# L11: ER modeling 4

CS3200 Database design (sp18 s2)

https://course.ccs.neu.edu/cs3200sp18s2/

2/15/2018

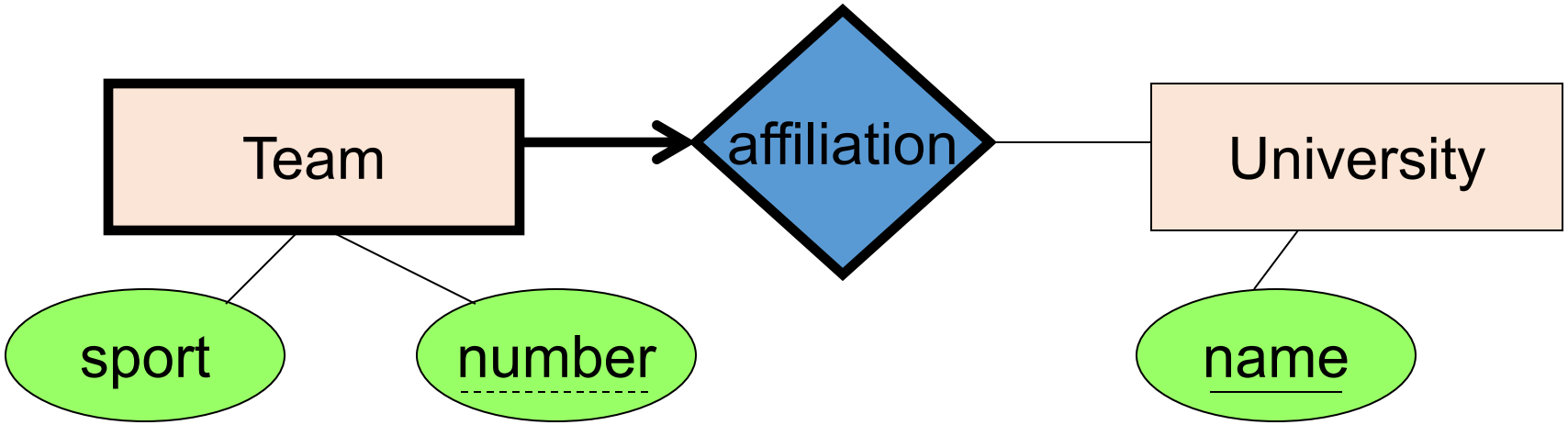
# Announcements!

- Keep coming with your name plates
- Mid-course feedback for instructor
  - Exam: Cheat sheet, Honor code, Database setup, what went wrong? how to avoid in future?
  - Class participation, interactive questions, name tags
  - "Where do I find?"
    - Required class content: Class (slides and spoken), HWs (including FMs), and Piazza (extended classroom)

- Outline
  - Continue with ER modeling: more interactive exercise
  - Next week: Normalization

# Weak Entity Sets

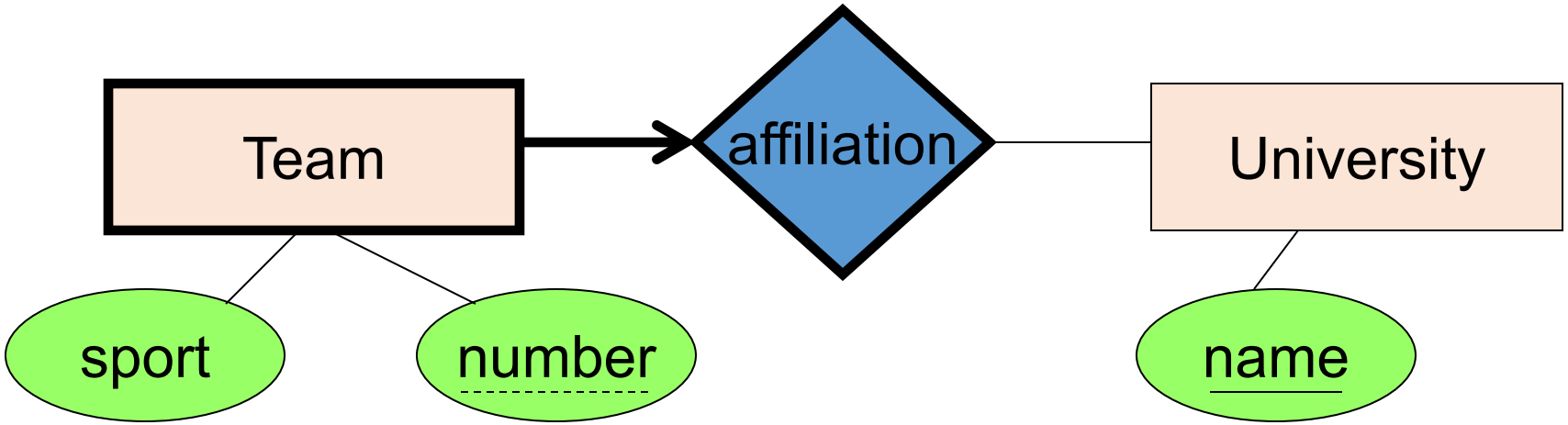
Entity sets are *weak* when their key comes from other classes to which they are related.




"Football team" v. "***The Northeastern*** Football team" *(E.g., BU has a football team too, sort of)*

# Weak Entity Sets

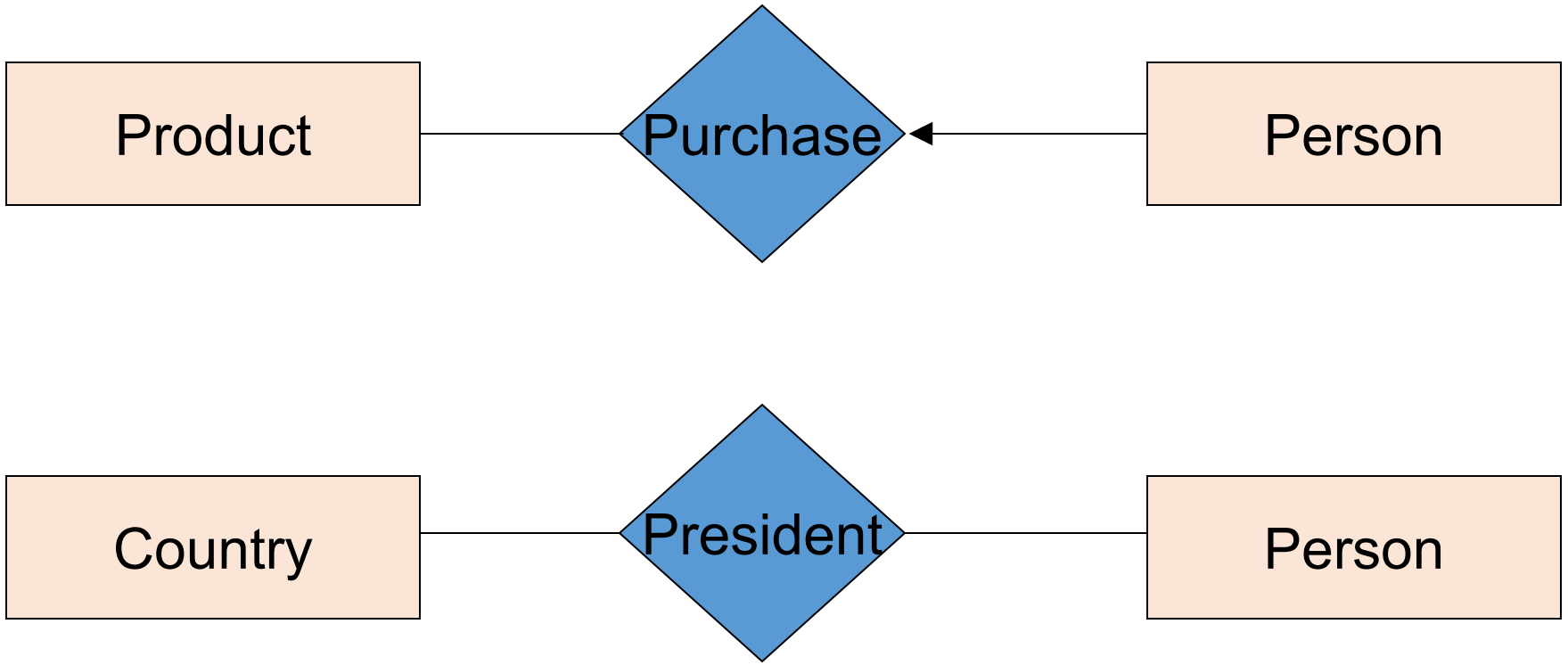
Entity sets are *weak* when their key comes from other classes to which they are related.




- number is a *partial key*. (denote with dashed underline).
- University is called the *identifying owner*.
- Participation in affiliation must be total. Why?

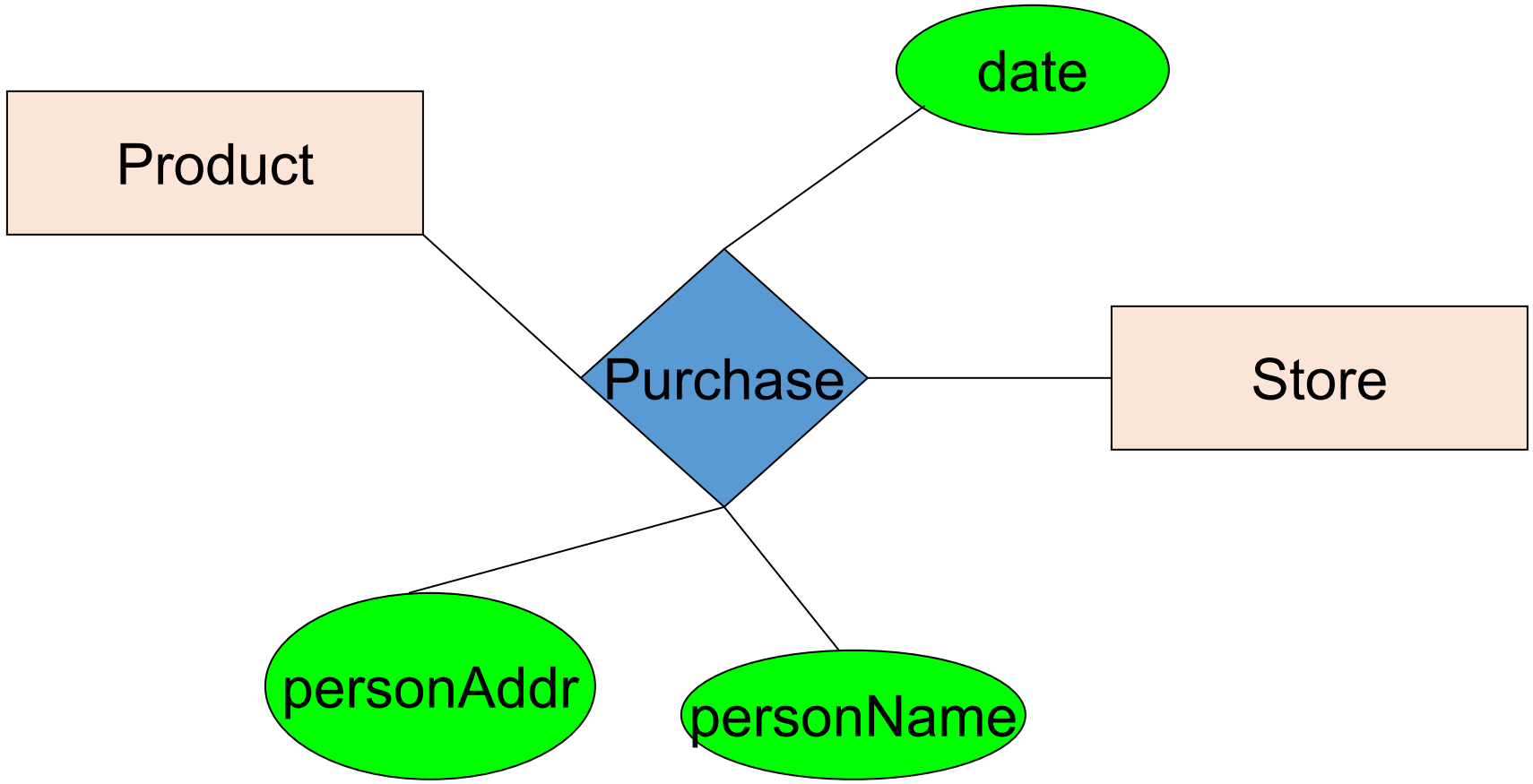
# Practice

# Design Principles: What's Wrong?

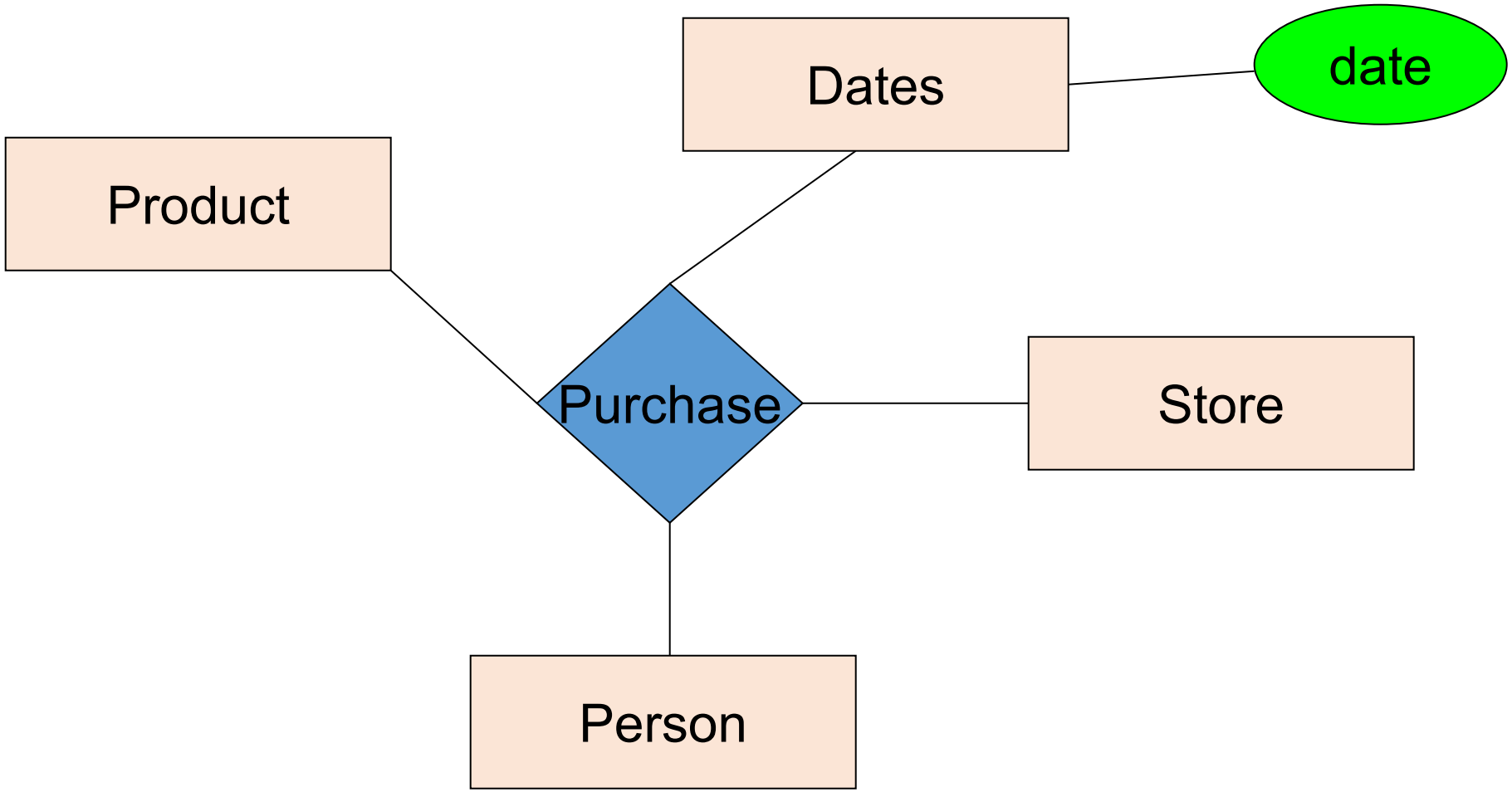



Moral: be faithful to the specifications of the app!

# Design Principles: What's Wrong?

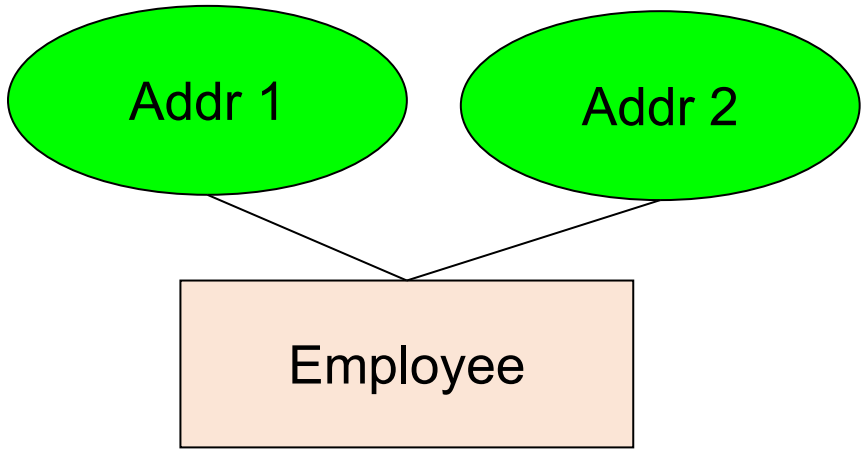


# Design Principles: What's Wrong?



# Examples: Entity vs. Attribute

Should address (A) be an attribute?

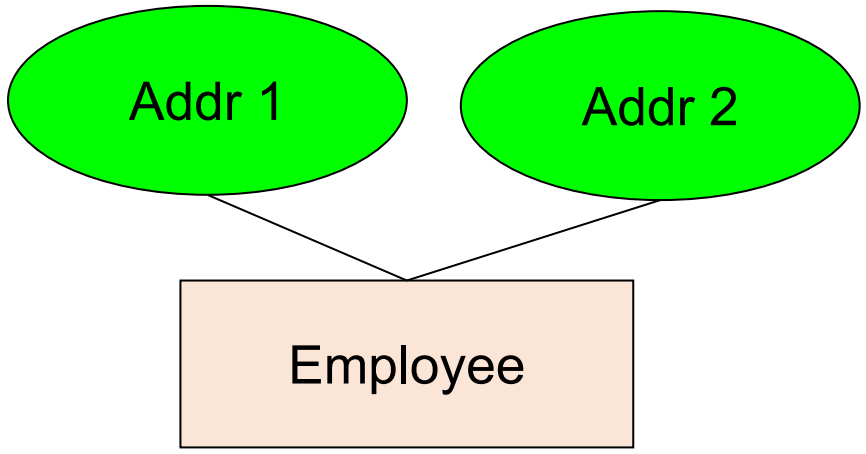



How do we handle employees with multiple addresses here?

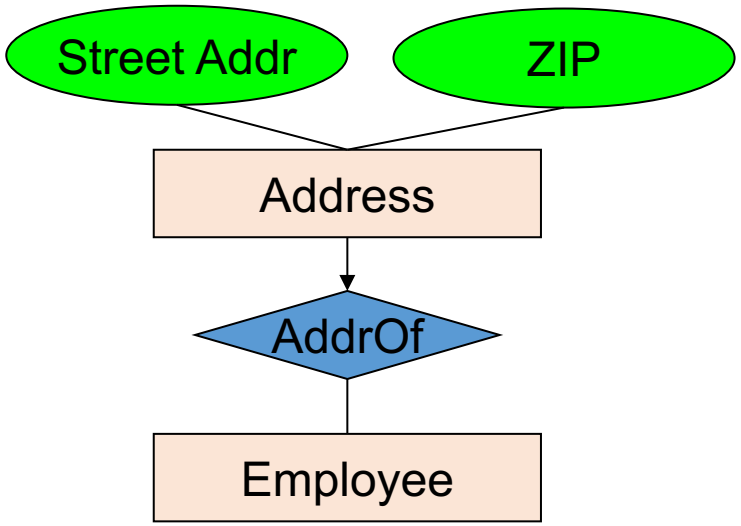
How do we handle addresses where internal structure of the address (e.g. zip code, state) is useful?

# Examples: Entity vs. Attribute

Should address (A)
be an attribute?

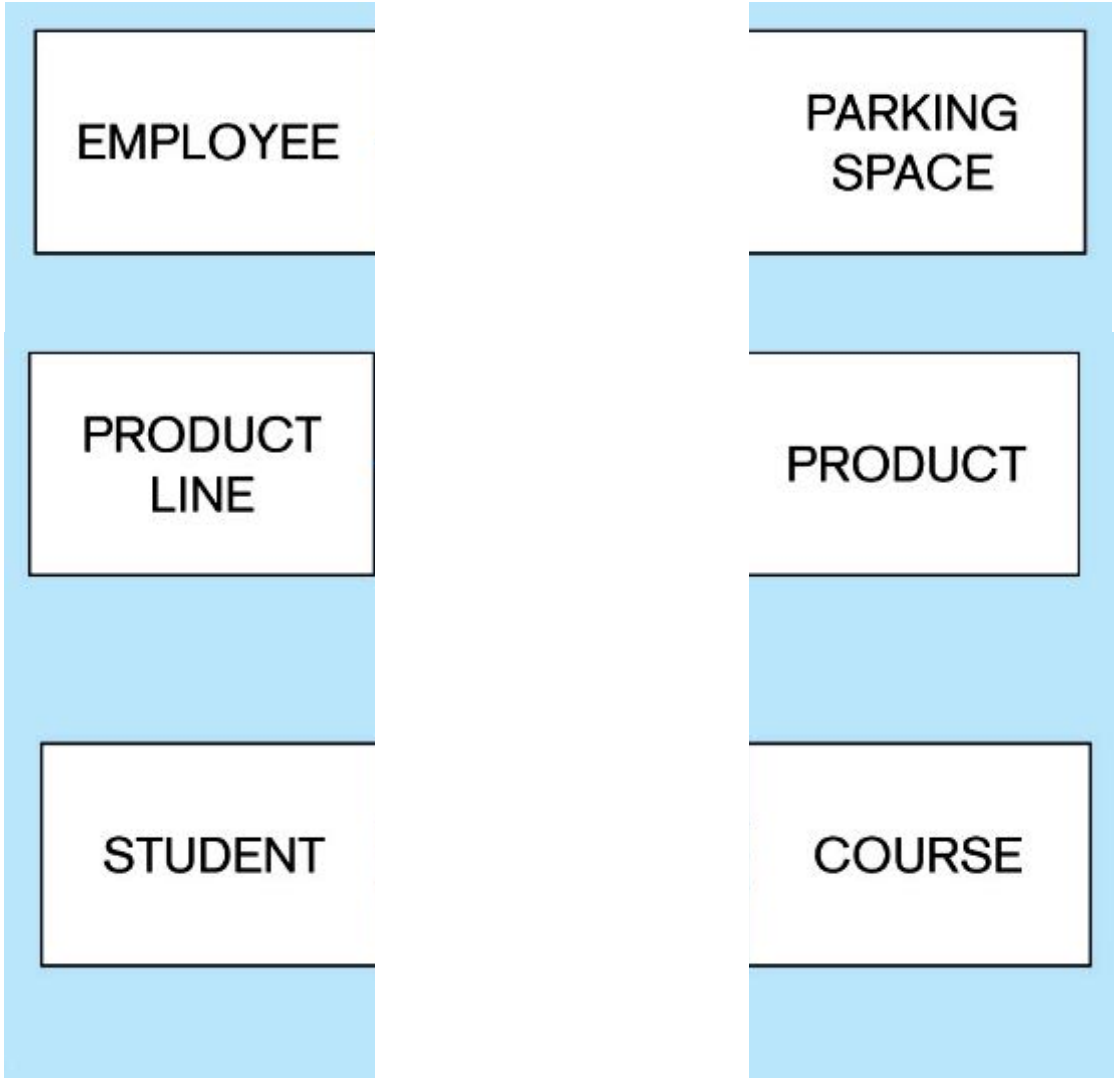



Or (B) be an entity?



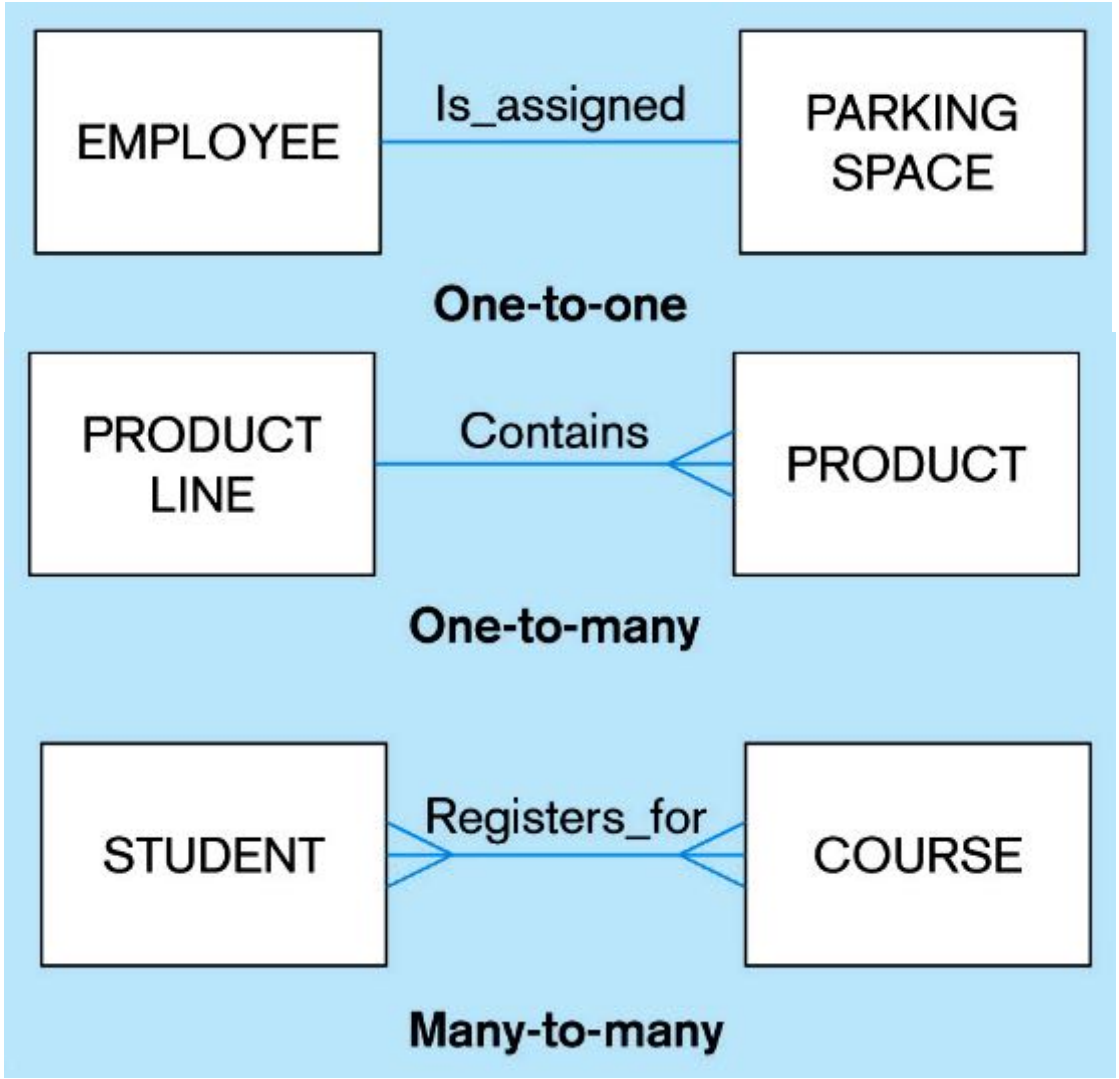

In general, when we want to record several values,
we choose new entity

# Example: Binary Relationships



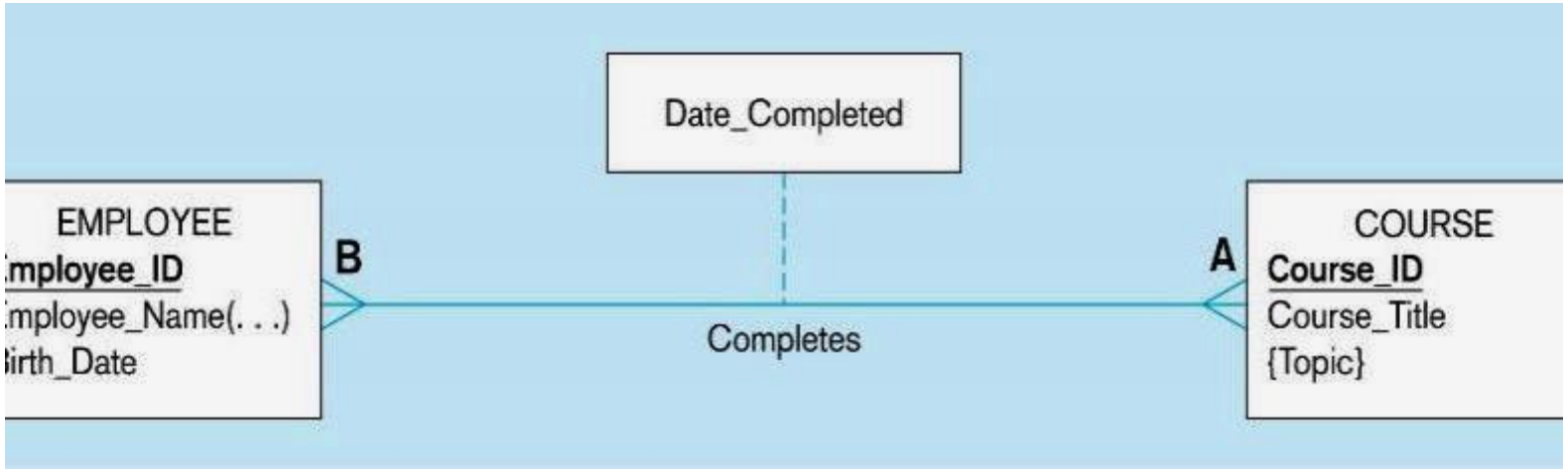

Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.

# Example: Binary Relationships




Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.

# Example: Binary Relationship With An Attribute




- The date completed attribute pertains specifically to the employee's completion of a course
- It is an attribute of the relationship, not either entity in isolation

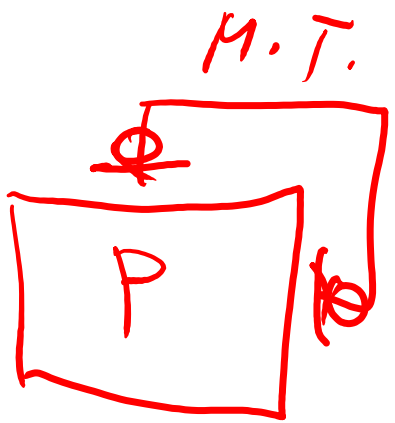
Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.

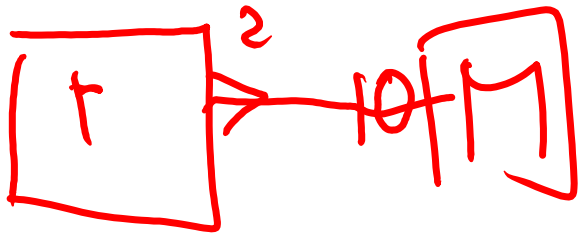
# Examples: Unary Degree Relationship

| Person | Employee | Team |
| --- | --- | --- |
| Is married to | Manages | Stands After |

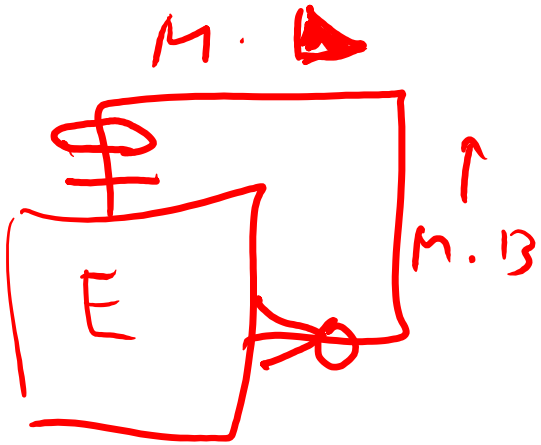
# Examples: Unary Degree Relationship (~~Focus only on cardinality for now~~)

Person
Is married to

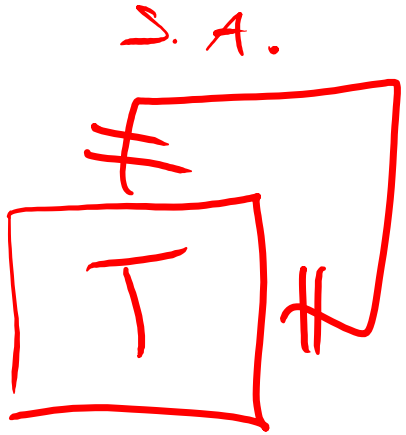



Employee
Manages




Team
Stands After

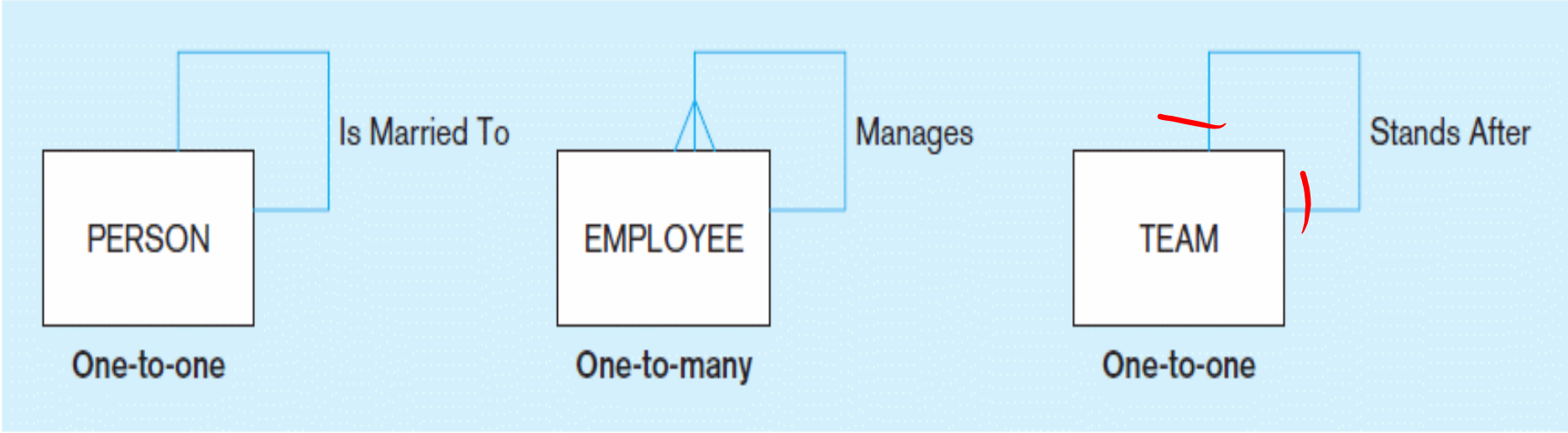


# Examples: Unary Degree Relationship (Focus only on cardinality for now)

Person
Is married to

Employee
Manages

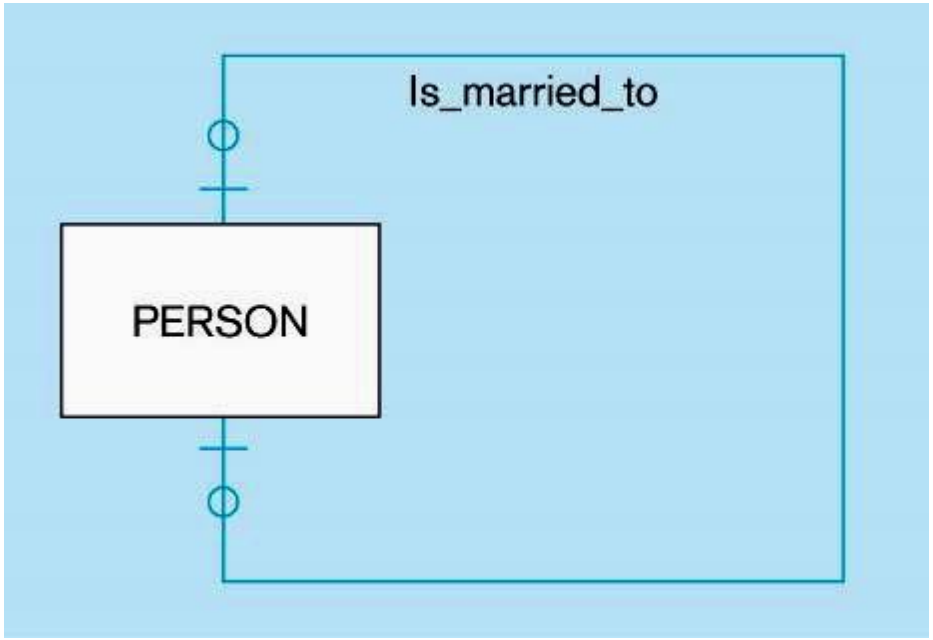
Team
Stands After



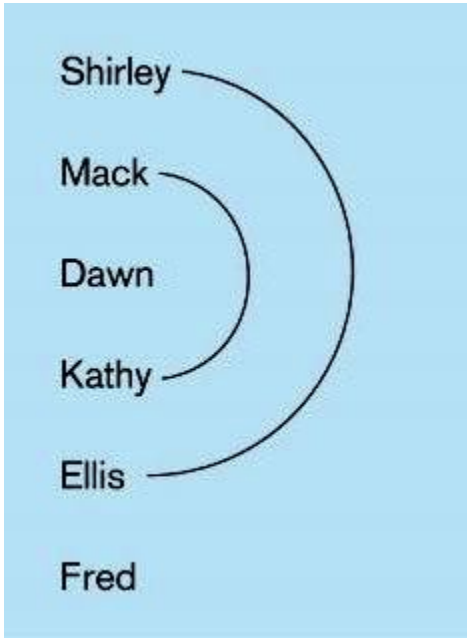

Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.

# Example: Married to with participation

type

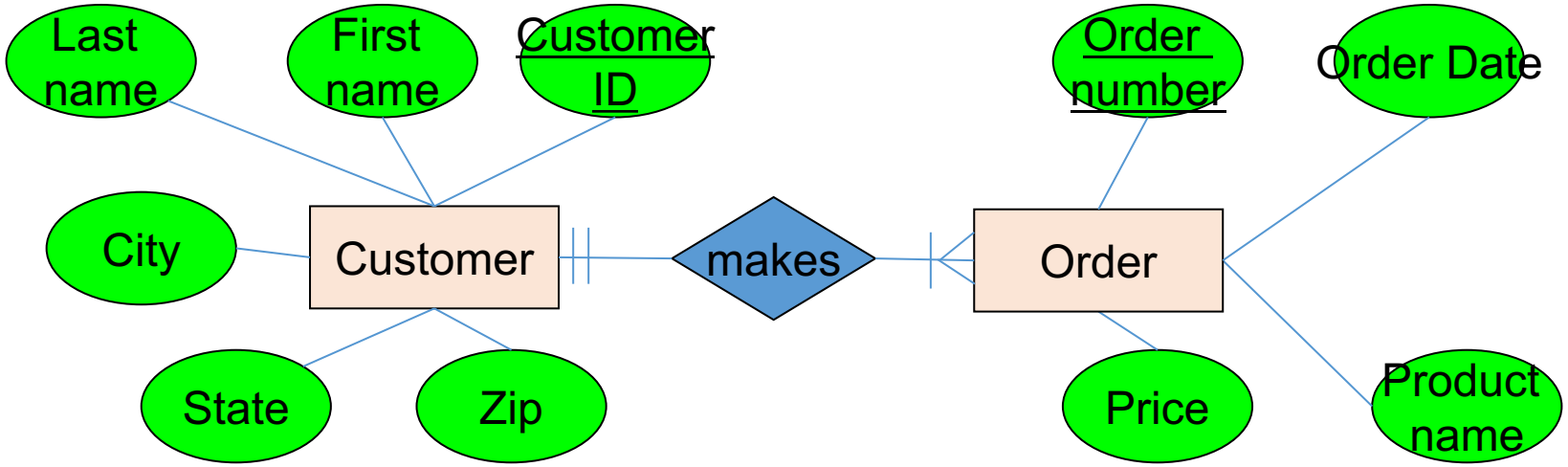



instance

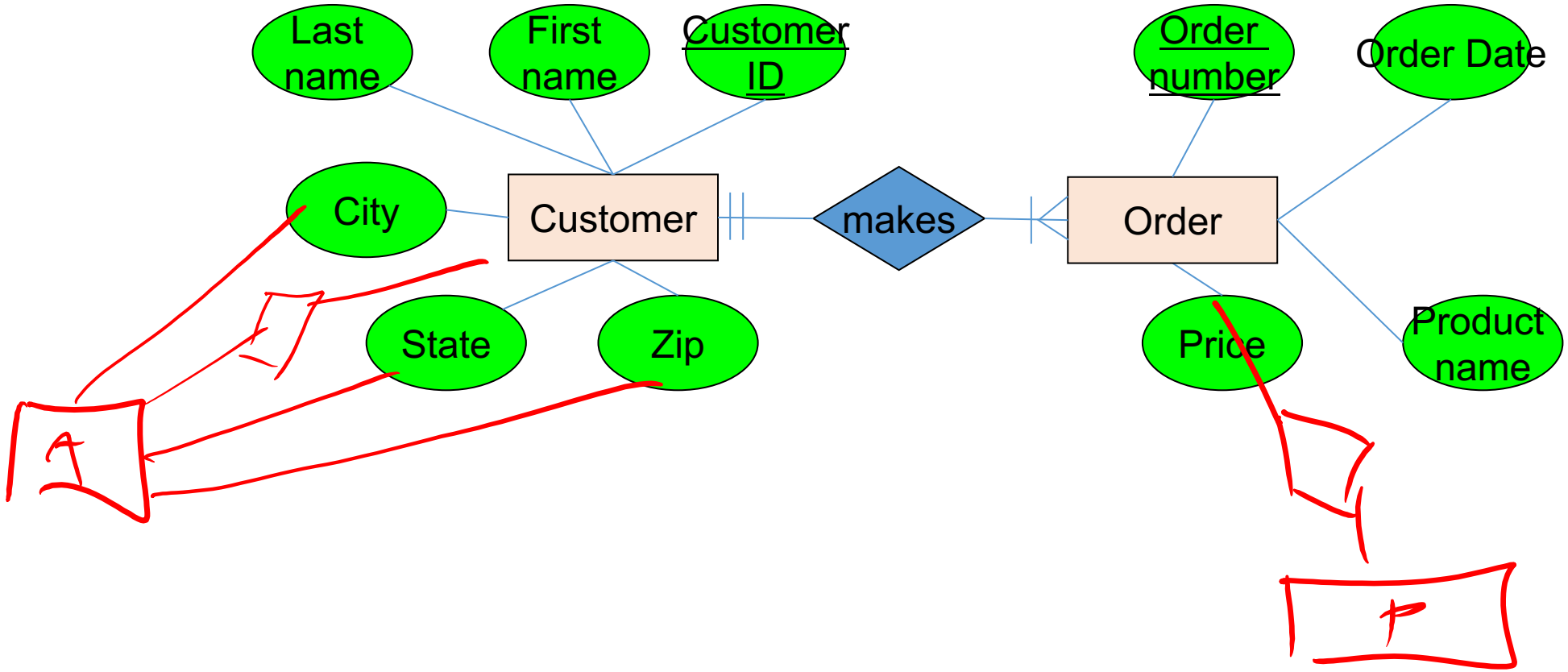



Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.

# There is a problem with our ERD



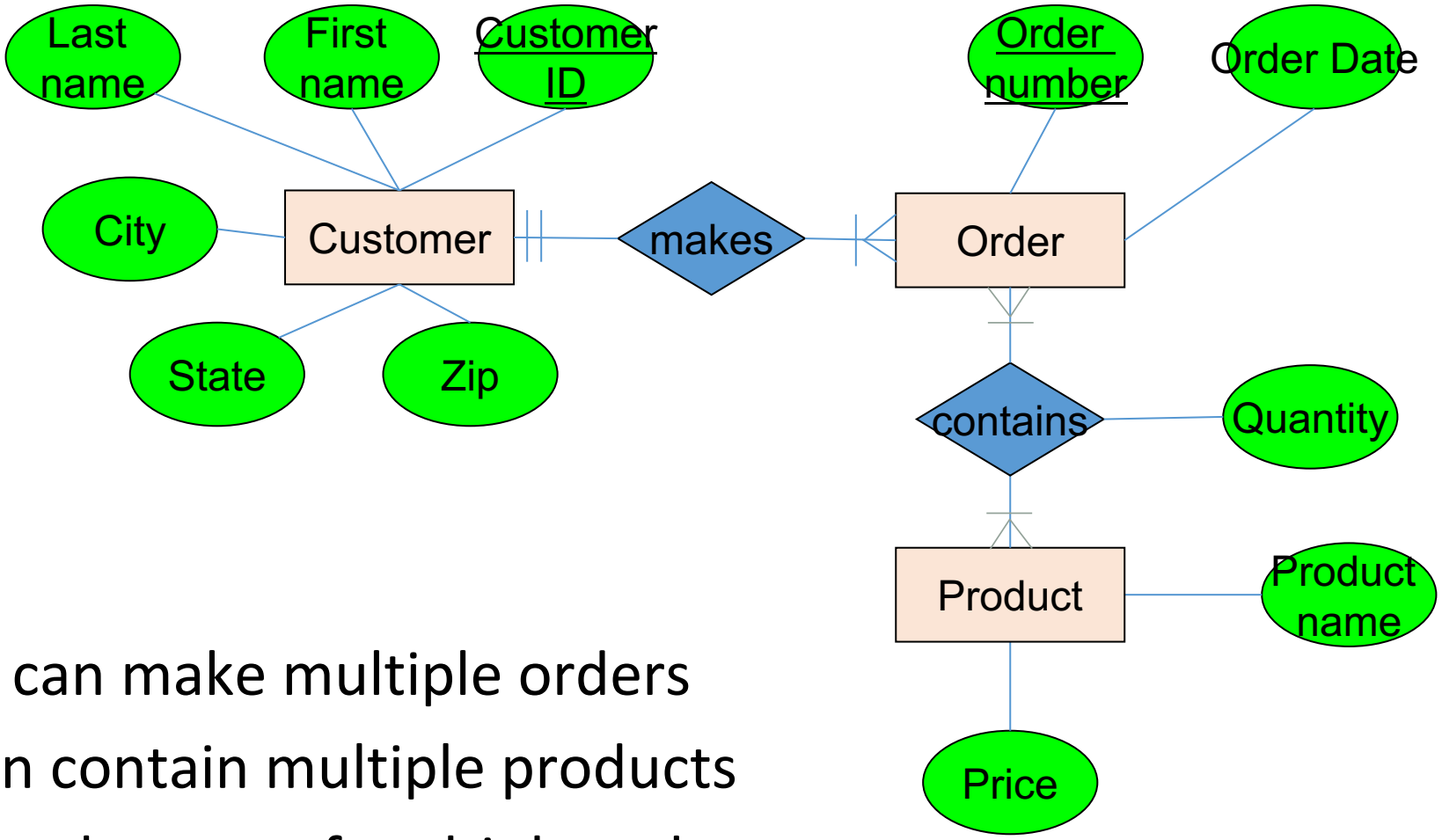
# There is a problem with our ERD




This assumes every order contains only one product.
So if I want two products, I have to make two orders!

The problem: Product is defined as an attribute, not an entity.
(Because we didn't define our requirements clearly enough?)

# Here is a solution




- Now
  - A customer can make multiple orders
  - An order can contain multiple products
  - A product can be part of multiple orders

# Example: multiple relationships

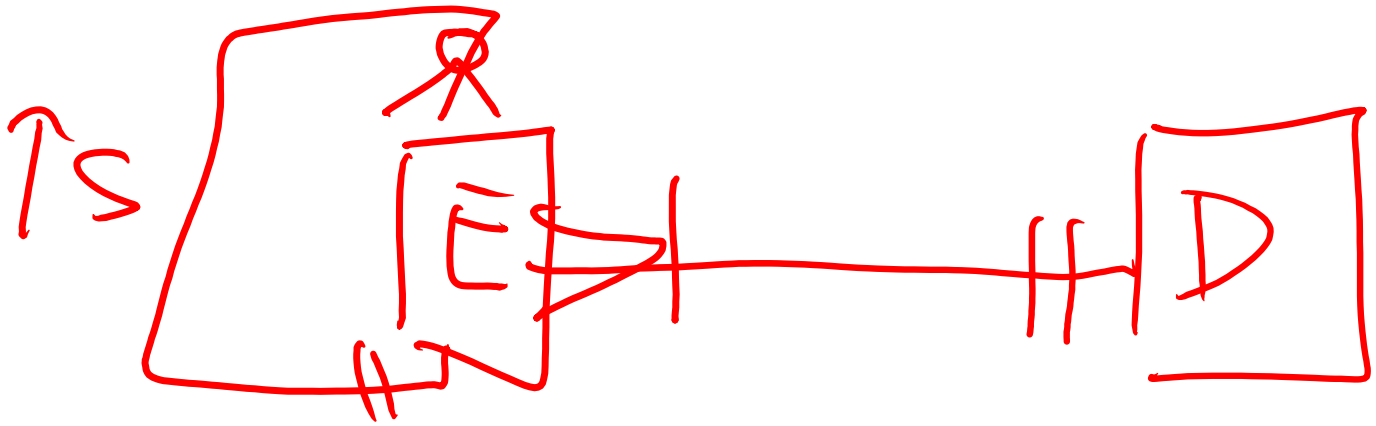
For this exercise, ignore attributes:

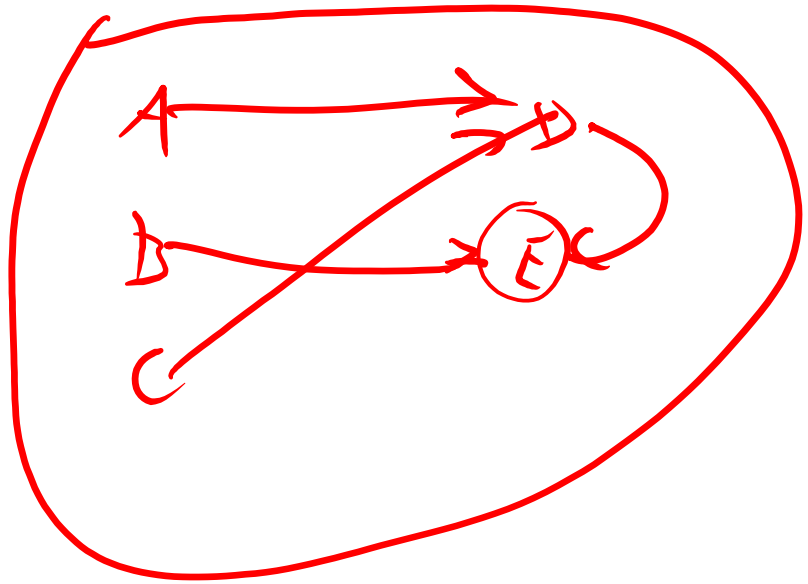
- Each employee is assigned to one department
- Each employee has one supervisor
- Each department is manged by one manager

# Example: multiple relationships

For this exercise, ignore attributes:

- Each employee is assigned to one department
- Each employee has one supervisor
- Each department is manged by one manager

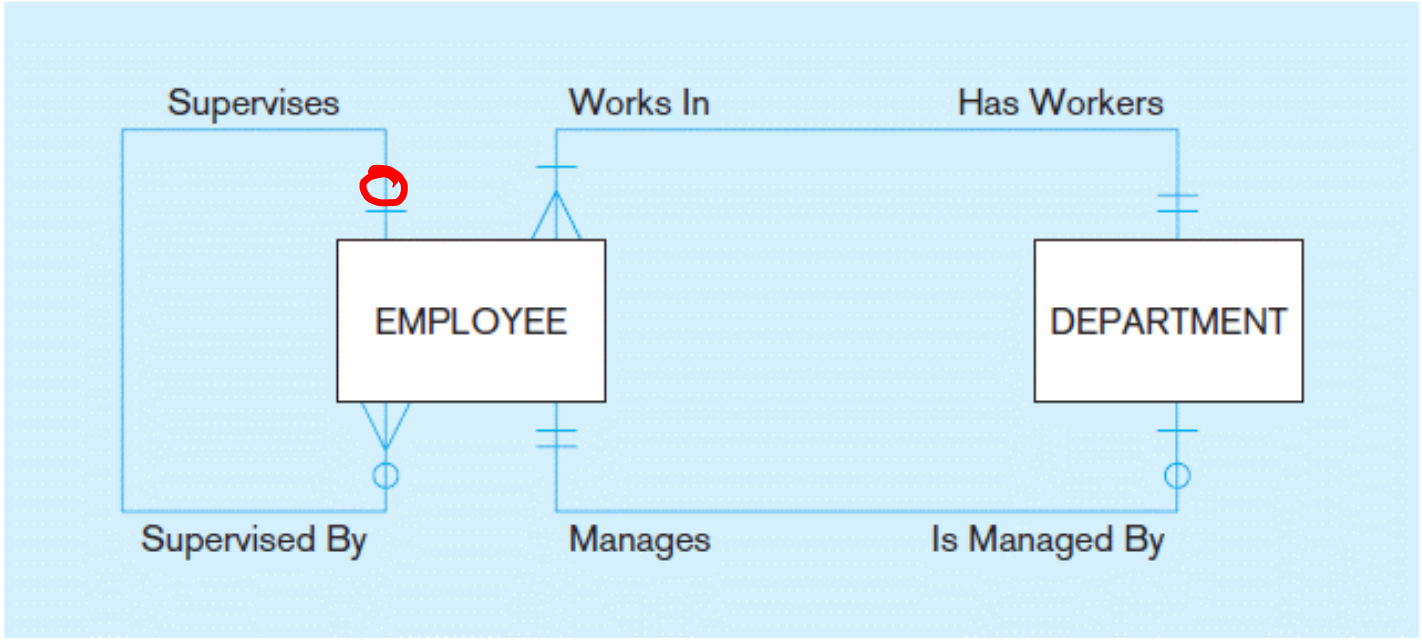


# Example: multiple relationships

For this exercise, ignore attributes:

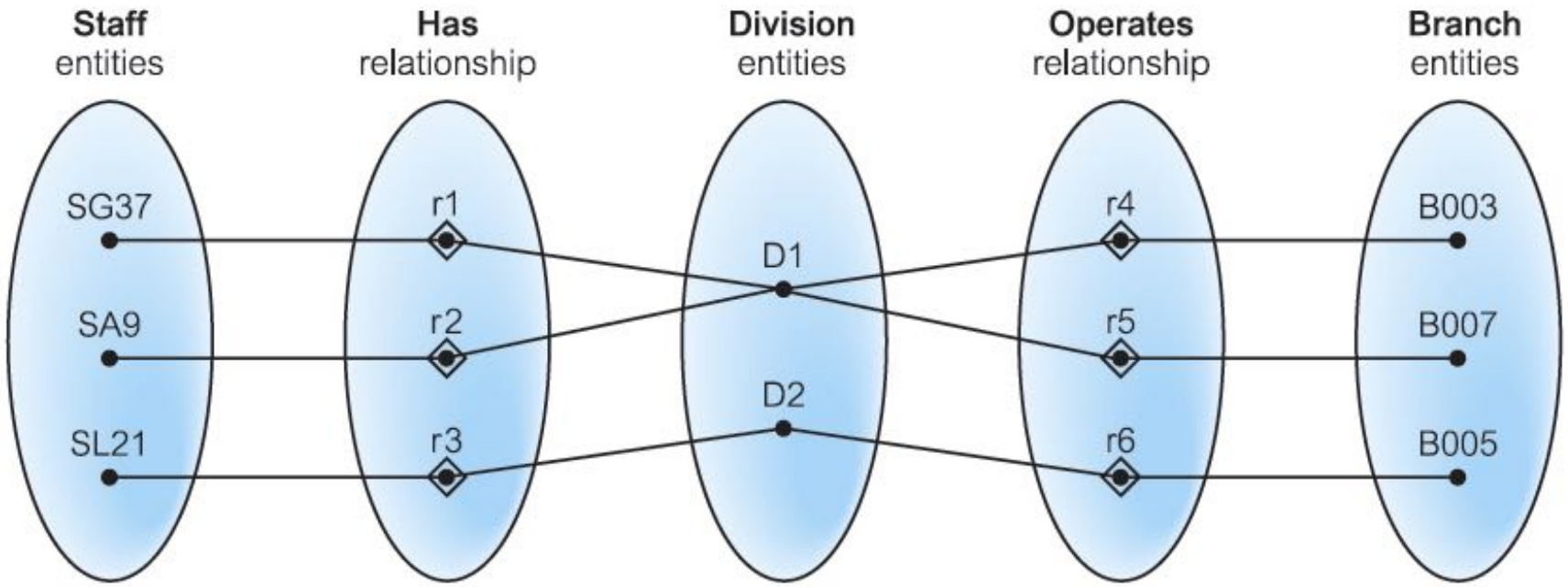
- Each employee is assigned to one department
- Each employee has one supervisor
- Each department is manged by one manager




Recall: Entities can be related to one another in more than one way

Source: Fig. 2-21, Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.

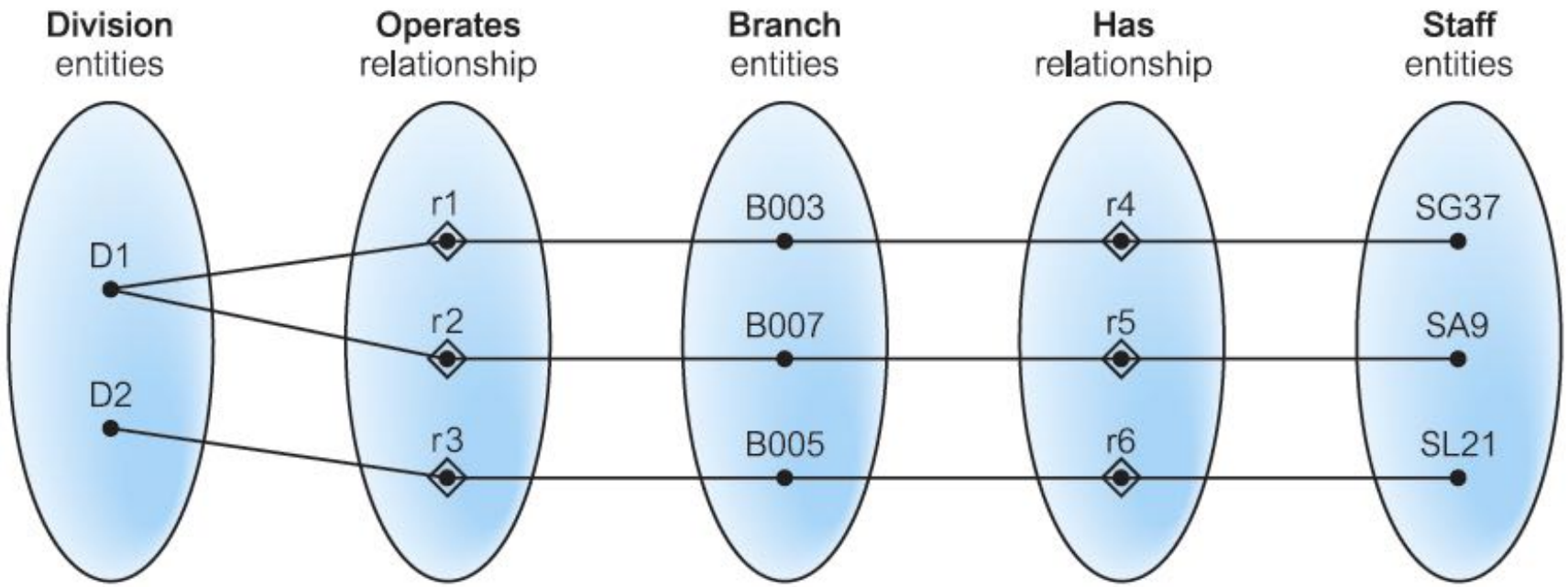
# We have a problem


**Fan Trap**: Where a model represents a relationship between entity types, but the pathway between certain entity occurrences is ambiguous. May exist when two or more 1:n relationships fan out from the same entity

Source: Connolly, Begg: Database systems, 4th ed, p. 364, 2005.

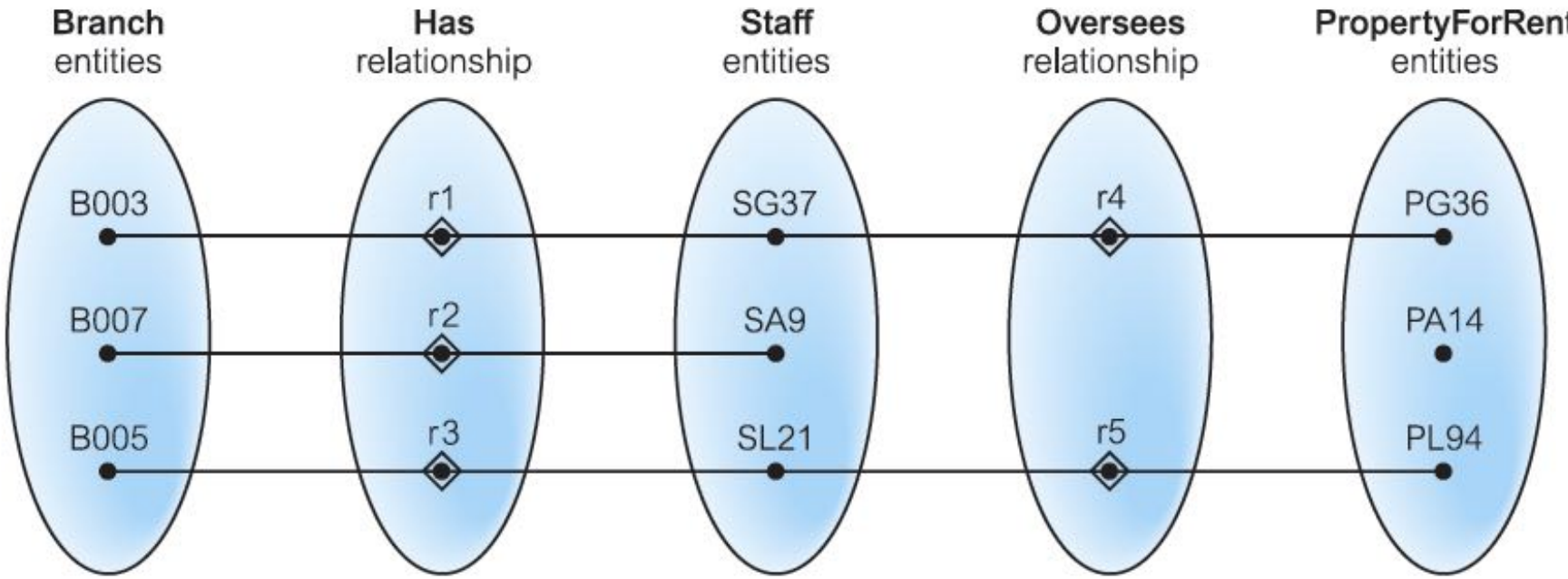
# Restructuring the model helps (in this case)


**Solution:** here restructuring helped. More general solution: add a new relationship

Source: Connolly, Begg: Database systems, 4th ed, p. 364, 2005.

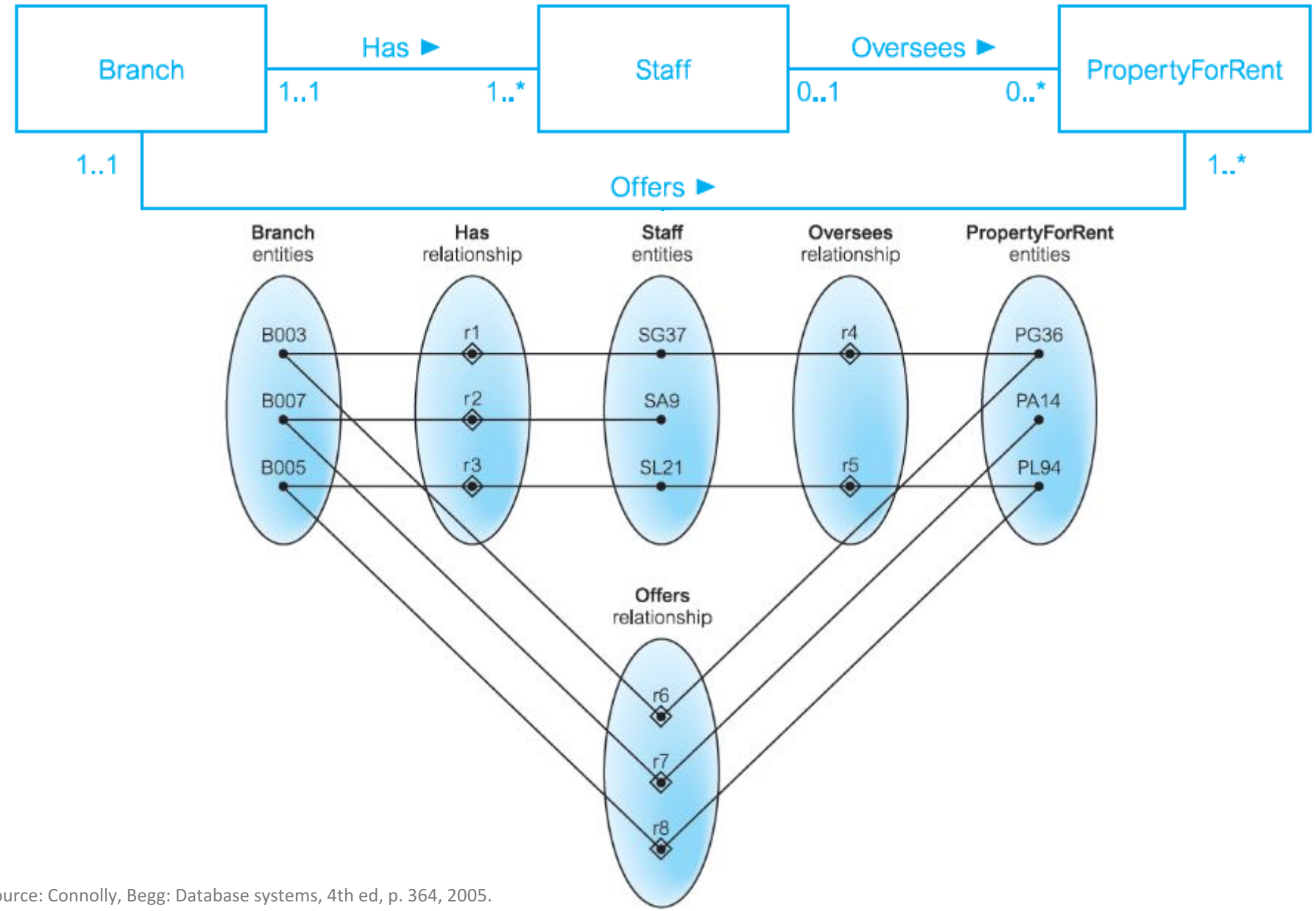
# We have another problem


**Chasm Trap**: Where a model suggests the existence of a relationship between entity types, but the pathway does not exist between certain entity occurrences. May exist when there is a relationship with optional participation between the related entities

Source: Connolly, Begg: Database systems, 4th ed, p. 364, 2005.

# Adding a relationship helps here

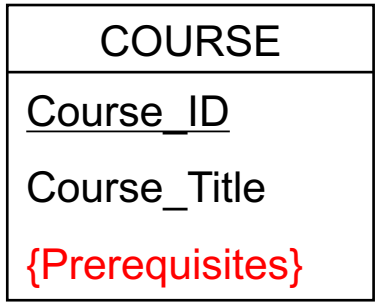


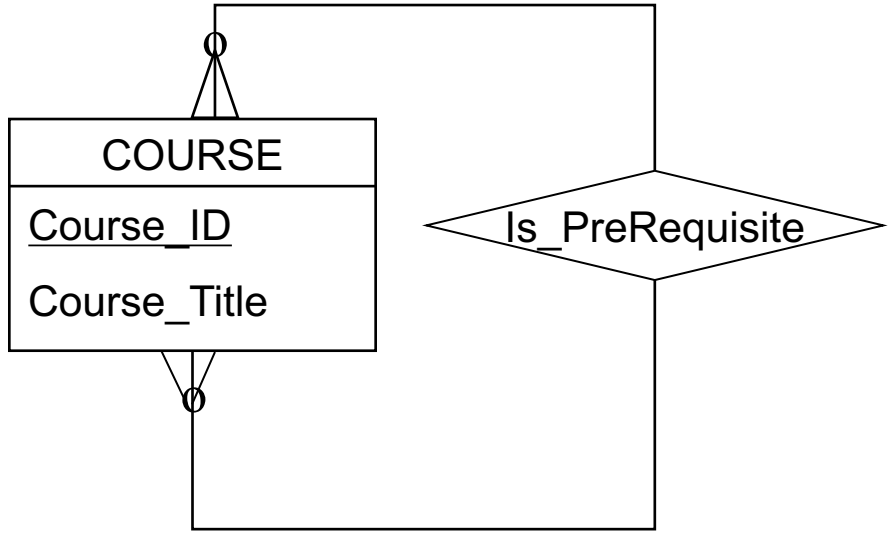

Source: Connolly, Begg: Database systems, 4th ed, p. 364, 2005.

# 1. Multivalued attributes represented as relationships

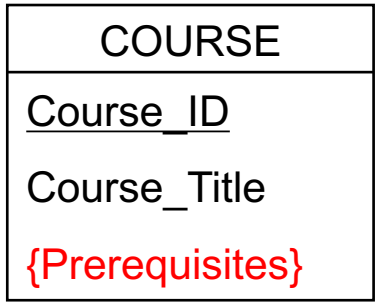
| COURSE |
| --- |
| Course_ID |
| Course_Title |
| {Prerequisites} |

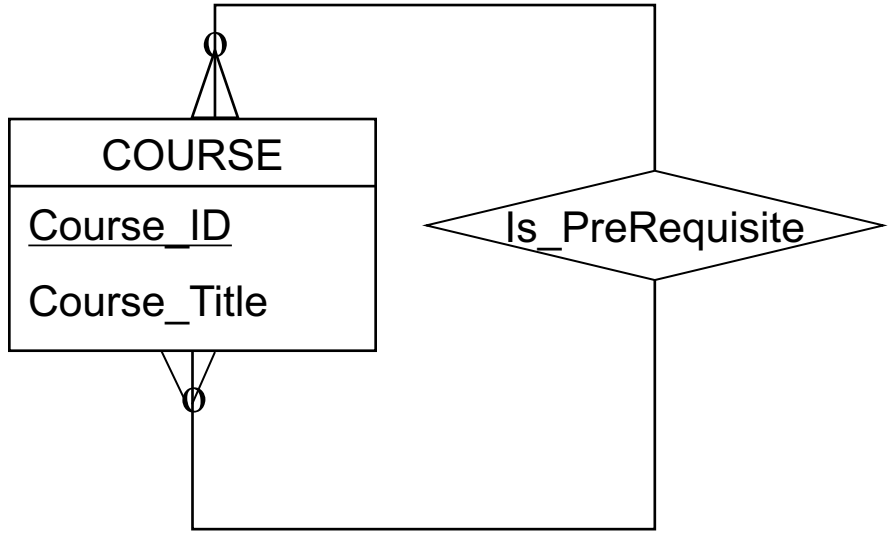
# 1. Multivalued attributes represented as relationships



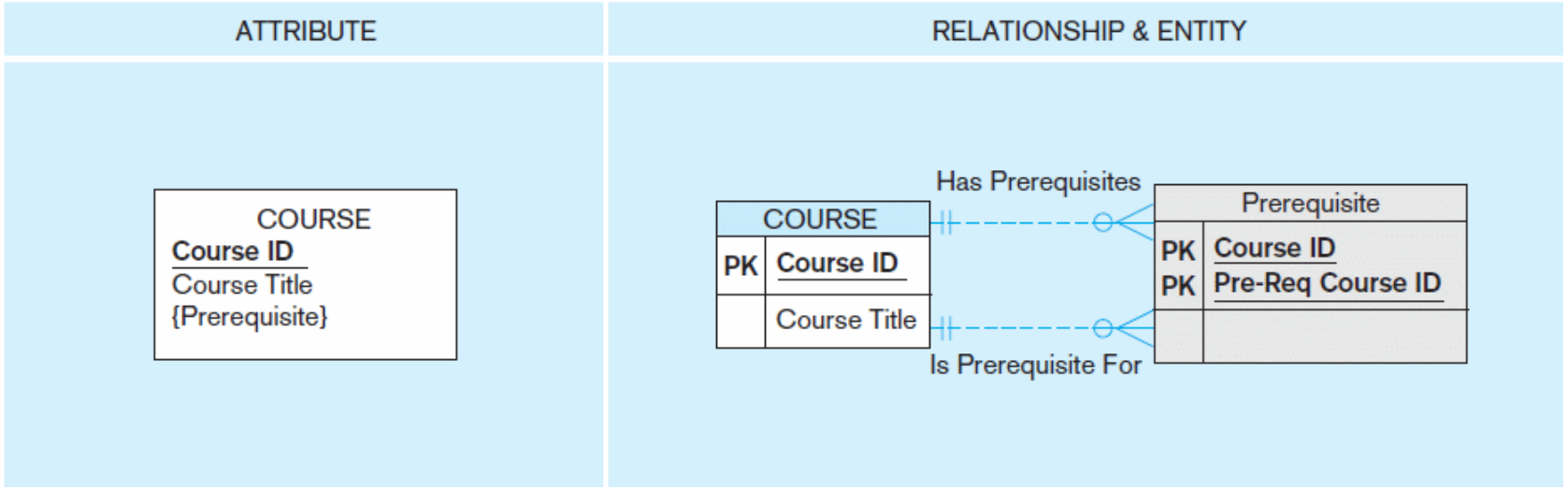



# 1. Multivalued attributes represented as relationships




Notation used by Microsoft Visio:



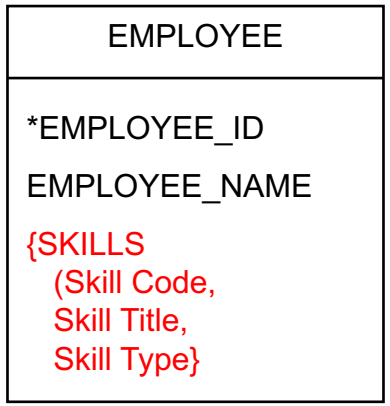

Source: Fig 2.15: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.

# 2. Multivalued attributes can be represented as entities

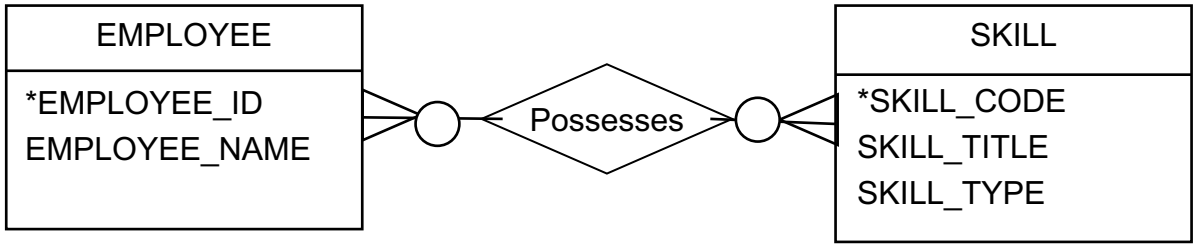
| EMPLOYEE |
| --- |
| *EMPLOYEE_ID<br>EMPLOYEE_NAME<br>{SKILLS<br>(Skill Code,<br>Skill Title,<br>Skill Type} |

composite

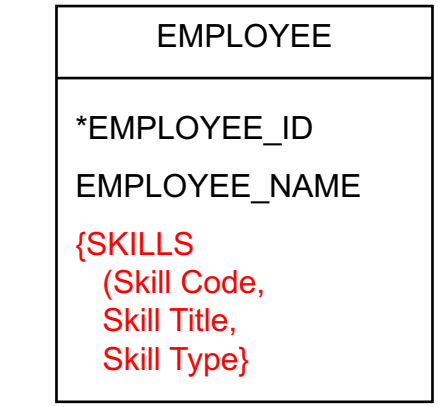
# 2. Multivalued attributes can be represented as entities



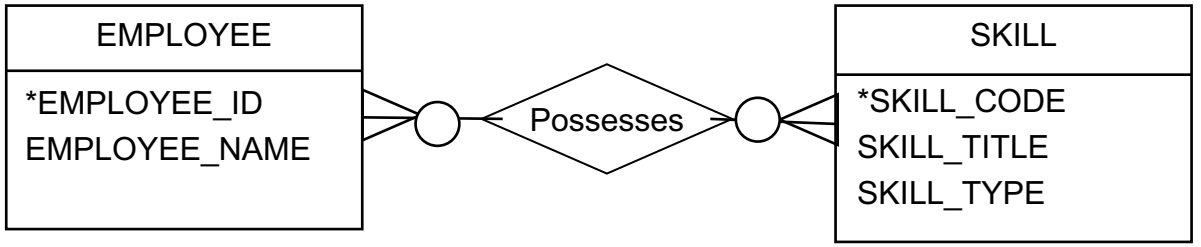

composite



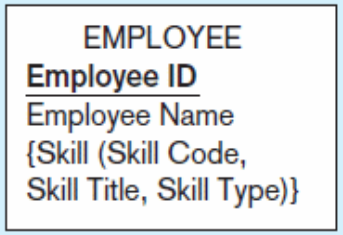
# 2. Multivalued attributes can be represented as entities

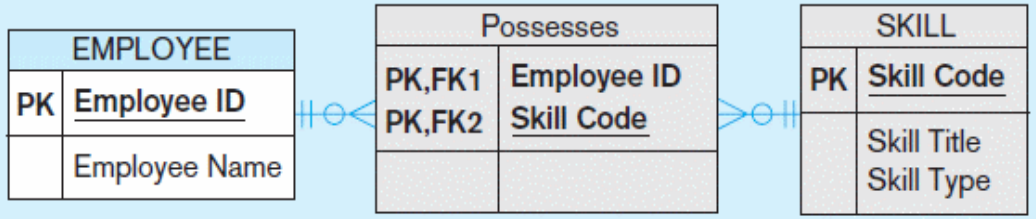



composite




Notation used by Microsoft Visio



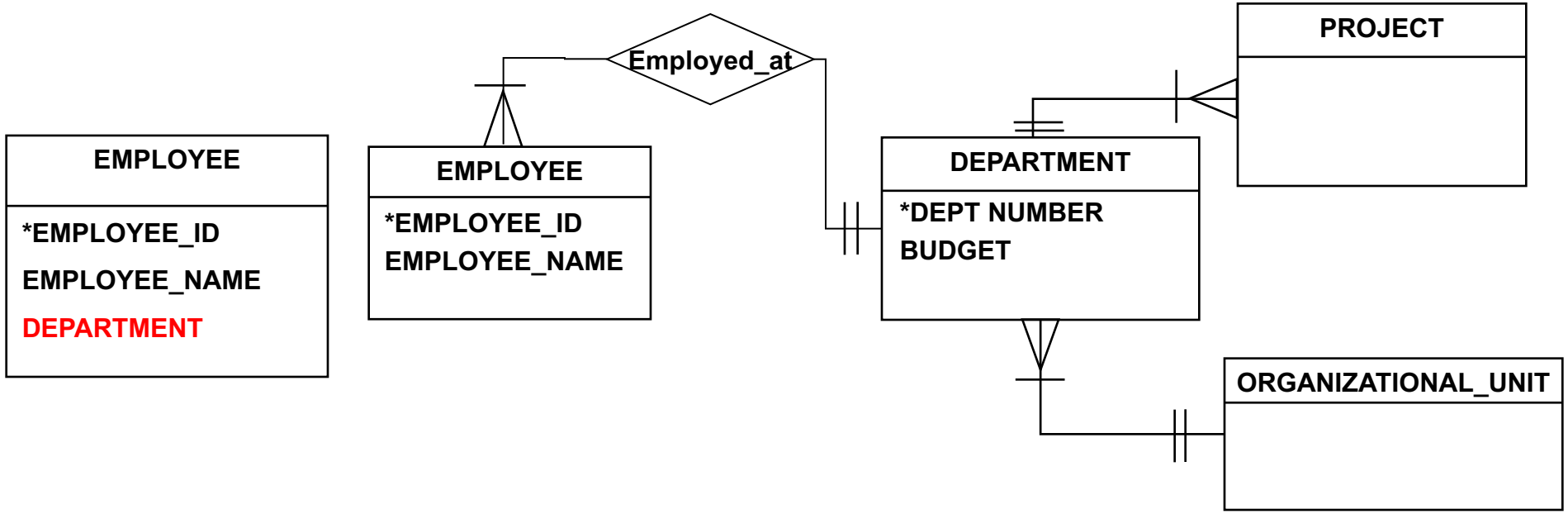




Source: Fig 2.15: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.

# 3. Attribute vs.

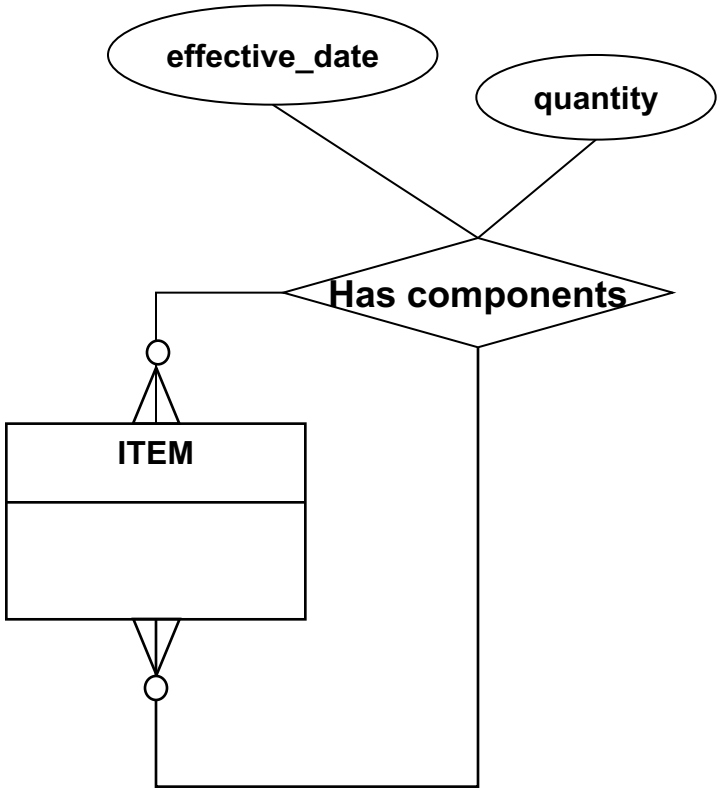
| EMPLOYEE |
| --- |
| *EMPLOYEE_ID<br>EMPLOYEE_NAME<br>DEPARTMENT |

# 3. Attribute vs. Multiple Entities



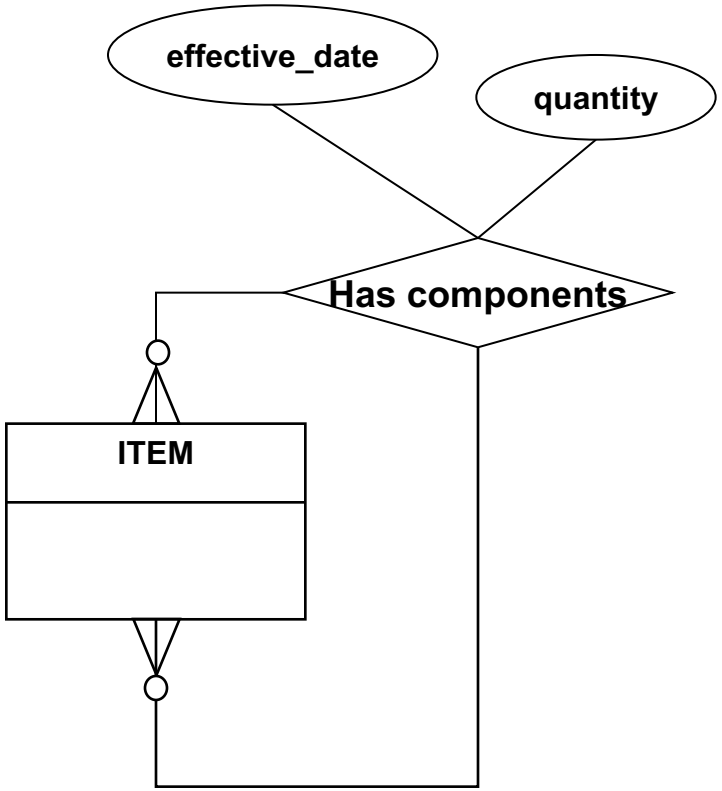
# Bill-of-materials (BOM) structure

**Relationship**

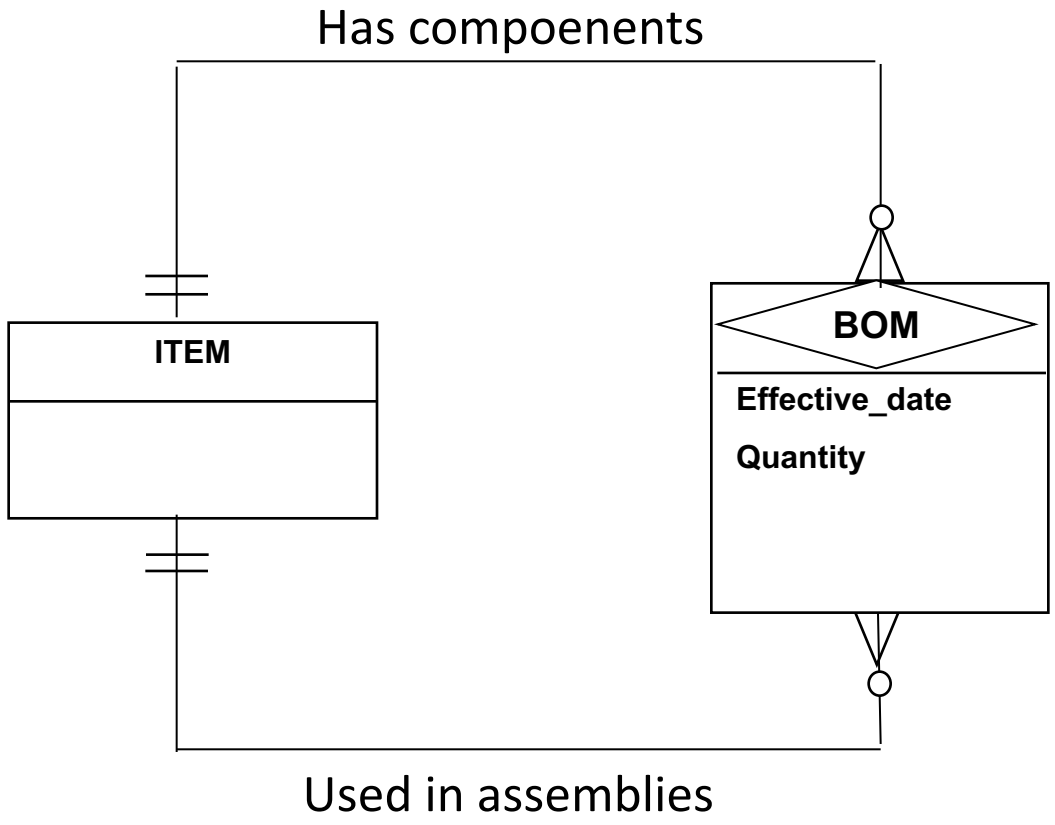


# Bill-of-materials (BOM) structure

## Relationship




## Associative entity



# In-Class Exercise (Part II): create an ERD

The following grade report below is mailed to students at the end of each semester. Prepare an ERD reflecting the data contained in the grade report (capturing Entities, Attributes, and Relationships). Assume that each course is taught by one instructor. Explain what you chose for the identifier of each entity type

MILLENNIUM COLLEGE
GRADE REPORT
FALL SEMESTER 200X

| | | |
|---|---|---|
| NAME: | Emily Williams | ID: 268300458 |
| CAMPUS ADDRESS: | 208 Brooks Hall | |
| MAJOR: | Information Systems | |

| COURSE ID | TITLE | INSTRUCTOR NAME | INSTRUCTOR LOCATION | GRADE |
|---|---|---|---|---|
| IS 350 | Database Mgt. | Codd | B104 | A |
| IS 465 | System Analysis | Parsons | B317 | B |

Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.

# In-Class Exercise (Part III): create an ERD

MILLENNIUM COLLEGE
GRADE REPORT
FALL SEMESTER 200X

NAME: Emily Williams ID: 268300458
CAMPUS ADDRESS: 208 Brooks Hall
MAJOR: Information Systems

| COURSE ID | TITLE | INSTRUCTOR NAME | INSTRUCTOR LOCATION | GRADE |
|---|---|---|---|---|
| IS 350 | Database Mgt. | Codd | B104 | A |
| IS 465 | System Analysis | Parsons | B317 | B |

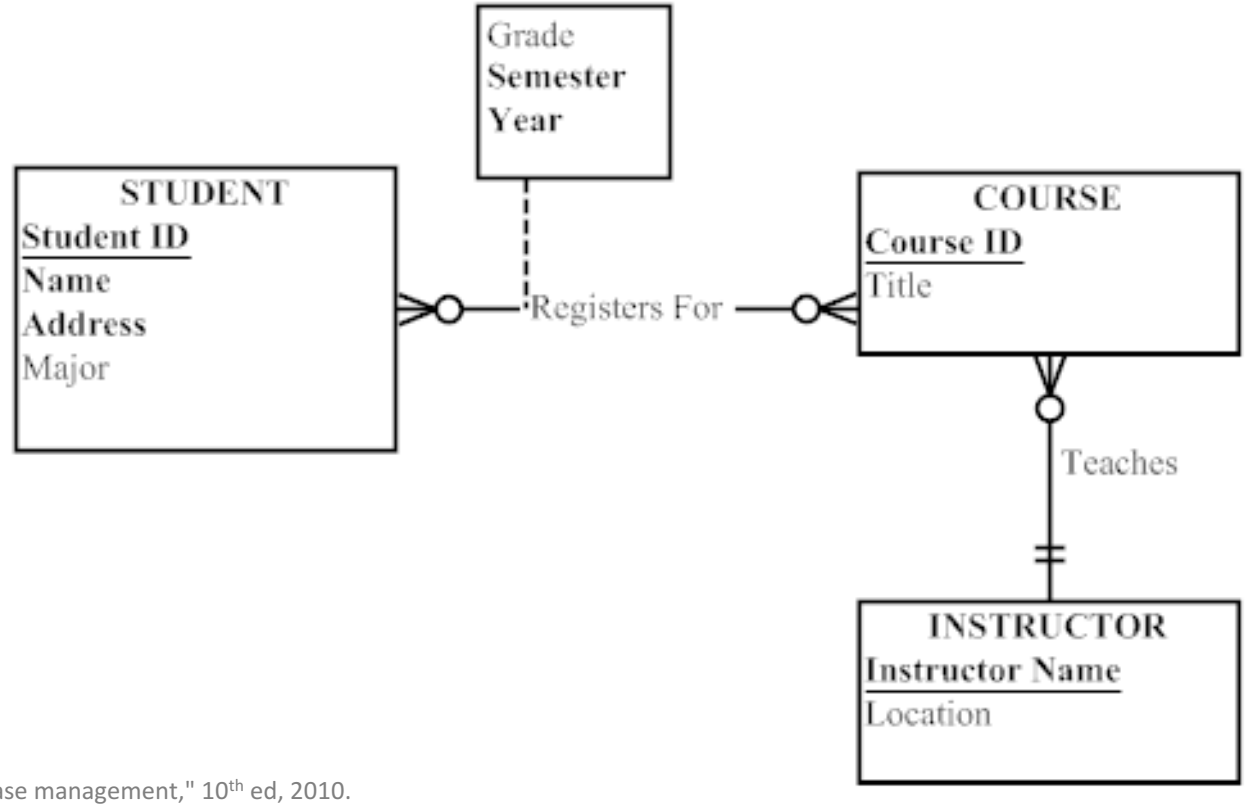
Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.

# In-Class Exercise (Part III): create an ERD

MILLENNIUM COLLEGE
GRADE REPORT
FALL SEMESTER 200X

NAME: Emily Williams ID: 268300458
CAMPUS ADDRESS: 208 Brooks Hall
MAJOR: Information Systems

| COURSE ID | TITLE | INSTRUCTOR NAME | INSTRUCTOR LOCATION | GRADE |
|---|---|---|---|---|
| IS 350 | Database Mgt. | Codd | B104 | A |
| IS 465 | System Analysis | Parsons | B317 | B |



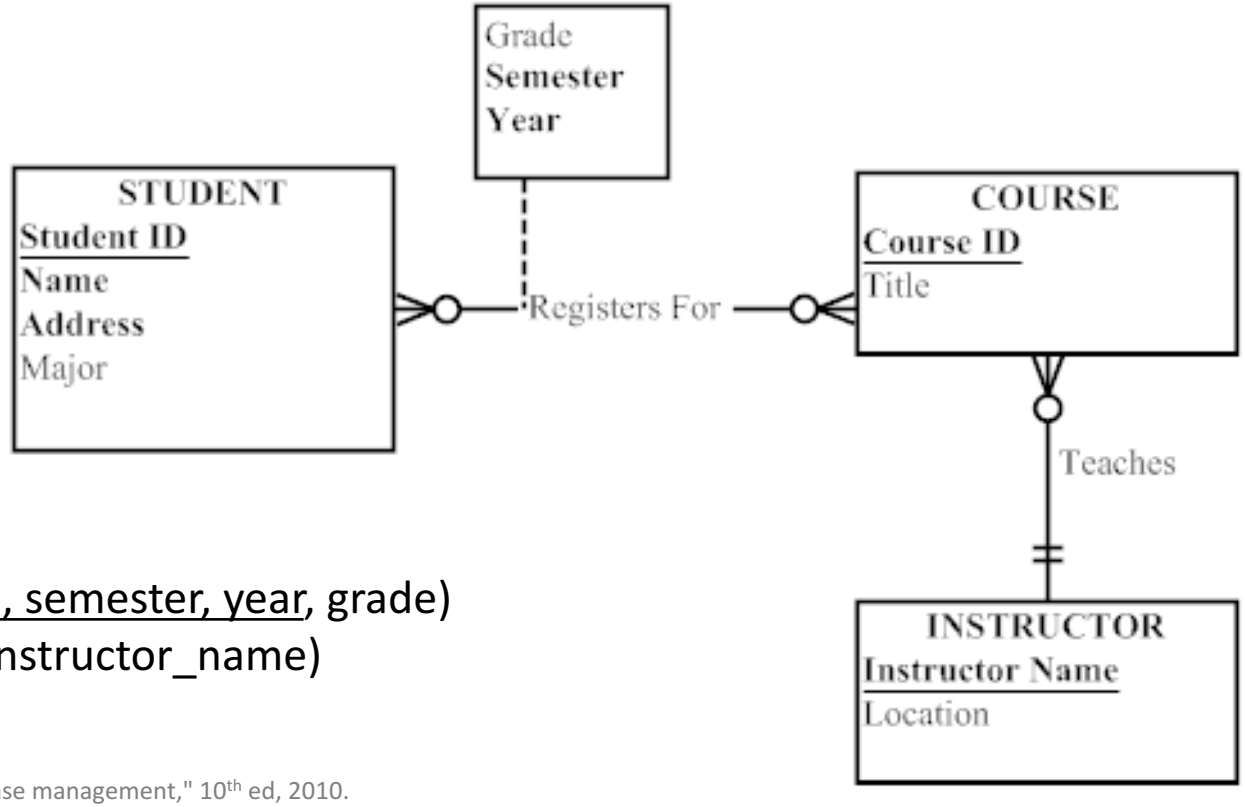

Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.

# In-Class Exercise (Part III): relational schema

MILLENNIUM COLLEGE
GRADE REPORT
FALL SEMESTER 200X

NAME: Emily Williams ID: 268300458
CAMPUS ADDRESS: 208 Brooks Hall
MAJOR: Information Systems

| COURSE ID | TITLE | INSTRUCTOR NAME | INSTRUCTOR LOCATION | GRADE |
|---|---|---|---|---|
| IS 350 | Database Mgt. | Codd | B104 | A |
| IS 465 | System Analysis | Parsons | B317 | B |




Register(@sid, @cid, semester, year, grade)
Course(cid, title, @instructor_name)

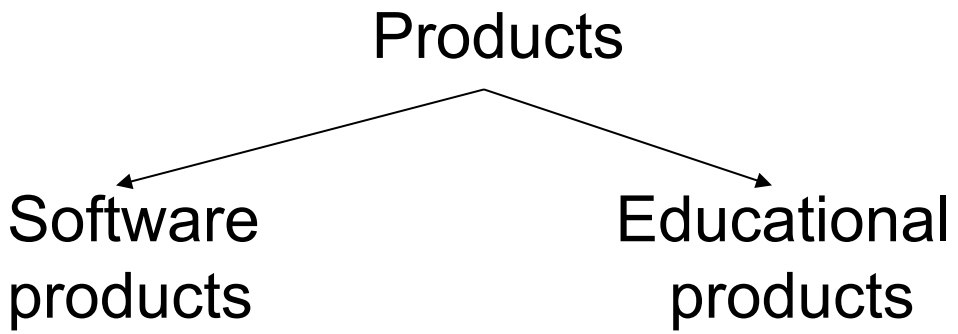
Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.

# EER (Enhanced ER)

## Subtypes
## in ER diagrams

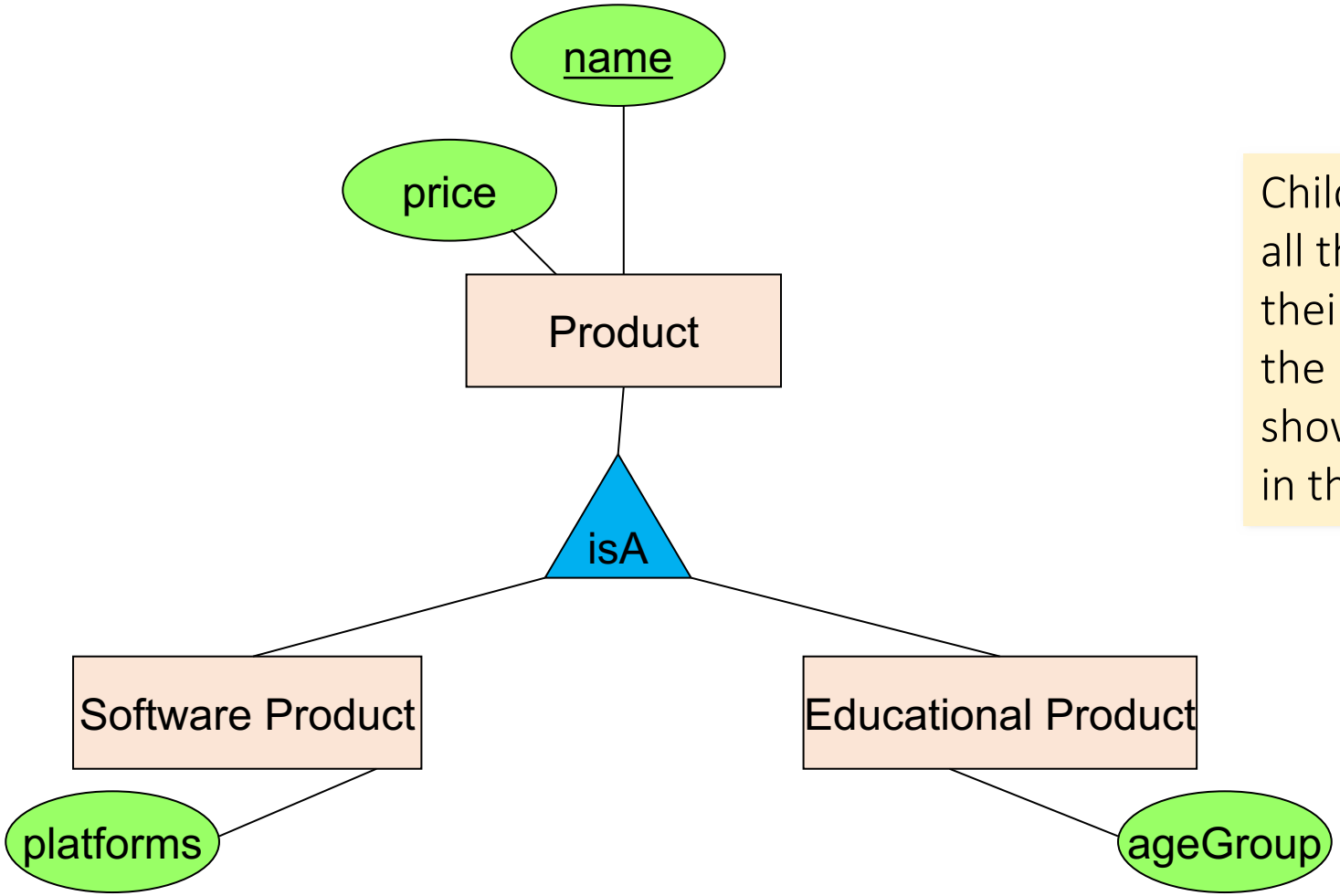
# Modeling Subclasses

- Some objects in a class may be special, i.e. worthy of their own class
  - Define a new class?
    - *But what if we want to maintain connection to current class?*
  - Better: define a *subclass*
    - *Ex:*




We can define **subclasses** in E/R!

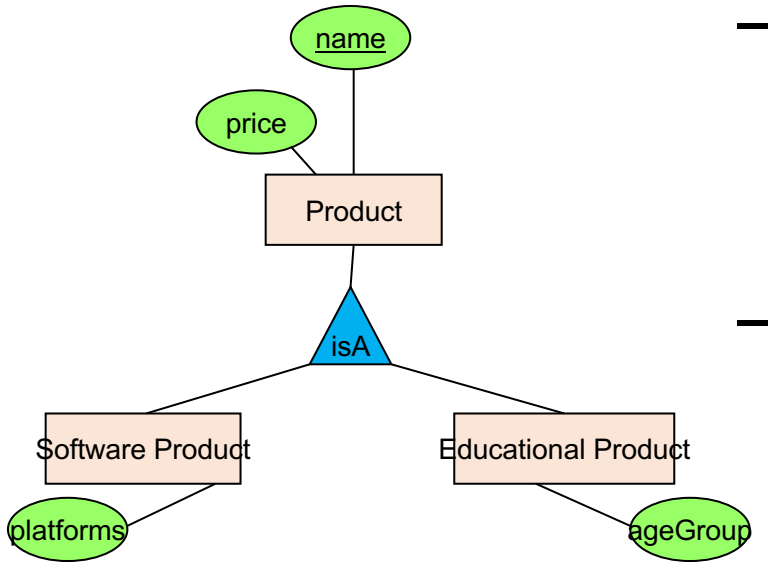
# Modeling Subclasses

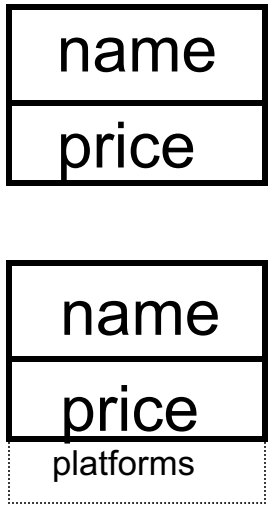


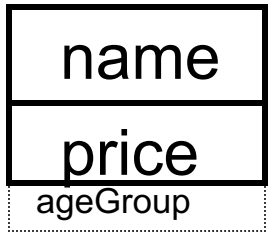

Child subclasses contain all the attributes of *all* of their parent classes **plus** the new attributes shown attached to them in the E/R diagram

# Understanding Subclasses

- Think in terms of records; ex:
  - Product
  - SoftwareProduct
  - EducationalProduct

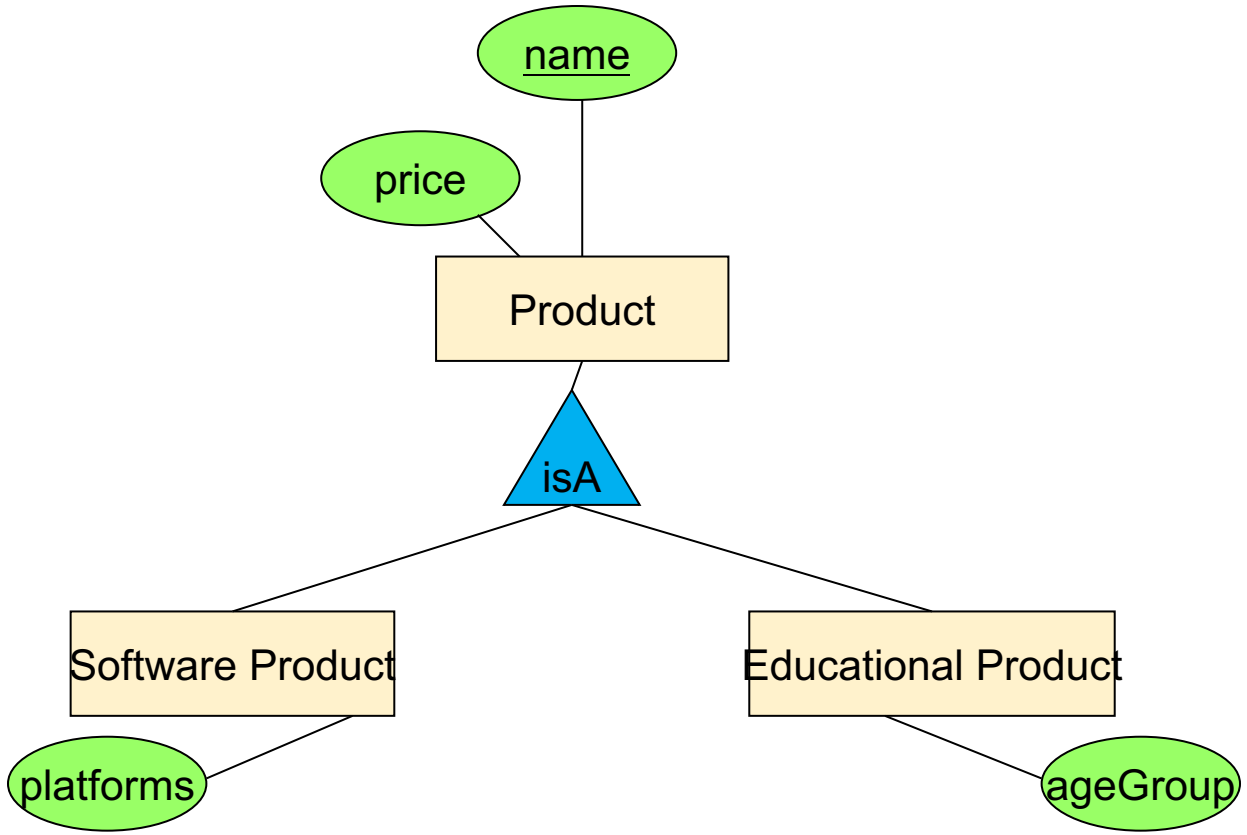



Child subclasses contain all the attributes of *all* of their parent classes **plus** the new attributes shown attached to them in the E/R diagram

# Think like tables…




Product

| name | price | category |
|---|---|---|
| Gizmo | 99 | gadget |
| Camera | 49 | photo |
| Toy | 39 | gadget |

Sw.Product

| name | platforms |
|---|---|
| Gizmo | unix |

Ed.Product

| name | ageGroup |
|---|---|
| Gizmo | toddler |
| Toy | retired |

# IsA Review

- If we declare ***A IsA B*** then every **A** is a **B**

- We use IsA to

  - Add descriptive attributes to a subclass

  - To identify entities that participate in a relationship

# Modeling UnionTypes With Subclasses

Person

FurniturePiece

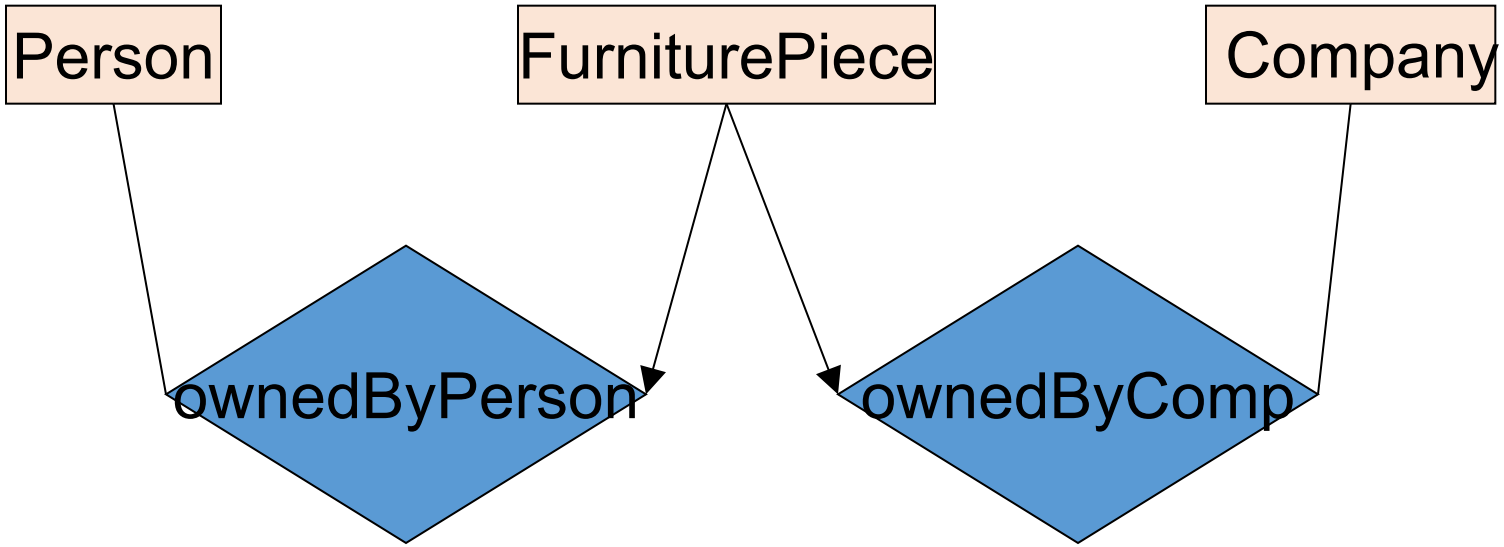
Company

Suppose each piece of furniture is owned either by a person, or by a company. *How do we represent this?*

# Modeling Union Types with Subclasses

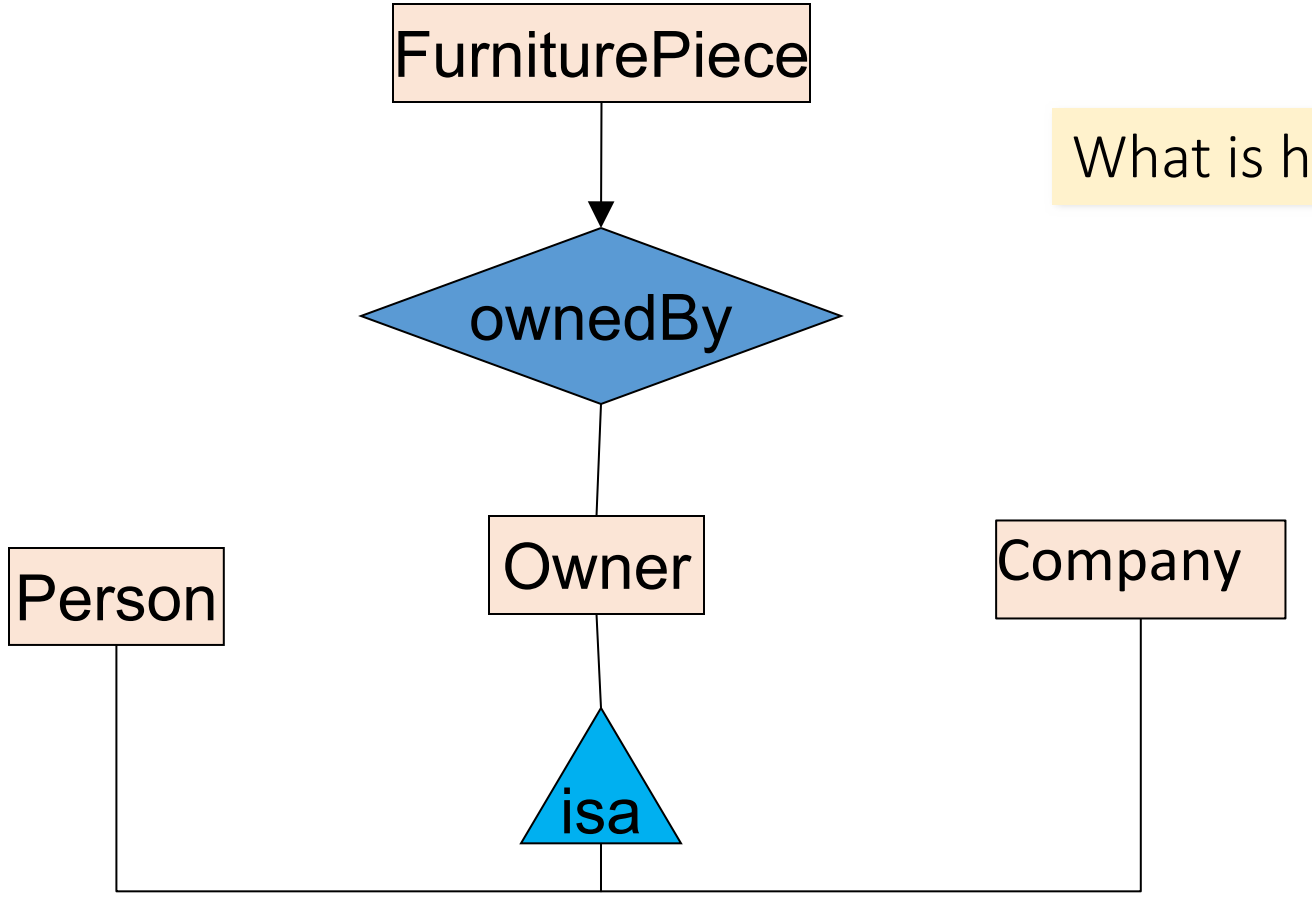
- Say: each piece of furniture is owned either by a person, or by a company

- Solution 1. Acceptable, but imperfect (What's wrong ?)



# Modeling Union Types with Subclasses

- Solution 2: better (though more laborious)




What is happening here?

# E/R Summary

- E/R diagrams are a visual syntax that allows technical and non-technical people to talk
  - For conceptual design

- Basic constructs: **entity**, **relationship**, and **attributes**

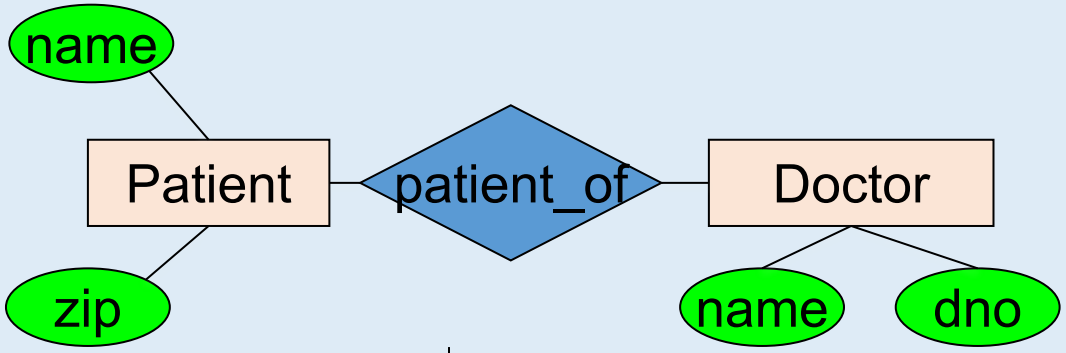
- A good design is faithful to the constraints of the application, but not overzealous

# From ERDs to Relations

# Data modeling and Database Design Process

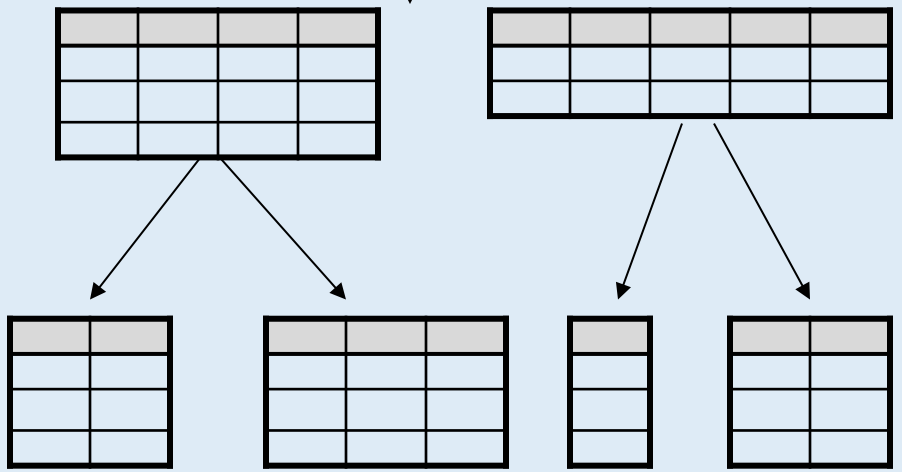
## 1. ER Diagram

Conceptual Model:
("technology independent")
describe main data items



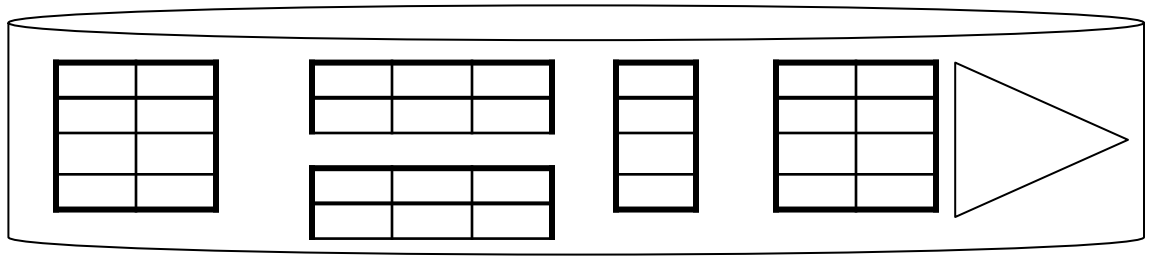

## 2. Relational Database Design

Logical Model
("for relational databases"):
Tables, Constraints
Functional Dependencies
Normalization:
Eliminates anomalies



## 3. Database Implementation

Physical Model
Physical storage details
Result: Physical Schema

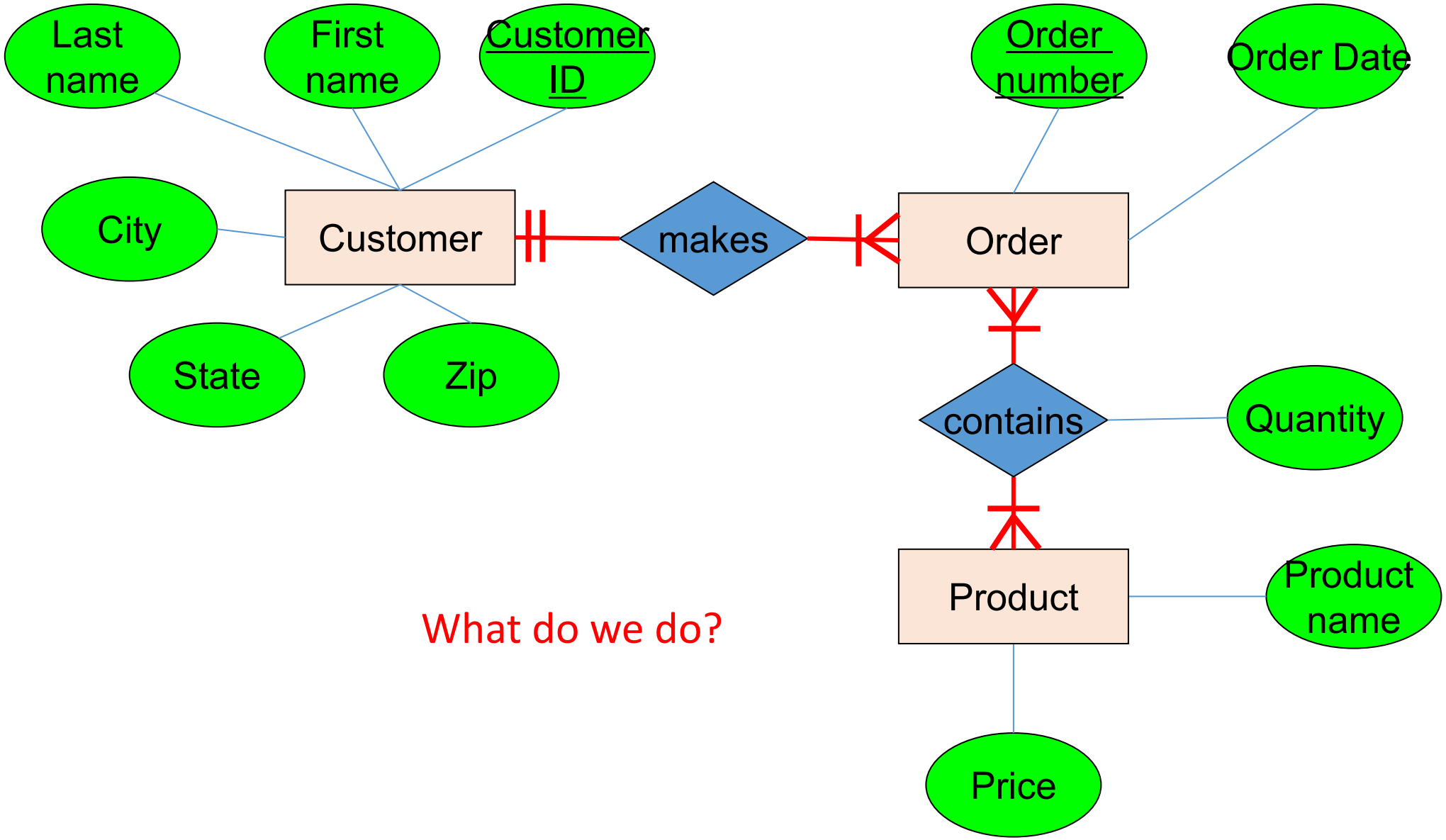


# From E/R Diagrams to Relational Schema

- Key concept
  - Entity sets become relations, Relationships can become relations (tables in RDBMS)
  - Tables are connected with foreign key constraints

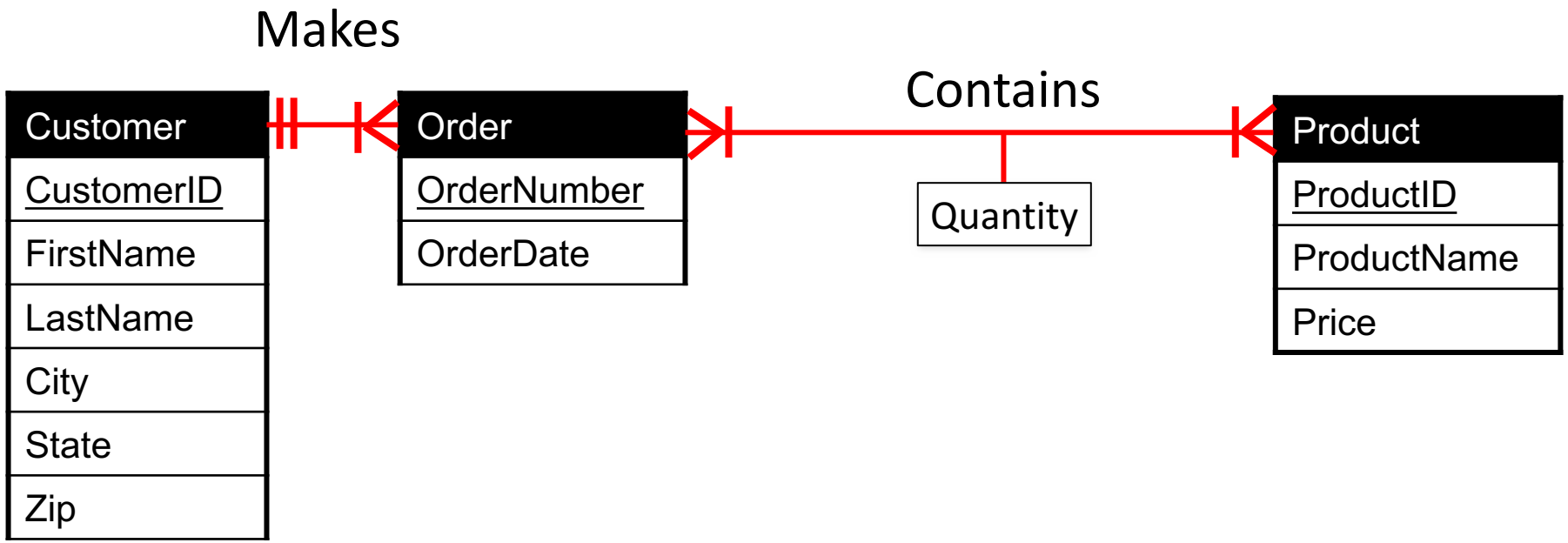
- A database schema
  - A map of the tables and fields (attributes) in the database
  - This is what is implemented in the database management system
  - Part of the “design” process

# Example: translate this ERD v1 into tables



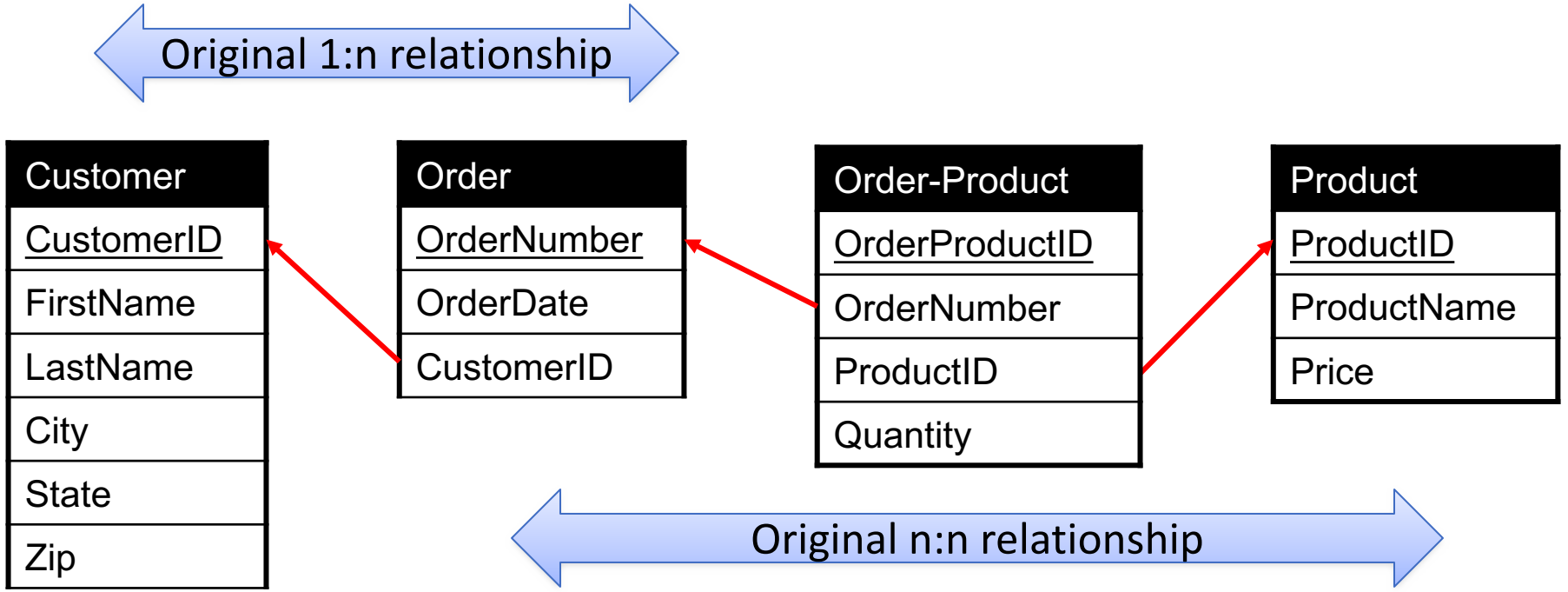

What do we do?

# Example: translate this ERD v2 into tables




What do we do?

# Example: Our Order Database schema



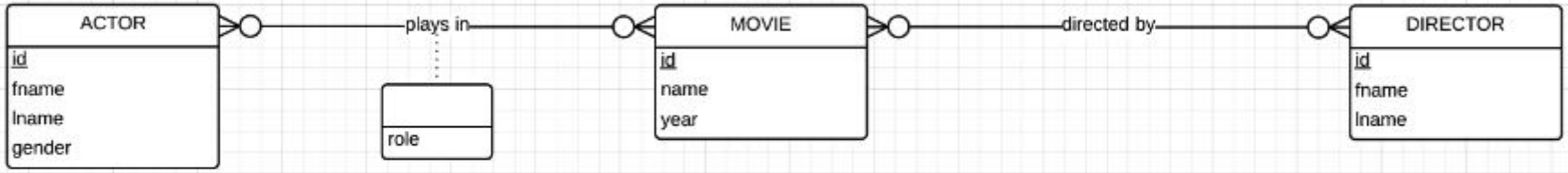

- Order-Product is a decomposed many-to-many relationship
  - Order-Product has a 1:n relationship with Order and Product
  - Now an order can have multiple products, and a product can be associated with multiple orders

# The Rules

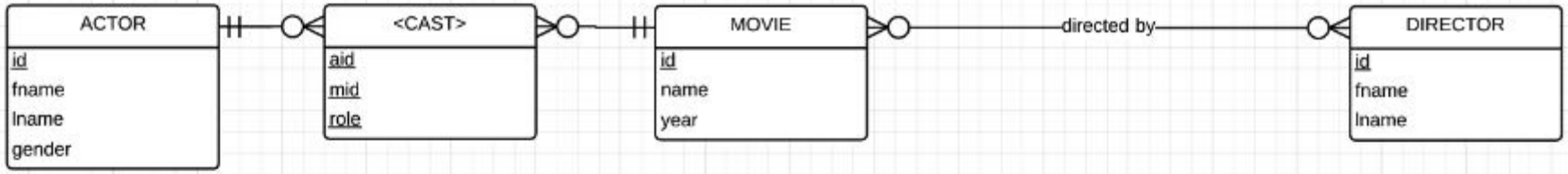
- Create a table for every **entity**
- Create table fields for every entity's **attributes**
- Implement **relationships** between the tables
  - 1:1 relationships: primary key field of one table put into other table as foreign key field
  - 1:many relationships: primary key field of "1" table put into "many" table as foreign key field
  - many:many relationships:
    - Create new table!
    - 1:many relationships with original table

# CAST in our IMDB movie database

ER diagram: don't forget identifiers, but no FKs




ER diagram: CAST as associative entity can be justified




Relational schema: don't forget PKs and FKs