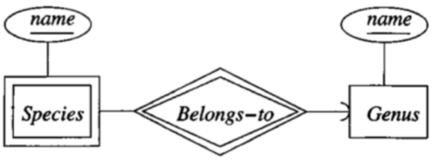
1. Falling into a hierarchy based on classifications unrelated to the isa-hierarchy
2. Connecting entity sets as a way to eliminate a multiway relationship

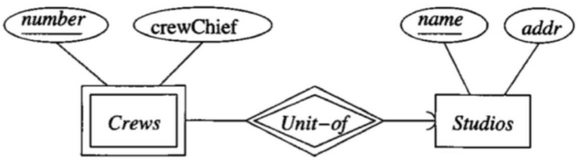
Illustrations are given in the next two slides

Weak Entity Sets (2): Cause 1

E.g., two different studios may have the same designation for crews, so *number* does not identify a crew uniquely (not a key)

E.g., same species names in different genera (plural of genus)





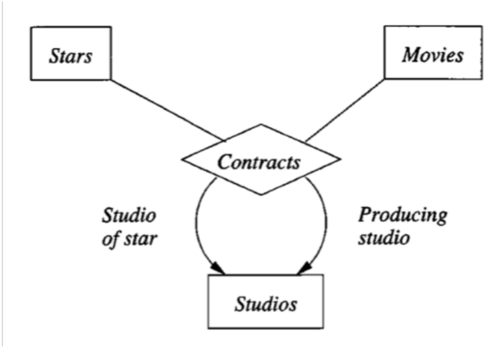
Does not correspond to a isa hierarchy

Notation for weak entity set and supporting relation:

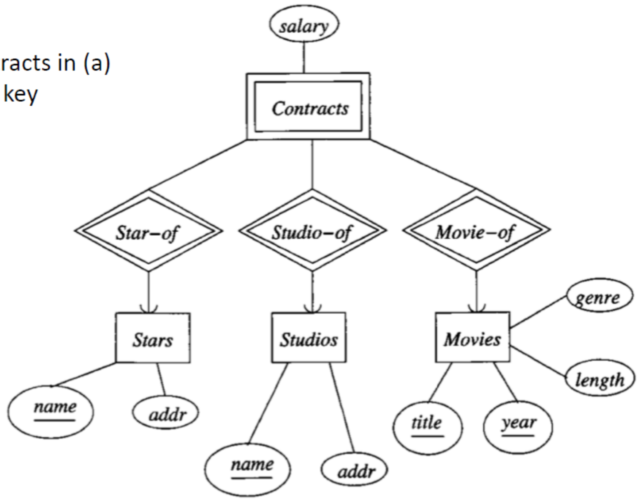
- The double-rectangle indicates a weak entity set
- The double-diamond indicates a many-one relationship that helps provide the key for the weak entity set

Weak Entity Sets (3): Cause 2

Contracts in (b) replaces the ternary relationship Contracts in (a)
Contracts has salary, but it does not contribute to the key



(a)



(b)

Another common source of weak entity sets is when a multi-way relationship is replaced by several binary relationships

Key for the *Contracts* consists of the name of the studio and the star involved, and the title and the year of the movie involved

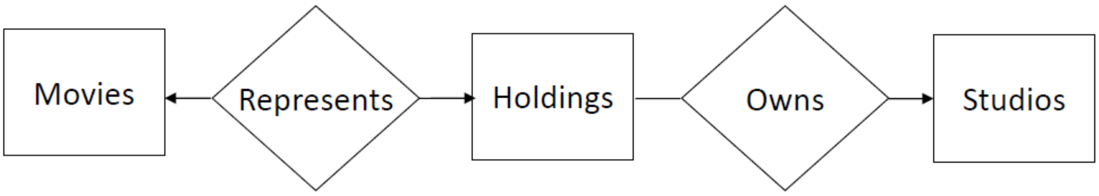
Conversion to SQL

Design Principles:

- Simplicity
- Avoiding redundancy
- Choosing the right elements and relationships
- Combining Relations

Simplicity

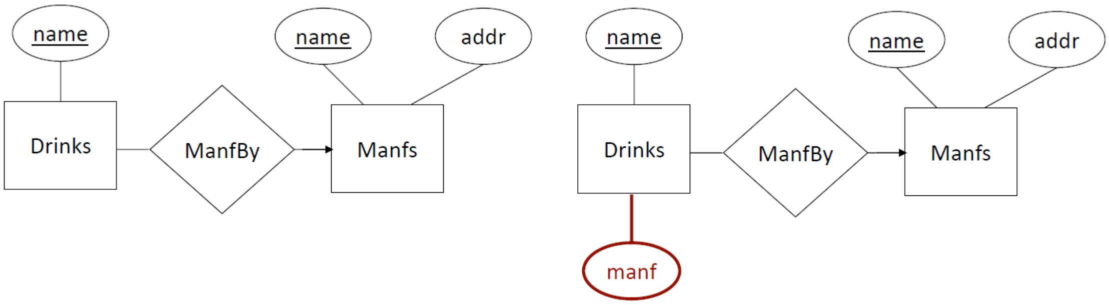
Avoid introducing more elements into your design than those absolutely necessary



A poor design with an unnecessary entity set

Avoiding Redundancy

Wastes space and encourages inconsistency

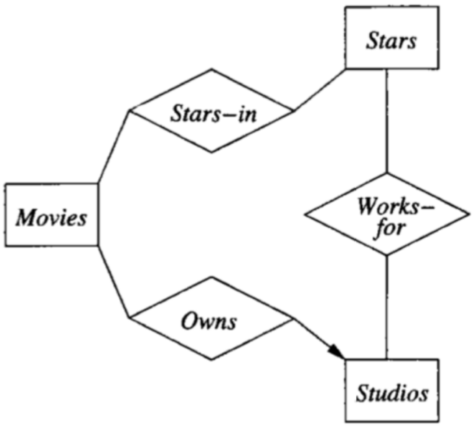


Choosing Relationships

Entity sets can be connected in various ways by relationships

Adding to our design every possible relationship: not a good idea

- Redundancy
- Update anomalies
- Deletion anomalies



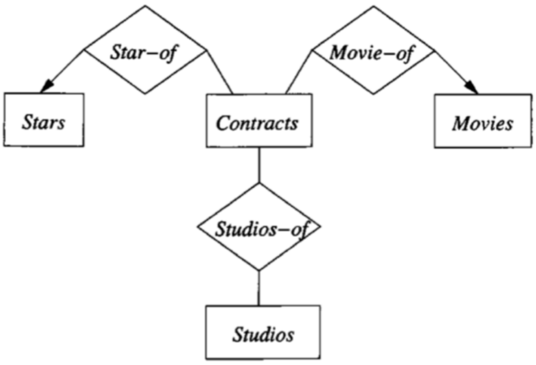
Adding a relationship between Stars and Studios

Choosing Elements

Many choices are between using attributes and using entity set/relationship combinations.

An attribute is simpler to implement than either an entity set or a relationship

Cannot make everything into attributes!



From E/R Diagrams to Relational Designs

To a first approximation, converting an E/R diagram to a relational database schema is straightforward:

- Turn each **entity set** into a relation with the same set of attributes
- Replace each **relationship** with a relation whose attributes are the keys for the connected entity sets

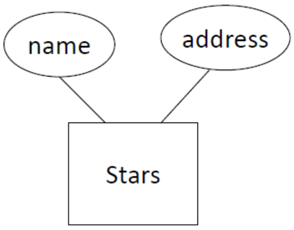
Special Situations

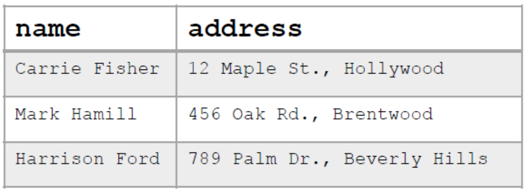
1. **Weak entity sets** cannot be translated straightforwardly to relations
2. **"Isa" relationships** and subclasses require careful treatment
3. Sometimes it is beneficial to **combine two relations**, especially an entity set E and the relation that comes from a many-one relationship R from E to some other entity set F

From Entity Sets to Relations

Consider **entity sets** that are not weak

- For each non-weak entity set, we shall create a relation of the same name and with the same set of attributes.
- No indication of the relationships





name	address
Carrie Fisher	12 Maple St., Hollywood
Mark Hamill	456 Oak Rd., Brentwood
Harrison Ford	789 Palm Dr., Beverly Hills

From E/R Relationships to Relations (1)

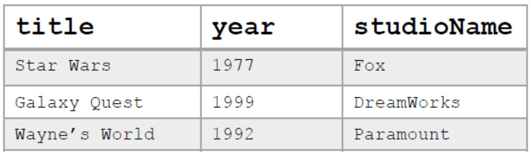
Relationships in the E/R model are also represented by relations

The relation for relationship R

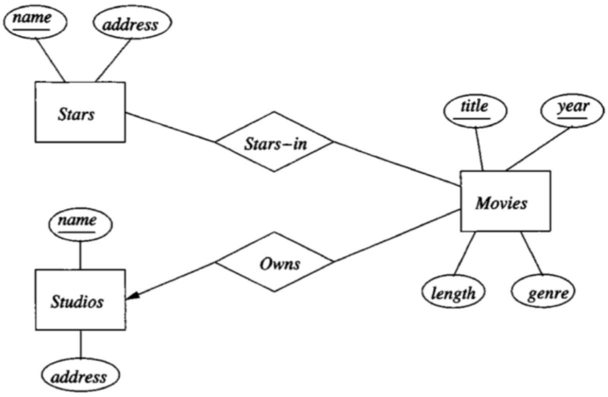
- Attributes:
  - For each entity set involved in relationship R: include its key attribute(s) as part of the schema of the relation for R
  - If the relationship has attributes: also attributes of relation R
  - If an entity set is involved several times in a relationship (in different roles): its key attributes each appear as many times as there are roles

From E/R Relationships to Relations (2)

Owns(title, year, studioName)



title	year	studioName
Star Wars	1977	Fox
Galaxy Quest	1999	DreamWorks
Wayne's World	1992	Paramount

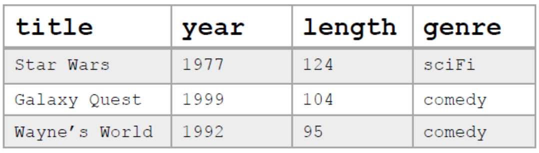


We included primary attributes from both entity sets connected by the relationship to obtain a relation (table) called *Owns*

Combining Relations (Tables) Together

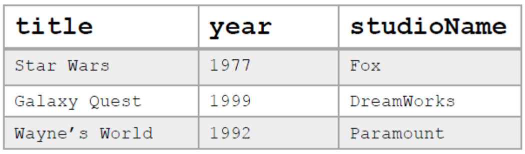
- All attributes of E
- The key attributes of F
- Any attributes belonging to relationship R

Movies:



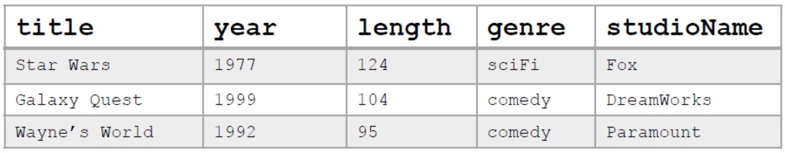
title	year	length	genre
Star Wars	1977	124	sciFi
Galaxy Quest	1999	104	comedy
Wayne's World	1992	95	comedy

Owns:



title	year	studioName
Star Wars	1977	Fox
Galaxy Quest	1999	DreamWorks
Wayne's World	1992	Paramount

Combining relation Movies
with relation Owns



title	year	length	genre	studioName
Star Wars	1977	124	sciFi	Fox
Galaxy Quest	1999	104	comedy	DreamWorks
Wayne's World	1992	95	comedy	Paramount

Handling Weak Entity Sets

The relation for weak entity set W

- Include not only the attributes of W but also the key attributes of the supporting entity sets
- The supporting entity sets are those reached by supporting (double-diamond) relationships **from** W

The relation for any relationship in which the weak entity set W appears

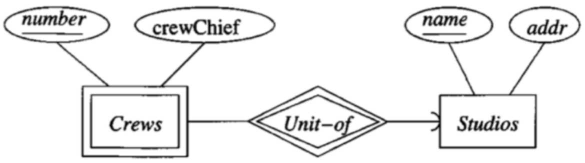
- Must use as a key for W all of its key attributes, including those of other entity sets that contribute to W's key

A supporting relationship R, from the W **to** a supporting entity set, need not be converted to a relation at all — its attributes will be in the relation for W

Handling Weak Entity Sets Example

```
Studios(name, addr)
Crews(number, studioName, crewChief)
```

~~Unit-of(number, studioName, name)~~



The crews example of a weak entity set

Relation *Studios* comes straightforwardly from the entity set
Relation *Crews* comes from the weak entity set
Attribute studioName in *Crews* corresponds to the attribute name in entity set *Studios*
Since *Unit-of* is a many-one rel., studioName must be the same as name
But then the set of attribute of *Unit-of* is a subset of the set of attributes of *Crews*, so *Unit-of* is not needed

Converting Subclass Structures to Relations

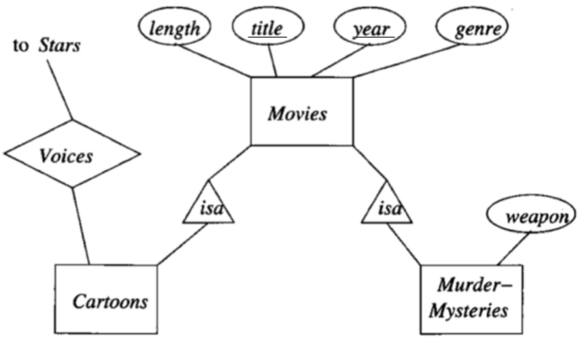
Conversion Strategies

1. Follow the E/R viewpoint
   - Create a relation that includes the key attributes from the root and any attributes belonging to entity set
2. Treat entities as objects belonging to a single class
   Example:
   - Movies alone.
   - Movies and Cartoons only.
   - Movies and Murder-Mysteries only.
   - All three entity sets.
3. Use Null Values
   Example:
   - Movie(title, year, length, genre, weapon)
     Movies that are not murder mysteries: NULL in the weapon component of their tuples

Converting Subclass Structures to Relations Example (1)

- E/R View

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



The movie hierarchy

Create a relation for each entity set as usual. The two keys for Movies are present in all these relations. Tuples for movies that are both cartoons and murder mysteries are present in all three rel.-s

Create a relation for relationship Voices as usual

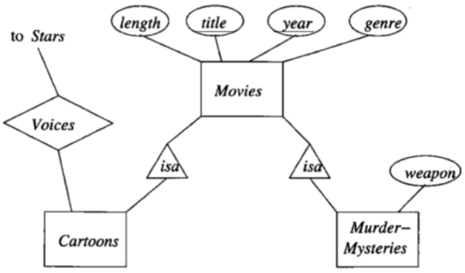
No relation is introduced for “isa” relationship

Converting Subclass Structures to Relations Example (2)

An alternative approach:

- Object-Oriented Approach

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

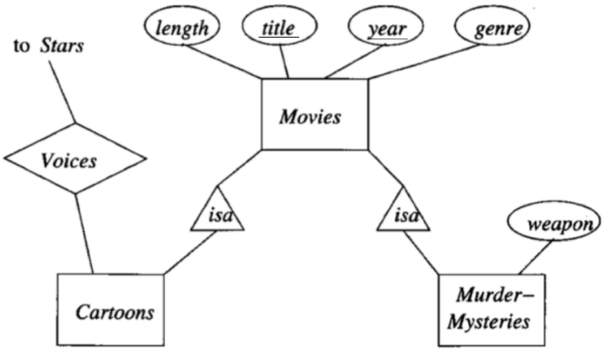


Converting Subclass Structures to Relations Example (3)

Yet another approach:

- Using Null Values

```
Movie(title, year, length, genre, weapon)
```



Create a single relation, and add NULLs where the attribute is not relevant,
e.g., weapon is only relevant in murder mysteries, NULL otherwise

Comparison of Approaches

We would like to reduce the cost of queries

- It's expensive to answer queries involving several relations
- We would like not to use too many relations
- We would like to minimize space and avoid repeating information

There are many issues to balance!

Summary of Conceptual Design

- Conceptual design follows requirements analysis,
  - Yields a high-level description of data to be stored
- E/R model is popular for conceptual design
  - Constructs are expressive, close to the way people think about their applications.
- Basic constructs: entities, relationships, and attributes (of entities and relationships).
- Some additional constructs: weak entities, Is-A hierarchies, and aggregation.
- Note: There are many variations on ER model.

Summary of ER (1)

- Several kinds of integrity constraints can be expressed in the ER model:
  - key constraints,
  - participation constraints, and
  - overlap/covering constraints for Is-A hierarchies.
- Some foreign key constraints are also implicit in the definition of a relationship set.
- Some constraints (notably, functional dependencies) cannot be expressed in the ER model.
- Constraints play an important role in determining the best database design for an enterprise.

Summary of ER (2)

- ER design is **subjective**. There are often many ways to model a given scenario. Analyzing alternatives can be tricky, especially for a large enterprise.

  Common choices include:
  - Entity vs. attribute, entity vs. relationship, binary or N-nary relationships, whether or not to use Is-A hierarchies, and whether or not to use aggregation.
- Ensuring good database design: resulting relational schema should be analyzed and refined further.
- Functional Dependencies information and normalization techniques are **especially useful**.

Acknowledgements

[1] Database Systems: The Complete Book, 2nd EditionHector Garcia-Molina, Jeffrey D. Ullman, Jennifer WidomPrentice Hall, 2009

[2] Database System Concepts, Seventh EditionAvi Silberschatz, Henry F. Korth, S. SudarshanMcGraw-Hill, March 2019www.db-book.com

Additional references and resources used in preparation of this course are listed on the course webpage or mentioned in slides.