

CSC343 Worksheet 14 Solution

July 13, 2020



1.

Correct Solution:



Notes:

- E/R Model
 - Means **Entity Relationship Model**
 - Entity Relationship Model(ER Modeling) is a graphical approach to database design.
 - Is comparable to class diagram in UML
 - Uses three principle element types:
 1. Entity sets
 - * Is an abstract object of some sort (i.e. entity)
 - * Is not used to represent class
 - * Is represented by rectangles



2. Attributes

- * Are properties of entities in a set (i.e. column name)
- * Each has its own primitive data types (e.g. String, integers, Reals)
- * Is represented by ovals



3. Relationships

- * Are connections among two or more entity sets (e.g. intermediary Relations like Stars In)
- * Is represented by diamond

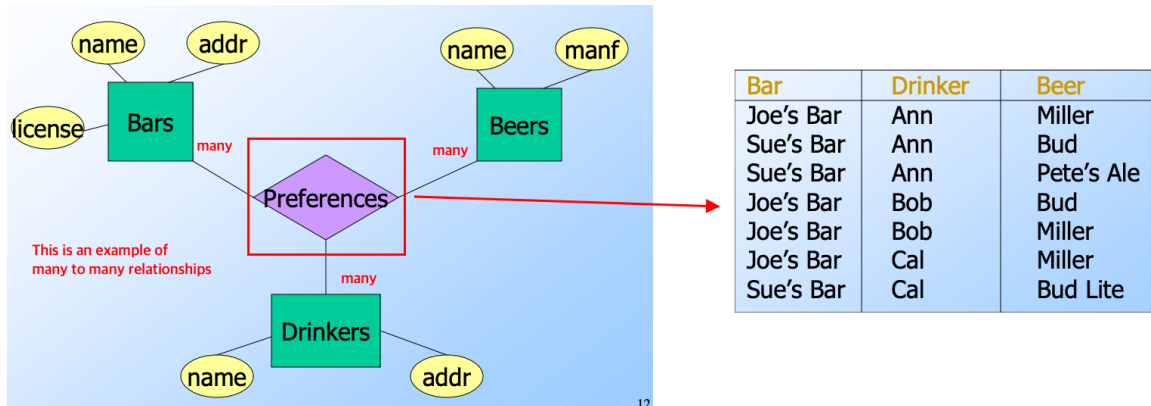


Example:



- Multiway Relationships
 - Connects more than two relationship sets
 - Enables to represent relationships that otherwise is difficult in binary relationship
 - Arrow → 'one'
 - No arrow → 'many'

Example:



Example 2:



Figure 4.4: A three-way relationship

- Roles in Relationships
 - Is the label of edges between the entity set and relationship
 - Are used to clarify the semantics of relationship

Example:



Figure 4.5: A relationship with roles

Example 2:

Figure 4.6: A four-way relationship

- Attributes on Relationships

- can be thought as a property of tuples in the relationship set (i.e. String, Integer, Float, Boolean)

Example:

- Can be removed by creating an entity set with the attribute

Example:

- Converting Multiway Relationships to Binary

Example:



- Subclasses in the E/R Model
 - Has its own special attributes and/or relationships
 - All 'isa' relationship is one to one
 - Is represented by triangle with label 'isa' followed by entity set

Example:



2. a)

Correct Solution:



Correct Solution:



c)

Correct Solution:



Correct Solution:





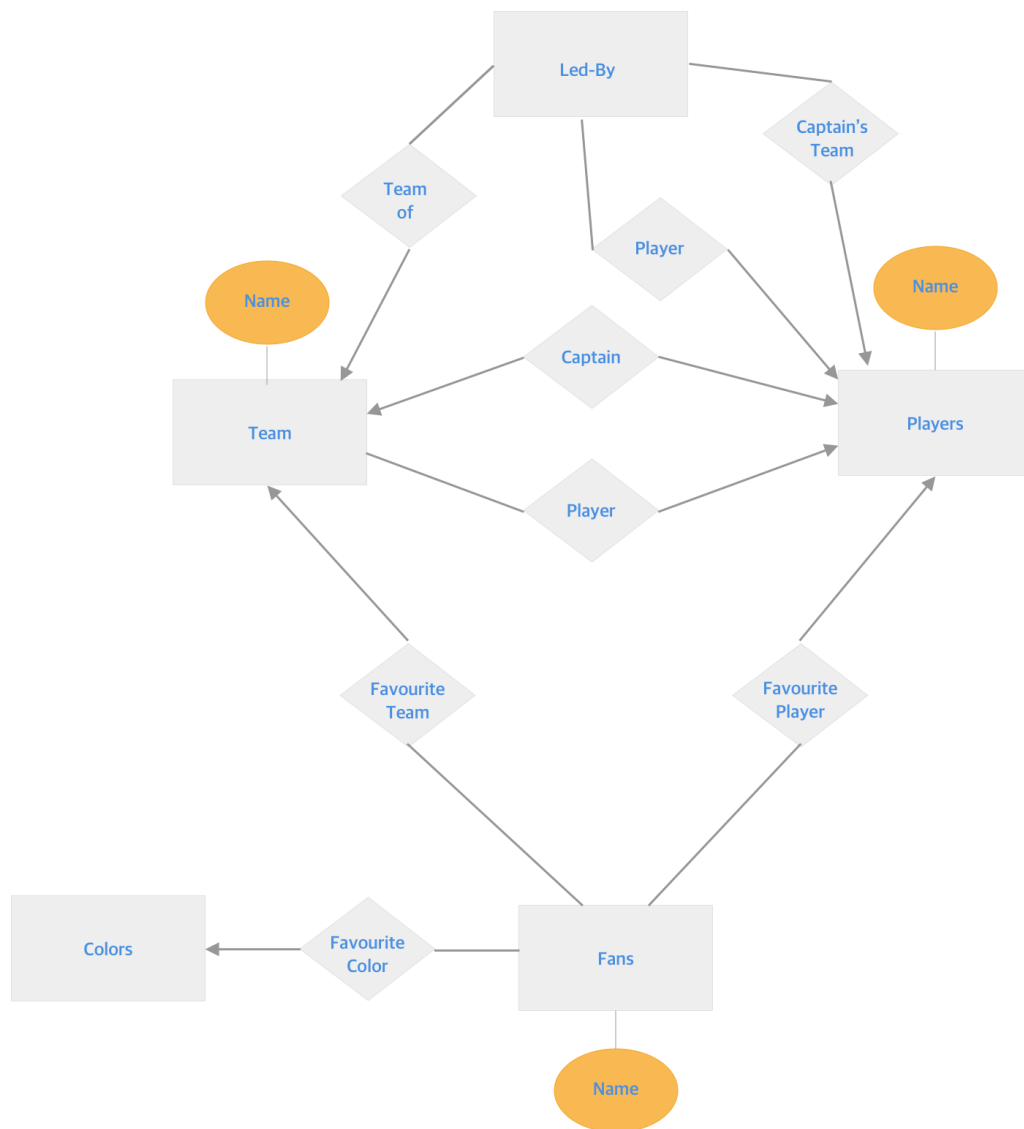
3.

Correct Solution





4. a)



b)

c) They are the same. (I need more work on providing reason).

Notes:

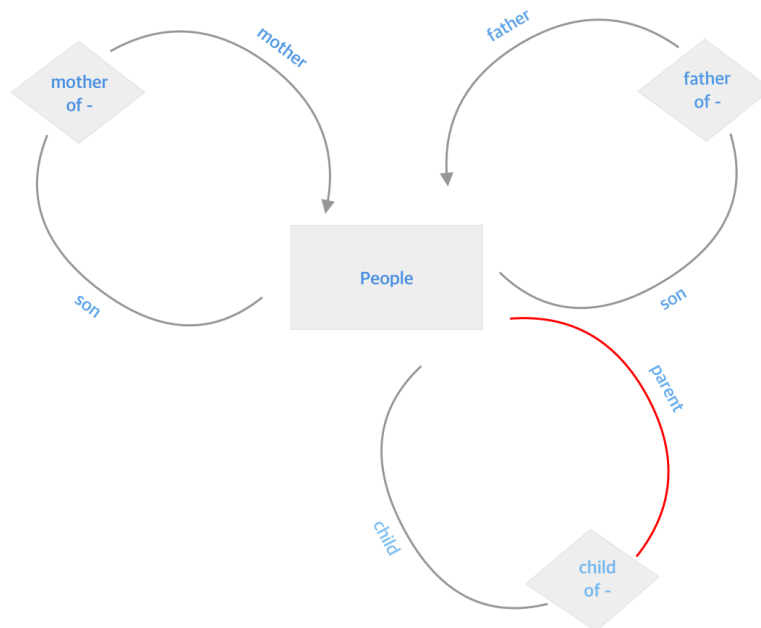
- I should ask professor about this :'(



5.



6.

Correct Solution:



7.

Notes:

- I feel the need to clarify with professor if two parent subclasses can exist
- I feel the need to ask professor whether this design is valid



8. a)



b)

Notes:

- I need to clarify with professor on one-to-many relationship.

Is it correct that the 'one' side of 'one-to-many' relationship represent foreign key in terms of SQL?

But how about the many side? What does it mean it to be many? so for example, ('Josh', 'Neville the father', 'Mary the mother'), ('Jay', 'Neville the father', 'Mary the mother'), is this one to many relationship?

In tabular terms / example what does one-to-many relationship represent in this context?



9.



10.

Correct Solution:



11. Simplicity count is violated. There is more than necessary number of entity sets and attributes for address and accounts.



Notes:

- Design Principles

1. Faithfulness

- means design should make sense and meet its specification
- e.g. Adding attribute *number-of-cylinders* to *Stars* → NONO

2. Avoiding Redundancy

- *Redundancy* means saying the same thing in two (or more) different ways

Example (The good example):



Example (The bad example):

3. Simplicity Counts

- Avoid adding more more elements than necessary

4. Choosing the Right Relationships

- Don't add relationships more than necessary

5. Picking the Right Kind of Element

- Many of the choices are between using attributes and using entity set / relationship combinations

12. They should be combined when each studio has unique president

13. Solution:

14. a) **Solution:**



b) **Solution:**



c) **Solution:**



Notes:

- Multivalued attributes is denoted by the following ^[1]



References:

- 1) OpenTextBC, The Entity Relationship Model, link

15. a) **Solution:**



i)



ii)

Notes:

- Keys in the E/R Model
 - *key* is an attribute or a group of attributes whose values can be used to uniquely identify an individual entity in an entity set ^[1]
 - Keys are represented using ‘underline’ under each attribute



- **Referential Integrity**

- Means value appearing in one context must also appear in another
- Functions like Foreign Key in SQL
- Is represented by a rounded arrow



References:

1) OpenTextBC, The Entity Relationship Model, link

b) **Solution:**



i)



ii)

c) **Solution:**



i)



ii)

16. **Solution:**



Correct Solution:



- Yes. grade is a part of a key for enrollent.

Notes:

- Weak Entity Sets
 - Is an entity set of which some or all of attributes belong to another entity set
 - The usual reason is that there is no global authority capable of creating unique ID's.
 - Is denoted by the following symbol:

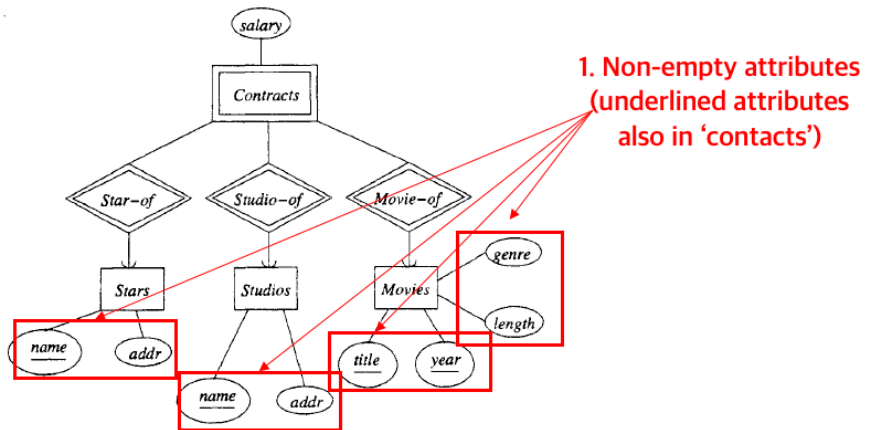


- Depends on a dominant entity, and it cannot exist without a strong entity. ^[1]
- Has attributes in both weak entity sets and the entity sets

Example:



- * Schemas for the strong entity sets
 - student (username)
 - assignment (shortname, due_date, url)
- * Schemas for the weak entity set
 - submission(username, shortname, version, submit_date, data)
- Requirements for Entity Sets
 - E is a weak entity if it consists of
 1. Zero or more of its own attributes, and
 2. One or more many-one relationships to other (supporting) entity sets. ^[2]



References:

- 1) StackOverflow, Example of a strong and weak entity types, [link](#)
- 2) Stanford, Entity-Relationship Model, [link](#)
- 3) Caltech, Converting E-R Diagrams to Relational Model, [link](#)



17.

18. a) **Solution:**



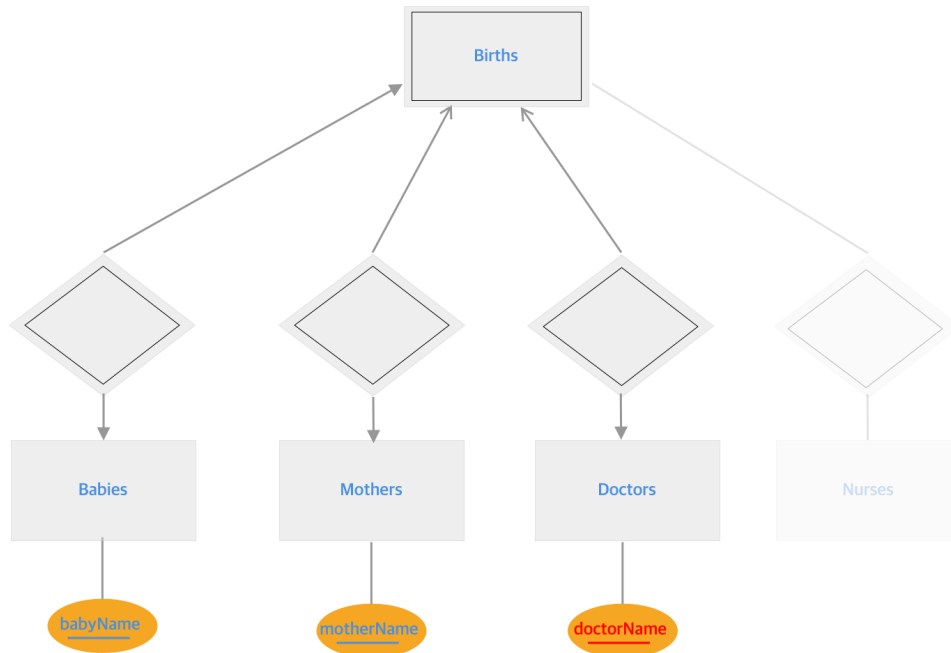
Correct Solution:



b)

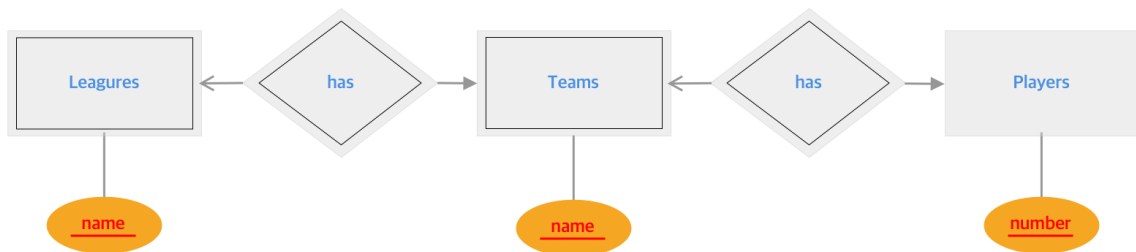
Correct Solution:

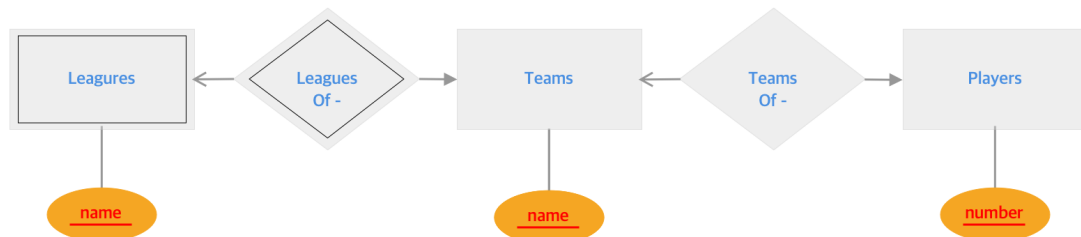
c)

Correct Solution:19. a) **Solution:**



b) **Solution:**



Correct Solution:

20. • Customers

Customer(SSNo, number, addr, phone)

• Flights

Flights(number, day, aircraft)

• Bookings

Bookings(SSNo, number, row, seat)

• toCust

toCust (SSNo) Can be eliminated since subset of Customers

• ToFit

toFit (number) Can be eliminated since subset of Flights

Notes:

- From entity sets to Relations
 - There are two basic steps
 1. Turn each entity set into a relation with the same set of attributes, and
 2. Replace a relationship by a relation whose attributes are the keys for the connected entity sets

Steps:

- a. Take key attributes in E and put to the relation of R
- b. Take attributes of R, and put into its relation
- c. **Notes:** Don't try to combine entity E with relationship R
 - * Doing so can lead to redundancy → requires normalization

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

Redundancies

- With more difficult steps
 1. Weak entity sets cannot be translated straightforwardly to relation
 2. 'Isa' relationship and subclasses require careful treatment
 3. Sometimes, we do well to combine two relations of different types (many-to-one → many-to-one) the connected entity sets

Example:

1. Turning each entity into relation



2. Turning E/R Relationships to Relations

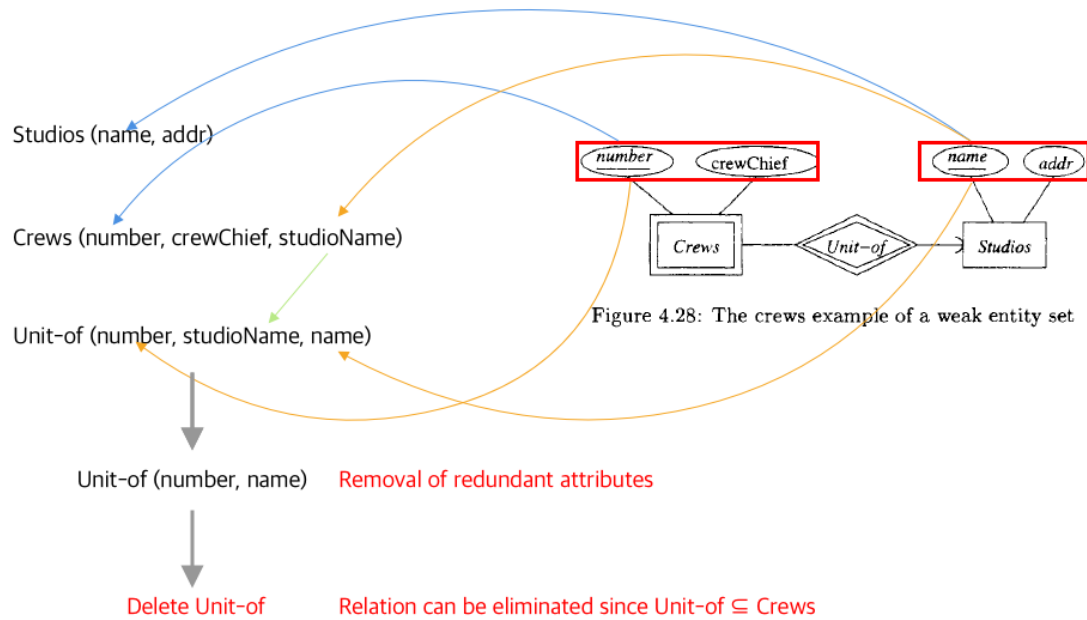
* The schema of the relation ‘Owns’ is:

Owns(title, year, studioName)



- Handling weak entity sets
 - Three things must be done differently
 1. The relation for the weak entity set W must include the its own attributes, but also the attributes of the supporting entity sets
 2. W should use key of it's attributes in addition to other entity sets that contribute to W's key
 3. Do not convert the supporting sets to relation

Example:



21. a)

- b)
- Bookings (number, day, row, seat)
 - Flights (day, number, aircraft)

No. It's not the same as the E/R in exercise 5.2.

22. • SisterOf(shipName, sisterName)

• Ships(name, yearLaunched)

23. a) • Stars

Stars(name, addr)

• Studios

Studios(name, addr)

• Movies

Studios(name, addr)

• Contracts

Contracts(title, year, length, genre)

• Movies

Movies(title, year, length, genre)

• StarOf

StarOf(name) Can be eliminated. Is a subclass of Stars

• StudioOf

StudioOf(name) Can be eliminated. Is a subclass of Studio

• StudioOf

StudioOf(title, year) Can be eliminated. Is a subclass of Movies

b) • Enrollment

Enrollment(grade, name)

• Student

Student(grade)

• Course

Course(name)

- studentOf
studentOf(grade) Can be eliminated. Is a subset of Student
- courseOf
courseOf(name) Can be eliminated. Is a subset of Course
- c) • Courses
Courses(number)
- Departments
Departments(name)
- Offers
Offers(number, name)
- d) • Players
Players(number)
- TeamsOf
TeamsOf(name, number)
- Teams
Teams(name, number)
- LeaguesOf
LeaguesOf(leagueName, teamsName) Can be eliminated since it's the subset of League
- Leagues
Leagues(name, teamsName)

Notes:

- I should clarify with professor on this question. Something doesn't feel right.
24. a) • Depts
Depts(name, chair)

- GivenBy

GivenBy(deptName, CoursesNumber) Can be eliminated since it's the subclass of Courses

- Courses

Courses(number, deptName, room)

- Lab Courses

LabCourses(number, deptName, computerAllocation)

Notes:

- Coverting Subclass Structure to Relations (Three different approaches)
 1. *E/R Style Conversion:*
 2. *An Object-Oriented Approach*
 3. *Use Null Values*
- E/R-Style Conversion
 - a) Create a relation that includes key attributes from it's parent
 - b) From a) include all attributes from its own

Example:

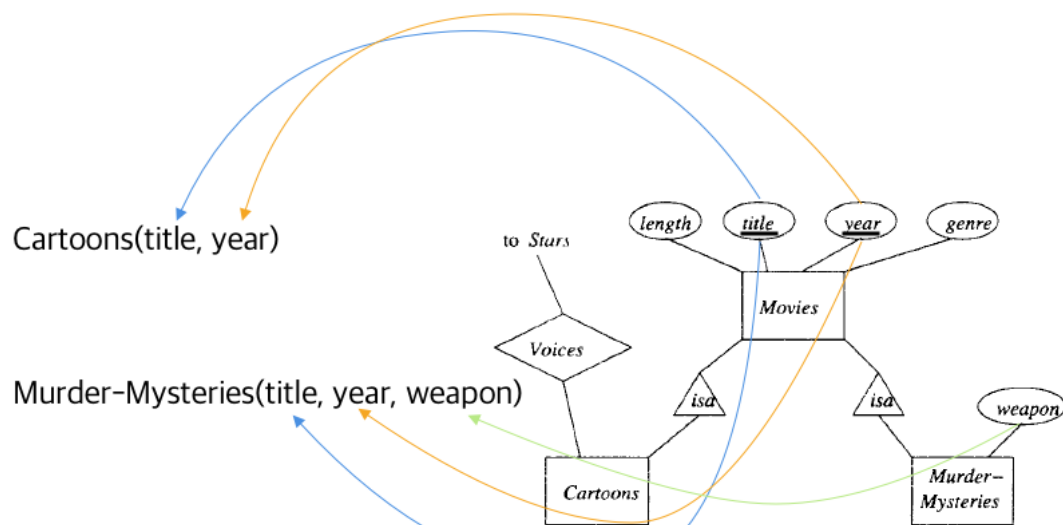
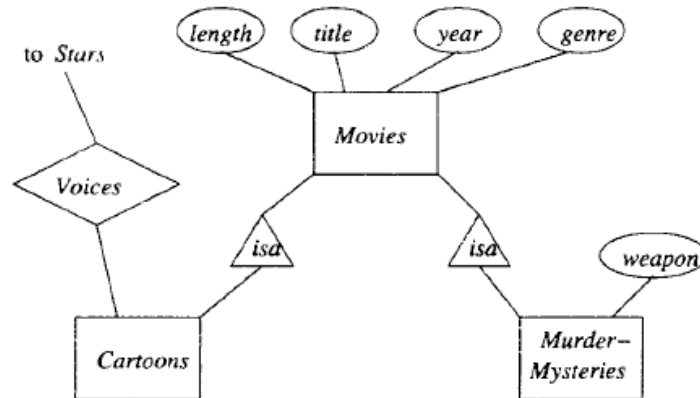


Figure 4.31: The movie hierarchy

- An Object-Oriented Approach

- A strategy is to enumerate all the possible subtrees of the hierarchy

Example:



1. *Movies* alone

Movies(title, year, length, genre)

2. *Movies* and *Cartoons* only

MoviesC(title, year, length, genre)

3. *Movies* and *Muder-Mysteries* only

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

4. All three entity-sets

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

5. Voice for MoviesCMM and MoviesMM

Voices(title, year, starName)

- Use Null values to Combine Relations
 - Has all attributes belonging to any entity set in the hierarchy
 - Use 'NULL' in each attribute that is not defined for that entity

Example:

Movies

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

↑
NULL

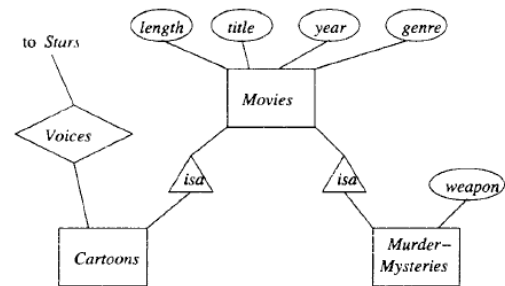
Cartoons

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

↑
NULL

Murder-Mysteries

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



b) • Depts

Depts(name, chair)

• GivenBy

GivenBy(deptName, courseNumber) Can be eliminated since it's the subclass of Courses

• Courses

Courses(number, deptName, room)

• Lab Courses

LabCourses(number, deptName, room, computerAllocation)

c) • Depts

Depts(name, chair)

• GivenBy

GivenBy(deptName, courseNumber) Can be eliminated since it's the subclass of Courses

• Courses

Courses(number, deptName, room, computerAllocation (NULL))

• Lab Courses

Courses(number, deptName, room, computerAllocation)

25. a) • Person

Person (name, address)

- Child

Child (name, address)

- Father

Father(name, address)

- Mother

Mother (name, address)

- ChildOf

ChildOf(nameOfPerson, addressOfPerson, nameOfChild, addressOfChild)

- FatherOf

ChildOf(nameOfChild, addressOfChild, nameOfFather, addressOfFather)

- Married

Married(nameOfHusband, nameOfHusband, nameOfWife, addressOfWife)

- MotherOf

MotherOf(nameOfChild, nameOfChild, nameOfMother, addressOfMother)

- b)
 - Person

Person (name, address)

- Child

PersonC (name, address)

- Father

PersonF(name, address)

- Mother

PersonM (name, address)

- Person-Child-Father

PersonCF (name, address)

- Person-Child-Mother

PersonCM (name, address)

- Person-Father-Mother

PersonFM (name, address)

- Person-Child-Father-Mother

PersonCFM (name, address)

- ChildOf (For PersonChild, PersonChildMother, PersonChildFather, PersonChildFatherMother)

ChildOf(nameOfPerson, addressOfPerson, nameOfChild, addressOfChild)

- FatherOf (For PersonFather, PersonChild, PersonChildFather, PersonChildFatherMother)

ChildOf(nameOfChild, addressOfChild, nameOfFather, addressOfFather)

- Married (For PersonFather, PersonMother, PersonFatherMother, PersonChildFatherMother)

Married(nameOfHusband, nameOfHusband, nameOfWife, addressOfWife)

- MotherOf (For PersonMother, PersonChild, PersonChildMother, PersonChildFatherMother)

MotherOf(nameOfChild, nameOfChild, nameOfMother, addressOfMother)

- c)
 - Person

Person (name, address)

- Child

Child (name, address)

- Father

Father(name, address)

- Mother

Mother (name, address)

- ChildOf

ChildOf(nameOfPerson, addressOfPerson, nameOfChild, addressOfChild)

- FatherOf

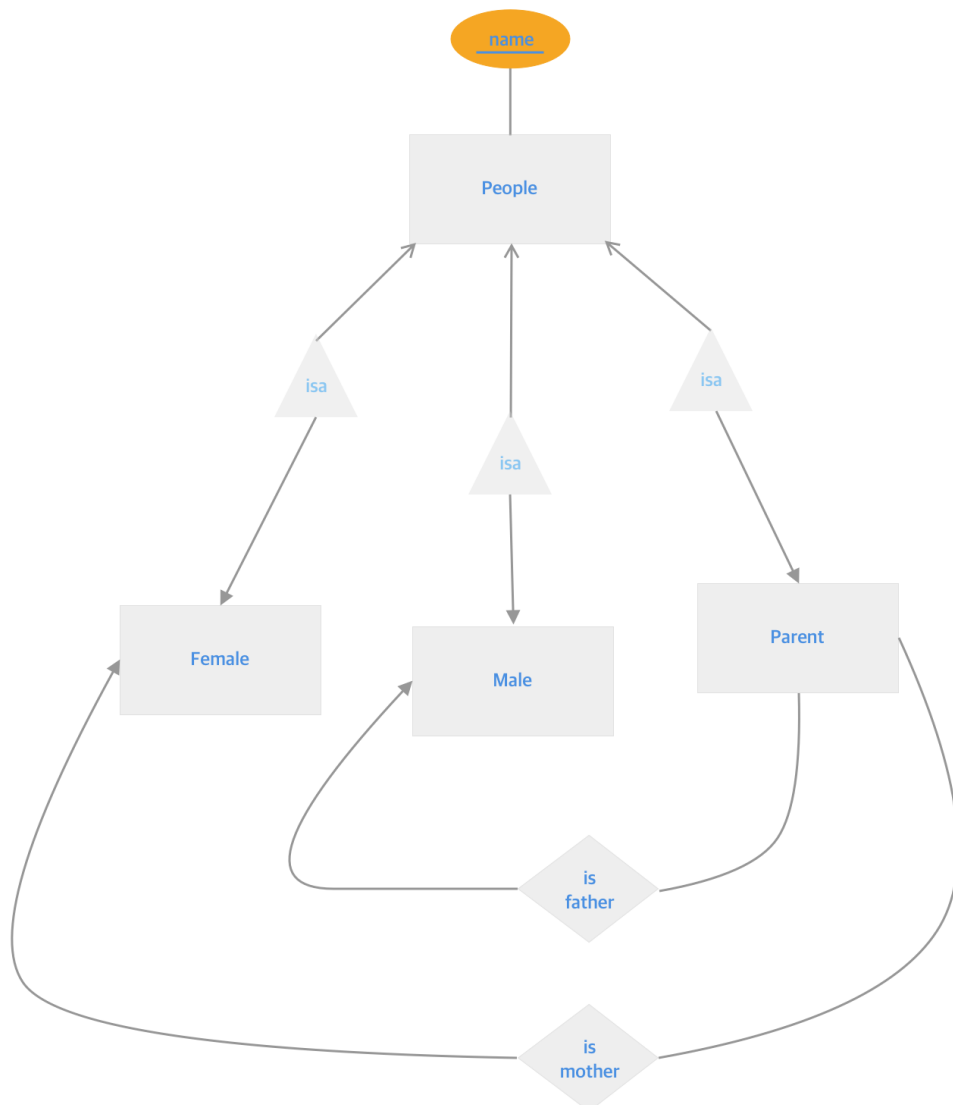
ChildOf(nameOfChild, addressOfChild, nameOfFather, addressOfFather)

- Married

Married(nameOfHusband, nameOfHusband, nameOfWife, addressOfWife)

- MotherOf

MotherOf(nameOfChild, nameOfChild, nameOfMother, addressOfMother)



a) • People

People(name)

• Female

Female(name)

• Male

Male(name)

• Parent

Parent(name)

• isFather

isFather(maleName, femaleName)

• isMother

isMother(femaleName, maleName)

b) • People

People(name)

• Female

PeopleF(name)

• Male

PeopleM(name)

• Parent

PeopleP(name)

• People-Female-Male

PeopleFM(name)

• People-Female-Parent

PeopleFP(name)

- People-Male-Parent

PeopleMP(name)

- People-Female-Male-Parent

PeopleFMP(name)

- isFather (For PeopleMale, PeopleParent, PeopleMaleParent, PeopleFemaleMaleParent)

isFather(maleName, femaleName)

- isMother (For PeopleFemale, PeopleParent, PeopleFemaleParent, PeopleFemaleMaleParent)

isMother(femaleName, maleName)