CS 6400 A

Database Systems Concepts and Design

Lecture 4
09/02/26

Announcement

- COMET is a shadow AI operator aimed at assisting satellite mission operations. During mission operations, COMET receives inbound telemetry, performs anomaly detection and then looks at the mission documentation to perform a root-cause analysis and provide remediation steps. We have an initial pipeline design for COMET and are looking for students to help scale the design for different satellite missions.
  US citizenship/residency preferred.

- Interested students contact: Vaibhav Bhosale: https://vaibhavb007.github.io

Agenda

1. E/R Basics: Entities & Relations
2. Constraints in E/R Diagram
3. E/R Diagram to Relational Schema
4. Advanced E/R Concepts (time permitting)

Reading Materials

Notation in this lecture follow these books (**Chen-style ER notation**)

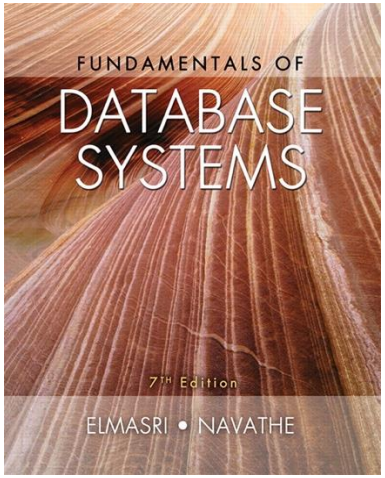
Please use the same notation for AS1!

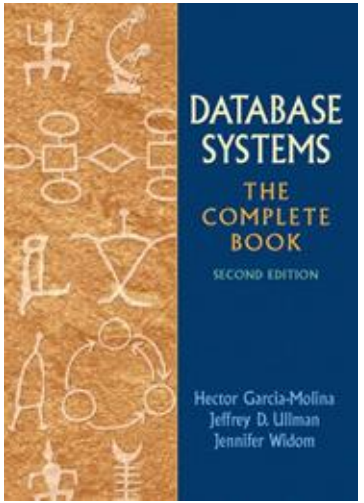
Database Systems: The Complete Book

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

Fundamentals of Database Systems

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





Acknowledgement: The following slides have been adapted from EE477 (Database and Big Data Systems) taught by Steven Whang and CS145 (Intro to Big Data Systems) taught by Peter Bailis.

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:

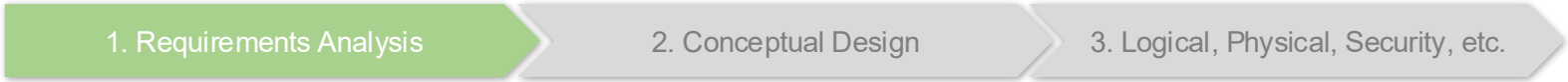
- What entities to model
- How entities are related
- What constraints exist in the domain
- How to achieve good designs

Several formalisms exist

- We discuss one flavor of E/R diagrams

"The Entity-Relationship model – toward a unified view of data" Peter Chen, 1976

Database Design Process



1. **Requirements analysis**
    - What is going to be stored?
    - How is it going to be used?
    - Who should access the data?

Technical and non-technical people are both involved

Database Design Process

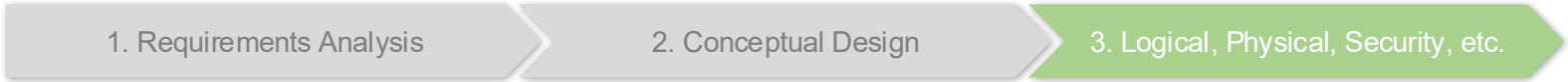


2. Conceptual Design

- 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



3. More:

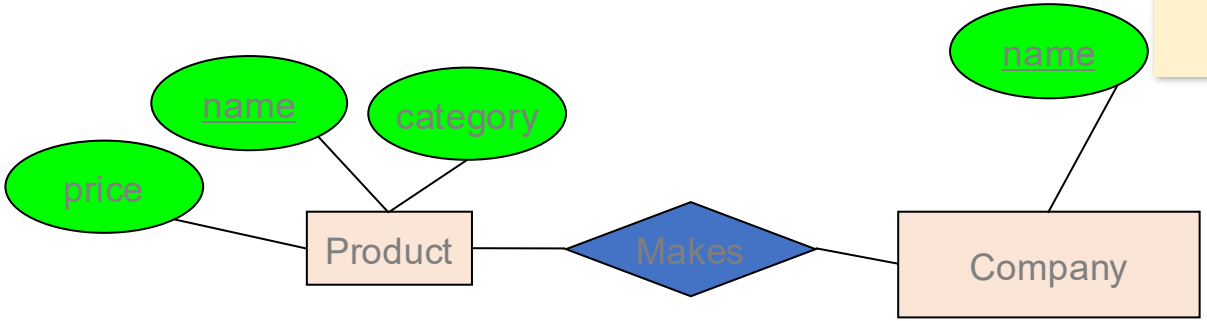
- Logical Database Design
- Physical Database Design
- Security Design

Database Design Process



E/R Model & Diagrams used

This process is iterated **many** times



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

Interlude: Impact of the ER model

- The E/R model is one of the most cited articles in Computer Science
  - *“The Entity-Relationship model – toward a unified view of data”* Peter Chen, 1976
- Used by companies big and small
  - You’ll know it soon enough



1. E/R Basics: Entities & Relations

The E in E/R: Entities and Entity Set

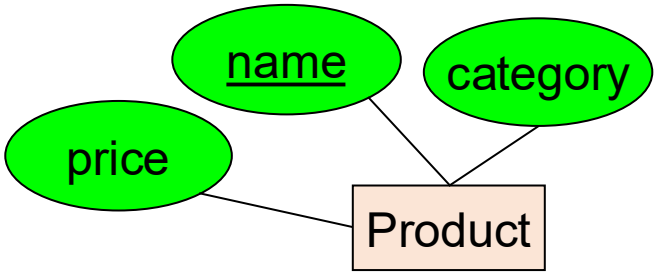
Entities are the individual objects (no associated methods)

Entity sets are collections of similar entities

- Represented by rectangles

An entity set has attributes

- Represented by ovals attached to an entity set

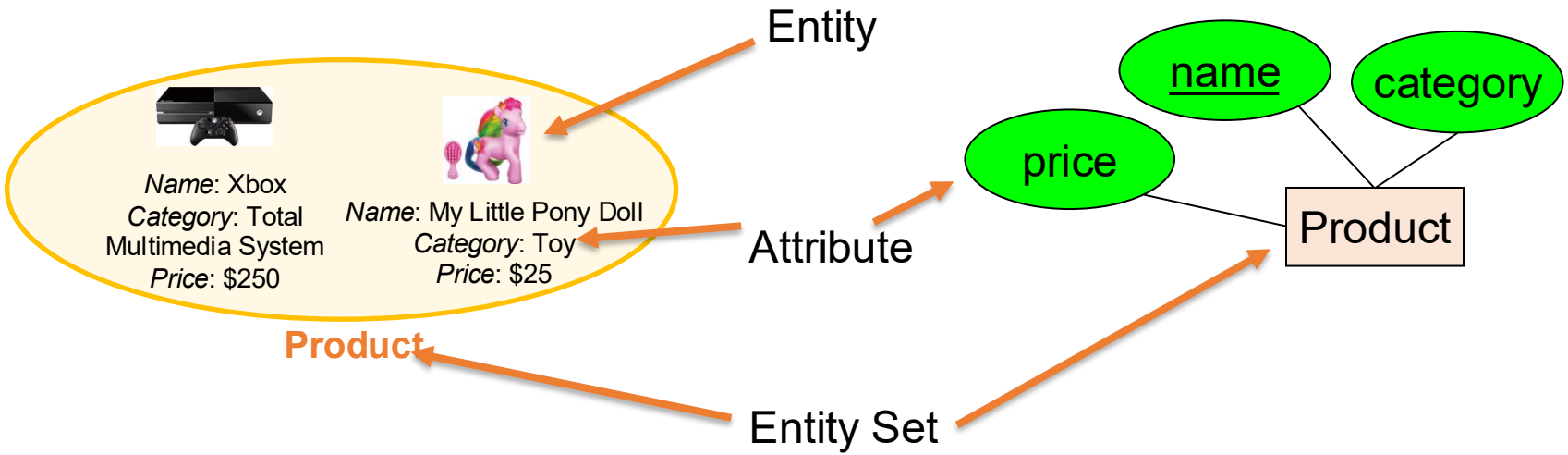


Shapes are important.
Colors are not.

Entities vs. Entity Sets

Entities are not explicitly represented in E/R diagrams!

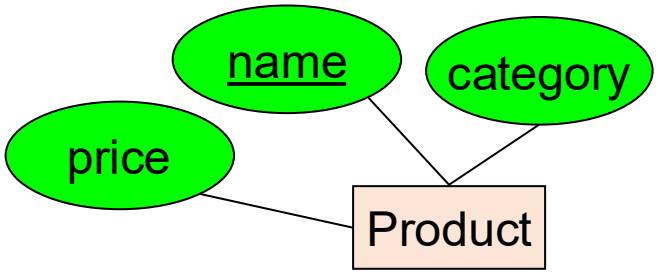
Example:



Keys

A *key* is a **minimal** set of attributes that uniquely identifies an entity.

Denote elements of the primary key by underlining.

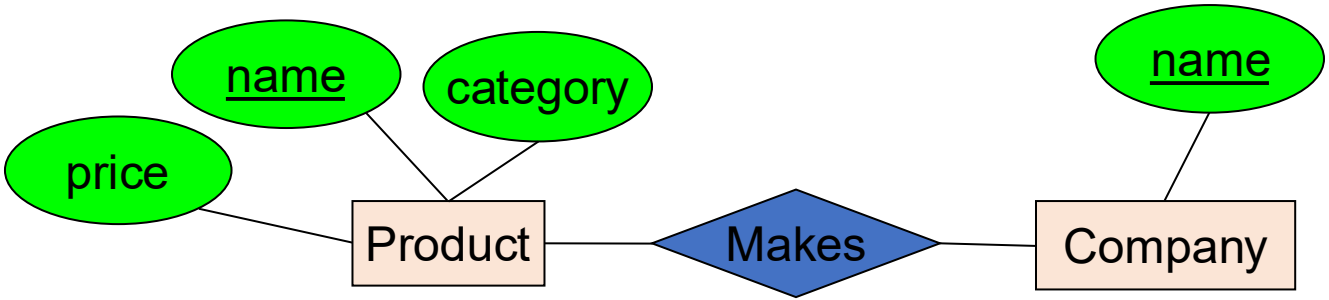


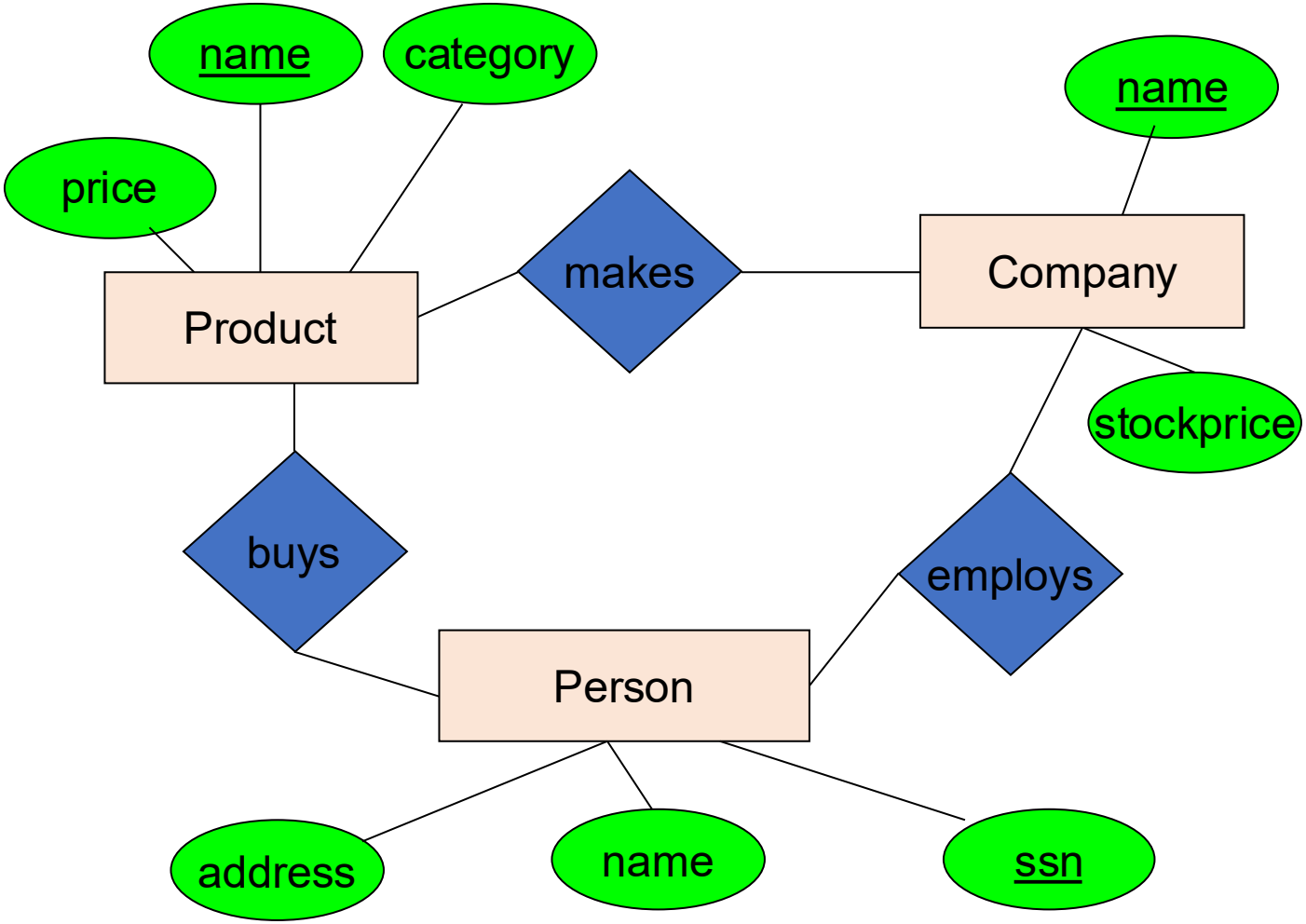
The E/R model forces us to designate a single primary key, though there may be multiple candidate keys

The R in E/R: **Relationships**

A **relationship** is between two entities

- Represented by diamonds

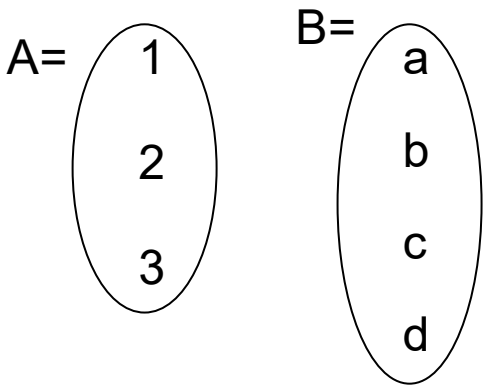



name
category
price
Product
makes
Company
name
stockprice
buys
employs
Person
address
name
ssn

What is a Relationship?

A mathematical definition:

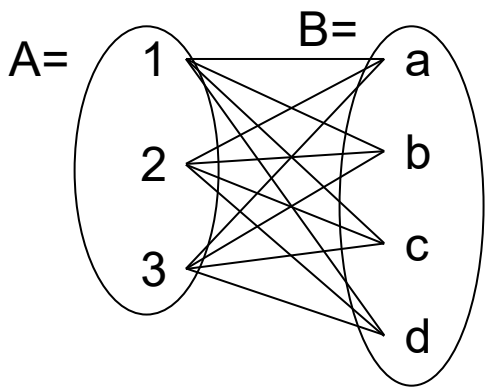
- Let A, B be sets
  - *A={1,2,3}, B={a,b,c,d}*



What is a Relationship?

A mathematical definition:

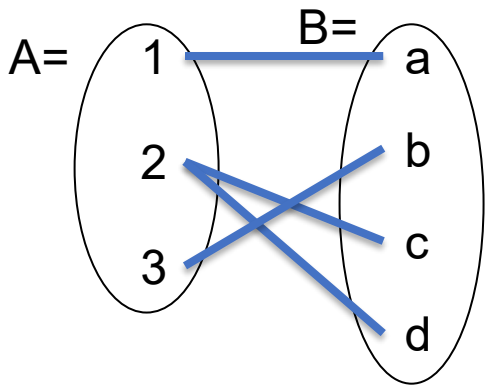
- Let A, B be sets
  - *A={1,2,3}, B={a,b,c,d}*
- A x B (the ***cross-product***) is the set of all pairs (a,b)
  - *A × B = {(1,a), (1,b), (1,c), (1,d), (2,a), (2,b), (2,c), (2,d), (3,a), (3,b), (3,c), (3,d)}*



What is a Relationship?

A mathematical definition:

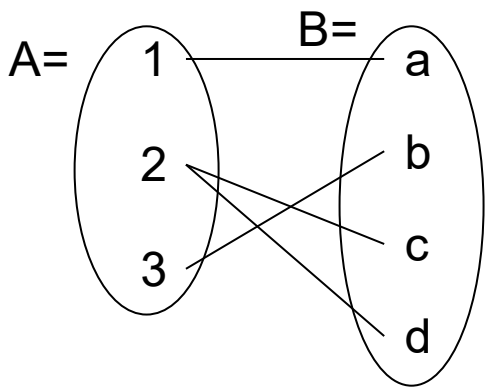
- Let A, B be sets
  - *A={1,2,3}, B={a,b,c,d},*
- A x B (the ***cross-product***) is the set of all pairs (a,b)
  - *A × B = {(1,a), (1,b), (1,c), (1,d), (2,a), (2,b), (2,c), (2,d), (3,a), (3,b), (3,c), (3,d)}*
- We define a relationship to be a subset of A x B
  - *R = {(1,a), (2,c), (2,d), (3,b)}*



What is a Relationship?

A mathematical definition:

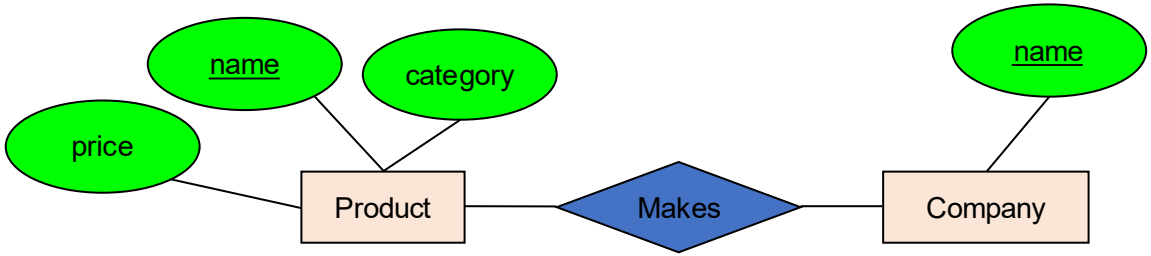
- Let A, B be sets
- A x B (the ***cross-product***) is the set of all pairs
- A <u>relationship</u> is a subset of A x B



Makes is relationship: it is a ***subset*** of **Product** × **Company**:



What is a Relationship?



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

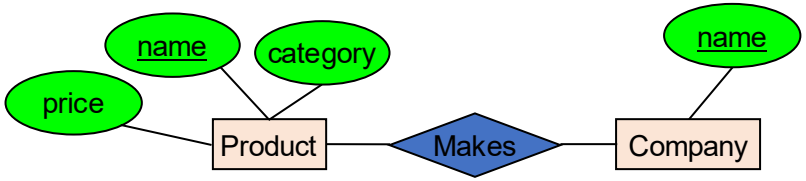
What is a Relationship?

Company

name
GizmoWorks
GadgetCorp

Product

name	category	price
Gizmo	Electronics	$9.99
GizmoLite	Electronics	$7.50
Gadget	Toys	$5.50



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

Company C × Product P

C.name	P.name	P.category	P.price
GizmoWorks	Gizmo	Electronics	$9.99
GizmoWorks	GizmoLite	Electronics	$7.50
GizmoWorks	Gadget	Toys	$5.50
GadgetCorp	Gizmo	Electronics	$9.99
GadgetCorp	GizmoLite	Electronics	$7.50
GadgetCorp	Gadget	Toys	$5.50

Makes

C.name	P.name
GizmoWorks	Gizmo
GizmoWorks	GizmoLite
GadgetCorp	Gadget

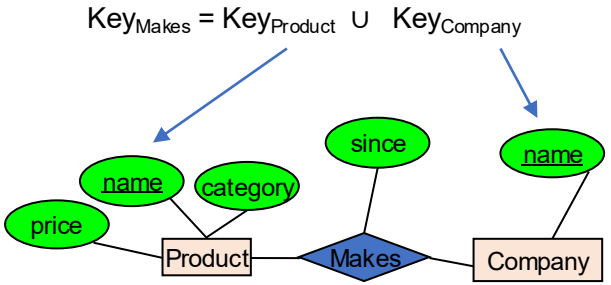
What is a Relationship?

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

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

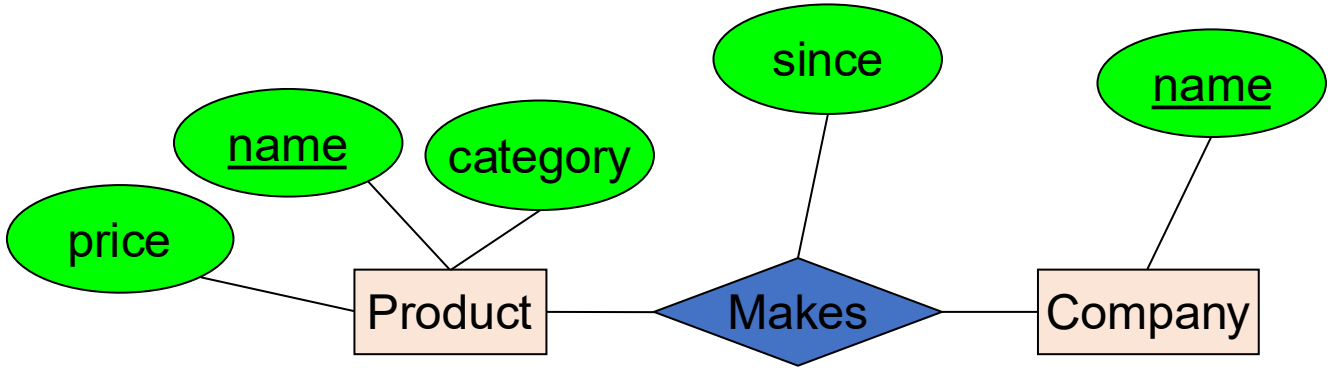
Example: the "key" for Makes is
{Product.name, Company.name}

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



Relationships and Attributes

Relationships may have attributes as well.



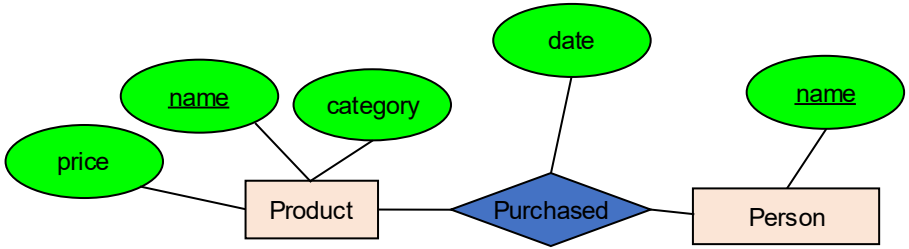
For example: “since” records when company started making a product

Note: “since” is implicitly unique per pair here! Why?

Note #2: Why not “how long”?

Decision: Relationship vs. Entity?

Q: What does this say?

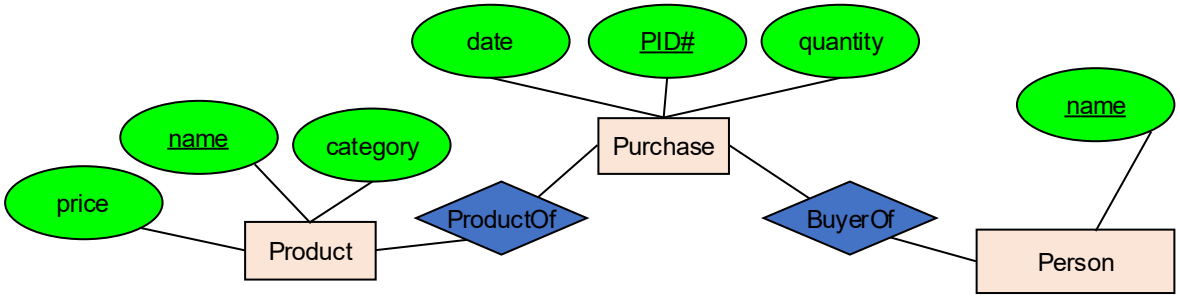


A: A person can only buy a specific product once (on one date)

Modeling something as a relationship makes it unique; what if not appropriate?

Decision: Relationship vs. Entity?

What about this way?

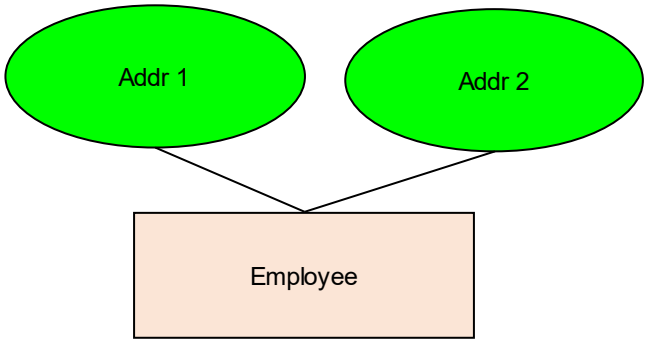


Now we can have multiple purchases per product, person pair!

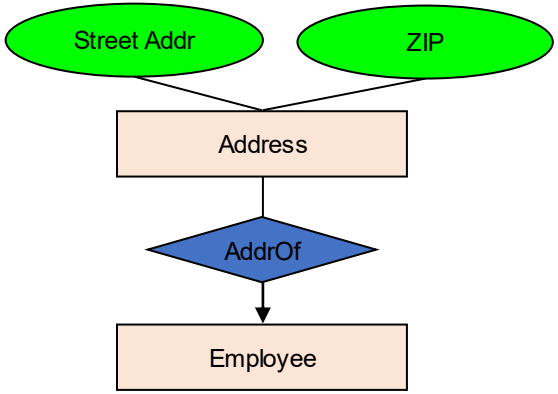
We can always use a **new entity** instead of a relationship. For example, to permit multiple instances of each entity combination!

Decision: Entity vs. Attribute?

Should address (A) be an attribute?

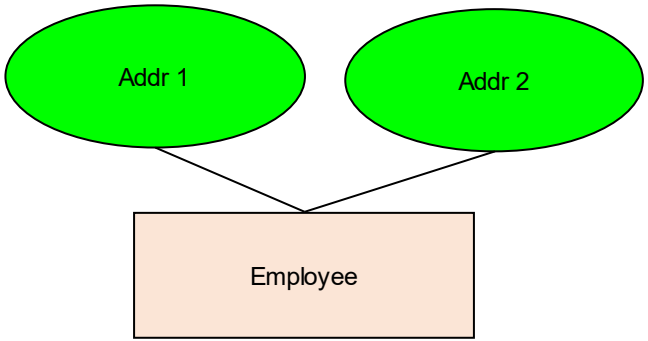


Or (B) be an entity?



Decision: Entity vs. Attribute?

Should address (A) be an attribute?

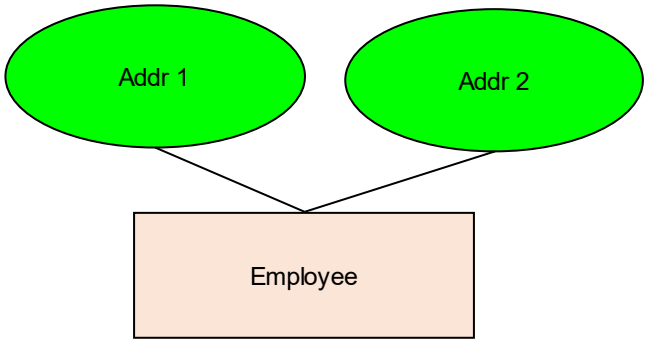


How do we handle employees with multiple addresses here?

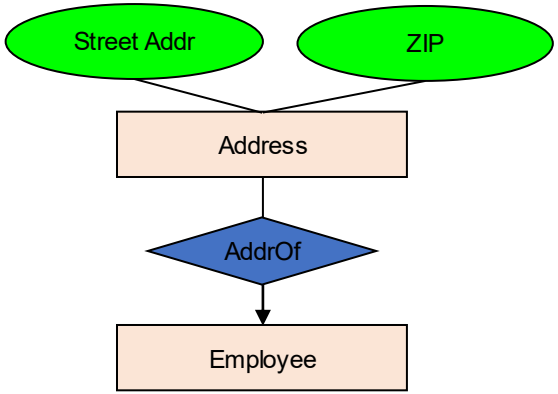
How do we handle addresses where internal structure of the address (e.g. zip code, state) is useful?

Decision: 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

2. Constraints in E/R Diagram

Constraints in E/R Diagrams

Commonly used constraints are:

Keys: Implicit constraints on uniqueness of entities

- Ex: An SSN uniquely identifies a person

Single-value constraints:

- Ex: An employee can work for only one department.

Participation constraints:

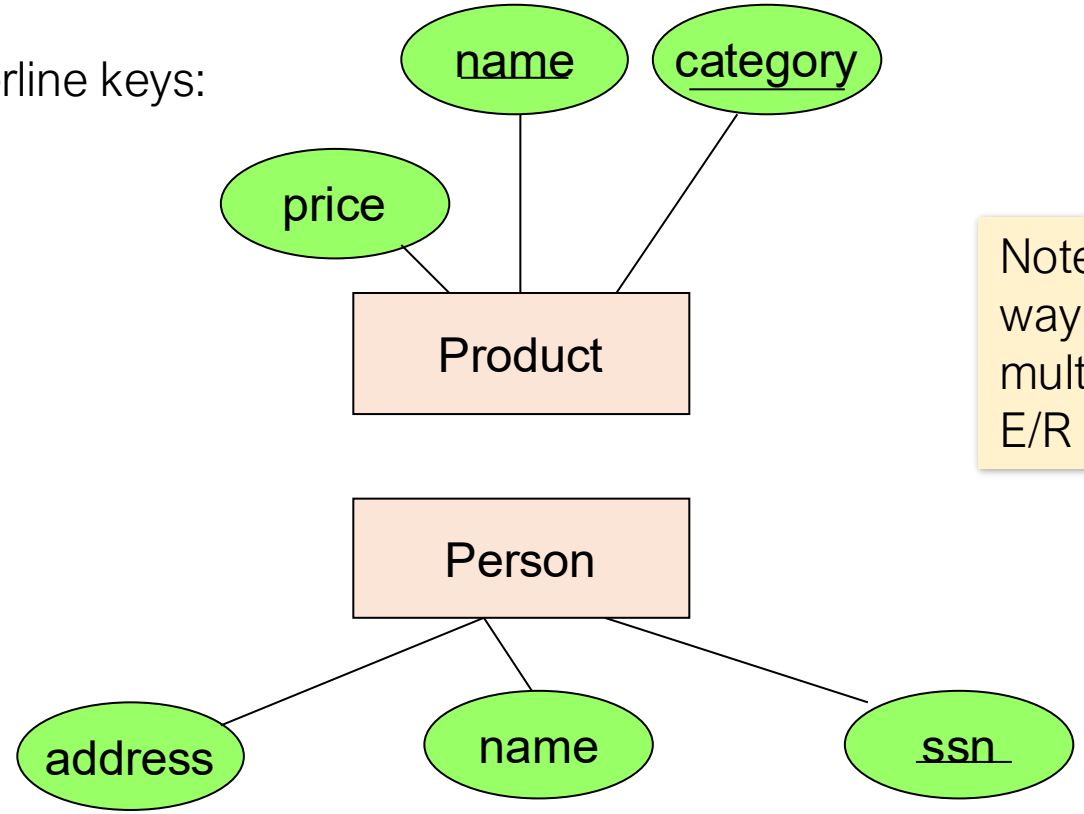
- Ex: every student must enroll in a class

Other constraints:

- Ex: peoples' ages are between 0 and 150

Key Constraints

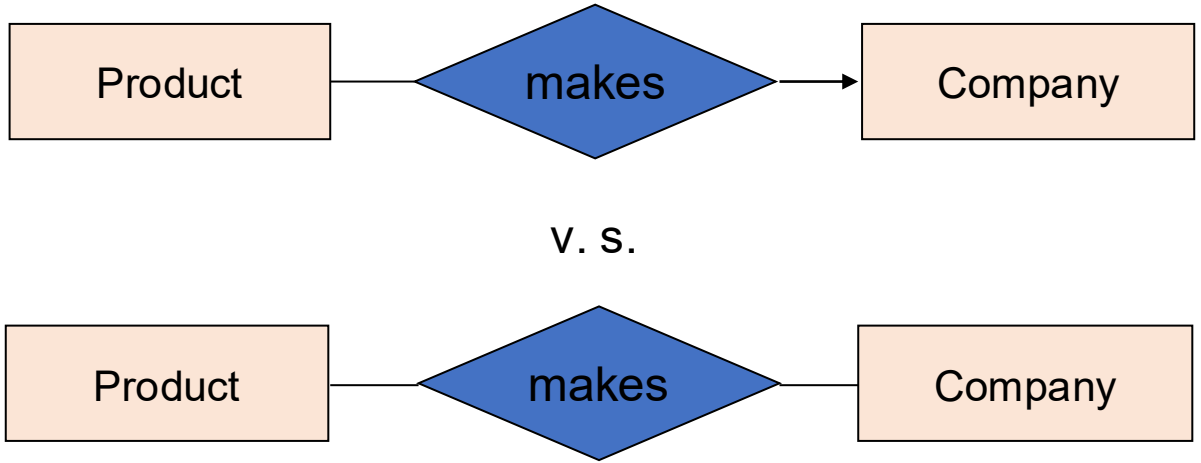
Underline keys:



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

Single Value Constraints

An arrow indicates “related to at most one entity”



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"

Arrow points to the "one" side of the relationship



A product is made by at most one company



A product is made by at most one company, and a company has at most one product

What kind of relationship does the bottom diagram imply?

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"

Arrow points to the "one" side of the relationship



A product is made by at most one company

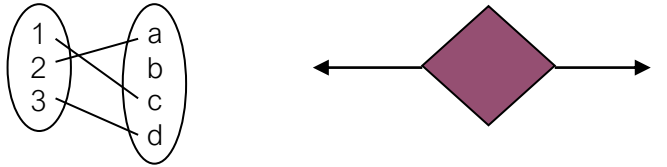


A product is made by at most one company, and a company has at most one product

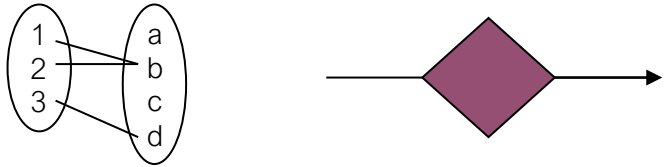
What kind of relationship does the bottom diagram imply?

Multiplicity of binary relationships

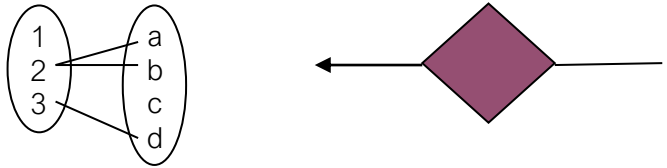
One-to-one:



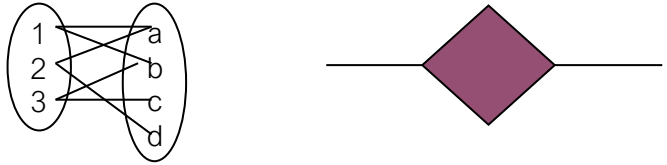
Many-to-one:



One-to-many:

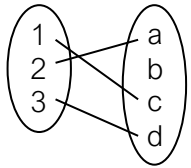


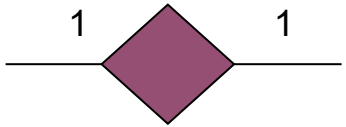
Many-to-many:



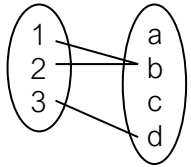
Multiplicity of binary relationships

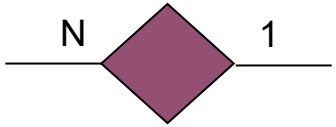
One-to-one:



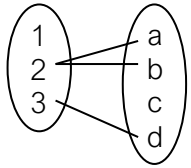


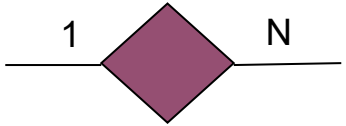
Many-to-one:



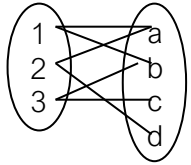


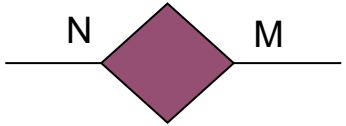
One-to-many:





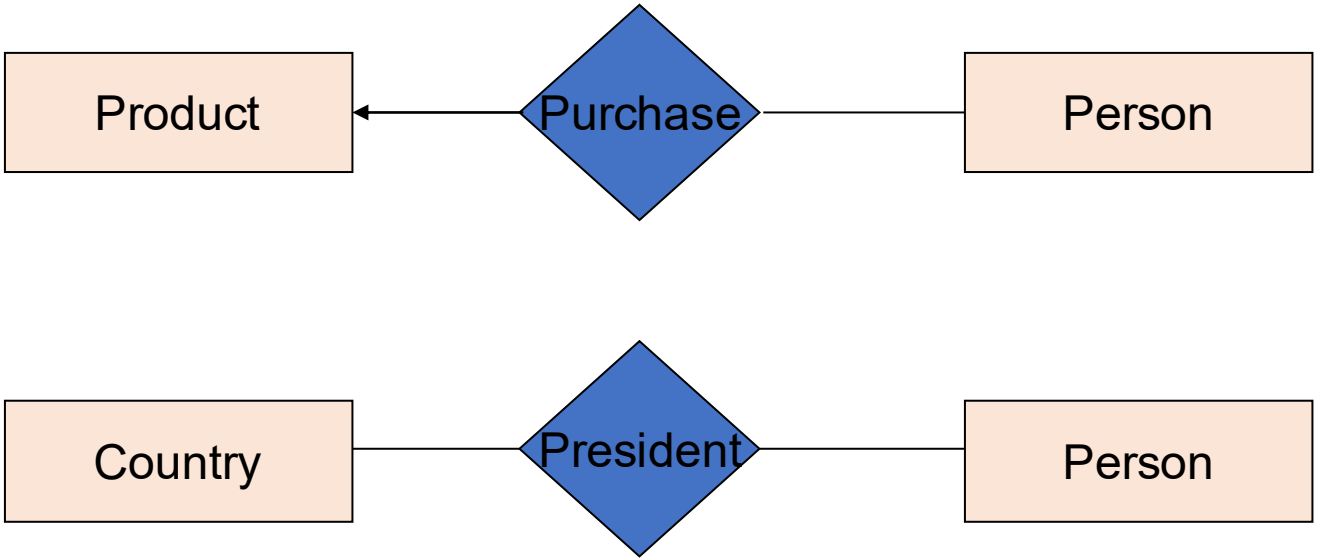
Many-to-many:



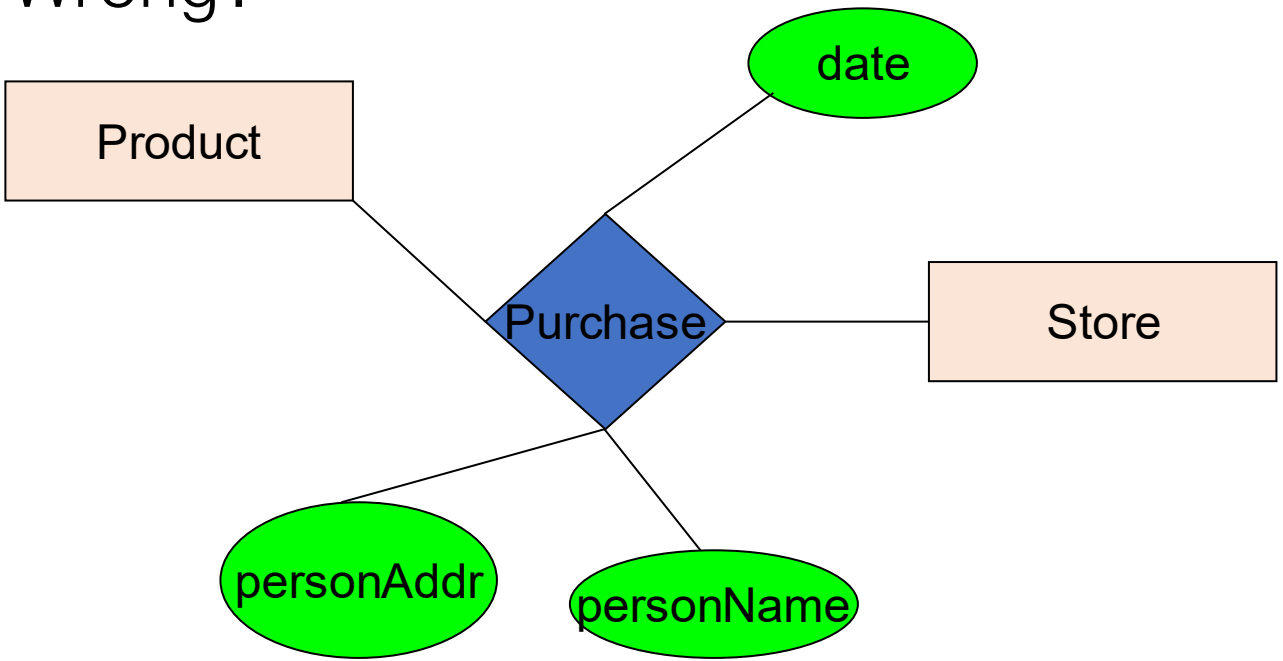


Equivalent notation (choose one of them)

Exercises:
What's Wrong?

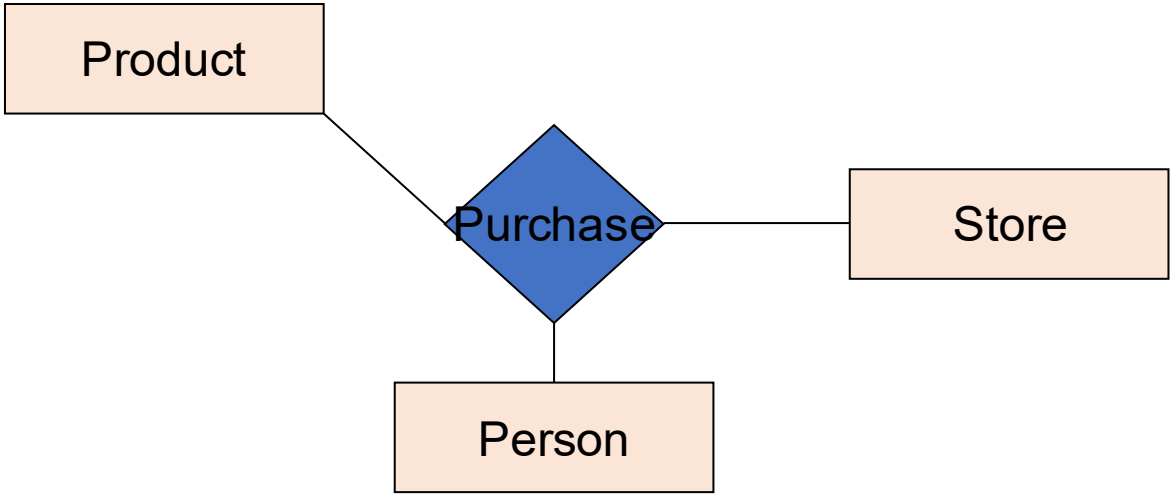


Exercises:
What's Wrong?



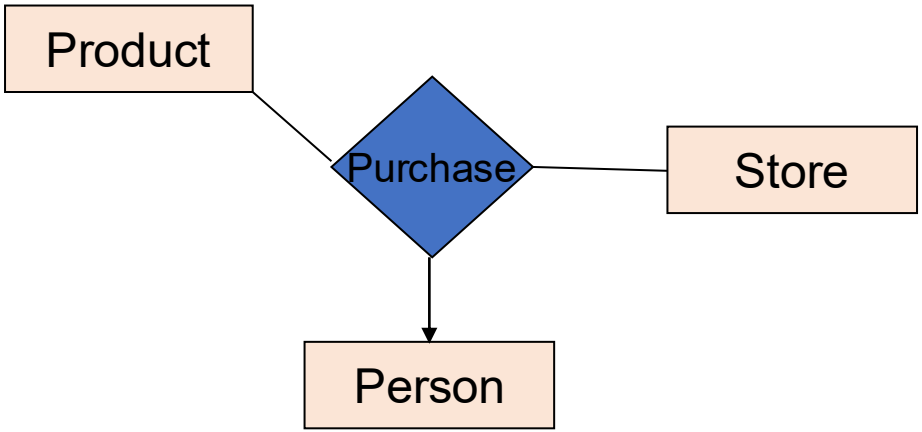
Multiway relationships

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



Arrows in Multiway Relationships

Q: What does the arrow mean ?



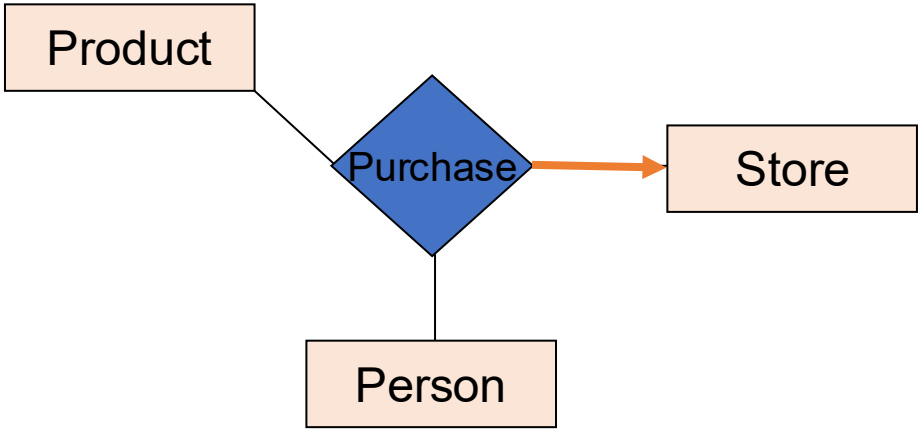
Arrow: if we select one entity from each of the other entity sets in the relationship, those entities are related to at most one entity in E.

For each (product, store), there is at most one person who have made that purchase

Q: Can a person purchase two different products the same store?
Q: Can a person purchase the same product at two different stores?

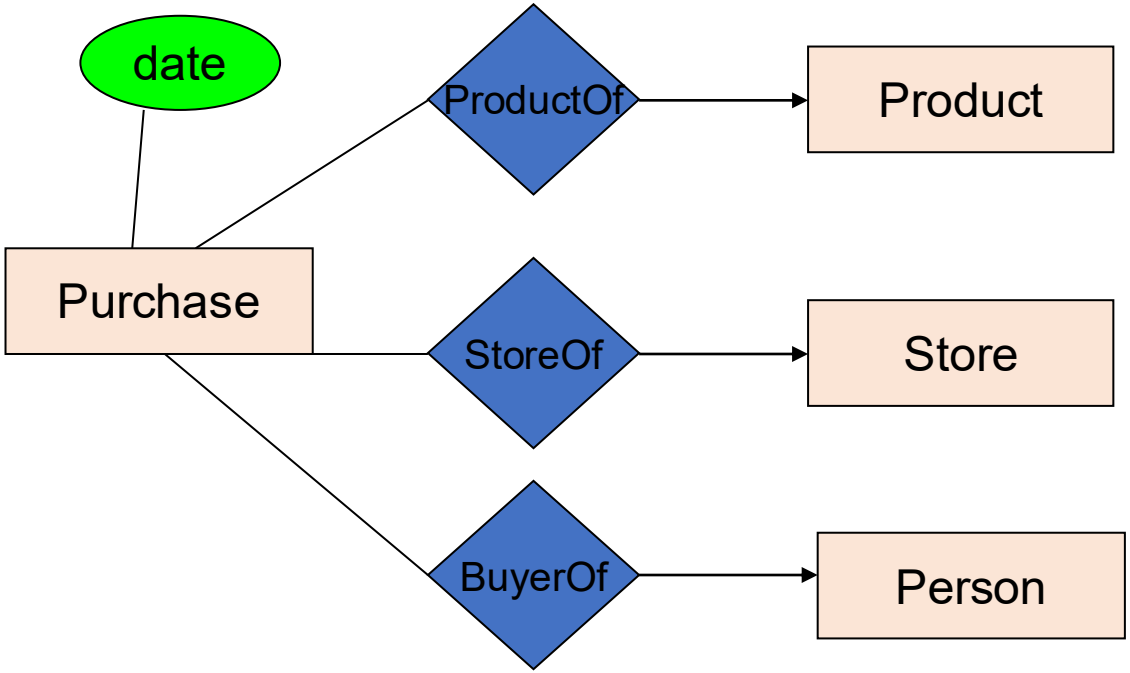
Arrows in Multiway Relationships

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

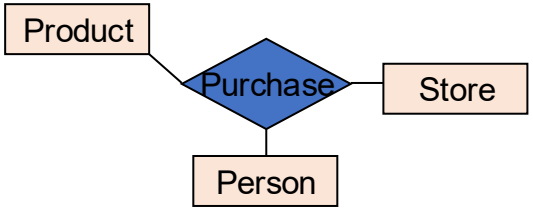


A: Cannot. This is the best approximation.
(Why only approximation ?)

Converting Multi-way Relationships to Binary

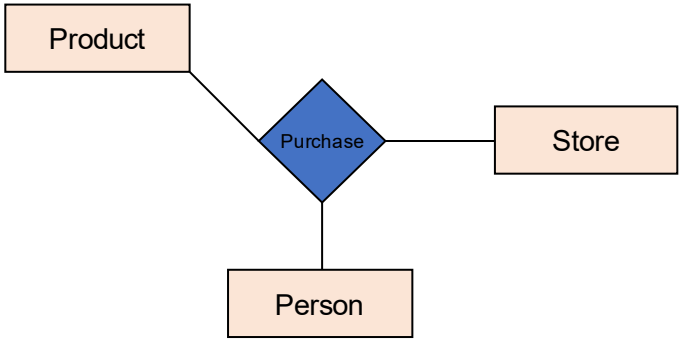


- Create a connecting entity set
- Introduce many-one relationships

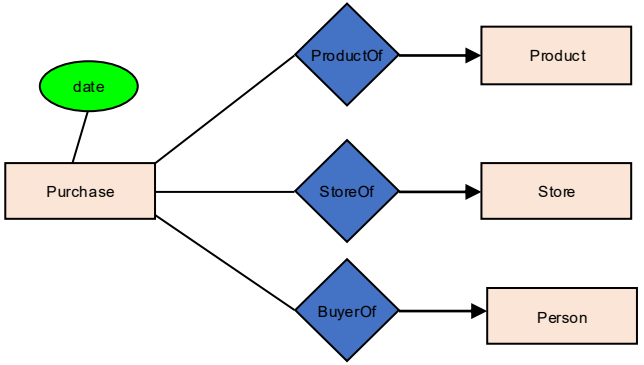


Decision: Multi-way or New Entity + Binary?

(A) Multi-way Relationship

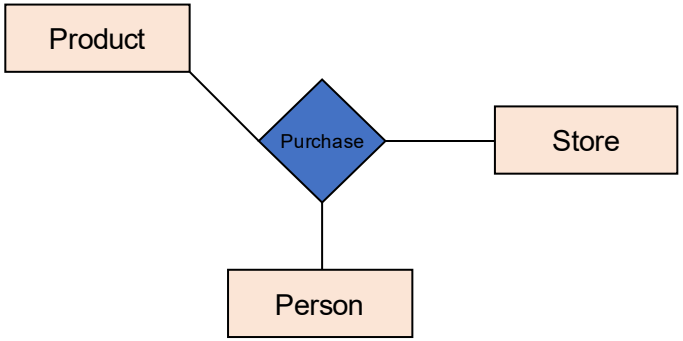


(B) Entity + Binary

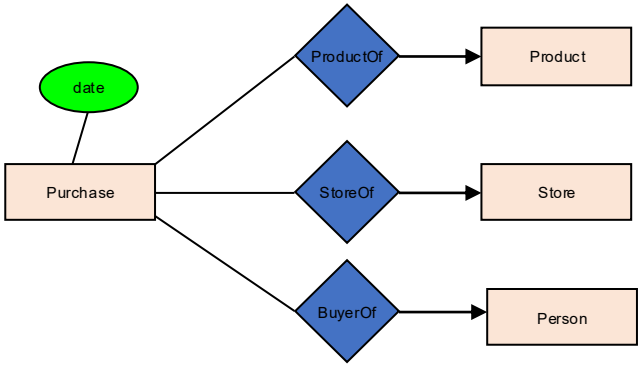


Decision: Multi-way or New Entity + Binary?

(A) Multi-way Relationship



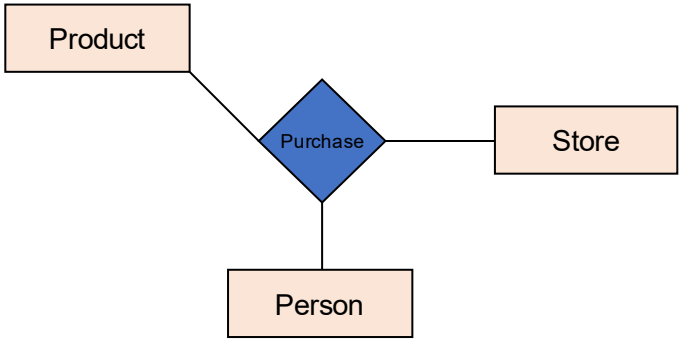
(B) Entity + Binary



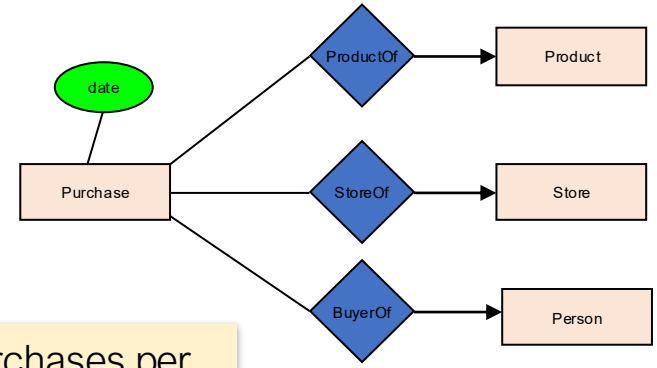
- (A) is useful when a relationship really is between multiple entities
  - Ex: A three-party legal contract

Decision: Multi-way or New Entity + Binary?

(A) Multi-way Relationship



(B) Entity + Binary

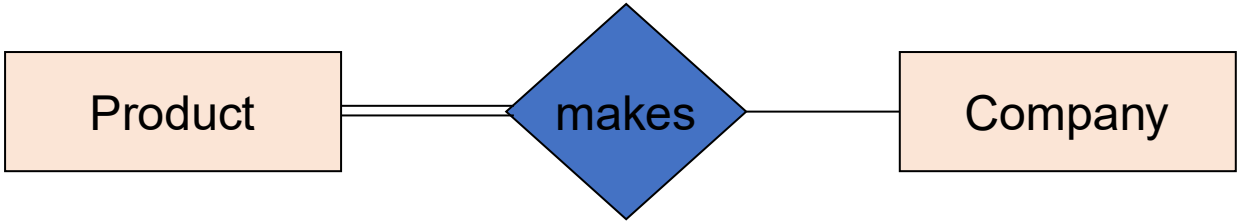


Multiple purchases per (product, store, person) combo possible here!

- Covered earlier: (B) is useful if we want to have multiple instances of the "relationship" per entity combination

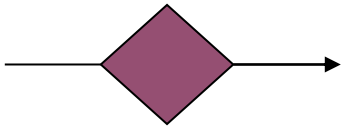
Participation Constraints: Partial v. Total

- **Total participation** (double line): Every instance in the entity must participate in the relationship
- **Partial participation** (single line): Some entities may exist without being associated with the relationship.



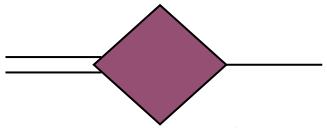
- Every product must be made by at least one company
- Some companies might not have any products.

Cardinality ratio vs Participation constraints



Cardinality ratio: at most how many?

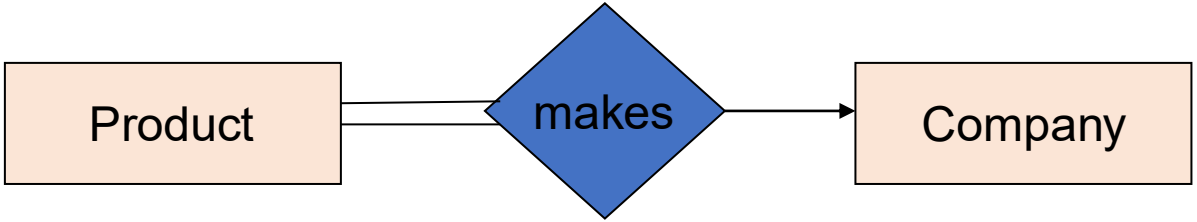
- The maximum number of entity instances that can be on each side of a relationship.
- For one instance of Entity A, what is the maximum number of instances of Entity B that can be associated with it
- Types: 1:1, 1:N, N:M



Participation constraints: at least how many?

- The minimum number of entity instances that must be on each side of a relationship
- For an instance of Entity A to exist, does it need be associated with at least one instance of Entity B
- Types: partial (minimum=0), total (minimum=1)

Cardinality ratio vs Participation constraints



What does this mean?

- Every product must participate (at least one company that makes it)
- Every product is associated with at most one company
- => Each product made by EXACTLY ONE company

3. E/R Diagram to Relational Schema

From E/R Diagrams to Relational Schema

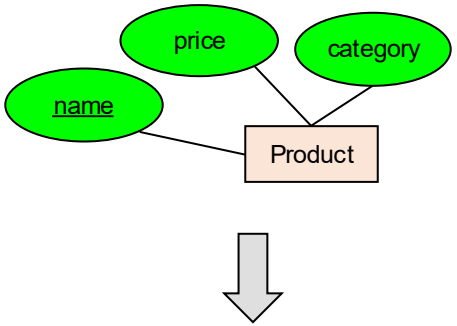
High-level idea:

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

From E/R Diagrams to Relational Schema

An entity set becomes a relation (multiset of tuples / table)

- Each tuple is one entity
- Each tuple is composed of the entity's attributes, and has the same primary key

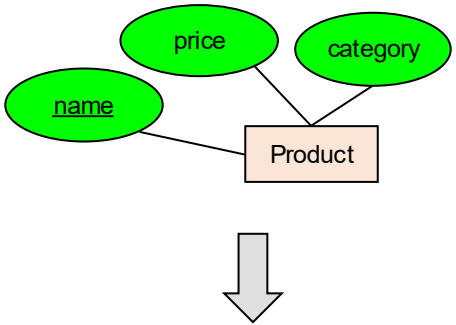


Product

name	price	category
Gizmo1	99.99	Camera
Gizmo2	19.99	Edible

From E/R Diagrams to Relational Schema

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

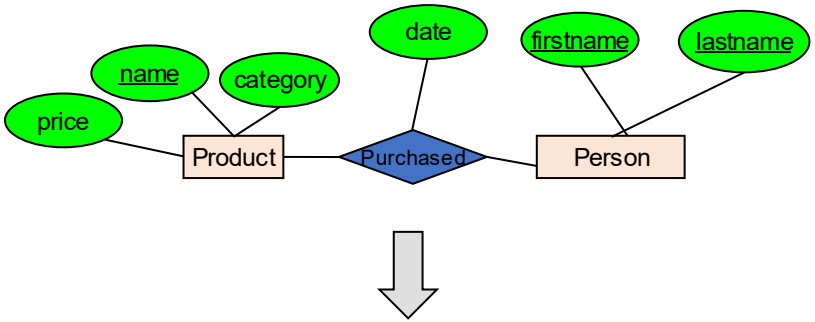


Product

name	price	category
Gizmo1	99.99	Camera
Gizmo2	19.99	Edible

From E/R Diagrams to Relational Schema

A relation between entity sets $A_1, \ldots, A_N$ also becomes a multiset of tuples / a table



- Each row/tuple is one relation, i.e. one unique combination of entities $(a_1,\ldots,a_N)$

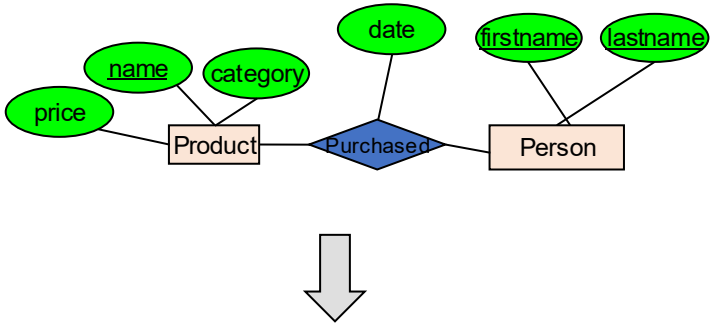
- Each row/tuple is
  - composed of the **union of the entity sets' keys**
  - has the union of the entity sets' keys as primary key
  - has the entities' primary keys as foreign keys

Purchased

name	firstname	lastname	date
Gizmo1	Bob	Joe	01/01/15
Gizmo2	Joe	Bob	01/03/15
Gizmo1	JoeBob	Smith	01/05/15

From E/R Diagrams to Relational Schema

```
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
)
```

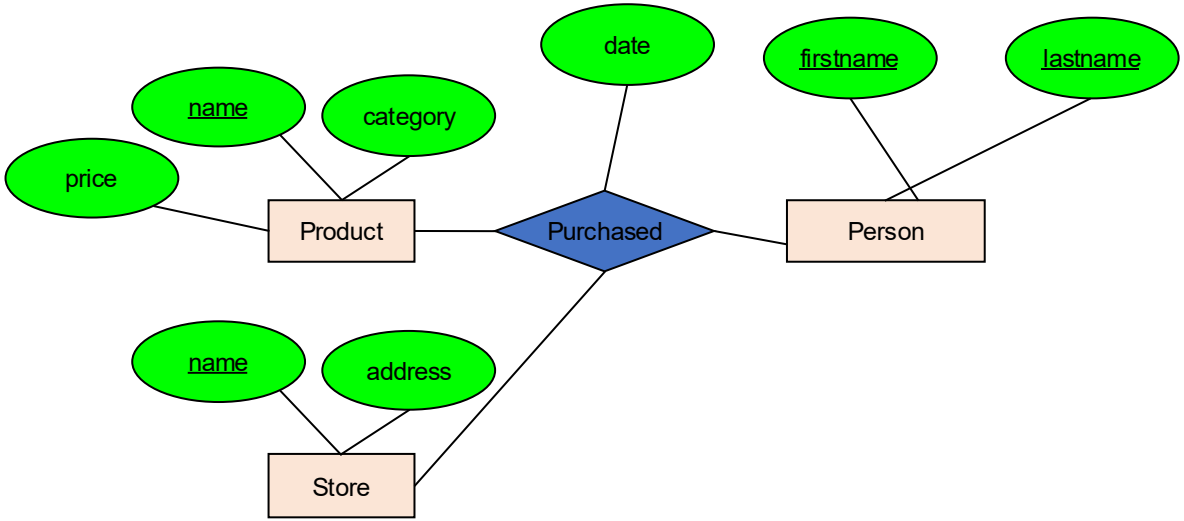


Purchased

name	firstname	lastname	date
Gizmo1	Bob	Joe	01/01/15
Gizmo2	Joe	Bob	01/03/15
Gizmo1	JoeBob	Smith	01/05/15

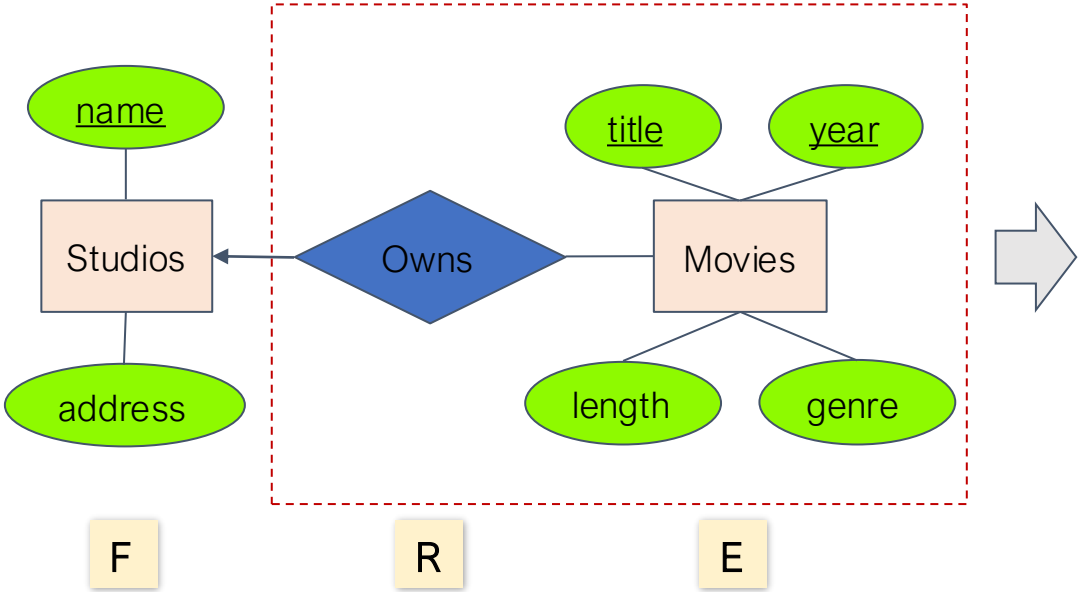
Exercise

How do we represent **Purchased** as a relational schema?



Combining relations

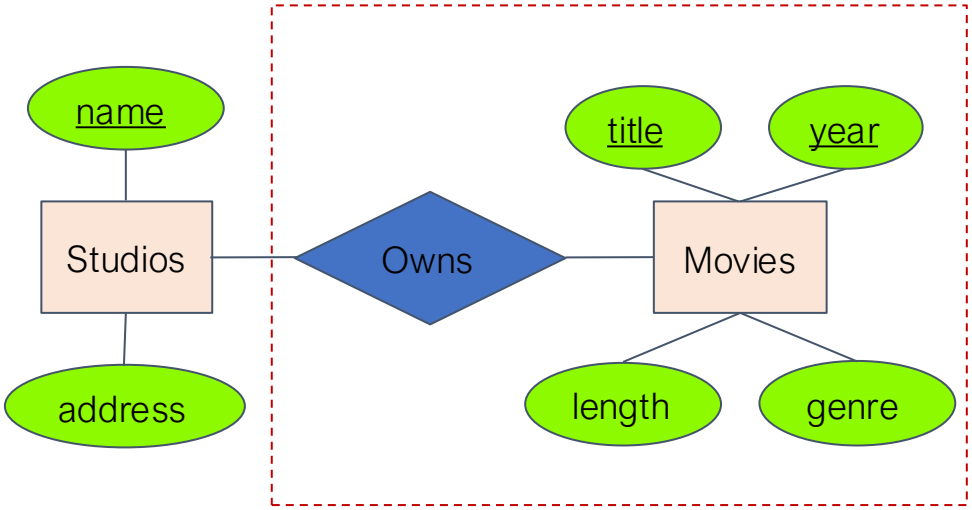
- If E is connected to F through a many-one relationship R, combine E and R
  - Attributes of E and R, and the key attributes of F
- Advantage: querying one relation is faster than querying several relations



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

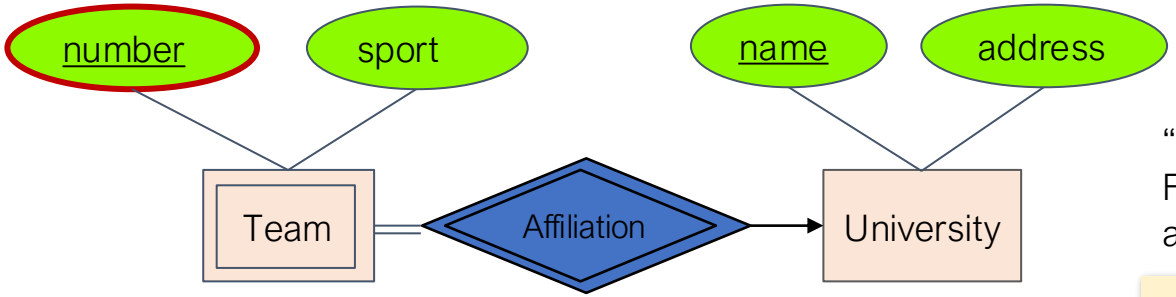
Owns

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

4. Advanced E/R Concepts

Weak entity set

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



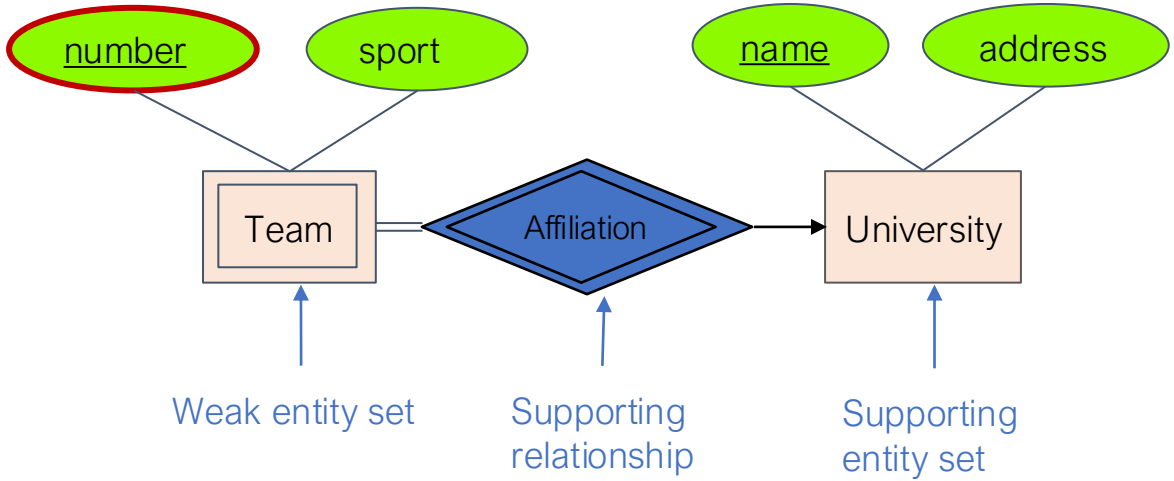
"Football team" v. "The GT Football team" (E.g., UGA has a football team too)

"has a" relationship

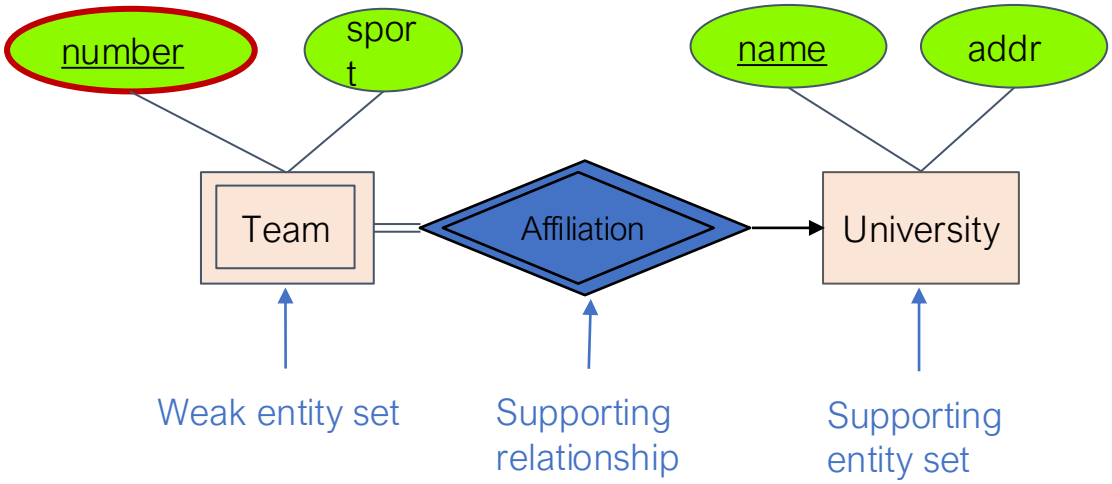
- number is a *partial key;* the key also contains keys of the University entity set
- Total participation: every weak entity must have an owner.
- Each weak entity has exactly one owner through that identifying relationship

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 to relations



University(univ_name, address)
Team(number, univ_name, sport)

Schema of weak entity set W:

- All attributes of W
- Owner's primary key (as a foreign key)
- attributes of the identifying relationship

Primary key = owner's key + W's partial key

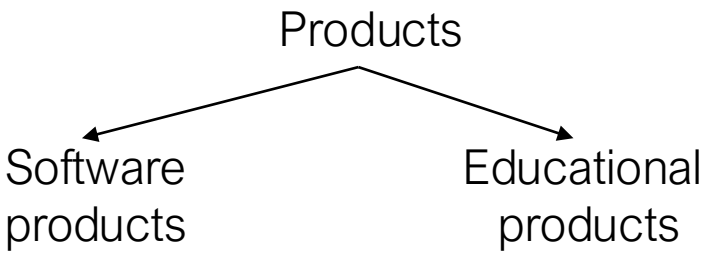
Do not construct a relation for the supporting relationship.

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?

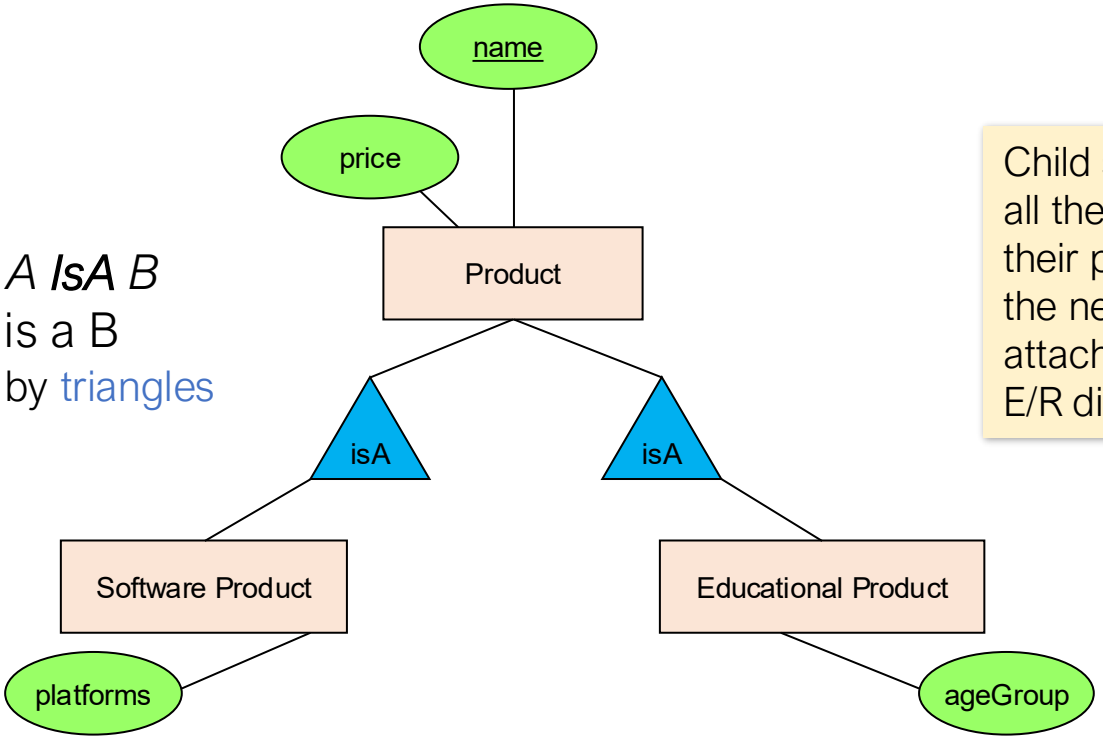
Better: define a subclass



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

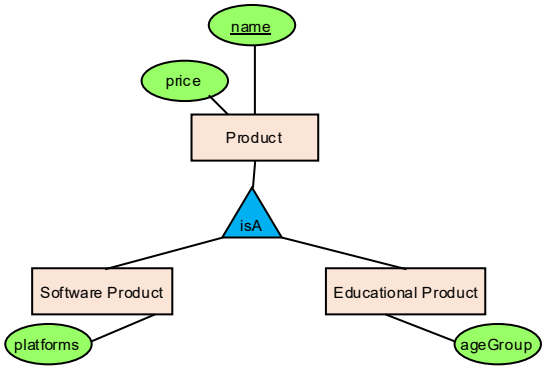
Modeling Subclasses

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



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

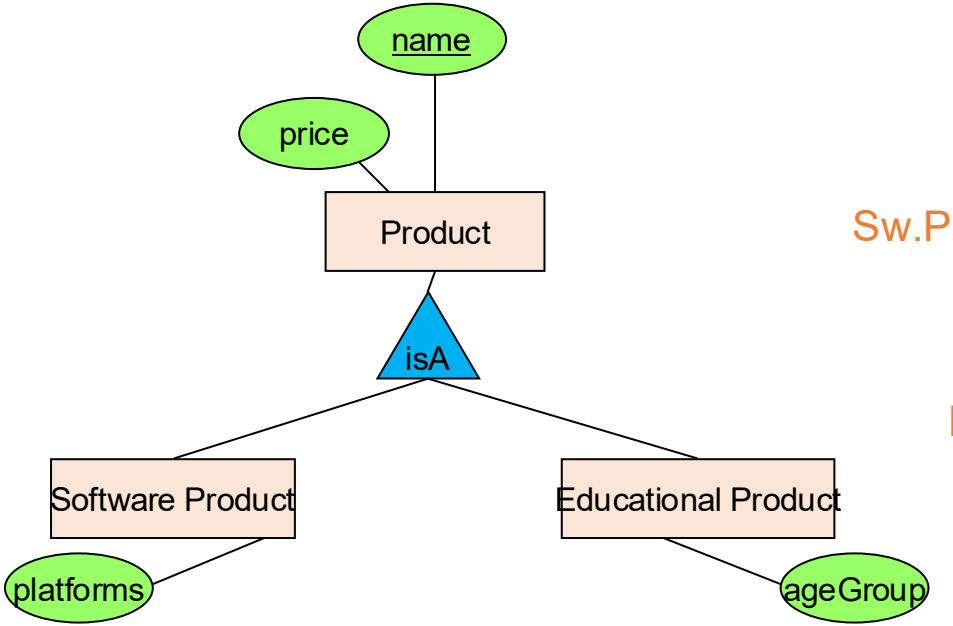
name
price
platforms

- EducationalProduct

name
price
ageGroup

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

Additional Reading

Requirements for weak entity set
- 4.4.2

Combining relationship
- 4.5.3

Converting weak entity set to relations
- 4.5.4

Converting subclasses to relations
- 4.6

