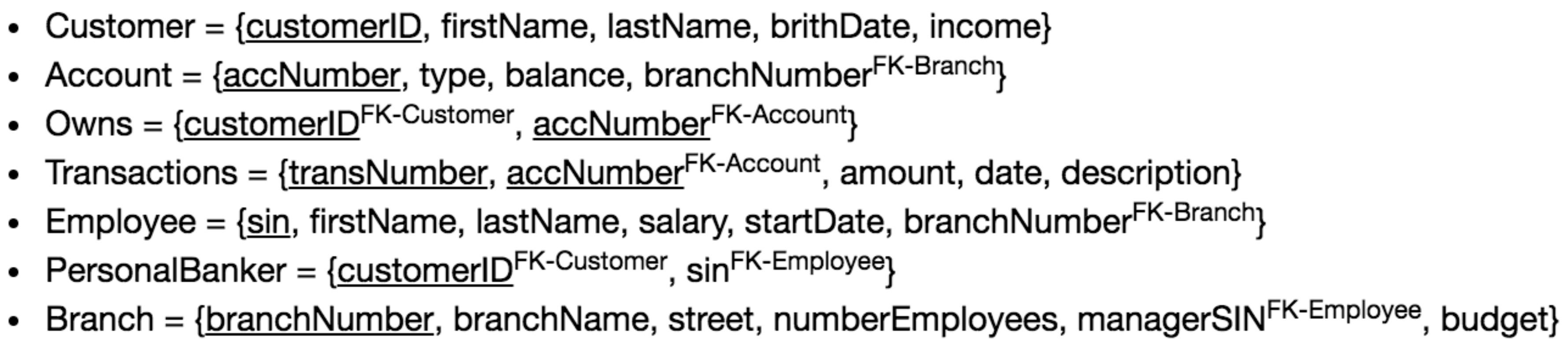
CMPT 354: Database System I

Lecture 8. The E/R Model

Motivation

- How to figure out this **database design**?

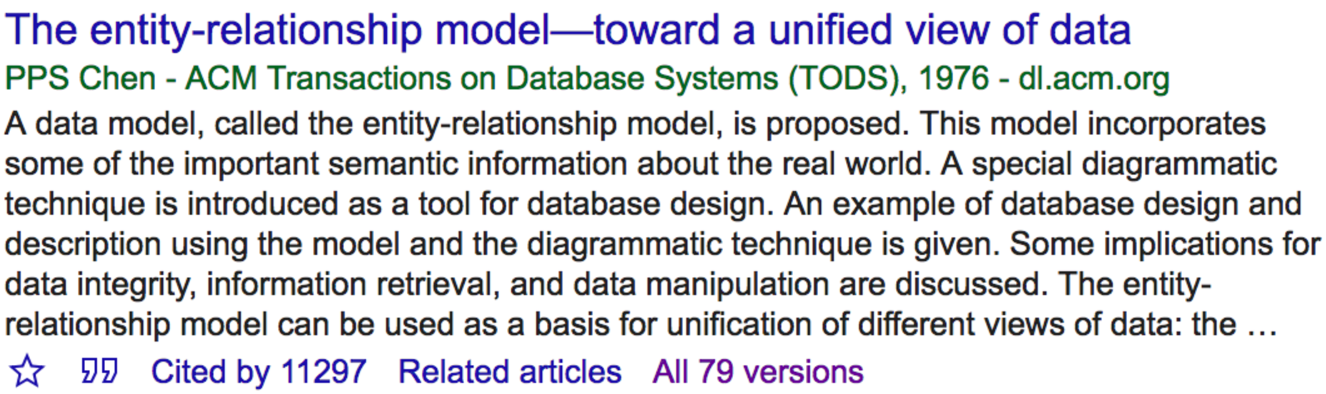


- Customer = {customerID, firstName, lastName, brithDate, income}
- Account = {accNumber, type, balance, branchNumber$^{\text{FK-Branch}}$}
- Owns = {customerID$^{\text{FK-Customer}}$, accNumber$^{\text{FK-Account}}$}
- Transactions = {transNumber, accNumber$^{\text{FK-Account}}$, amount, date, description}
- Employee = {sin, firstName, lastName, salary, startDate, branchNumber$^{\text{FK-Branch}}$}
- PersonalBanker = {customerID$^{\text{FK-Customer}}$, sin$^{\text{FK-Employee}}$}
- Branch = {branchNumber, branchName, street, numberEmployees, managerSIN$^{\text{FK-Employee}}$, budget}

- What **tables** to create?
- Which **attributes** should be added to each table?
- What are the **relationships** between the tables?

History of E/R Model

- E/R Model (Entity-Relationship Modeling)
  - Codd wrote a long letter criticizing paper
  - Many people suggested him to give up this idea



The entity-relationship model—toward a unified view of data

PPS Chen - ACM Transactions on Database Systems (TODS), 1976 - dl.acm.org

A data model, called the entity-relationship model, is proposed. This model incorporates some of the important semantic information about the real world. A special diagrammatic technique is introduced as a tool for database design. An example of database design and description using the model and the diagrammatic technique is given. Some implications for data integrity, information retrieval, and data manipulation are discussed. The entity-relationship model can be used as a basis for unification of different views of data: the …

☆ ❞ Cited by 11297 Related articles All 79 versions

Dr. Peter Chen

- Why not build RDBMS based on E/R Model?
  - No query language proposed
  - Relational DBMS in the 1970's

Outline

- E/R Basics: Entities & Relationships

- E/R Design considerations

- Advanced E/R Concepts

Outline

- **E/R Basics: Entities & Relationships**
  - **Database Design**
  - **Entities/Entity sets/Keys/Relationships**

- E/R Design considerations

- Advanced E/R Concepts

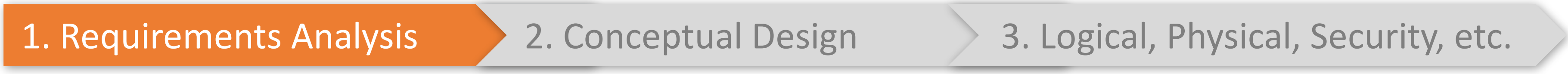
Database Design

- **Database design: Why do we need it?**
  - Agree on structure of the database before deciding on a particular implementation

- **Consider issues such as:**
  - What entities to model
  - How entities are related
  - What constraints exist in the domain
  - How to achieve <u>good</u> designs

- **Several formalisms exist**
  - We discuss one flavor of E/R diagrams

Database Design Process


1. Requirements Analysis

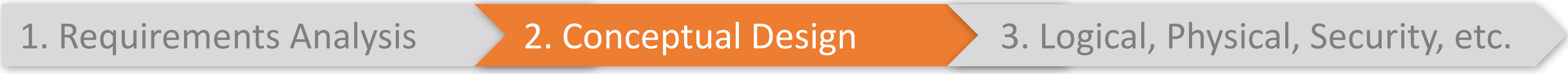
2. Conceptual Design

3. Logical, Physical, Security, etc.

1. **Requirements analysis**
   - What data is going to be stored?
   - What are we going to do with the data?
   - Who should access the data?

Technical and non-technical people are involved

Database Design Process



1. Requirements Analysis

2. Conceptual Design

3. Logical, Physical, Security, etc.

2. Conceptual Design

- A high-level description of the database
- Sufficiently precise that technical people can understand it
- But, not so precise that non-technical people can't participate

This is where E/R fits in.

Database Design Process

1. Requirements Analysis

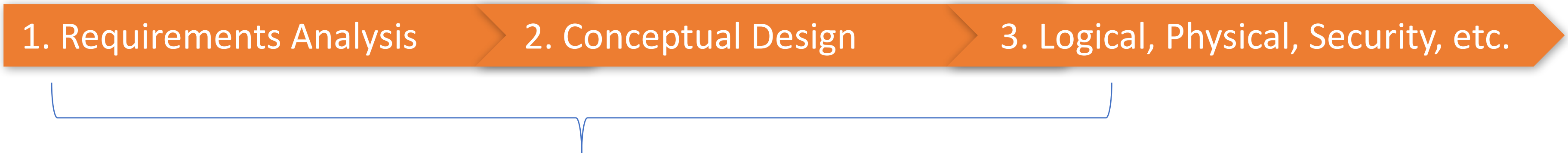
2. Conceptual Design

3. Logical, Physical, Security, etc.

3. More:

- Logical Database Design
- Physical Database Design
- Security Design

Database Design Process

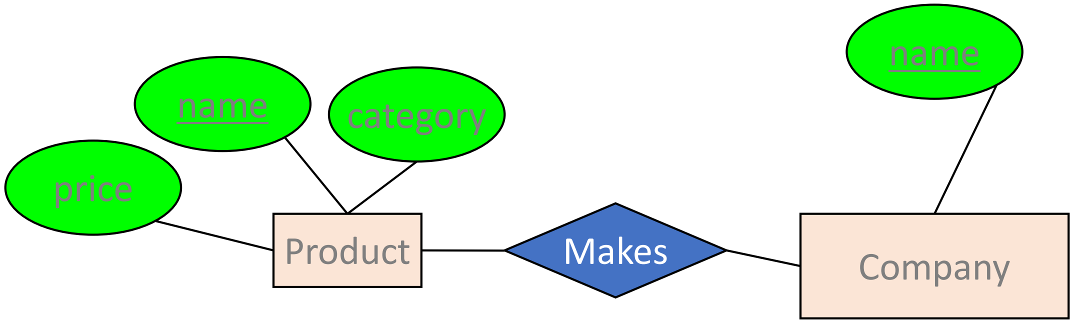


1. Requirements Analysis

2. Conceptual Design

3. Logical, Physical, Security, etc.

E/R Model & Diagrams used

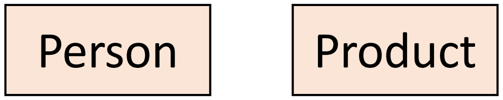


E/R is a *visual syntax* for DB design which is ***precise enough*** for technical points, but ***abstracted enough*** for non-technical people

Entities and Entity Sets

- An entity is an individual object
  - Eg: A specific person or product

- An entity set is a collection of entities of the same type
  - *These are what is shown in E/R diagrams - as rectangles*
  - Eg: Person, Product

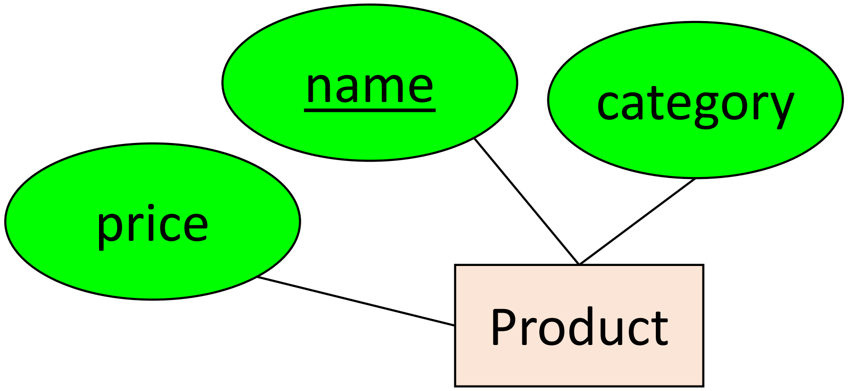


Person

Product

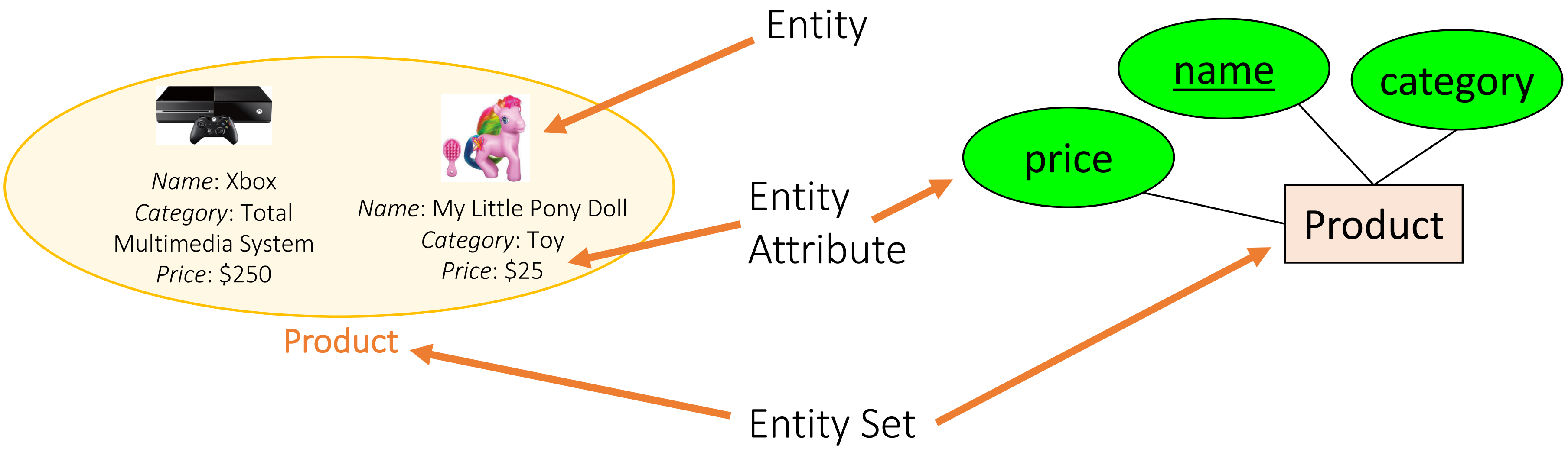
Attributes

- An entity set has **attributes**
  - Represented by ovals attached to an entity set



Example

Entities are **<u>not</u>** explicitly represented in E/R diagrams!



Entity

Name: Xbox
Category: Total Multimedia System
Price: $250

Name: My Little Pony Doll
Category: Toy
Price: $25

Product

Entity Attribute

Entity Set

<u>name</u>

category

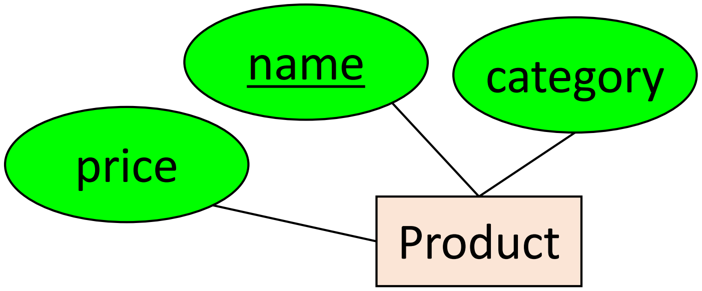
price

Product

Keys

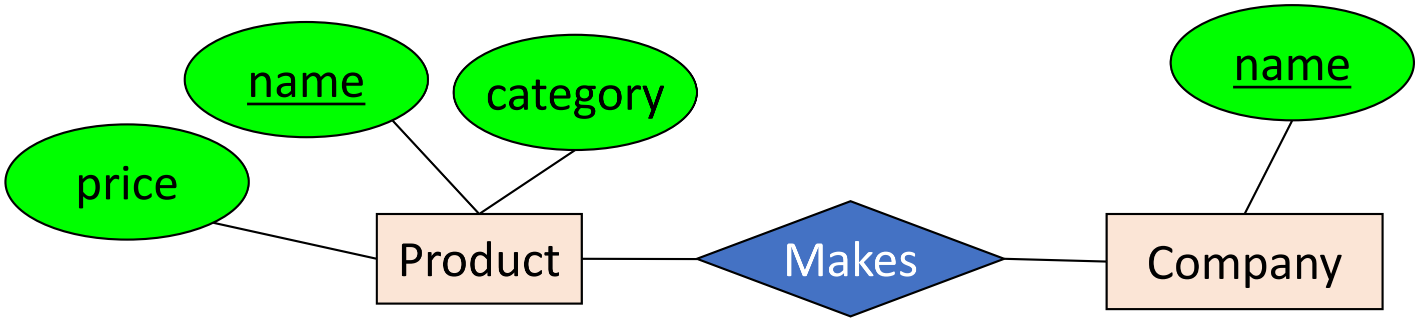
- A *key* is a set of attributes that uniquely identifies an entity.
- Every entity set must have a key

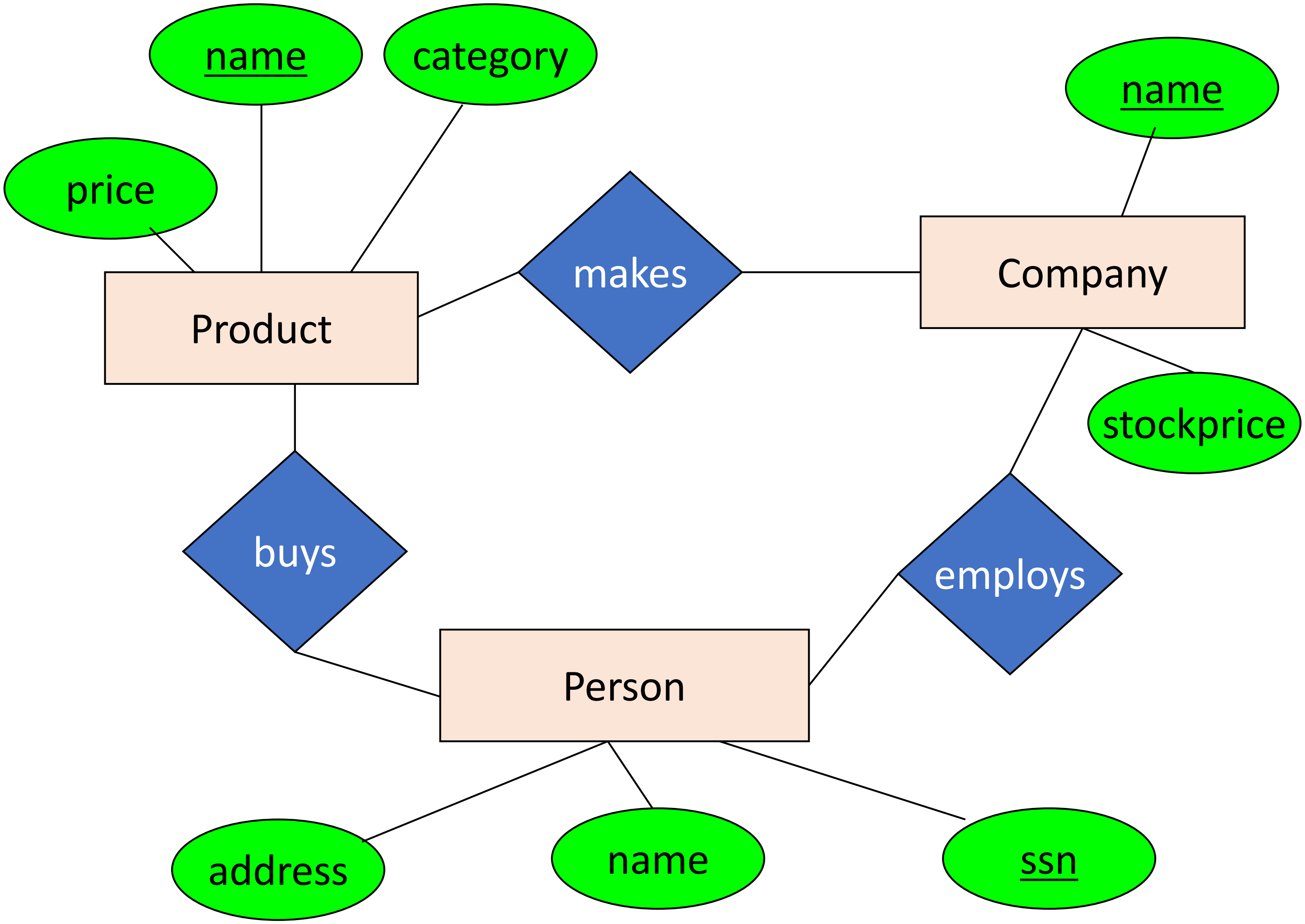
- Denote elements of the primary key by underlining.



The R in E/R: Relationships

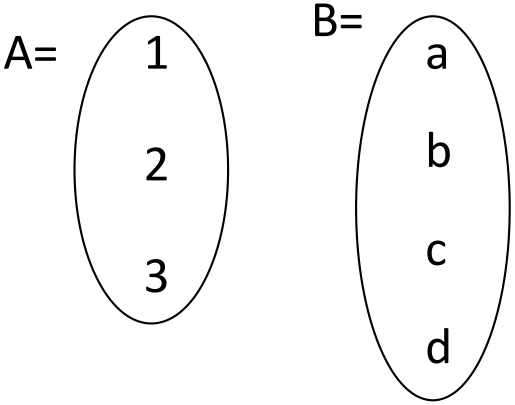
- A **relationship** is between two entities




name
category
price
makes
name
Company
Product
stockprice
buys
employs
Person
address
name
ssn

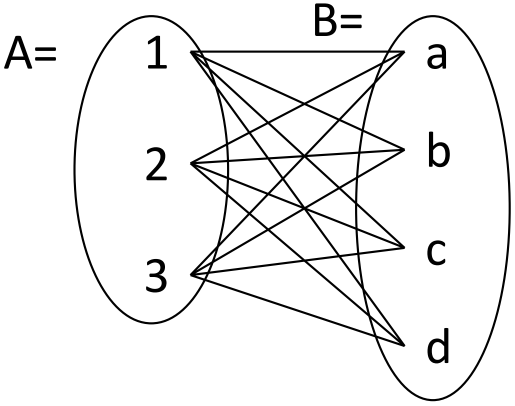
What is a Relationship?

- ***A mathematical definition:***
  - Let A, B be sets
    - *A={1,2,3},   B={a,b,c,d}*



What is a Relationship?

- ***A mathematical definition:***
  - Let A, B be sets
    - *A={1,2,3}, B={a,b,c,d}*
  - A x B (the ***cross-product***) is the set of all pairs (a,b)
    - *$A \times B$ = {(1,a), (1,b), (1,c), (1,d), (2,a), (2,b), (2,c), (2,d), (3,a), (3,b), (3,c), (3,d)}*



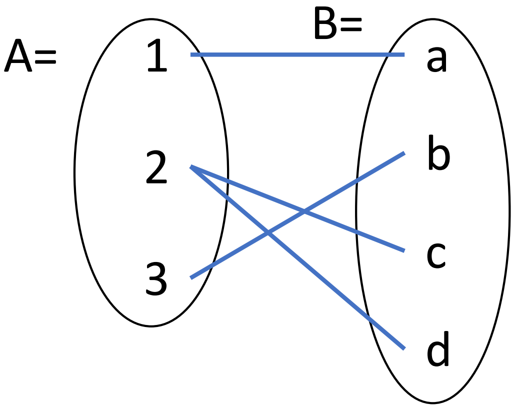
What is a Relationship?

- ***A mathematical definition:***

  - Let A, B be sets
    - *A={1,2,3}, B={a,b,c,d},*

  - A x B (the ***cross-product***) is the set of all pairs (a,b)
    - *A × B = {(1,a), (1,b), (1,c), (1,d), (2,a), (2,b), (2,c), (2,d), (3,a), (3,b), (3,c), (3,d)}*

  - **We define a <u>relationship</u> to be a subset of A x B**
    - *R = {(1,a), (2,c), (2,d), (3,b)}*



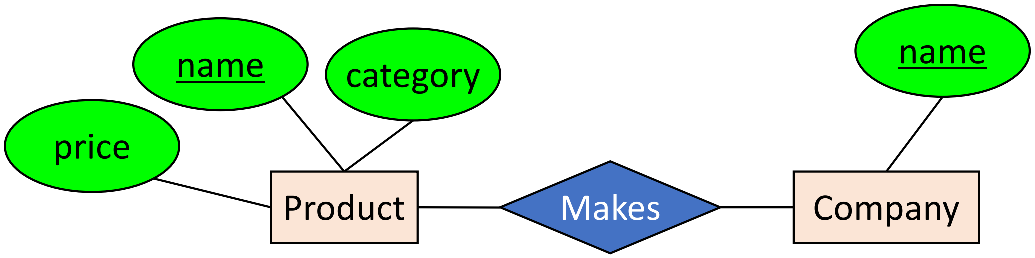
What is a Relationship?

Company

name
Apple
Microsoft

Product

name	category	price
iPhone 8	Electronics	$700
iPad 4	Electronics	$300
Office	Software	$120



A <u>**relationship**</u> between **entity sets P and C** is a *subset of all possible pairs of entities in P and C*, with tuples uniquely identified by *P and C's keys*

What is a Relationship?

Company

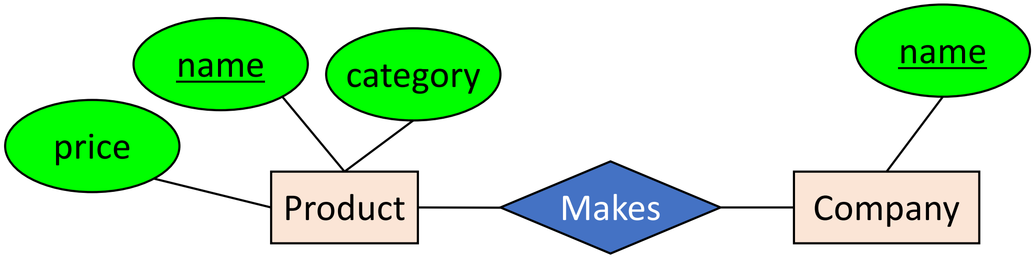
name
Apple
Microsoft

Product

name	category	price
iPhone 8	Electronics	$700
iPad 4	Electronics	$300
Office	Software	$120

Company C × Product P

C.name	P.name	P.category	P.price
Apple	iPhone 8	Electronics	$700
Apple	iPad 4	Electronics	$300
Apple	Office	Toys	$120
Microsoft	iPhone 8	Electronics	$700
Microsoft	iPad 4	Electronics	$300
Microsoft	Office	Toys	$120



A **relationship** between **entity sets P and C** is a ***subset of all possible pairs of entities in P and C***, with tuples uniquely identified by ***P and C's keys***

What is a Relationship?

Company

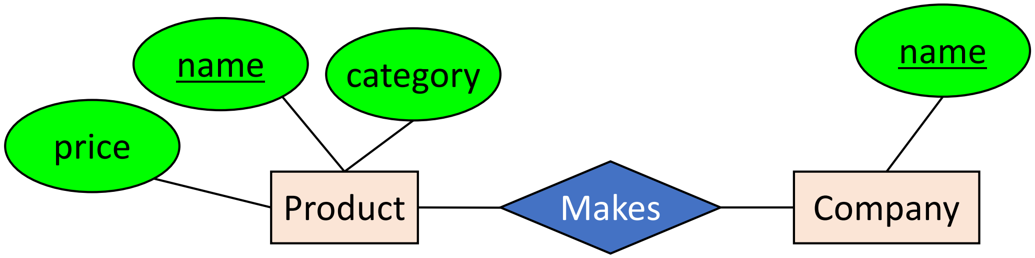
name
Apple
Microsoft

Product

name	category	price
iPhone 8	Electronics	$700
iPad 4	Electronics	$300
Office	Software	$120

Company C × Product P

C.name	P.name	P.category	P.price
Apple	iPhone 8	Electronics	$700
Apple	iPad 4	Electronics	$300
Apple	Office	Software	$120
Microsoft	iPhone 8	Electronics	$700
Microsoft	iPad 4	Electronics	$300
Microsoft	Office	Software	$120



Makes

C.name	P.name
Apple	iPhone 8
Apple	iPad 4
Microsoft	Office

A **relationship** between **entity sets P and C** is a ***subset of all possible pairs of entities in P and C,*** with tuples uniquely identified by ***P and C's keys***

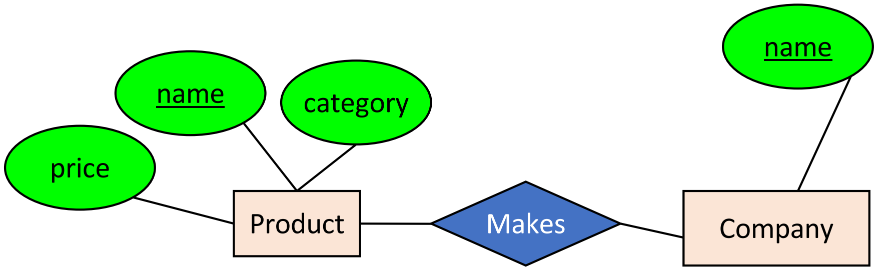
What is a Relationship?

- There can only be **one relationship for every unique combination of entities**

This follows from our mathematical definition of a relationship- it's a SET!

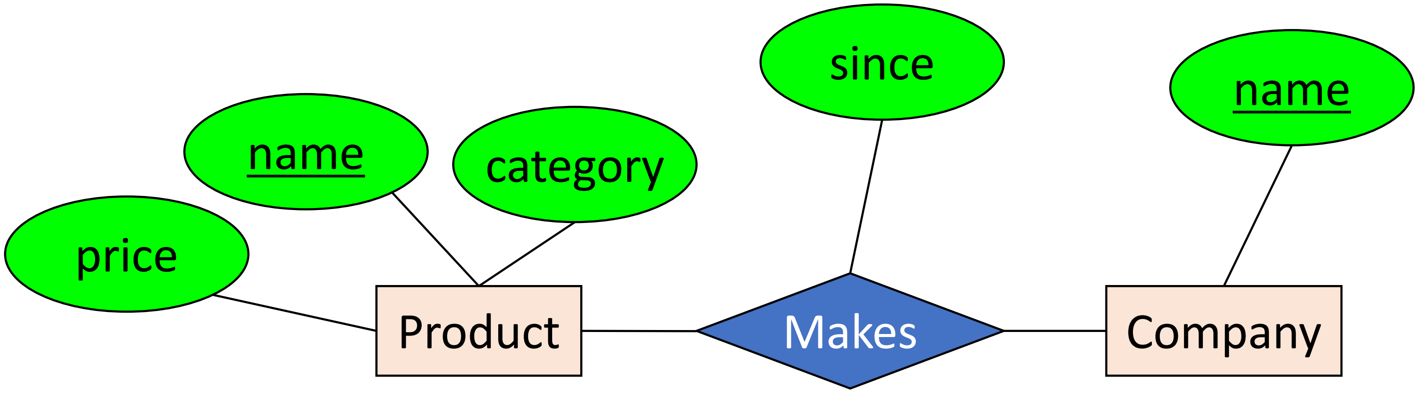
- This also means that **the relationship is uniquely determined by the keys of its entities**

- Example: the "key" for Makes (to right) is {Product.name, Company.name}



Relationships and Attributes

- Relationships may have attributes as well.



For example: “since” records when company started making a product

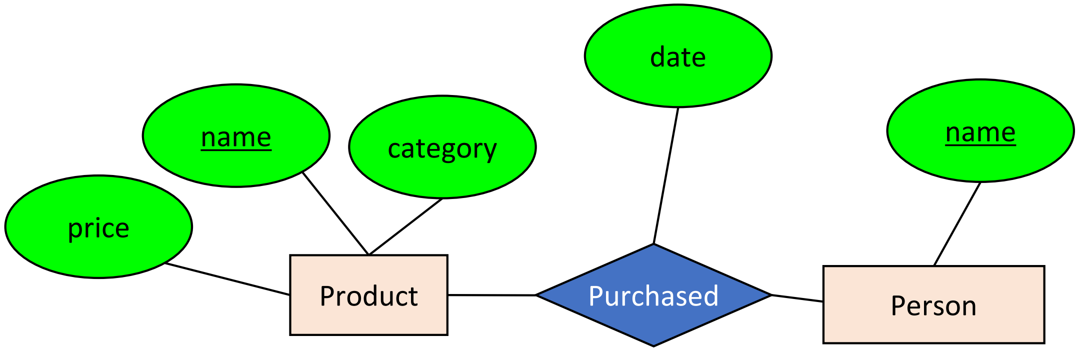
Makes

C.name	P.name	Since
Apple	iPhone 8	2018.09.01
Apple	iPhone 8	2017.09.01

✗

Decision: Relationship vs. Entity?

- **Q:** What does this say?



- **A:** A person can only buy a specific product once

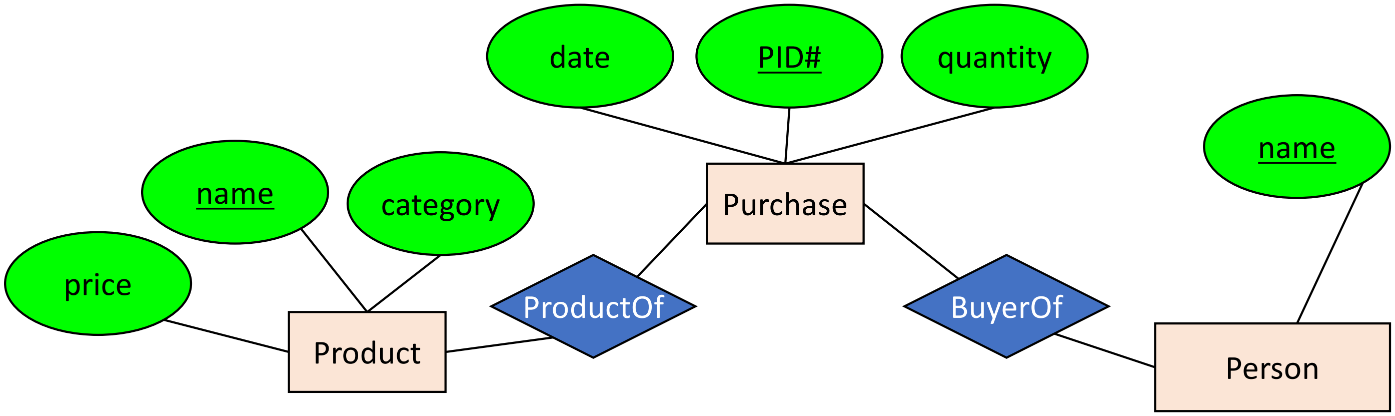
Purchase

Person.name	Product.name	Date
Jiannan	iPhone 8	2018.10.01
Jiannan	iPhone 8	2018.12.01

✗

Decision: Relationship vs. Entity?

- What about this way?



- *Now we can have multiple purchases per product, person pair!*

We can always use **a new entity** instead of a relationship. For example, to permit multiple instances of each entity combination!

Exercise -1

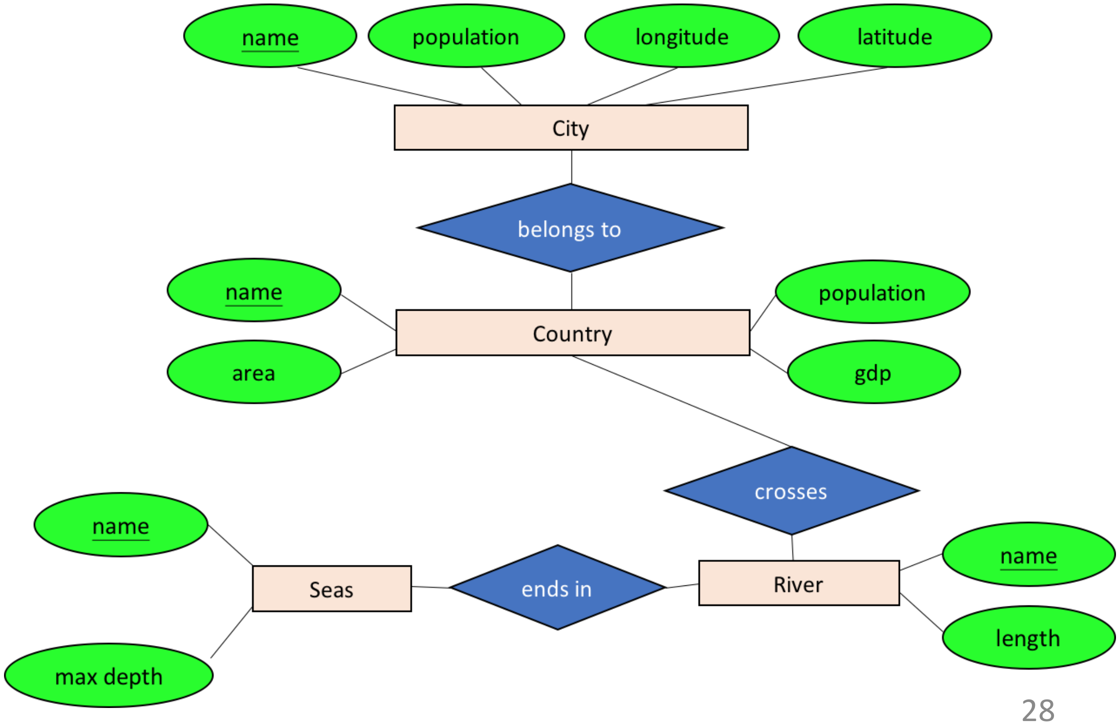
Draw an E/R diagram for geography

Entities

- Country: name, area, population, gdp
- City: name, population, longitude, latitude
- River: name, length
- Sea: name, max depth

Relationships

- City belongs to Country
- River crosses Country
- River ends in Sea



Outline

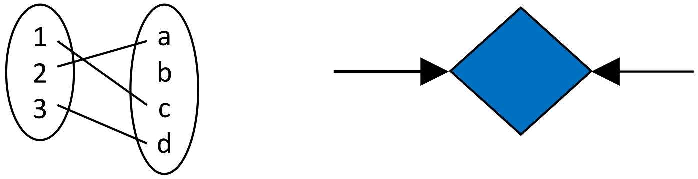
- E/R Basics: Entities & Relationships
  - Database Design
  - Entities/Entity sets/Keys/Relationships

- **E/R Design considerations**
  - **Relationships cond’s: multiplicity, multi-way**
  - **Design considerations**
  - **Conversion to SQL**

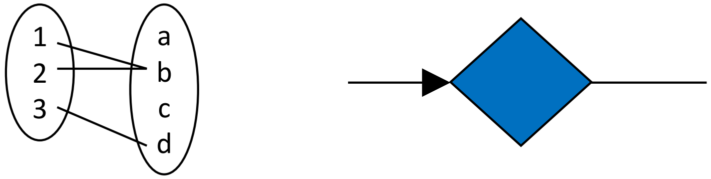
- Advanced E/R Concepts

Multiplicity of E/R Relationships

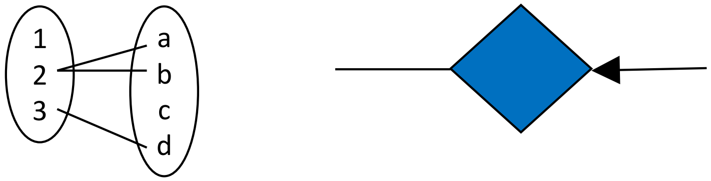
One-to-one:



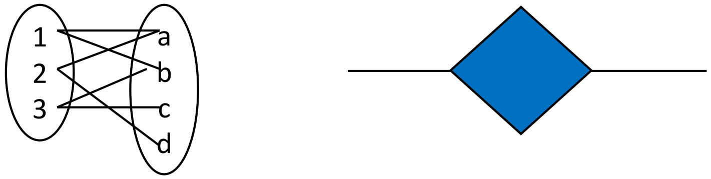
Many-to-one:



One-to-many:

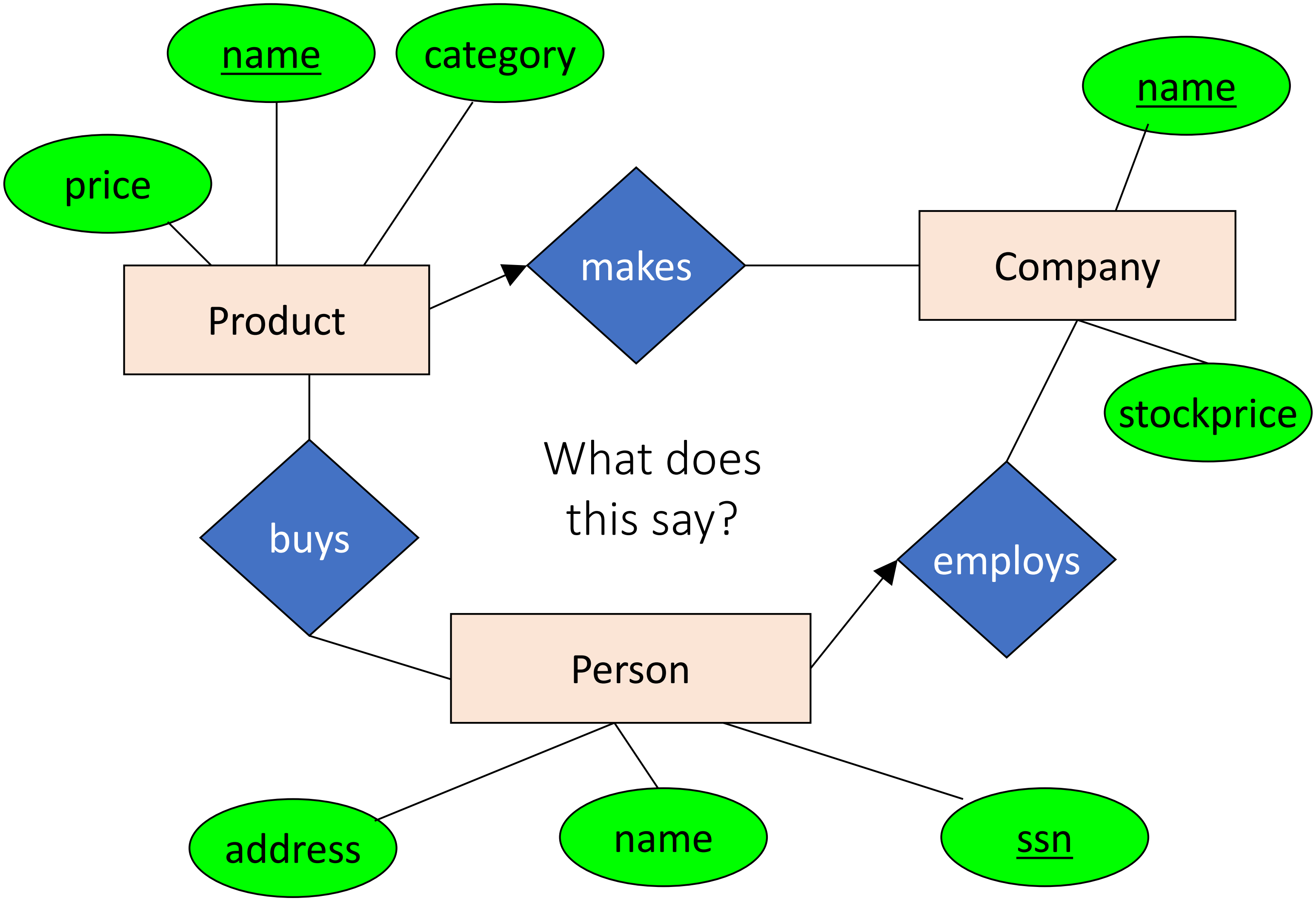


Many-to-many:



Indicated using arrows

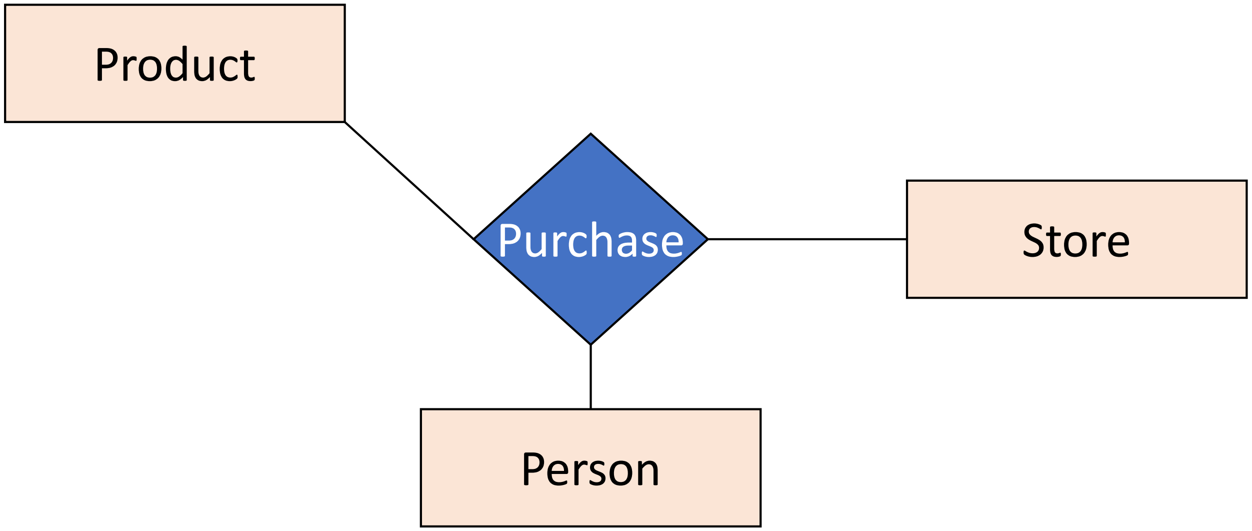
X -> Y means there exists a function mapping from X to Y (*recall the definition of a function)*



What does this say?

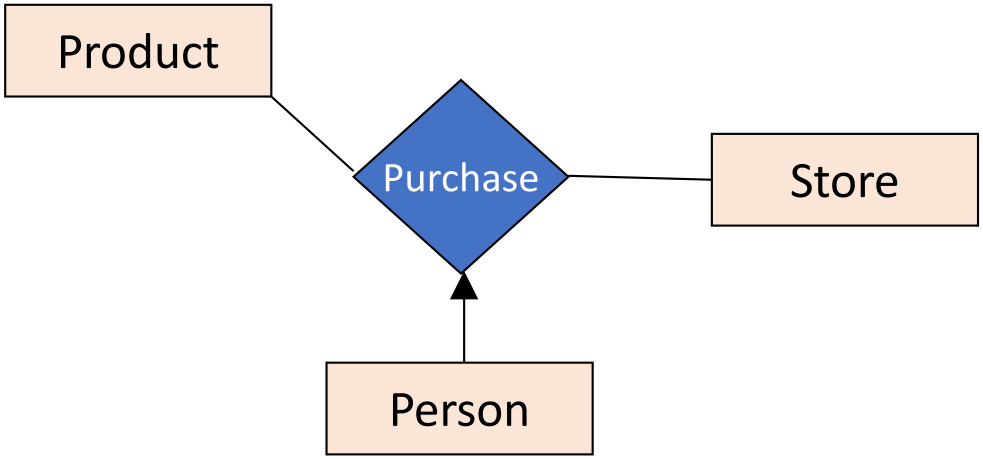
Multi-way Relationships

How do we model a purchase relationship between buyers, products and stores?



Arrows in Multiway Relationships

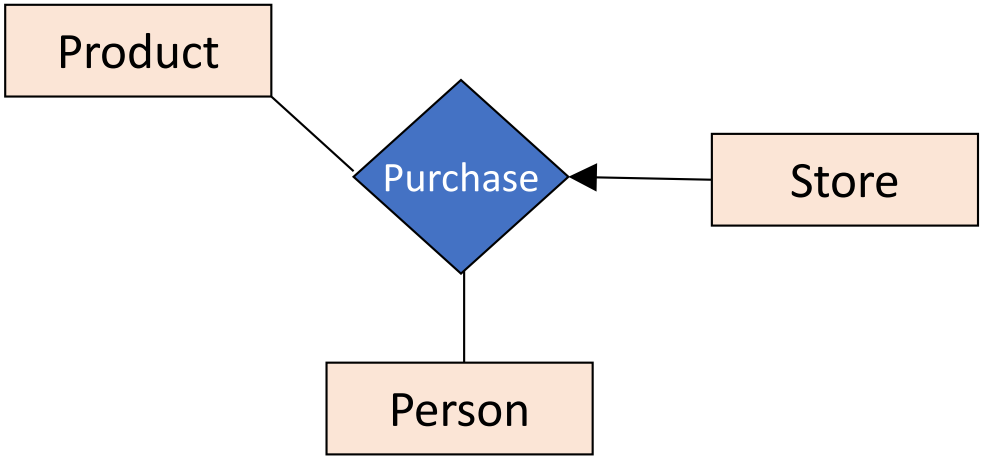
Q: What does the arrow mean ?



given a person, can determine what they bought and the store where they bought it

Arrows in Multiway Relationships

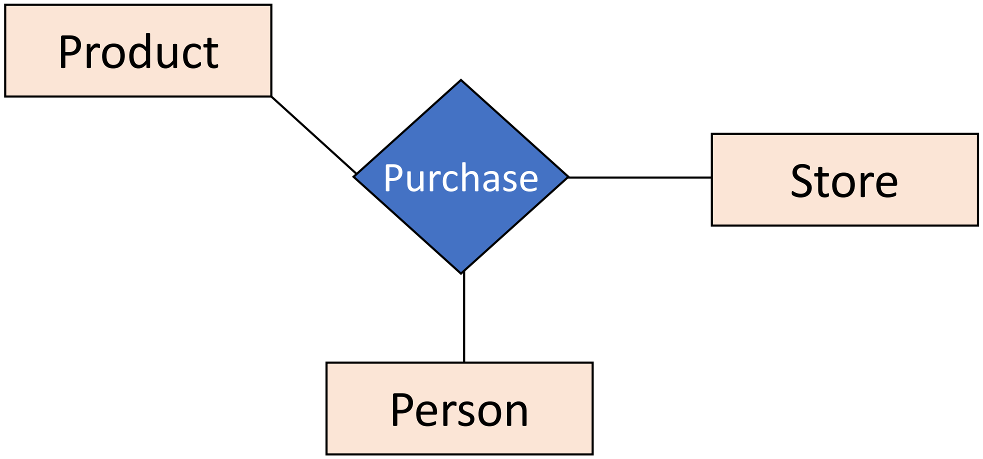
Q: What does the arrow mean ?



given a store, can determine who shopped there and the product they bought
each store sells one product and to one person, ever

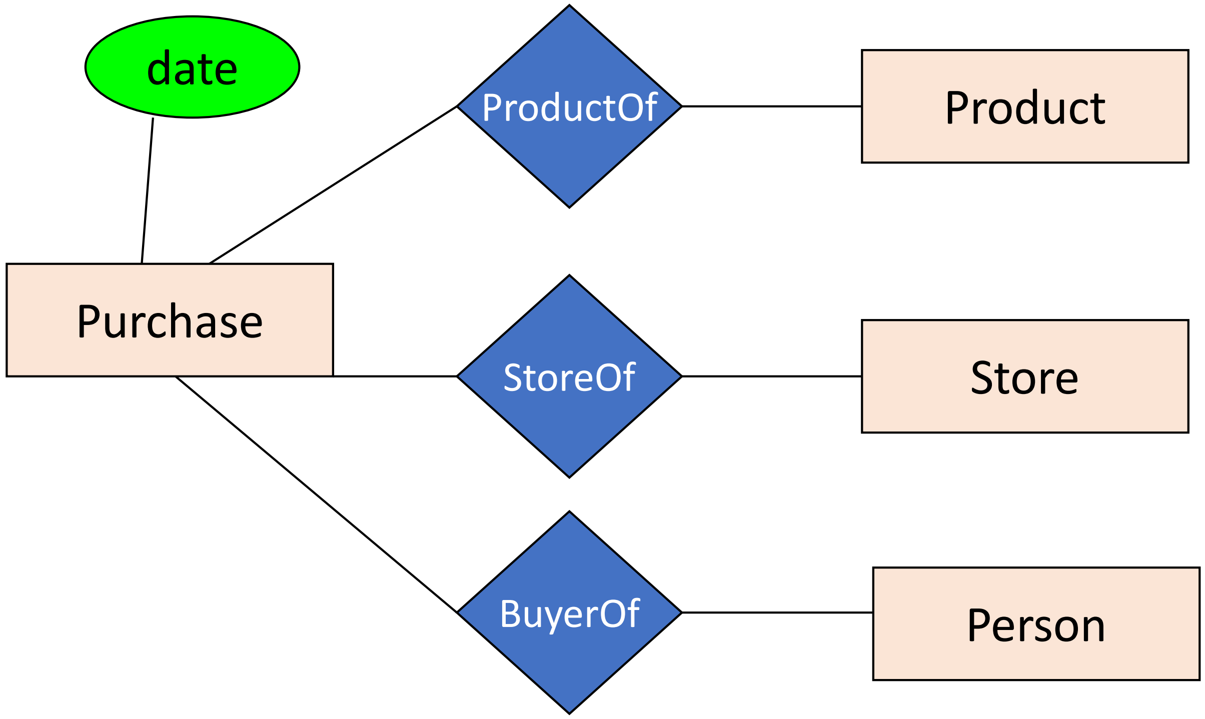
Arrows in Multiway Relationships

Q: How do we say that every person shops in at most one store ?



A: Cannot. This is the best approximation.
(Why only approximation ?)

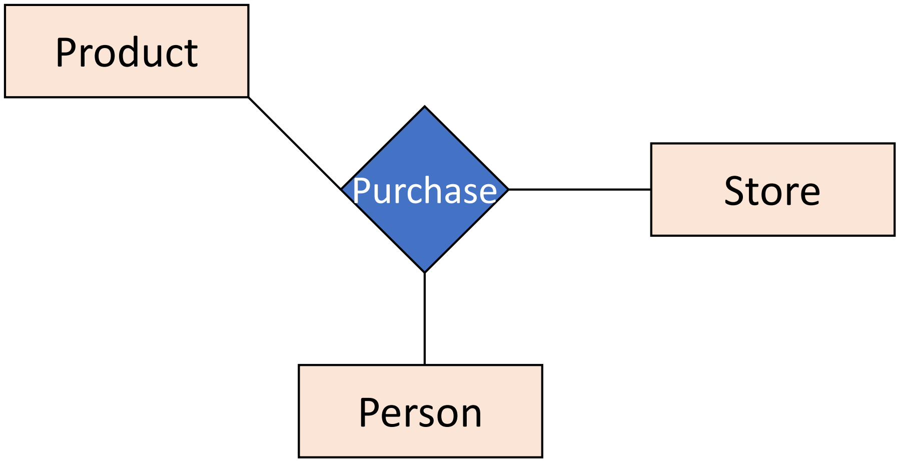
Converting Multi-way Relationships to Binary



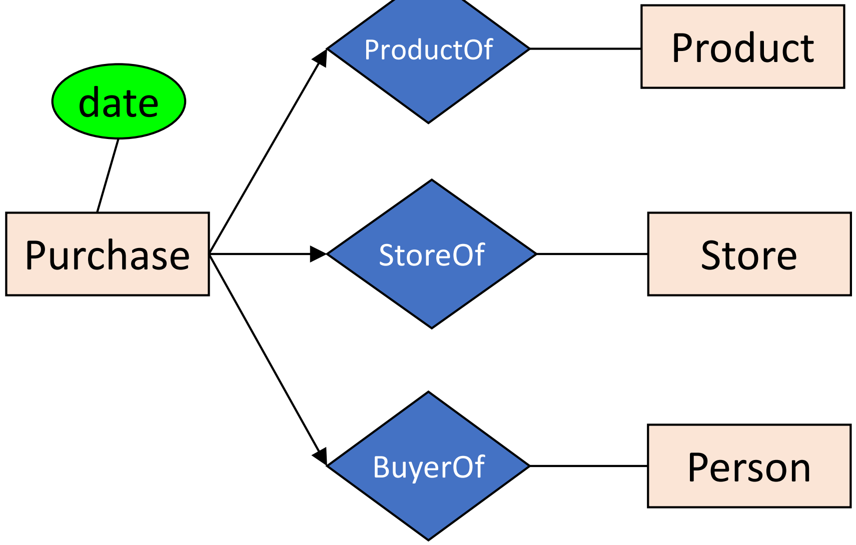
From what we had on previous slide to this - what did we do?

Decision: Multi-way or New Entity + Binary?

Multi-way Relationship



Entity + Binary

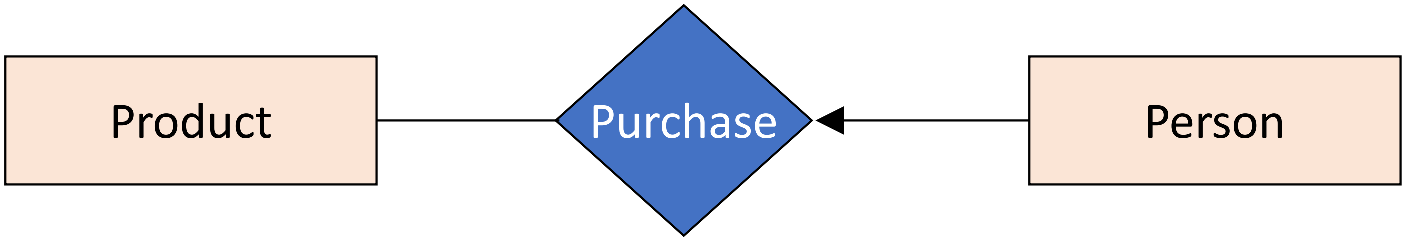


- (B) is also useful when we want to add details (constraints or attributes) to the relationship
  - “A person who shops in at most one store”
  - “How long a person has been shopping at a store”

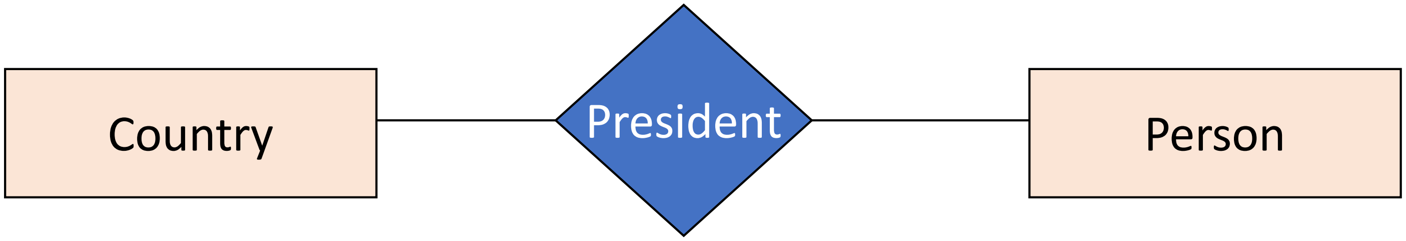
- (A) is useful when a relationship really is between multiple entities
  - *Ex: A three-party legal contract*

Design Principles

What’s wrong with these examples?

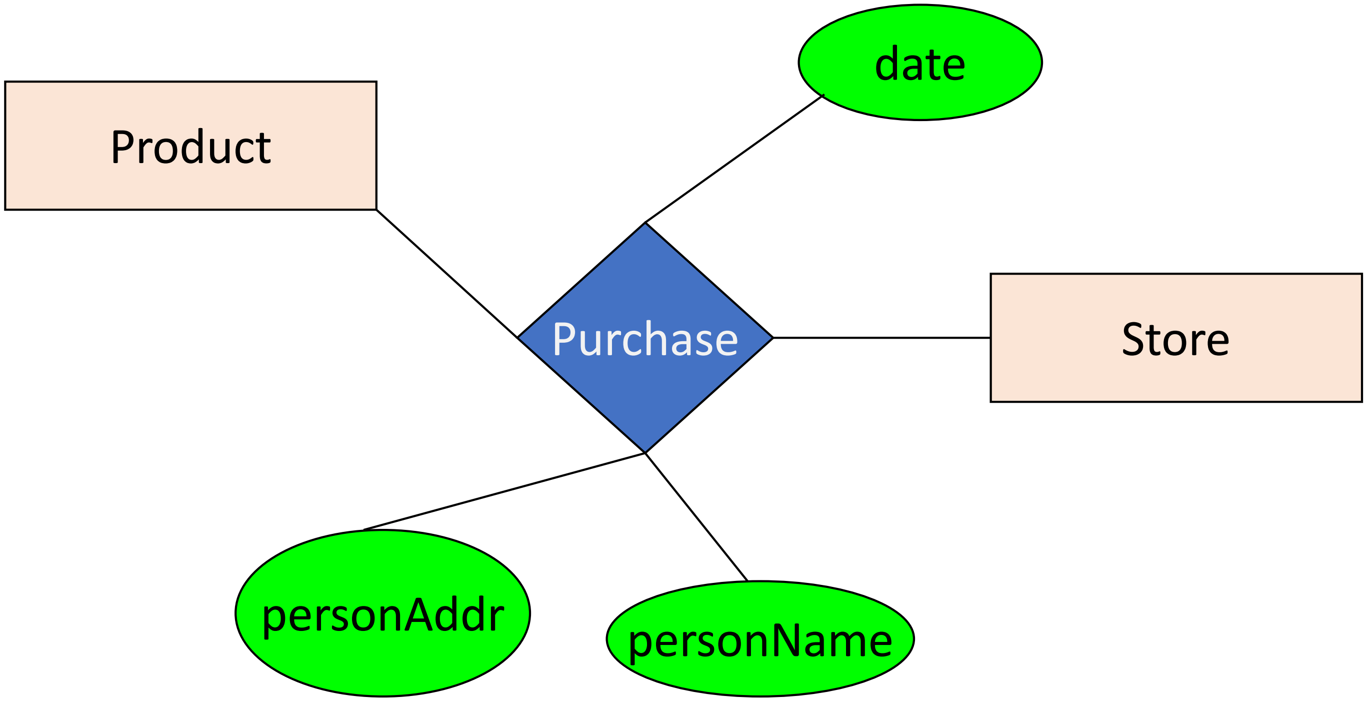


product buys only one product, then out



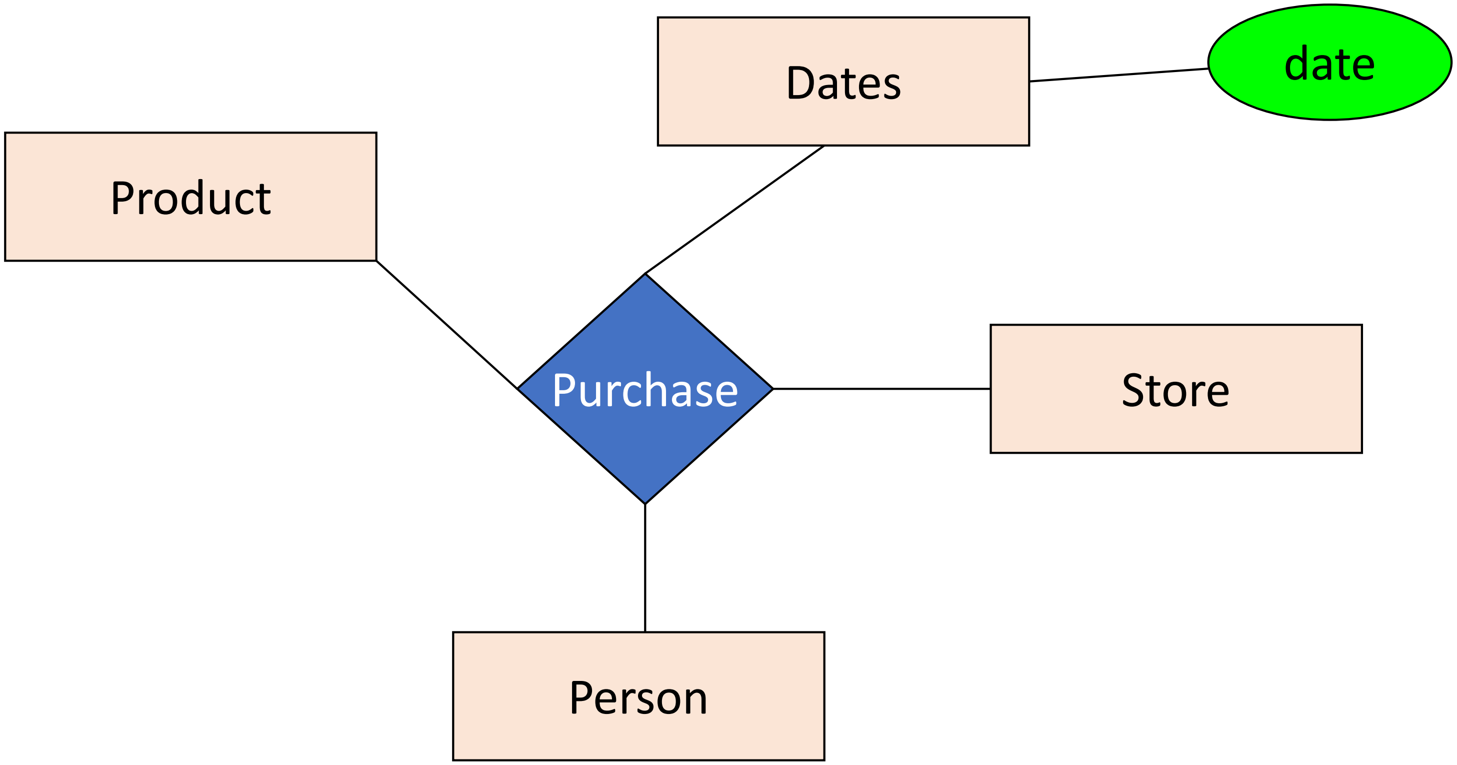
multiple presidents, also may want to require country to have president

Design Principles:
What's Wrong?



maybe people should be entities!

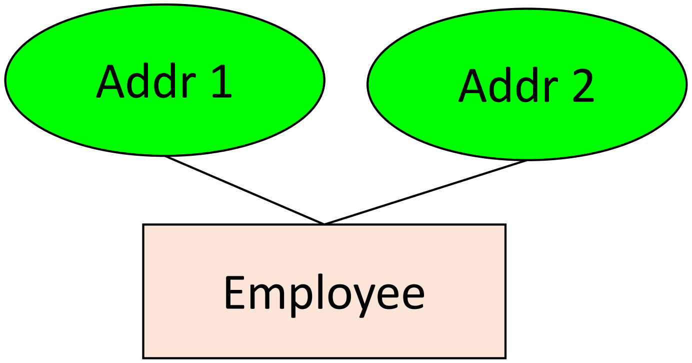
Design Principles: What’s Wrong?



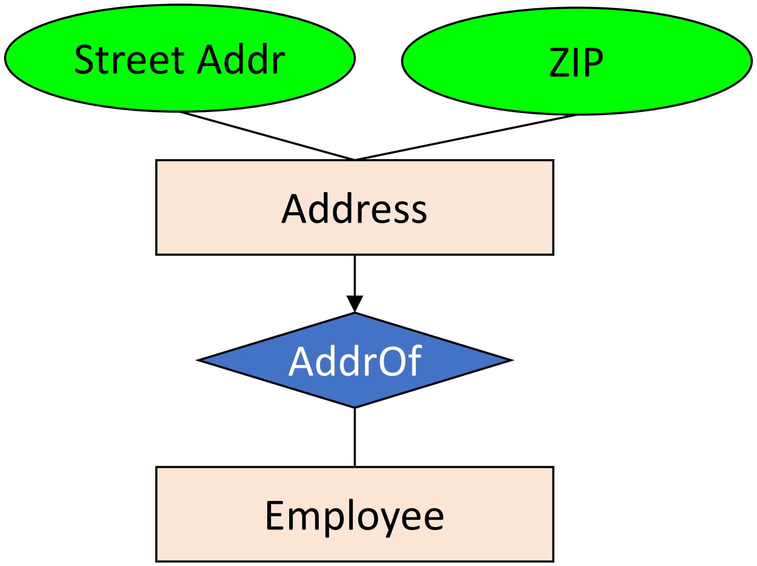
dates don’t need to be an entity by themselves

Examples: Entity vs. Attribute

Should address (A)
be an attribute?

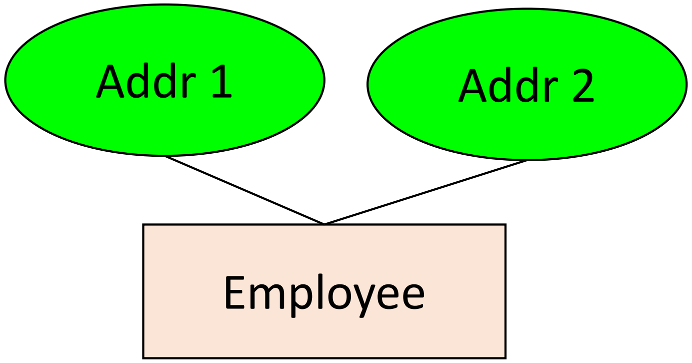


Or (B) be an entity?



Examples: Entity vs. Attribute

Should address (A) be an attribute?

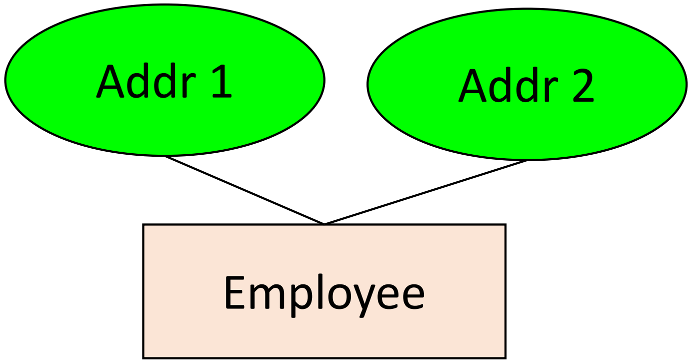


How do we handle employees with multiple addresses here?

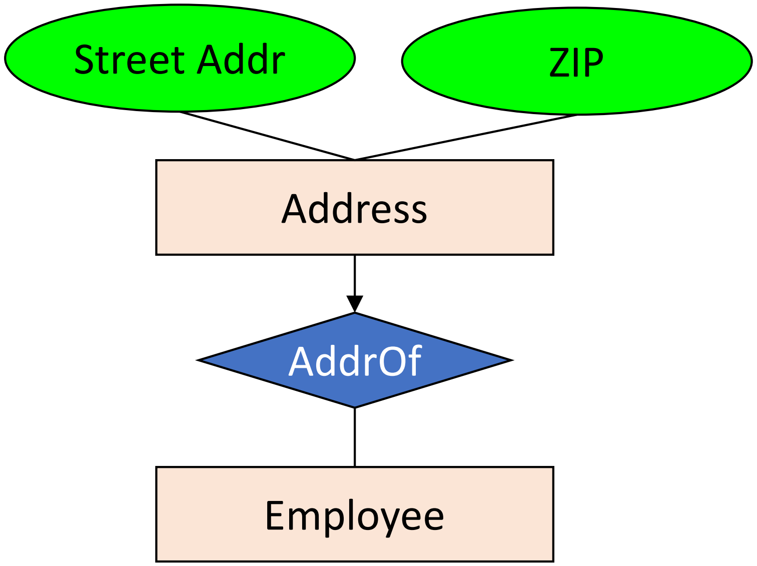
How do we handle addresses where internal structure of the address (e.g. zip code, state) is useful?

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

Exercise -2

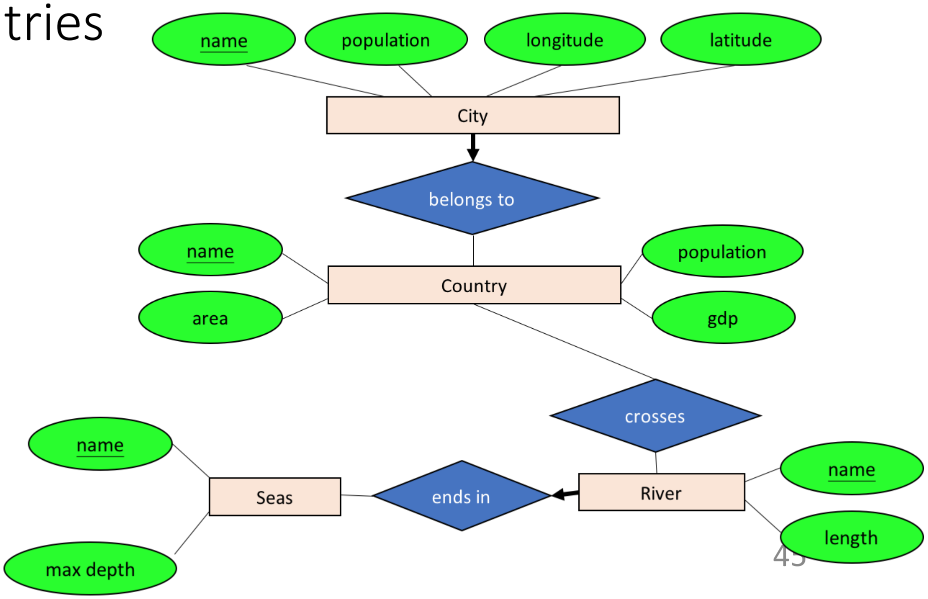
Draw an E/R diagram for geography

Entities

- Country: name, area, population, gdp
- City: name, population, longitude, latitude
- River: name, length
- Sea: name, max depth

Relationships

- Each city belongs to a single country
- Each river crosses one or several countries
- Each river ends in a single sea



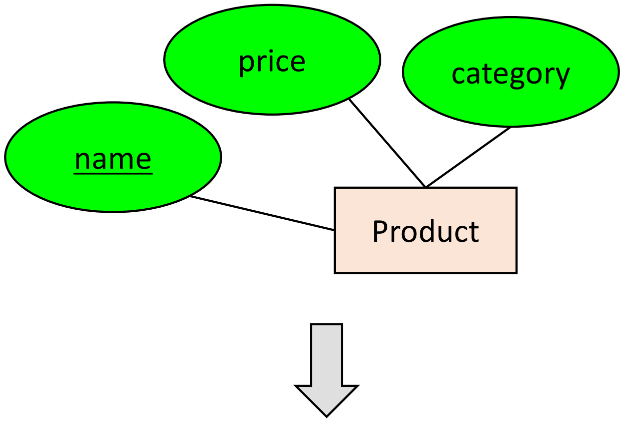
From E/R Diagrams to Relational Schema

- Key concept:

  Both ***Entity sets*** and ***Relationships*** become relations (tables in RDBMS)

From E/R Diagrams to Relational Schema

- An entity set becomes a table
  - Each row is one entity
  - Each row is composed of the entity’s attributes, and has the same primary key



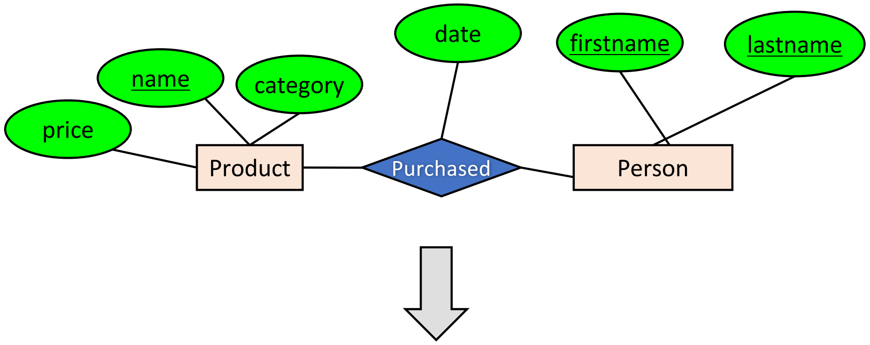
Product

name	price	category
iPhone	700	Electronics
Office	150	Software

```
CREATE TABLE Product(
  name     CHAR(50) PRIMARY KEY,
  price    DOUBLE,
  category VARCHAR(30)
)
```

From E/R Diagrams to Relational Schema

- A relationship *also* becomes a table
  - Add Primary Key
  - Add Foreign Key



```
CREATE TABLE Purchased(
  name      CHAR(50),
  firstname CHAR(50),
  lastname  CHAR(50),
  date      DATE,
  PRIMARY KEY (name, firstname, lastname),
  FOREIGN KEY (name)
        REFERENCES Product,
  FOREIGN KEY (firstname, lastname)
        REFERENCES Person
)
```

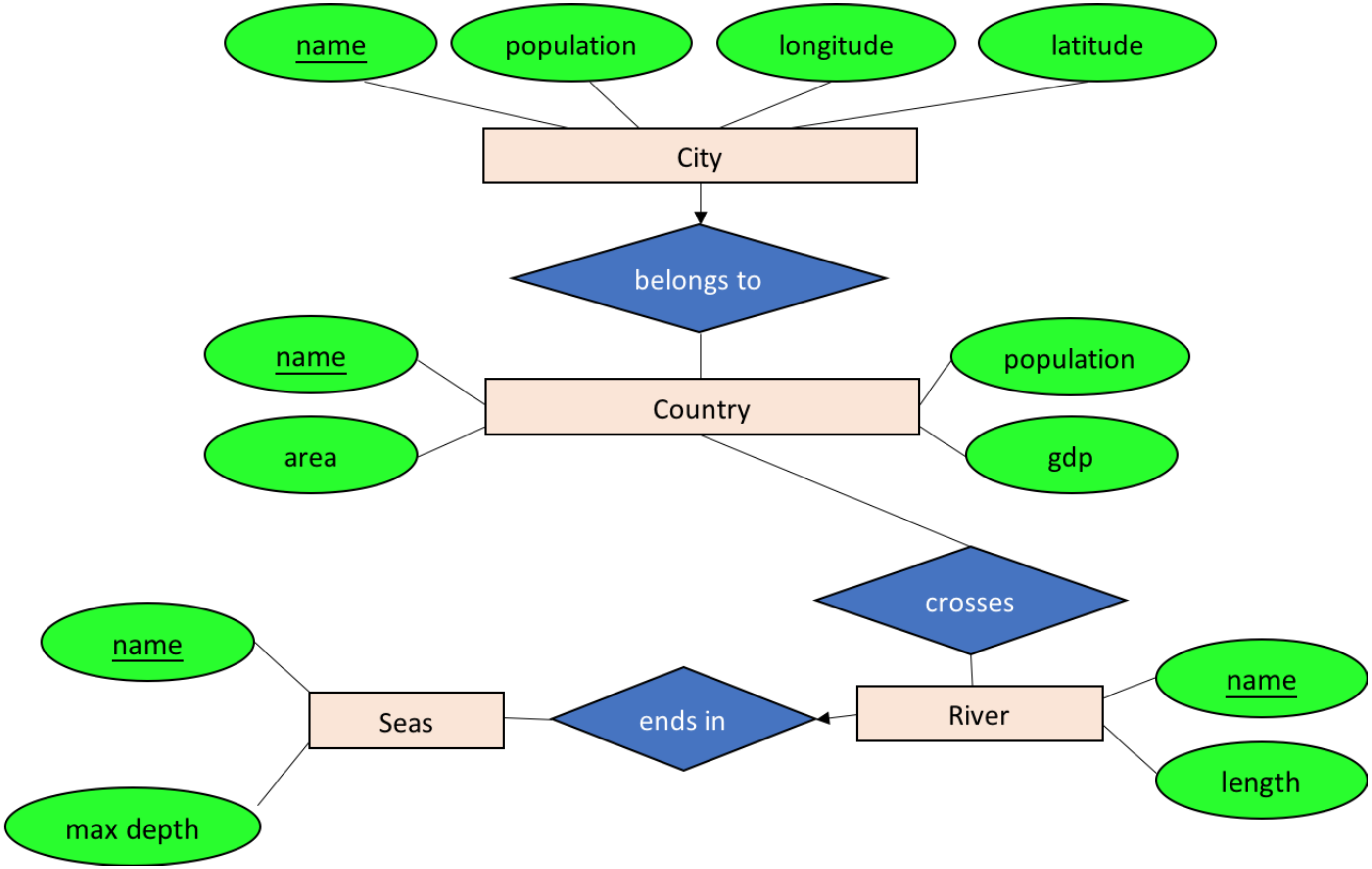
Purchased

name	firstname	lastname	date
iPhone	Mike	Jordan	01/01/18
iPhone	Jiannan	Wang	01/03/18
iPad	John	Smith	01/05/18

Exercise -3

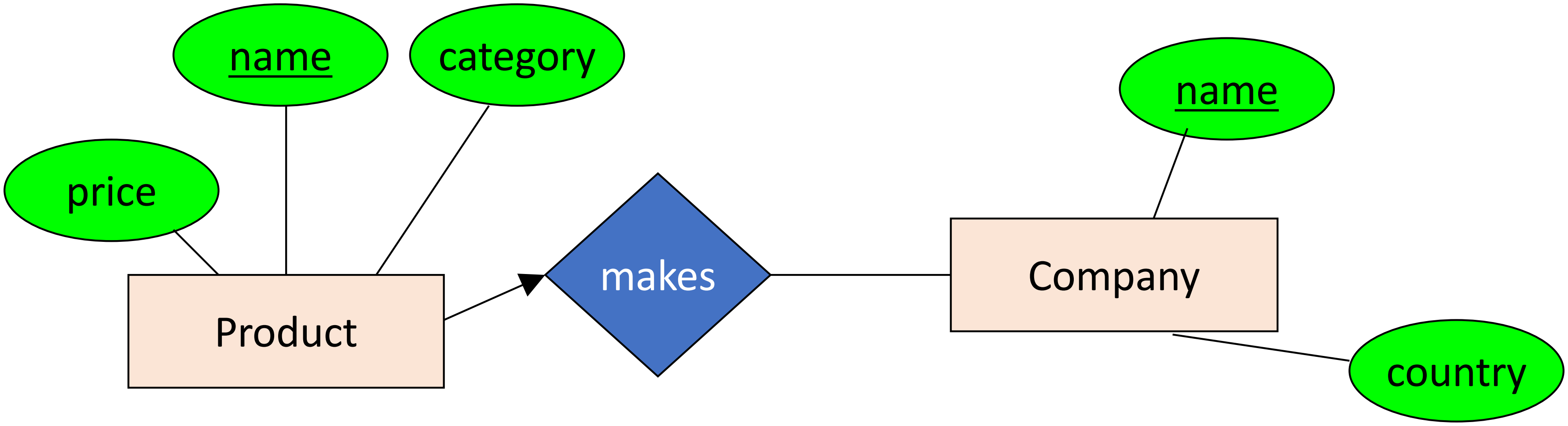
From E/R Diagram to Relational Schema

How do we represent this as a relational schema?



Outline

- **E/R Basics: Entities & Relationships**
    - Database Design
    - Entities/Entity sets/Keys/Relationships
- **E/R Design considerations**
    - Relationships cond’s: multiplicity, multi-way
    - Design considerations
    - Conversion to SQL

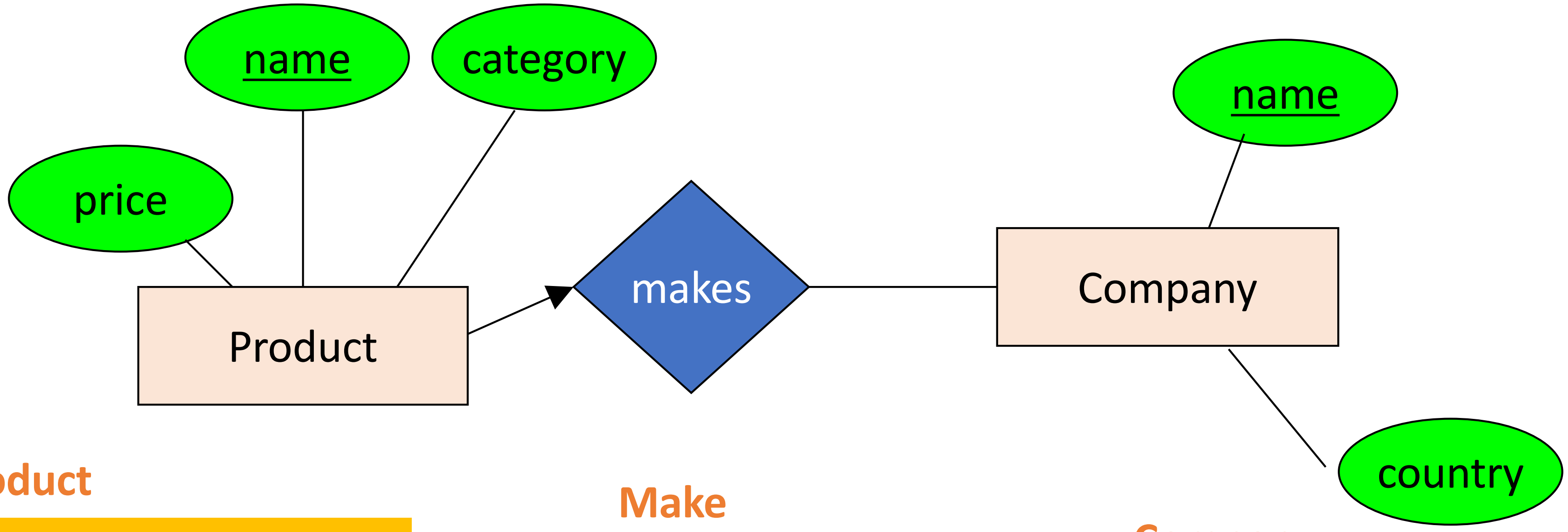


Outline

- E/R Basics: Entities & Relationships
  - Database Design
  - Entities/Entity sets/Keys/Relationships
- E/R Design considerations
  - Relationships cond's: multiplicity, multi-way
  - Design considerations
  - Conversion to SQL
- **Advanced E/R Concepts**
  - Combing Relations
  - Constraints
  - Subclass
  - Weak Entity Sets

Combing Relations

- For many-to-one relationships



Product

name	category	price
iPhone 8	Electronics	$700
iPad 4	Electronics	$300
Office	Software	$120

Make

P.name	C.name
iPhone 8	Apple
iPad 4	Apple
Office	Microsoft

Company

name	country
Apple	US
Microsoft	US

Combing Relations

Product

name	category	price
iPhone 8	Electronics	$700
iPad 4	Electronics	$300
Office	Software	$120

Make

P.name	C.name
iPhone 8	Apple
iPad 4	Apple
Office	Microsoft

Company

name	country
Apple	US
Microsoft	US

P.name	C.name	category	price
iPhone 8	Apple	Electronics	$700
iPad 4	Apple	Electronics	$300
Office	Microsoft	Software	$120

name	country
Apple	US
Microsoft	US

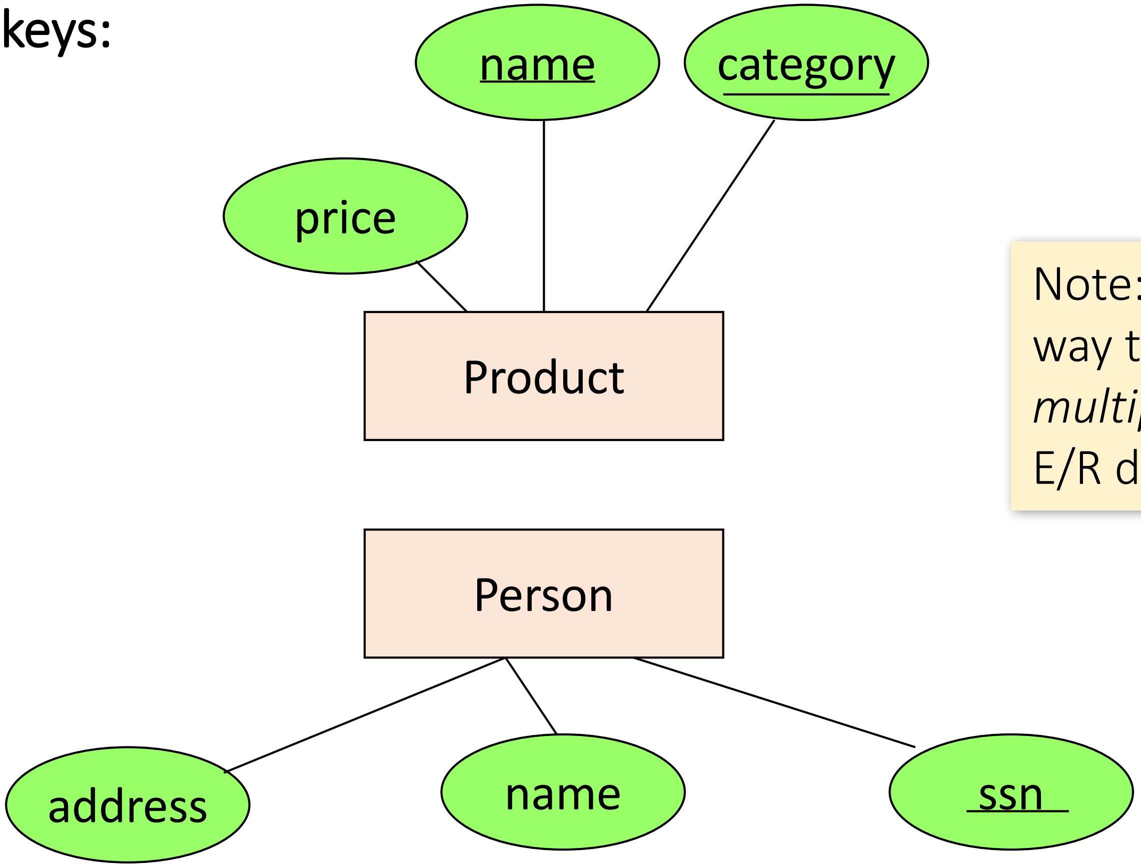
- Remember: no separate relations for many-one relationship

Constraints in E/R Diagrams

- Finding constraints is part of the E/R modeling process. Commonly used constraints are:
  - Keys
    - *Ex: A product name uniquely identifies a product*
  - Single-value constraints:
    - *Ex: a product made by exactly one company*
  - Participation constraints:
    - *Ex: all products are made by a company*

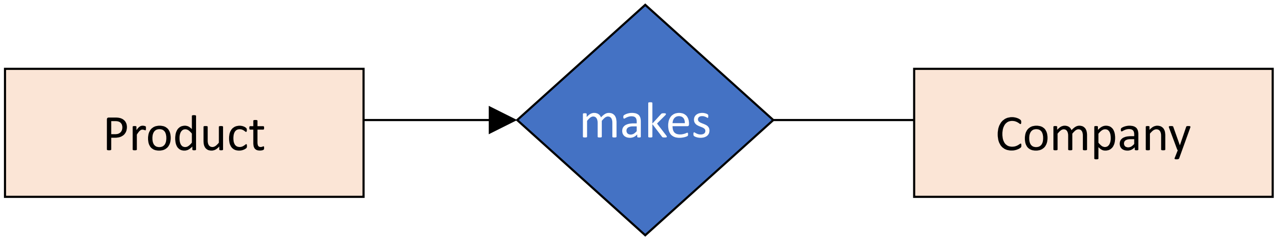
Keys in E/R Diagrams

Underline keys:

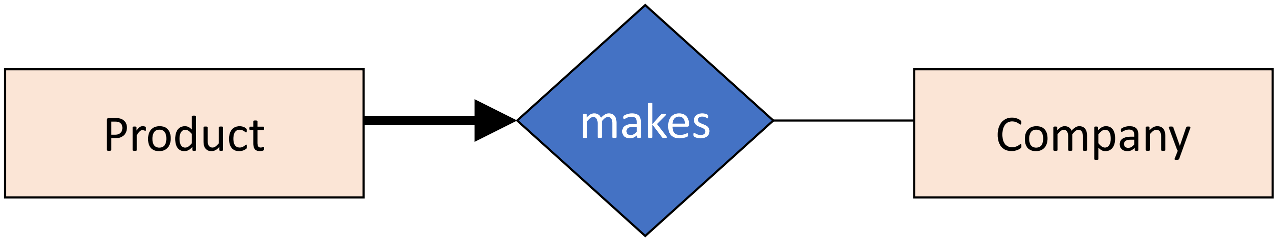


Note: no formal way to specify *multiple* keys in E/R diagrams…

Single-Value Constraints

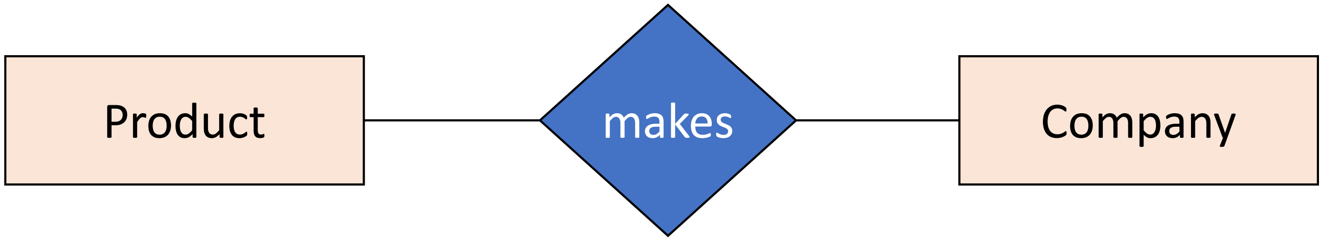


Each product made by at most one company.
Some products made by no company?

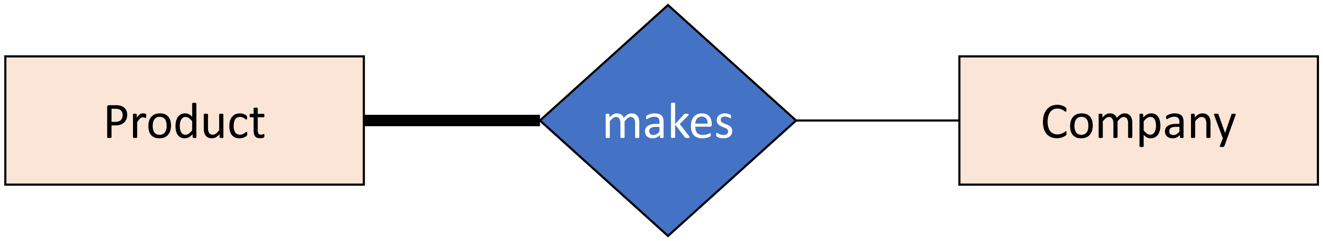


Each product made by *exactly* one company.

Participation Constraints: Partial v. Total



Are there products made by no company?
Companies that don’t make a product?

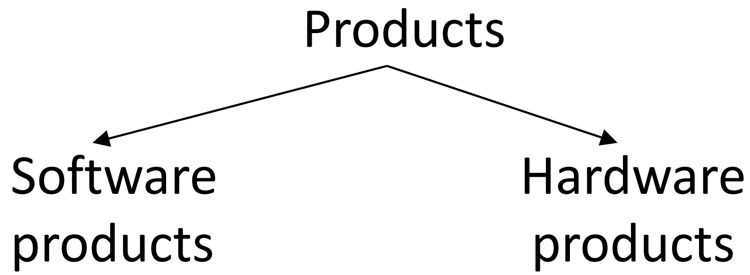


Bold line indicates *total participation* (i.e. here: all products are made by a company)

Modeling Subclasses

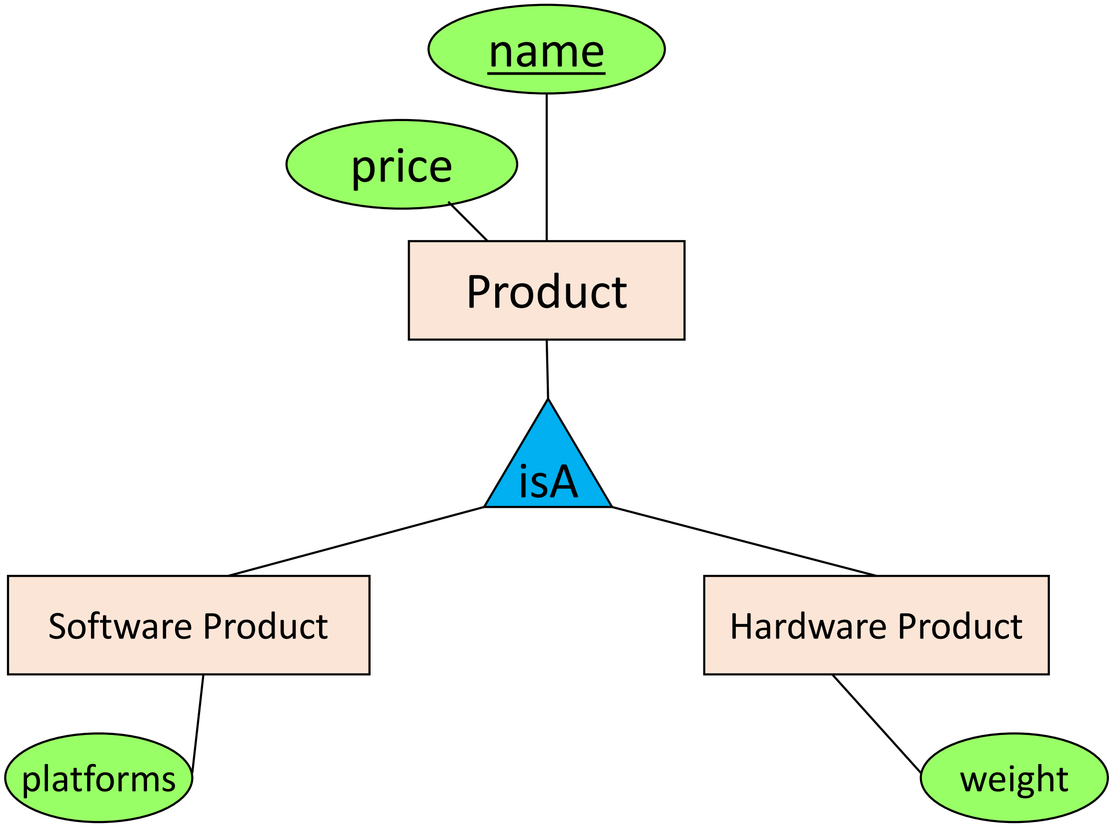
Some objects in a class may be special

- Define a new class?
  - *But what if we want to maintain connection to current class?*
- <u>Better: define a *subclass*</u>
  - *Ex:*



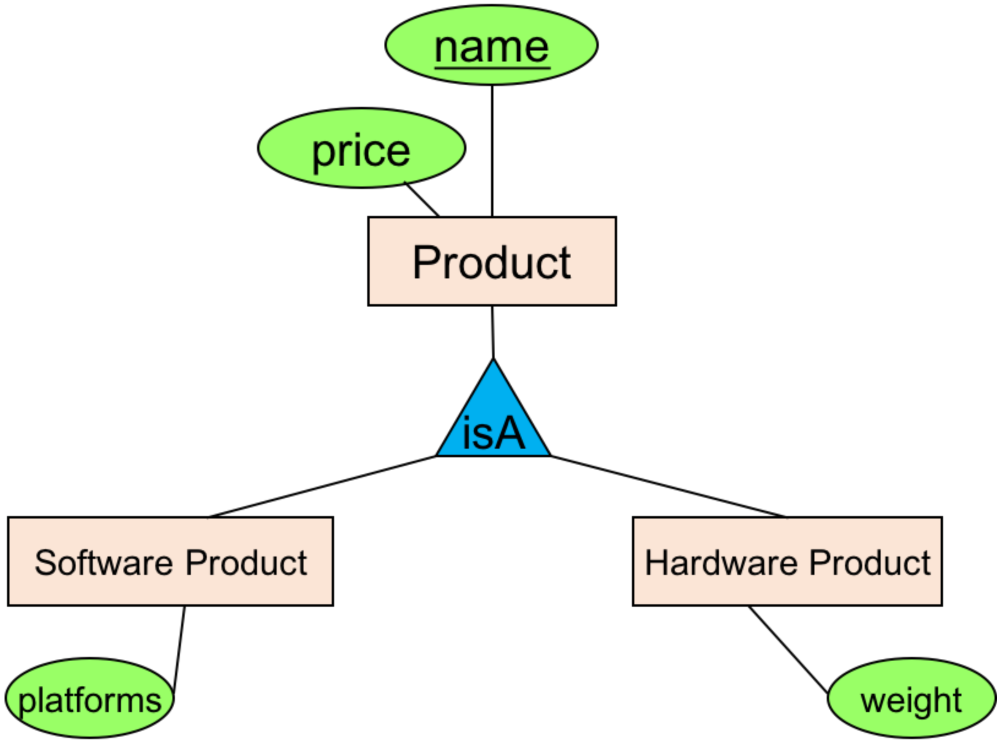
We can define **subclasses** in E/R!

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

    name
    price

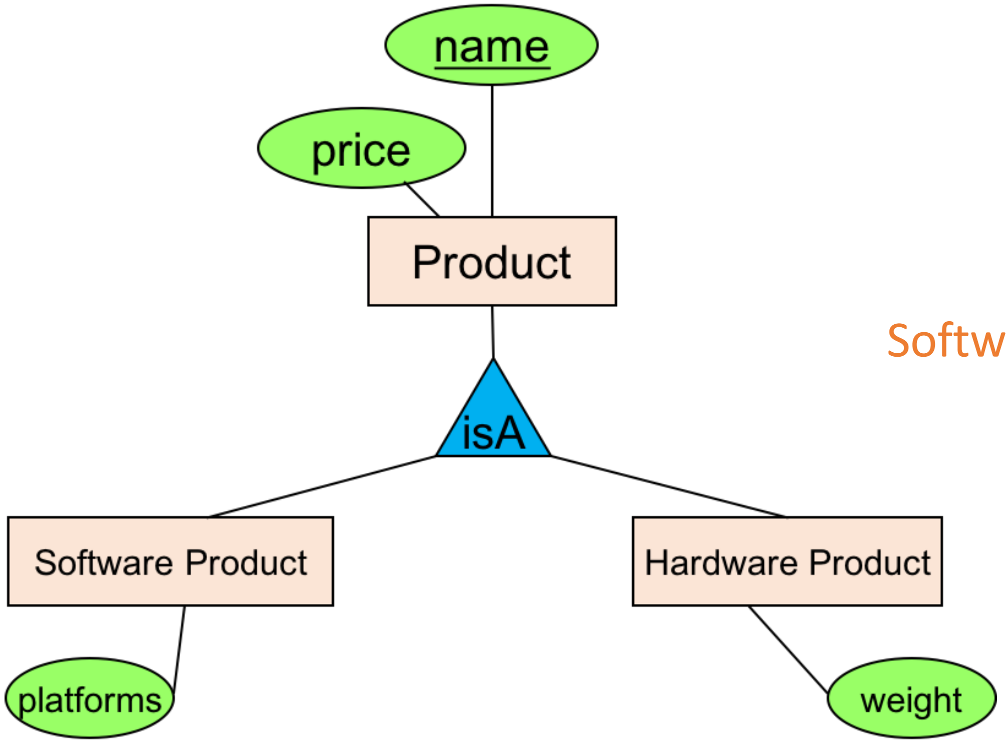
  - SoftwareProduct

    name
    price
    platforms

  - HardwareProduct

    name
    price
    weight

Subclasses to Relations



Product

name	price
iphone 8	700
iPad 4	300
office	100

SoftwareProduct

name	platforms
office	windows

HardwareProduct

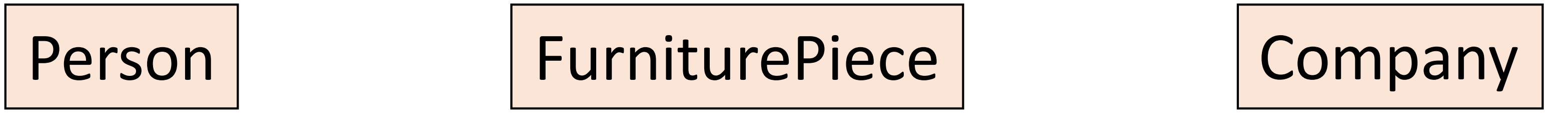
name	weight
iphone 8	148 g
ipad 4	650 g

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

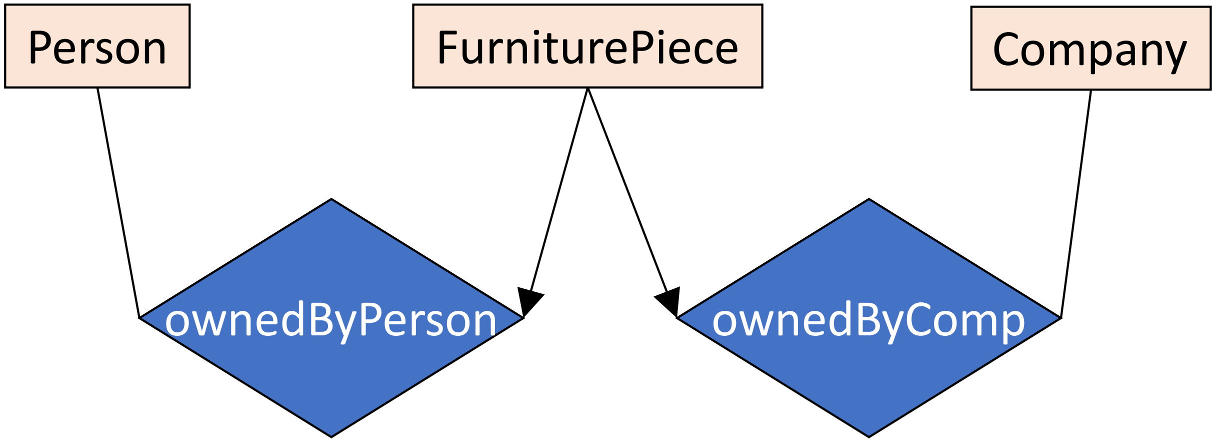
Company

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

Modeling UnionTypes With Subclasses

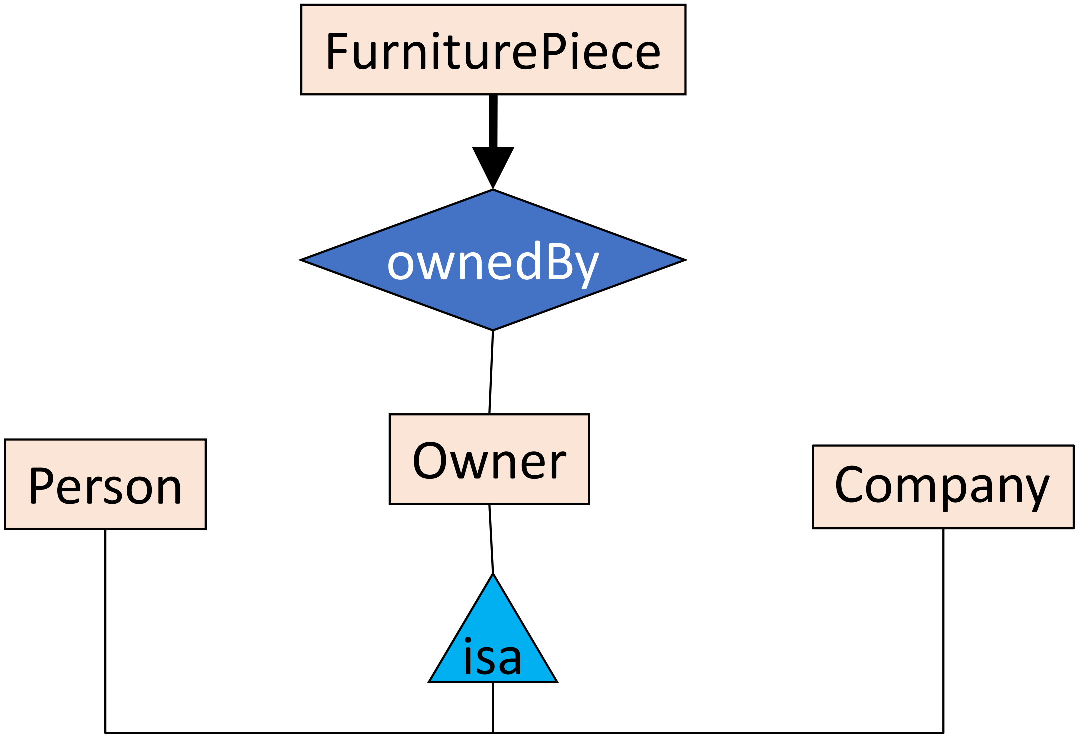
Say: each piece of furniture is owned either by a person or by a company

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



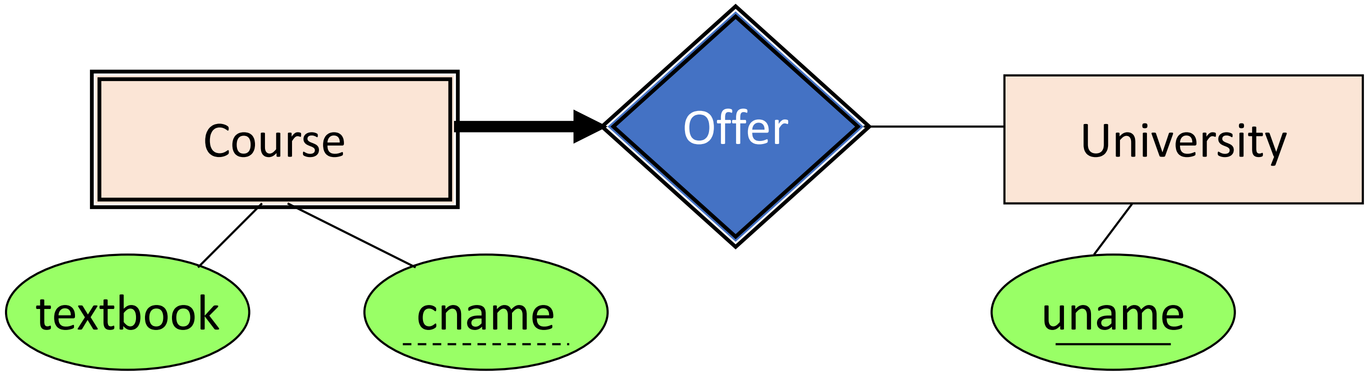
Modeling UnionTypes With Subclasses

Solution 2: better (though more laborious)



Weak Entity Sets

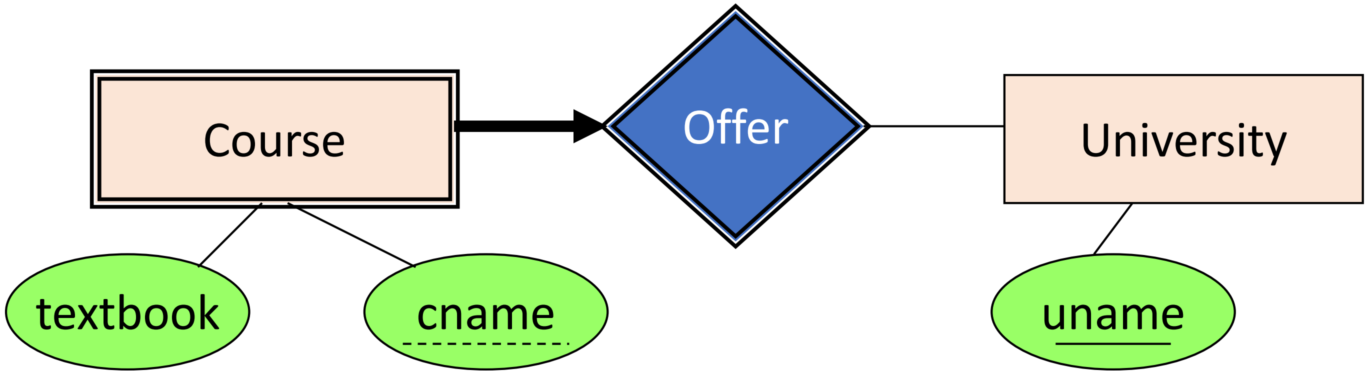
Entity sets are *weak* when their key comes from other classes to which they are related.



"Introduction to database" vs. "***The SFU introduction to database***"

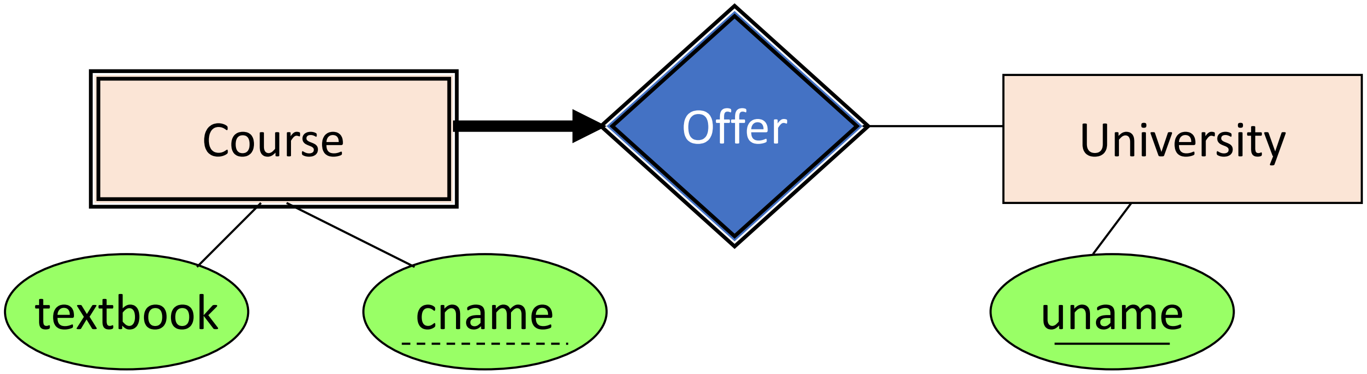
Weak Entity Sets

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



- cname is a *partial key* (denote with dashed underline).
- University is called the *supporting entity set*
- Offer is called the supporting relationship

Weak Entity Sets to Relations



Course(cname, uname, textbook)

University(uname)

~~Offering(cname, Course.uname, University.uname)~~

E/R Summary

- **E/R Basics: Entities & Relationships**
  - Database Design
  - Entities/Entity Sets/Keys/Relationships
- **E/R Design considerations**
  - Relationships cond’s: multiplicity, multi-way
  - Design considerations
  - Conversion to SQL
- **Advanced E/R Concepts**
  - Combing Relations
  - Constraints
  - Subclass
  - Weak Entity Sets

Acknowledge

- Some lecture slides were copied from or inspired by the following course materials
    - “W4111: Introduction to databases” by Eugene Wu at Columbia University
    - “CSE344: Introduction to Data Management” by Dan Suciu at University of Washington
    - “CMPT354: Database System I” by John Edgar at Simon Fraser University
    - “CS186: Introduction to Database Systems” by Joe Hellerstein at UC Berkeley
    - “CS145: Introduction to Databases” by Peter Bailis at Stanford
    - “CS 348: Introduction to Database Management” by Grant Weddell at University of Waterloo