Introduction to Data Management
CSE 414

Unit 6: Conceptual Design

E/R Diagrams

Integrity Constraints

BCNF

(3 lectures)

Introduction to Data Management
CSE 414

E/R Diagrams

Class Overview

- Unit 1: Intro
- Unit 2: Relational Data Models and Query Languages
- Unit 3: Non-relational data
- Unit 4: RDMBS internals and query optimization
- Unit 5: Parallel query processing
- Unit 6: DBMS usability, conceptual design
  - E/R diagrams
  - Schema normalization
- Unit 7: Transactions
- Unit 8: Advanced topics (time permitting)

Database Design

What it is:

- Starting from scratch, design the database schema: relation, attributes, keys, foreign keys, constraints etc

Why it's hard

- The database will be in operation for a very long time (years). Updating the schema while in production is very expensive (why?)

Database Design

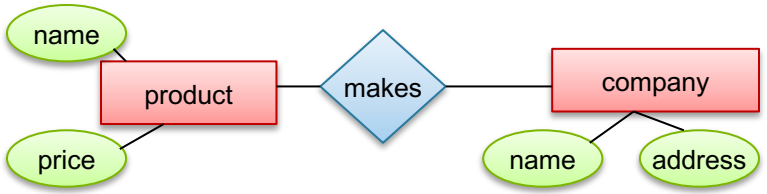
- Consider issues such as:
  - What entities to model
  - How entities are related
  - What constraints exist in the domain

- Several formalisms exists
  - We discuss E/R diagrams entities and relations!
  - UML, model-driven architecture

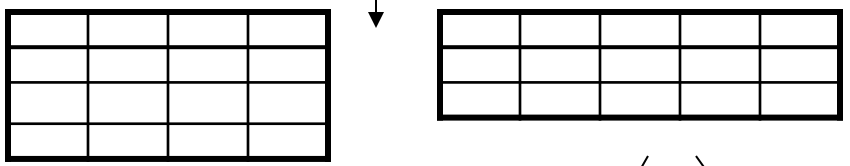
- Reading: Sec. 4.1-4.6

Database Design Process

Conceptual Model:

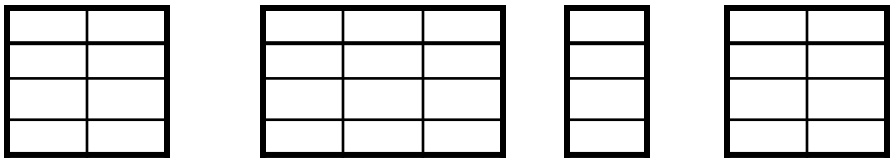


Relational Model:
Tables + constraints
And also functional dep.



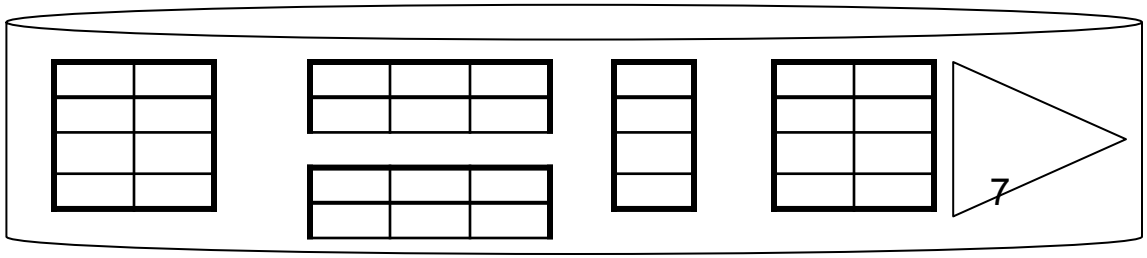
Normalization:
Eliminates anomalies

how are tables
going to be joined
helps us understand
how to break them up



Conceptual Schema

Physical storage details

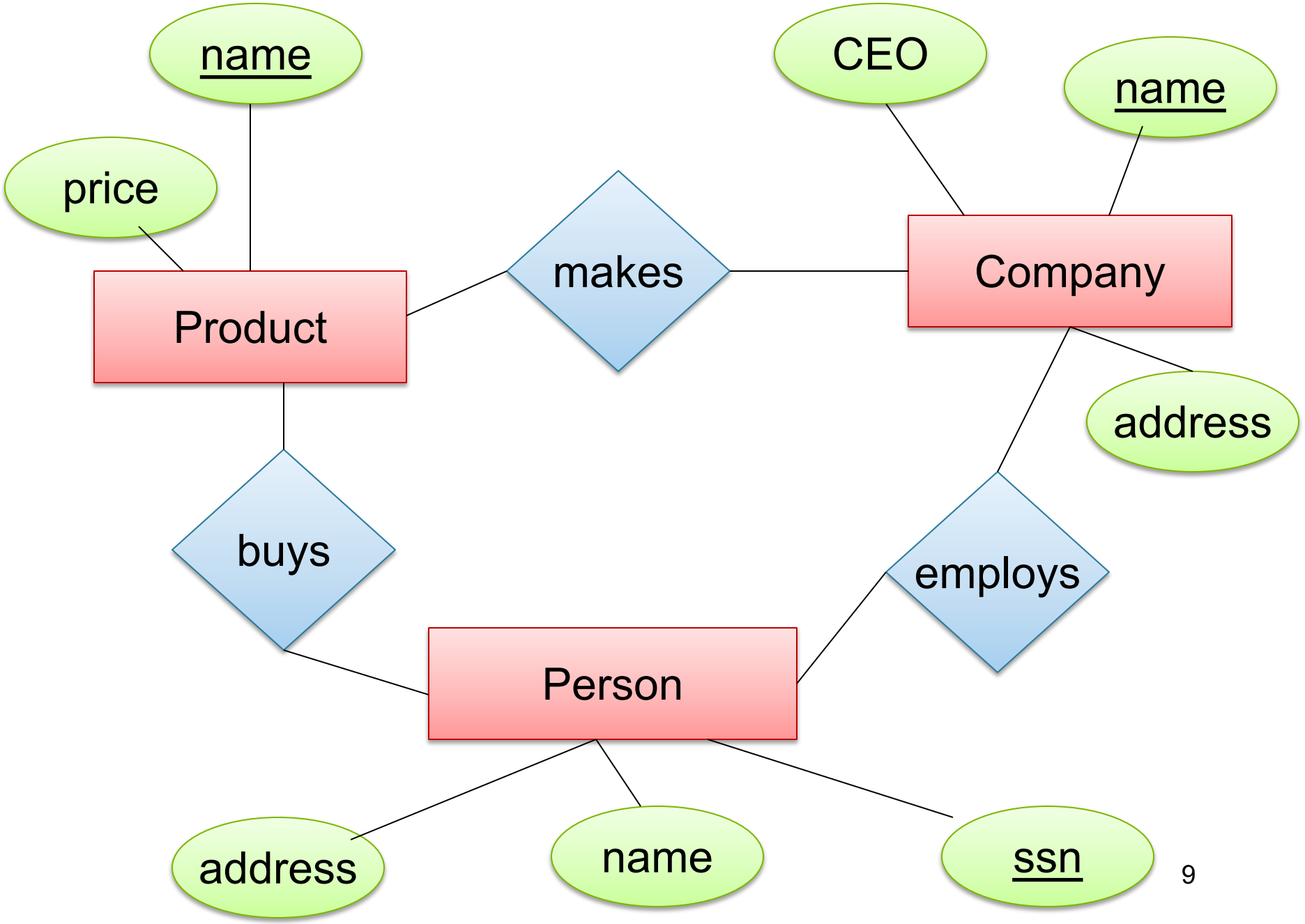


Physical Schema

Where do you store these tables on disk to be most efficient?
e.g. Row order vs Column order

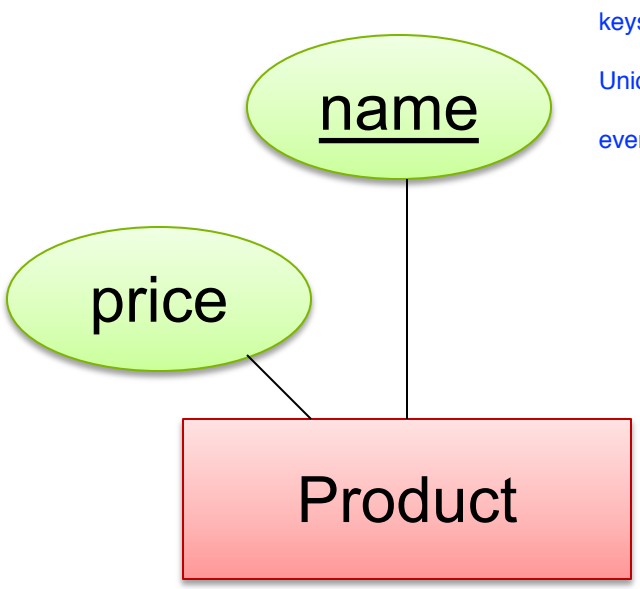
Entity / Relationship Diagrams

- Entity set = a class
  - An entity = an object
- Attribute
- Relationship


name
price
Product
makes
CEO
name
Company
address
buys
employs
Person
address
name
ssn

Keys in E/R Diagrams

- Every entity set must have a key



keys are underline;;;; not the same as a table key

Uniquely defines each object;

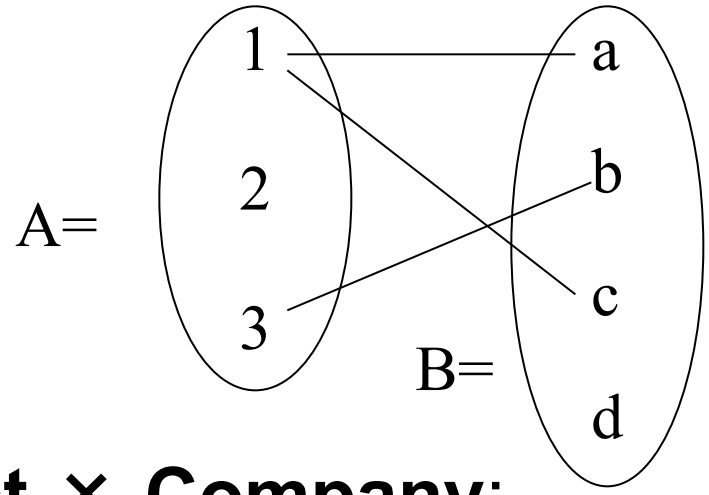
every entity MUST have a key

What is a Relation ?

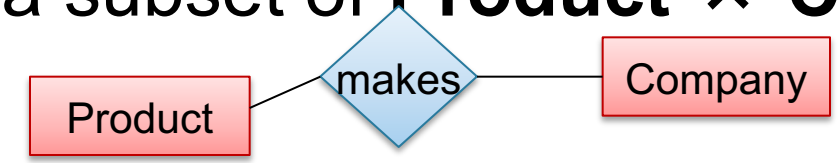
- A mathematical definition:
  - if A, B are sets, then a relation R is a subset of A × B
- A={1,2,3}, B={a,b,c,d},

  A × B = {(1,a),(1,b), . . ., (3,d)}

  R = {(1,a), (1,c), (3,b)}



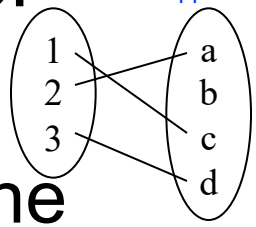
- **makes** is a subset of **Product × Company**:

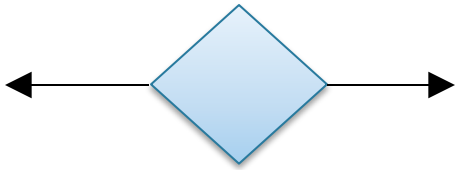


Multiplicity of E/R Relations

- one-one:

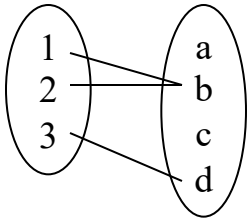
No elements (on either side) are mapped to the same value

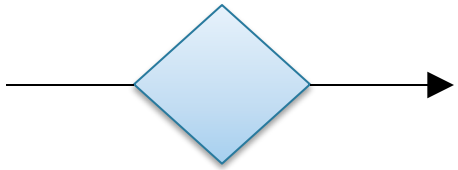




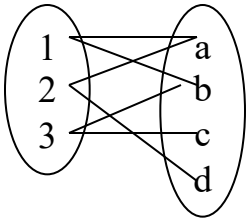
- many-one

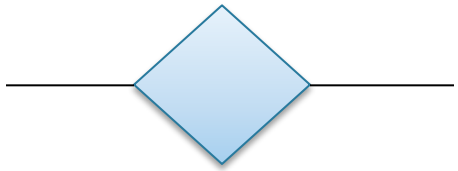
some elements on one side mapped to same value on other side, but no elements on other side are mapped to same value

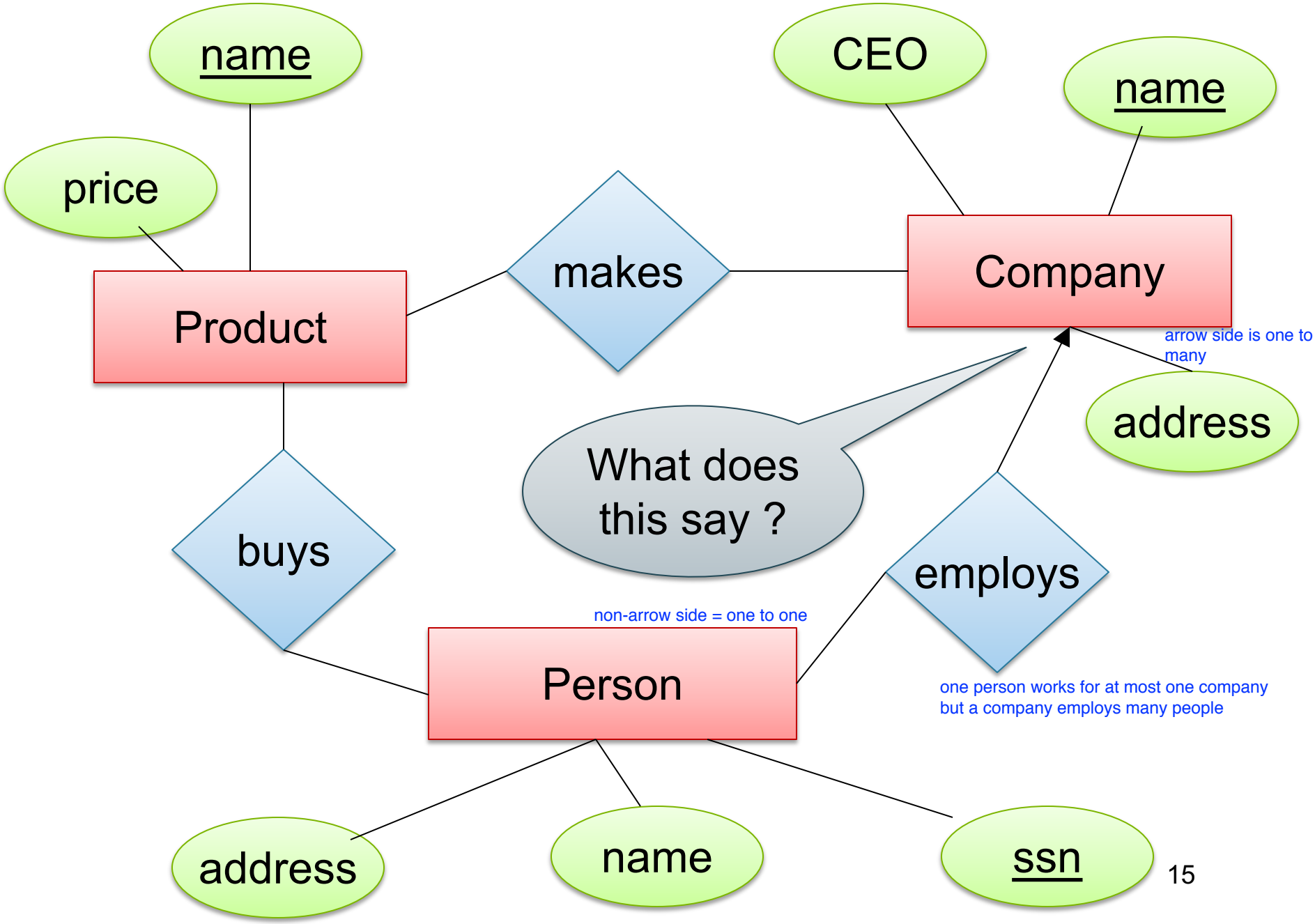




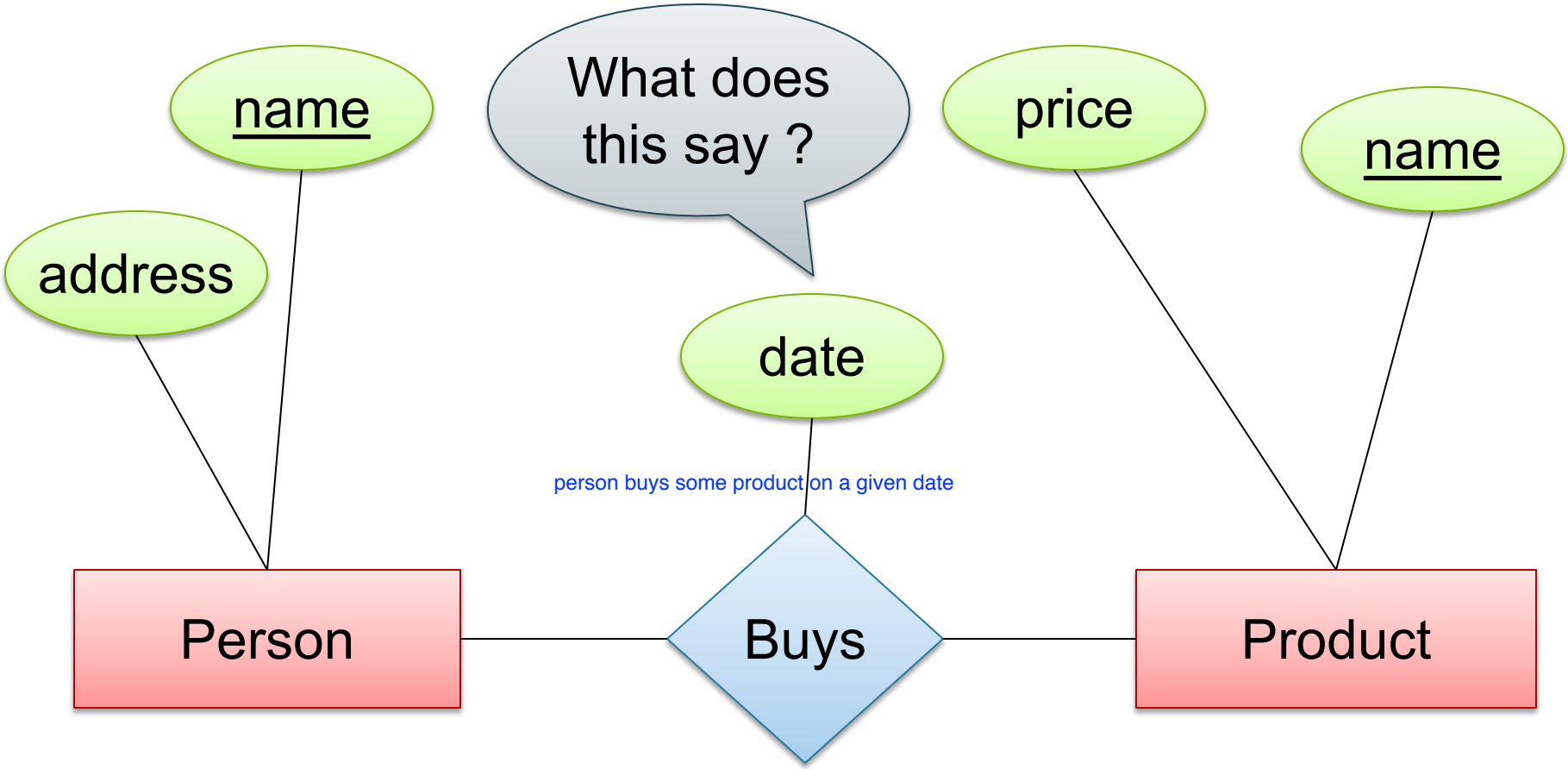
- many-many





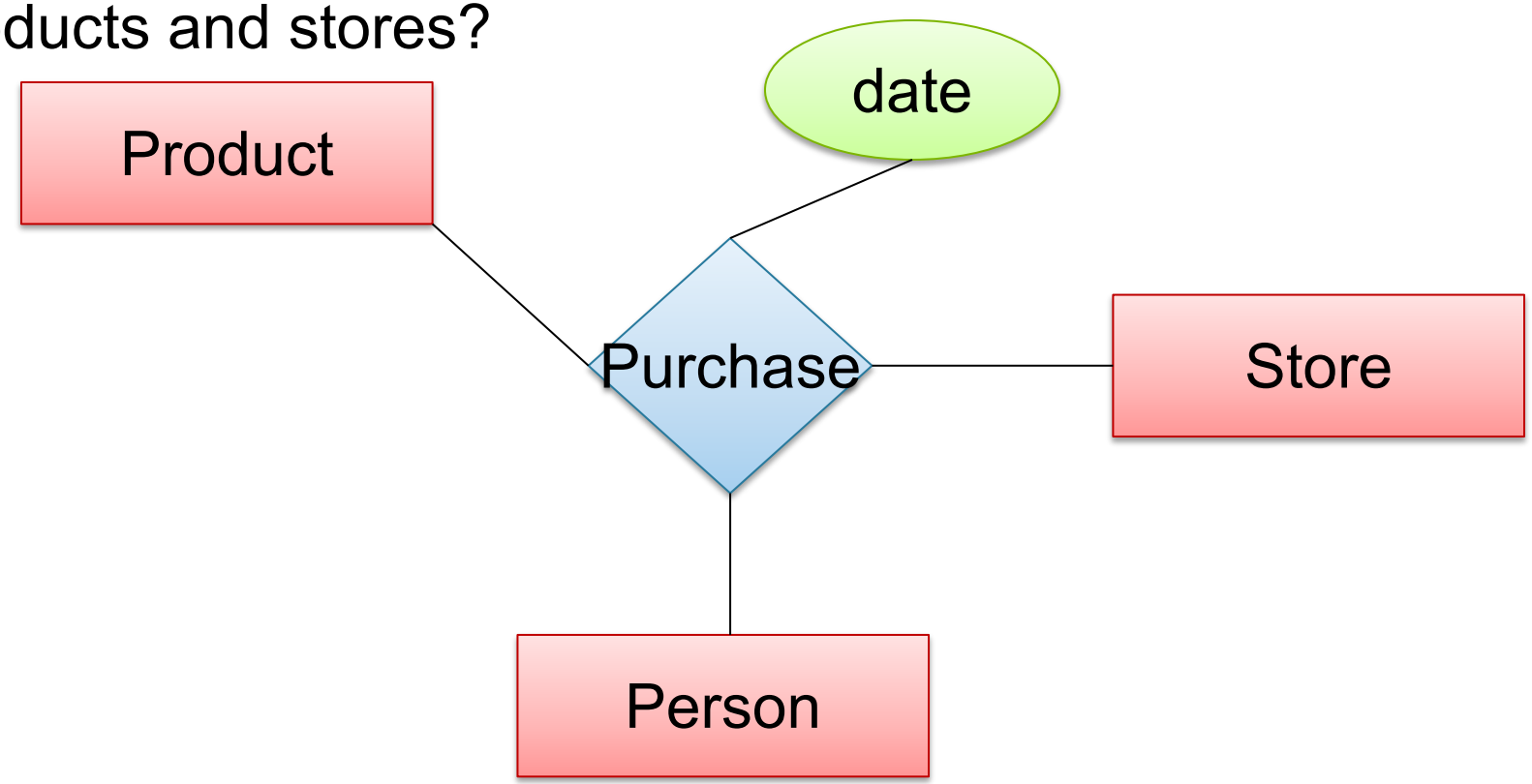

name
price
Product
makes
CEO
name
Company
arrow side is one to many
address
What does this say ?
buys
employs
non-arrow side = one to one
Person
one person works for at most one company
but a company employs many people
address
name
ssn

Attributes on Relationships



Multi-way Relationships

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

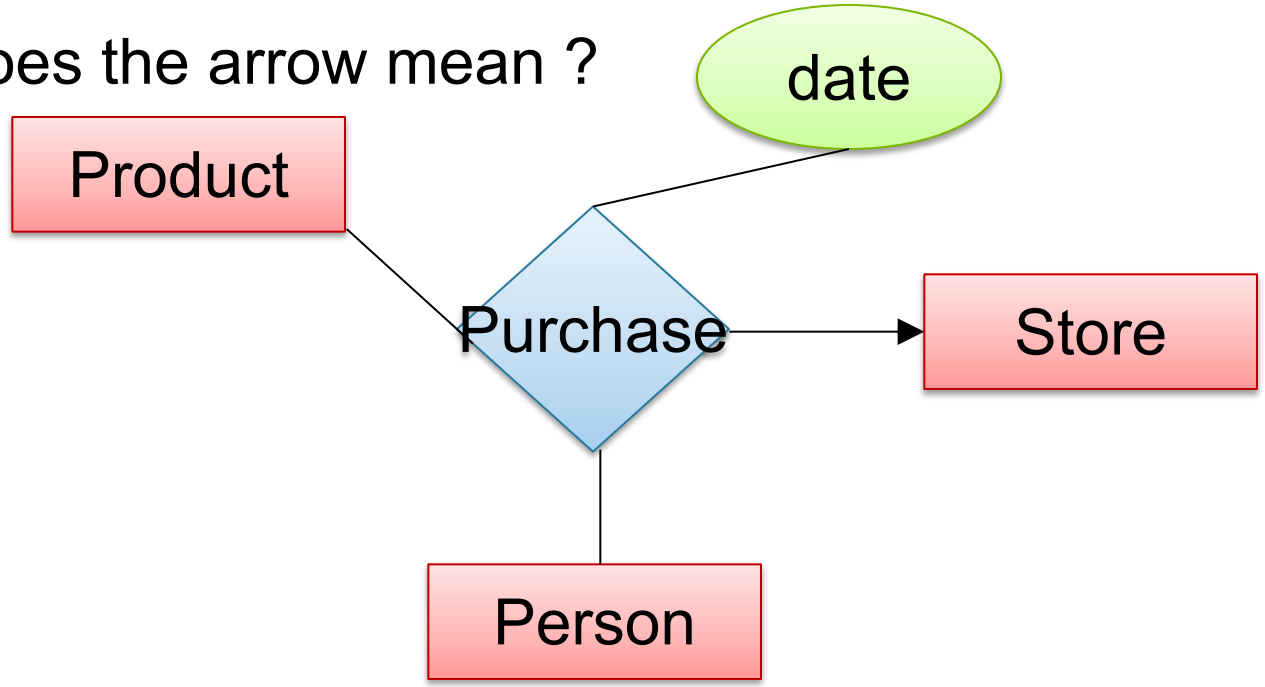


Can still model as a mathematical set (How?)

As a set of triples $\subseteq$ Person $\times$ Product $\times$ Store

Arrows in Multiway Relationships

Q: What does the arrow mean ?

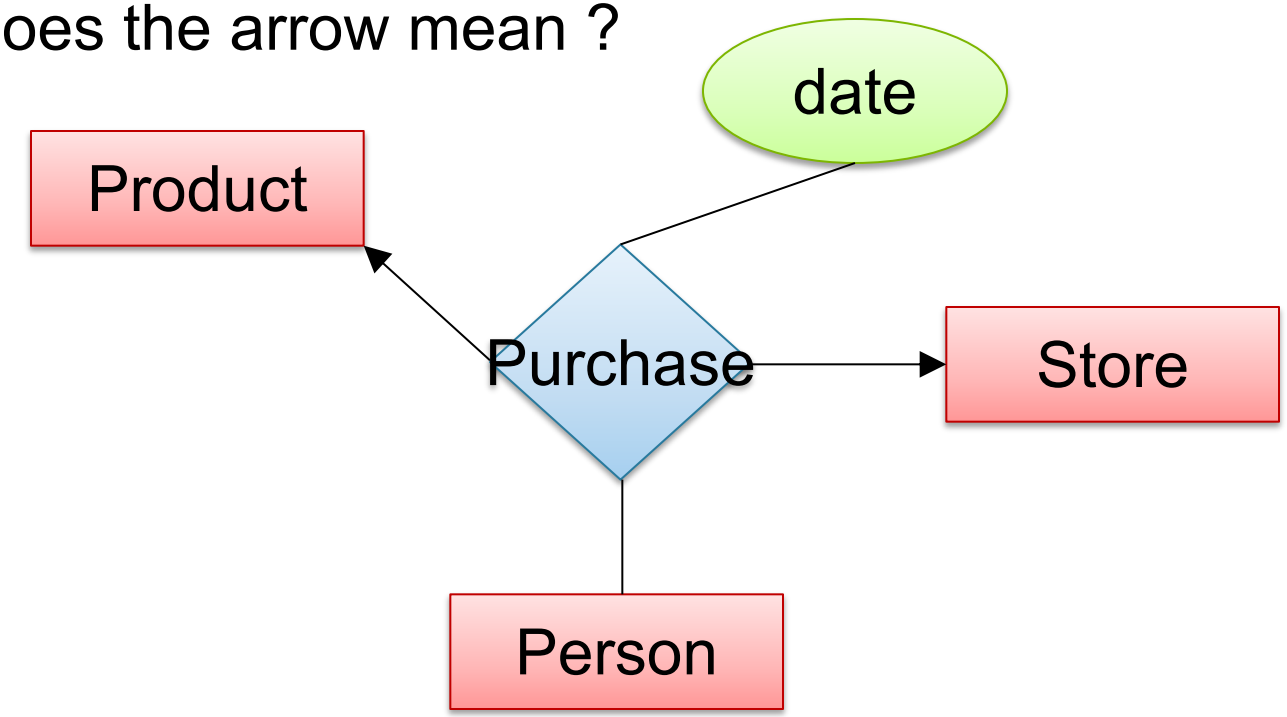


A: Any person buys a given product from at most one store

[Fine print: Arrow pointing to E means that 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]

Arrows in Multiway Relationships

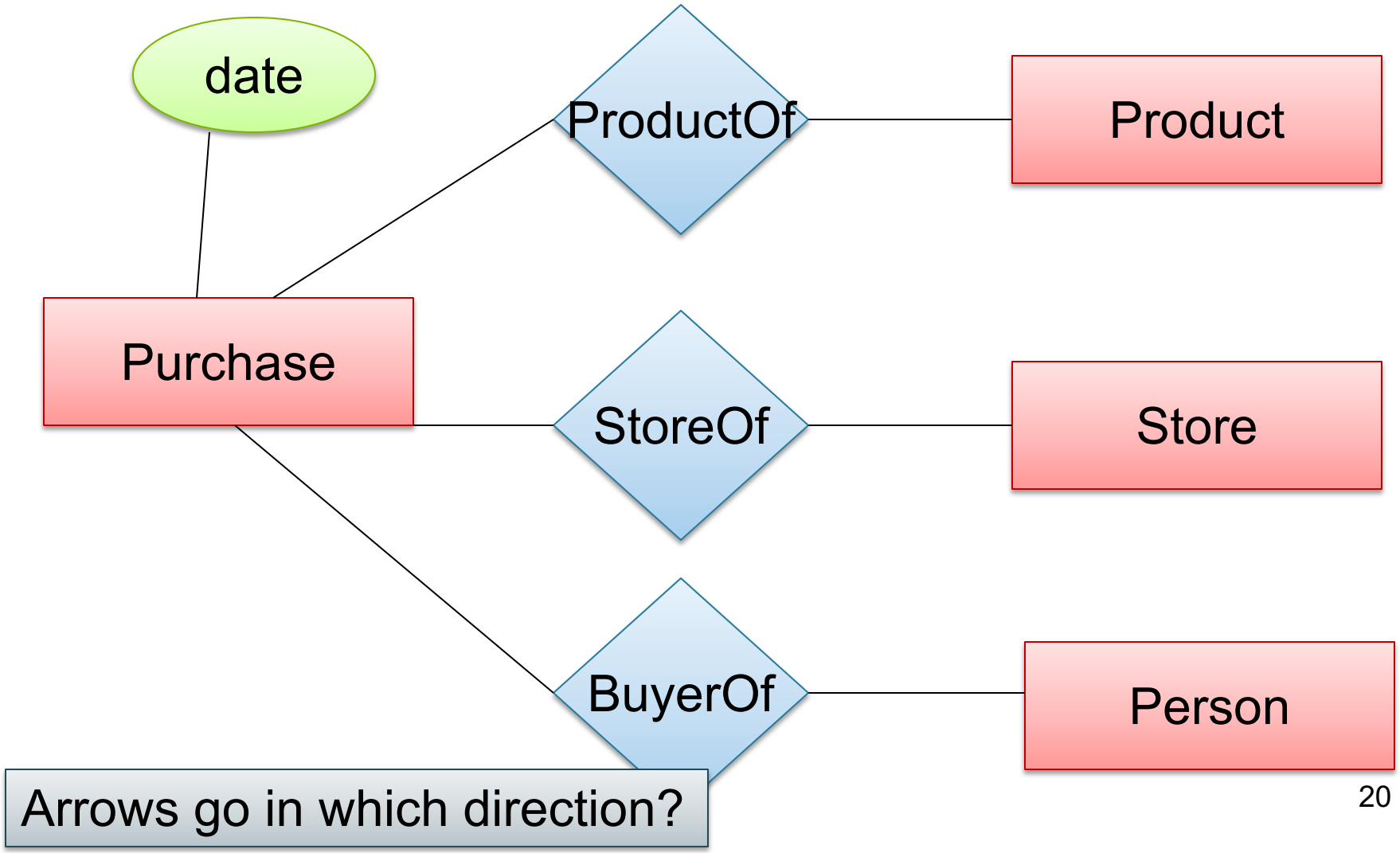
Q: What does the arrow mean ?



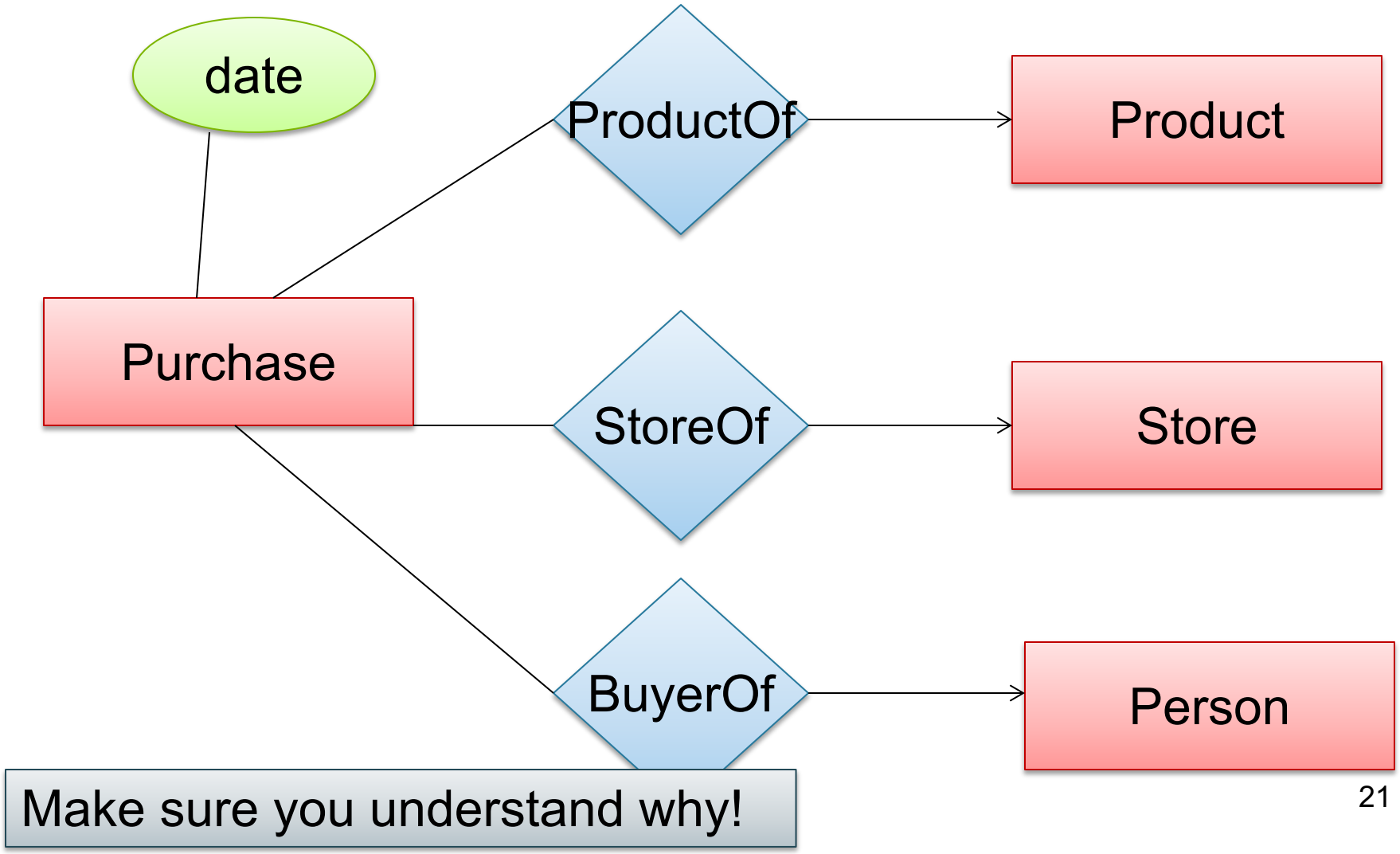
A: Any person buys a given product from at most one store AND every store sells to every person at most one product

Converting Multi-way Relationships to Binary

break a relationship into
something more linear
by converting relation to entity!

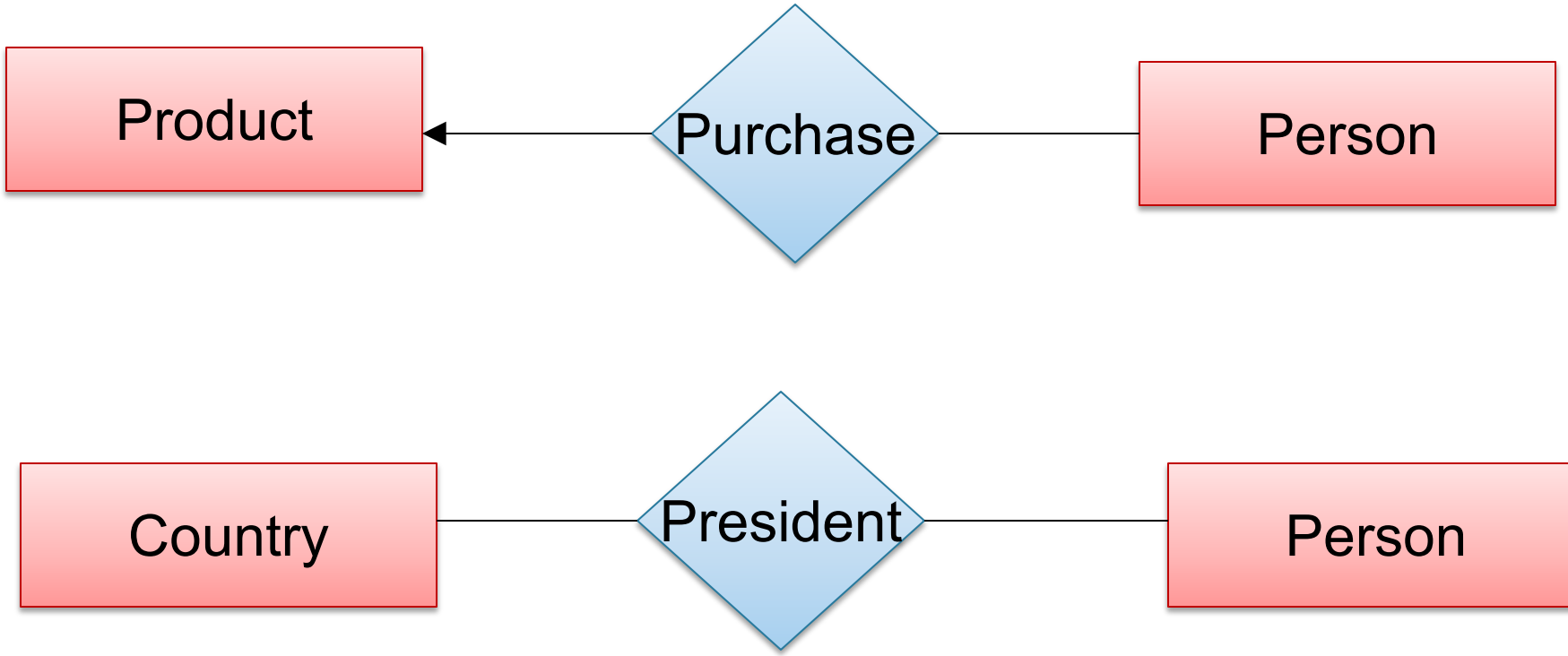


Converting Multi-way Relationships to Binary



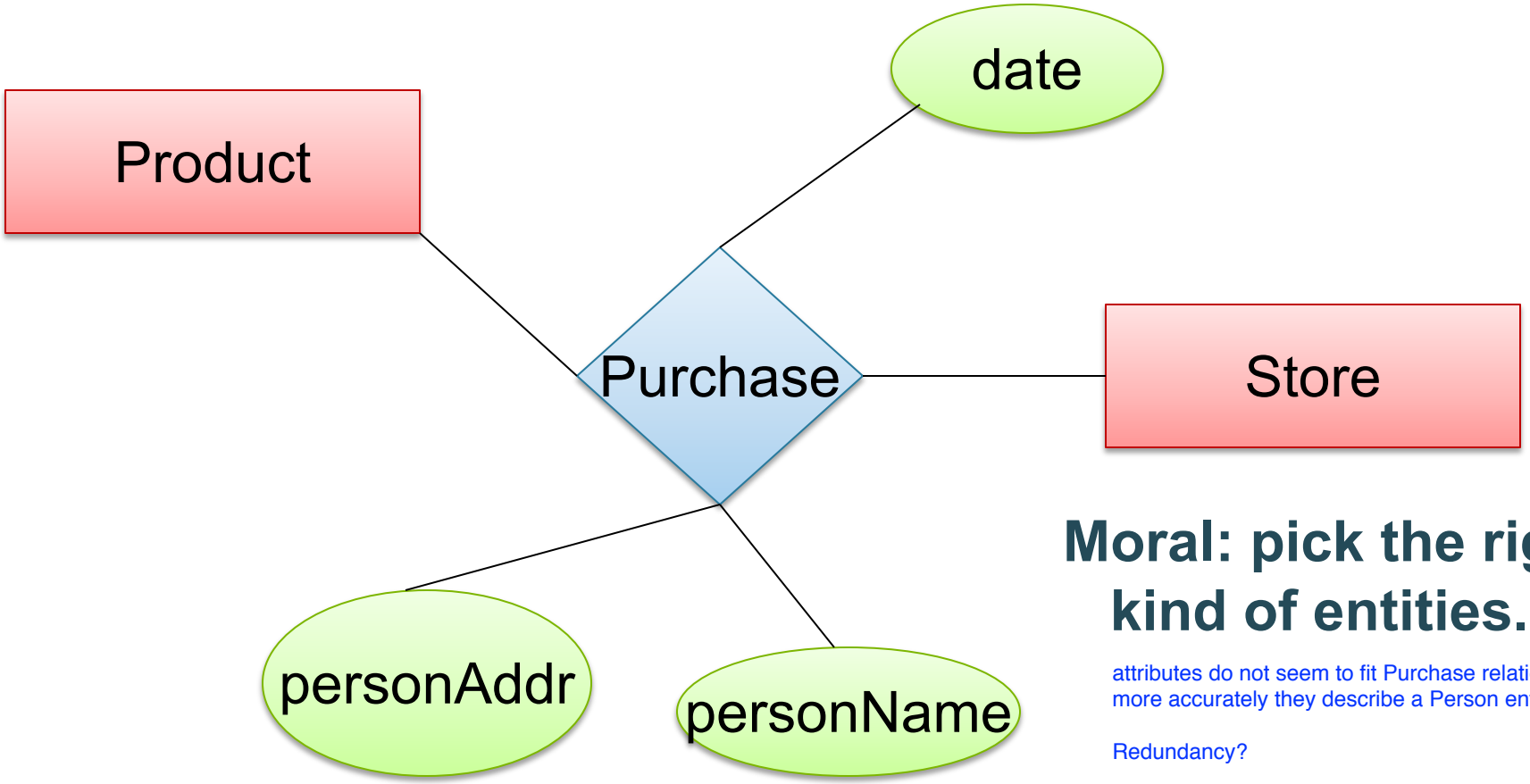
3. Design Principles

What's wrong? many people can purchase one product, but each person can only purchase at most one product



Moral: Be faithful to the specifications of the application!

Design Principles: What's Wrong?

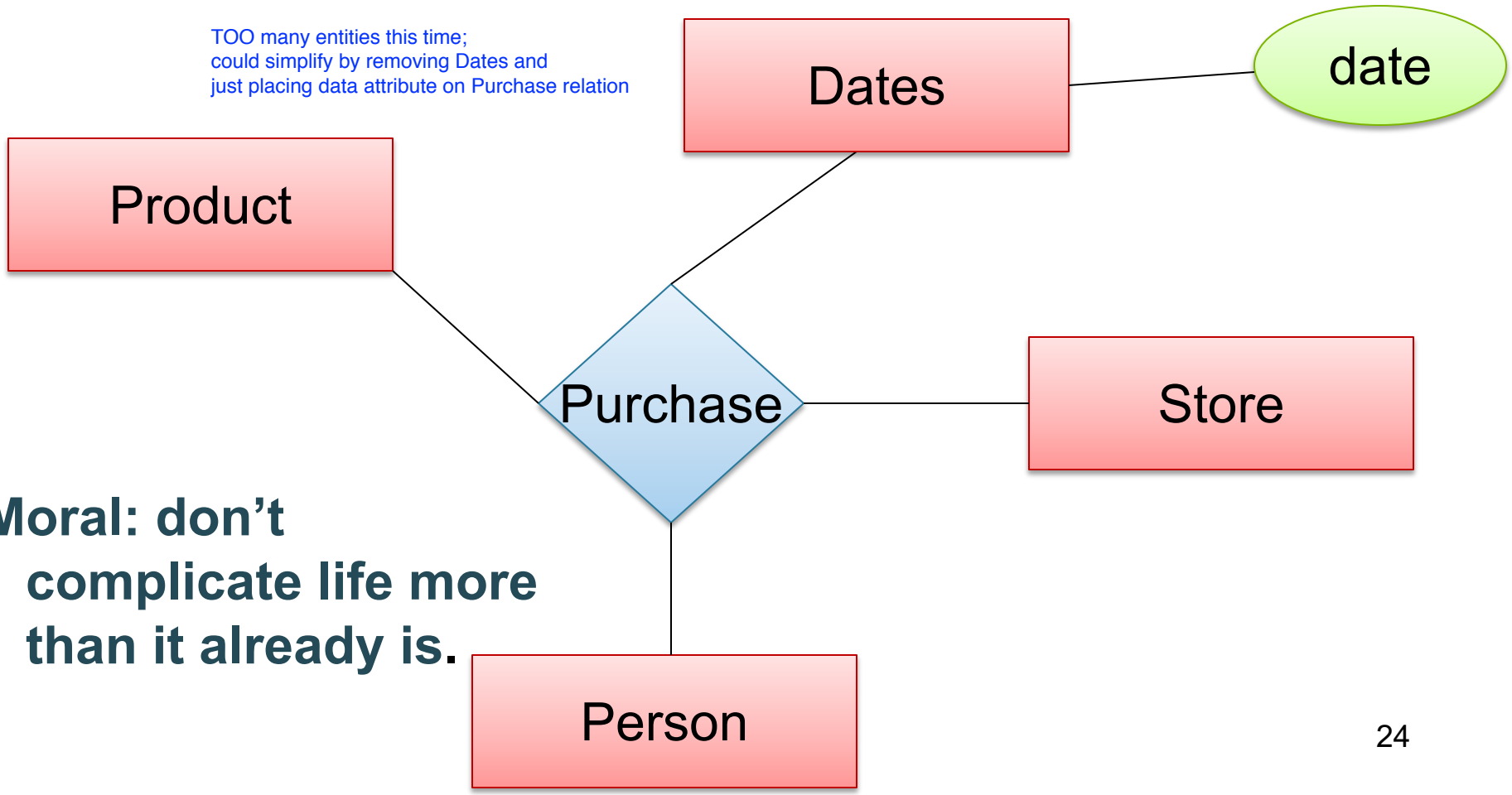


Moral: pick the right kind of entities.

attributes do not seem to fit Purchase relation well; more accurately they describe a Person entity

Redundancy?

Design Principles: What's Wrong?



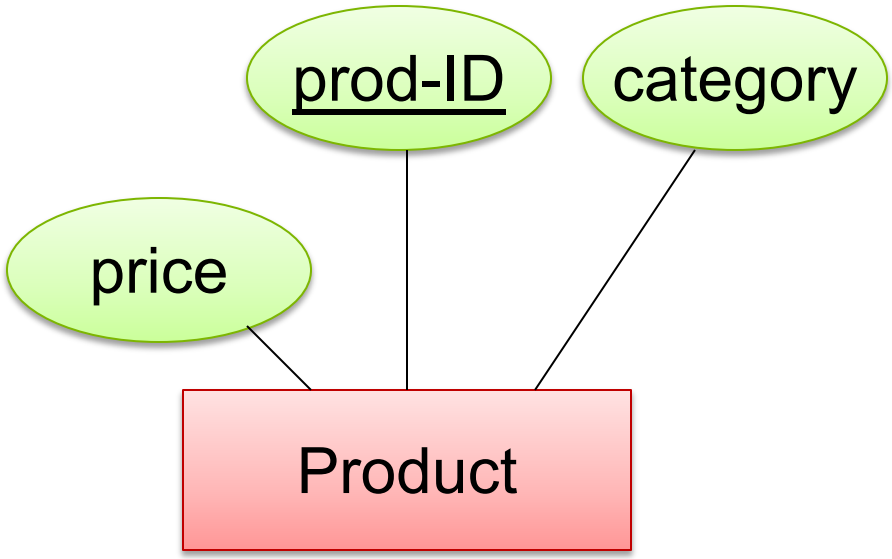
Moral: don't complicate life more than it already is.

From E/R Diagrams to Relational Schema

Converting these E/R Diagrams into relations we can store in tables

- Entity set → relation
- Relationship → relation

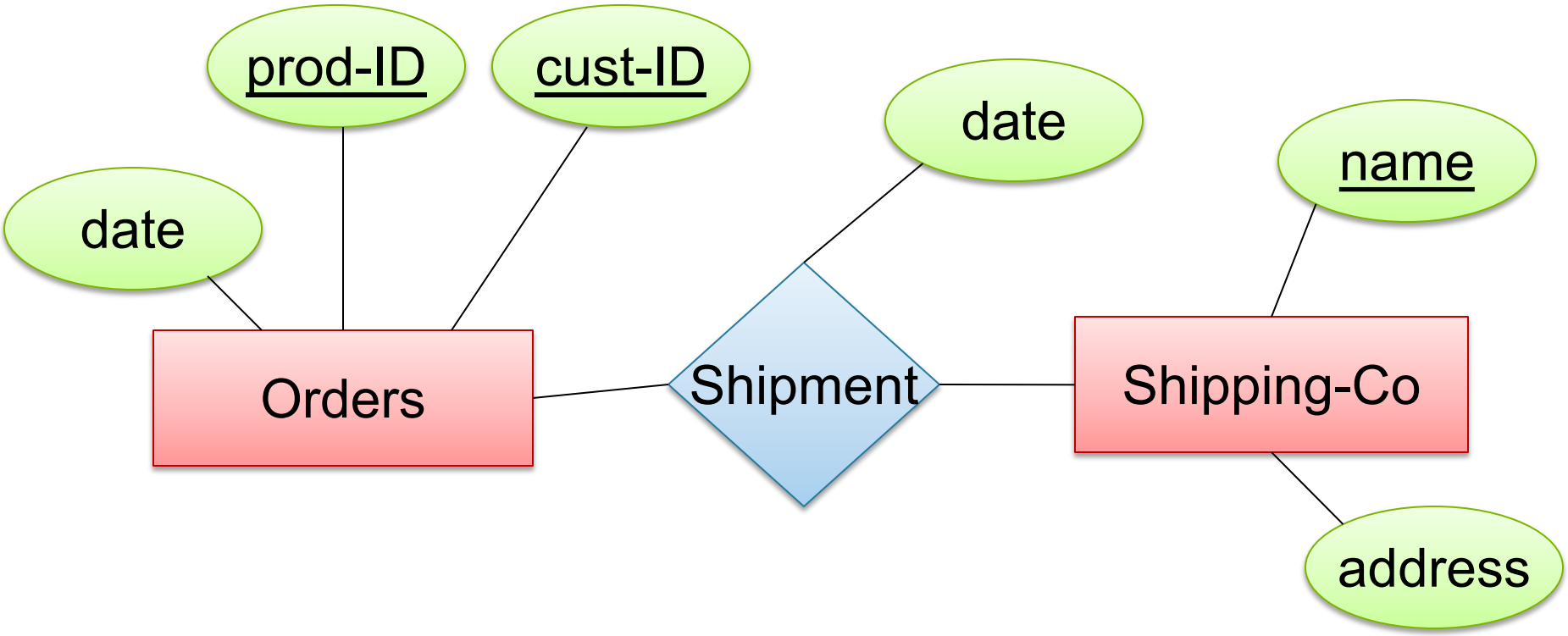
Entity Set to Relation



Product(prod-ID, category, price)

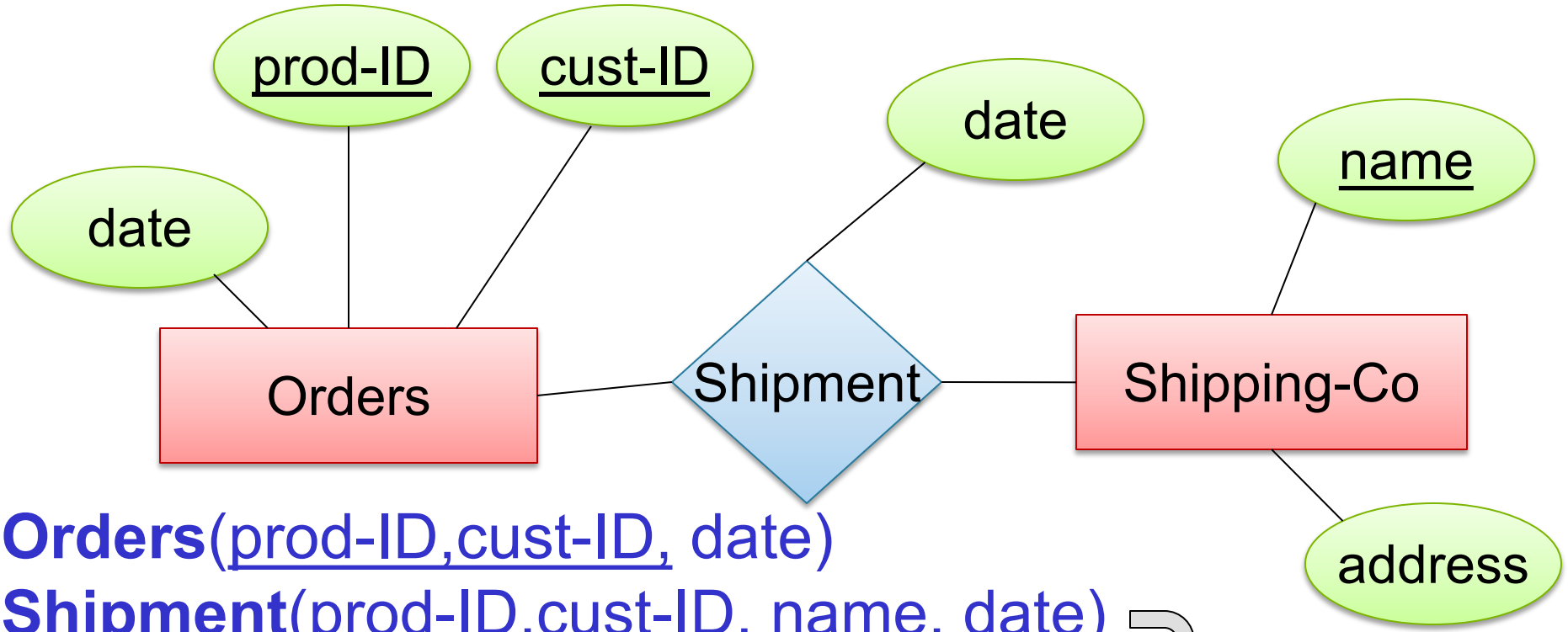
prod-ID	category	price
Gizmo55	Camera	99.99
Pokemn19	Toy	29.99

N-N Relationships to Relations



Represent this in relations

N-N Relationships to Relations

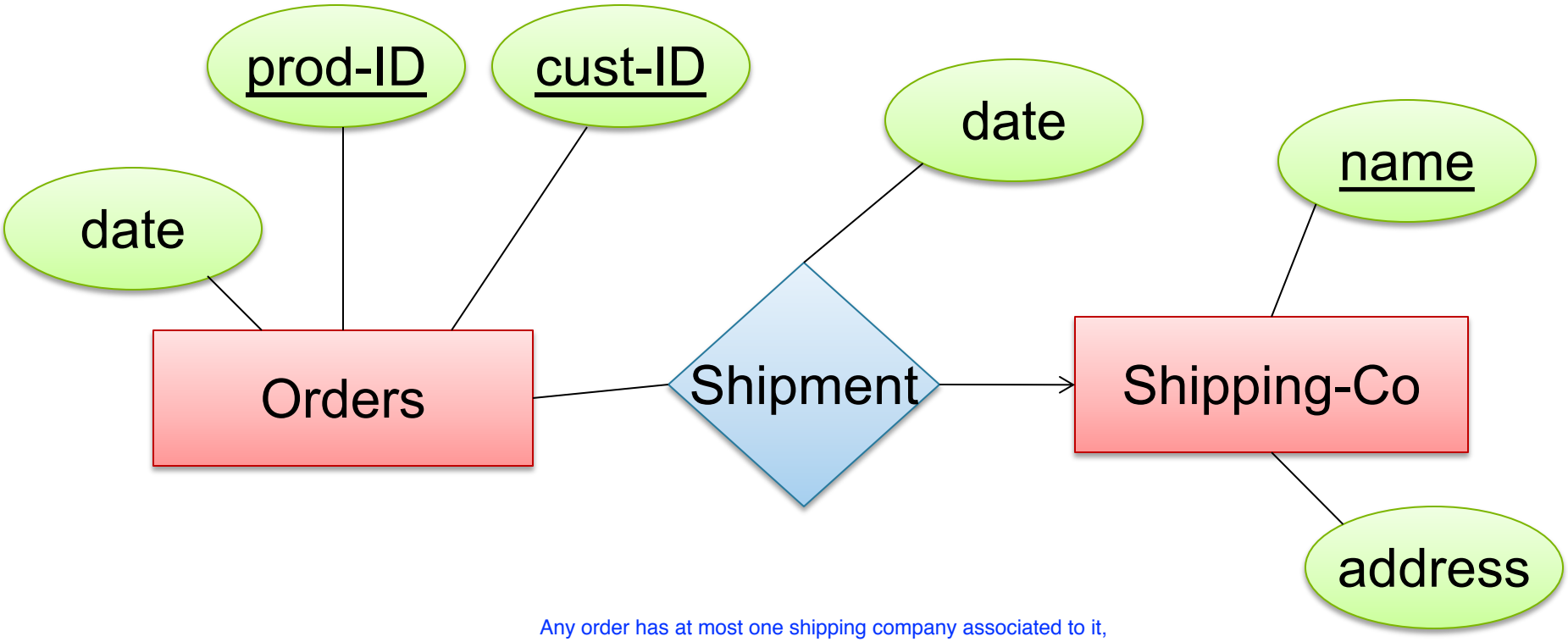


Orders(prod-ID,cust-ID, date)
Shipment(prod-ID,cust-ID, name, date)
Shipping-Co(name, address)

prod-ID	cust-ID	name	date
Gizmo55	Joe12	UPS	4/10/2011
Gizmo55	Joe12	FEDEX	4/9/2011

many to one

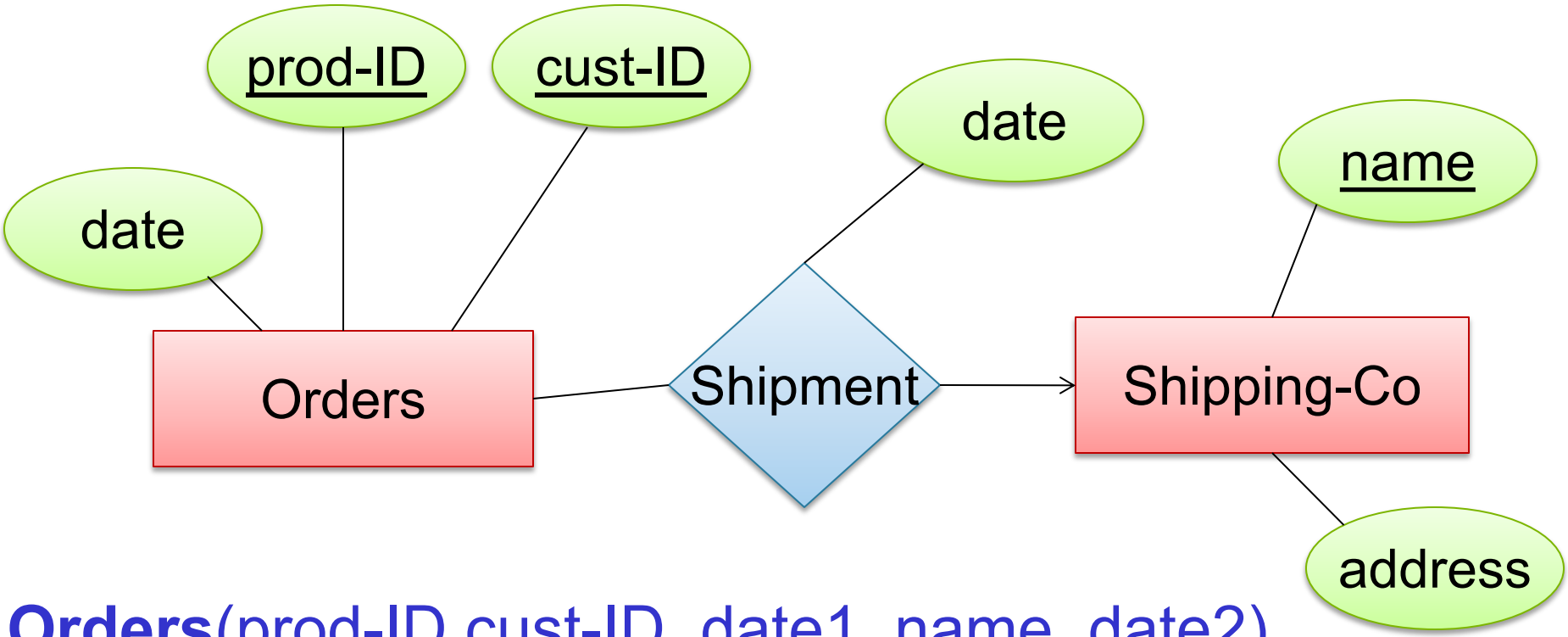
N-1 Relationships to Relations



Any order has at most one shipping company associated to it,
but shipping company may ship many orders

Represent this in relations

N-1 Relationships to Relations



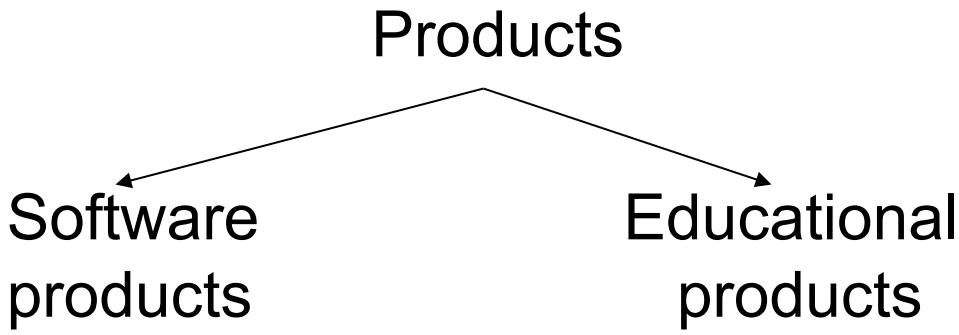
Orders(prod-ID,cust-ID, date1, name, date2)
Shipping-Co(name, address)

Remember: no separate relations for many-one relationship

Modeling Subclasses

Product

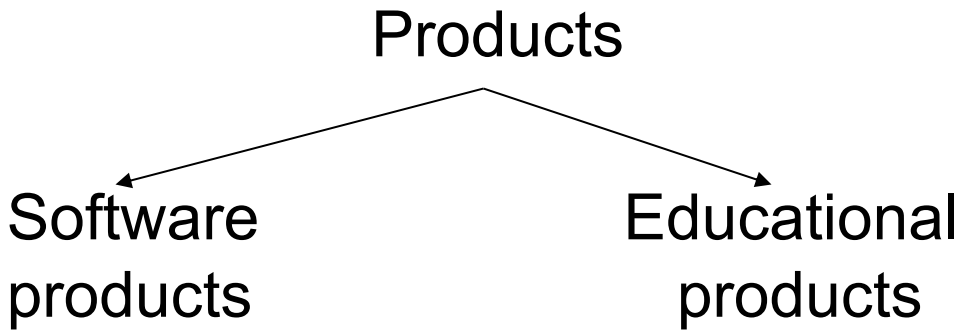
Name	Price	Category	Platforms	Age-group
Gizmo	99	gadget	unix	NULL
Camera	49	photo	NULL	NULL
Toy	39	gadget	NULL	infant



Modeling Subclasses

Product

Name	Price	Category	Platforms	Age-group
Gizmo	99	gadget	unix	NULL
Camera	49	photo	NULL	NULL
Toy	39	gadget	NULL	infant



Modeling Subclasses

Product

Name	Price	Category
Camera	49	photo

Software Product

Name	Price	Category	Platforms
Gizmo	99	gadget	unix

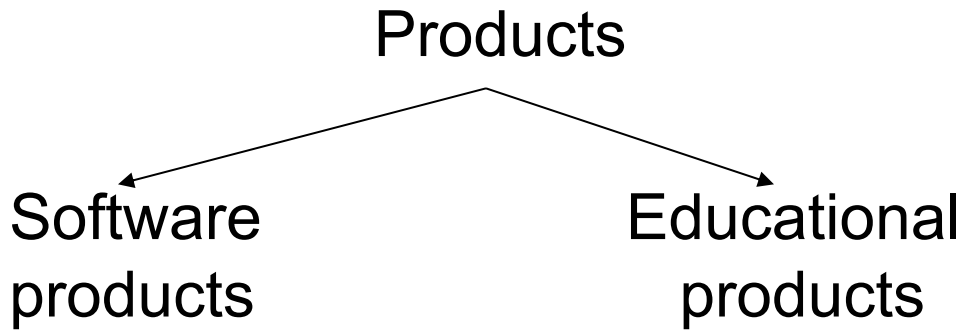
Educational Product

Name	Price	Category	Age-group
Toy	39	gadget	infant

Modeling Subclasses

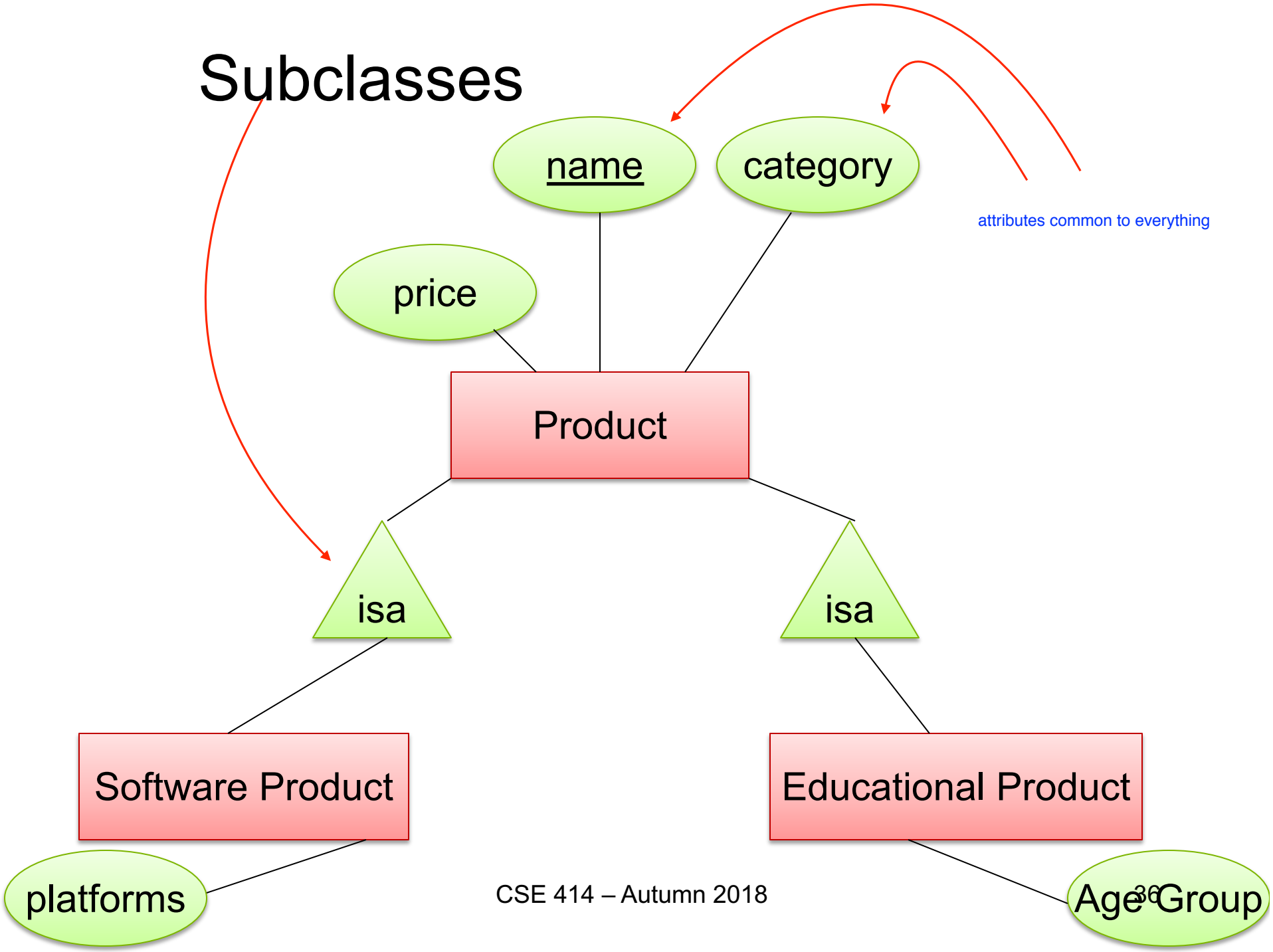
Some objects in a class may be special

- define a new class
- better: define a *subclass*

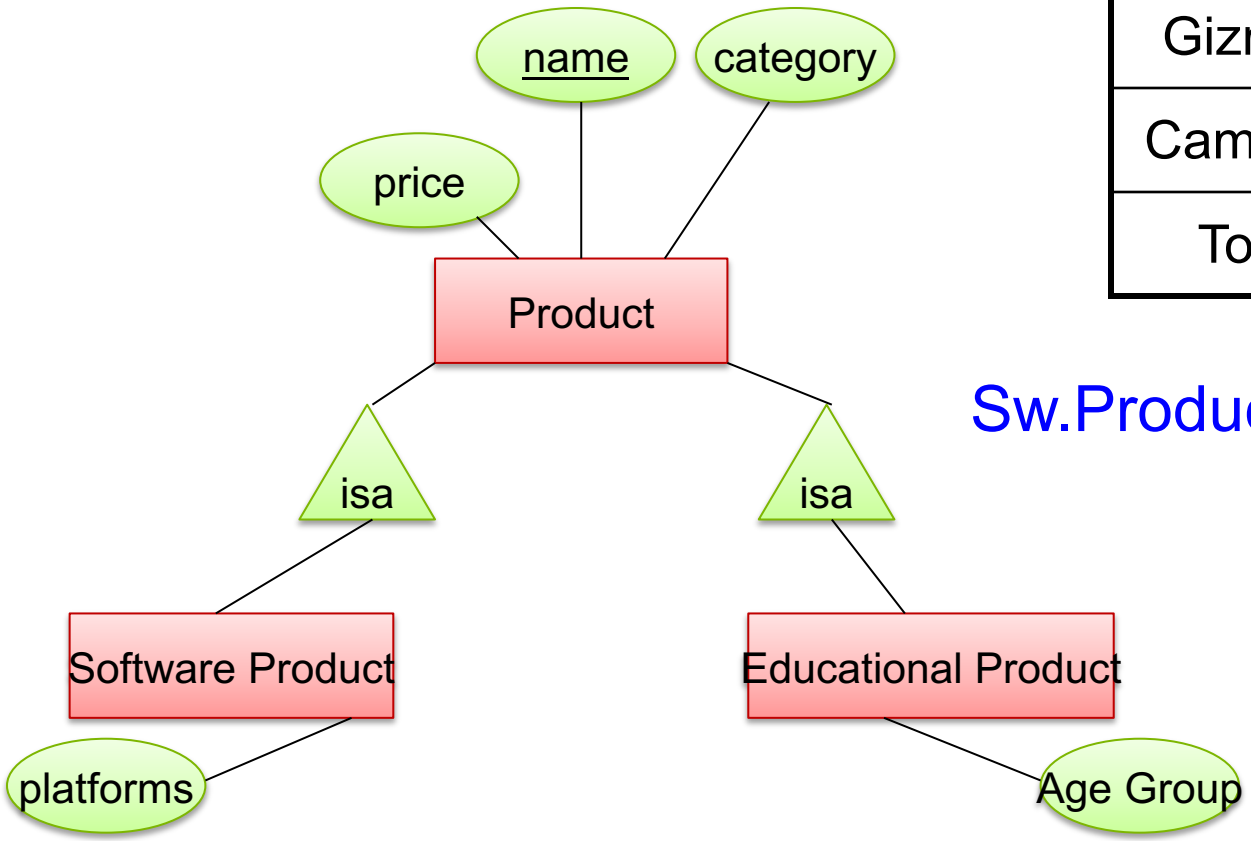


So --- we define subclasses in E/R

Subclasses


name
category
attributes common to everything
price
Product
isa
isa
Software Product
Educational Product
platforms
Age Group

Subclasses to Relations



Product

Name	Price	Category
Gizmo	99	gadget
Camera	49	photo
Toy	39	gadget

Sw.Product

Name	platforms
Gizmo	unix

Ed.Product

Name	Age Group
Gizmo	toddler
Toy	retired

Other ways to convert are possible

Modeling Union Types with Subclasses

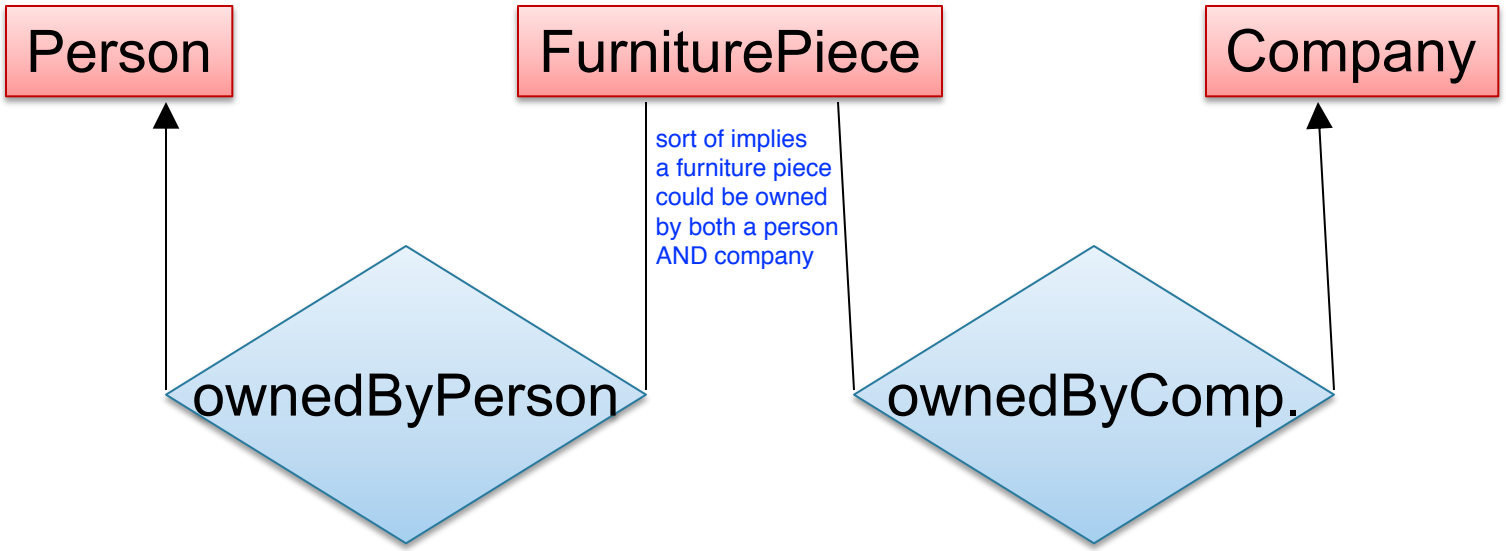
each object (furniture piece) is owned by either one person or one company

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

Modeling Union Types with Subclasses

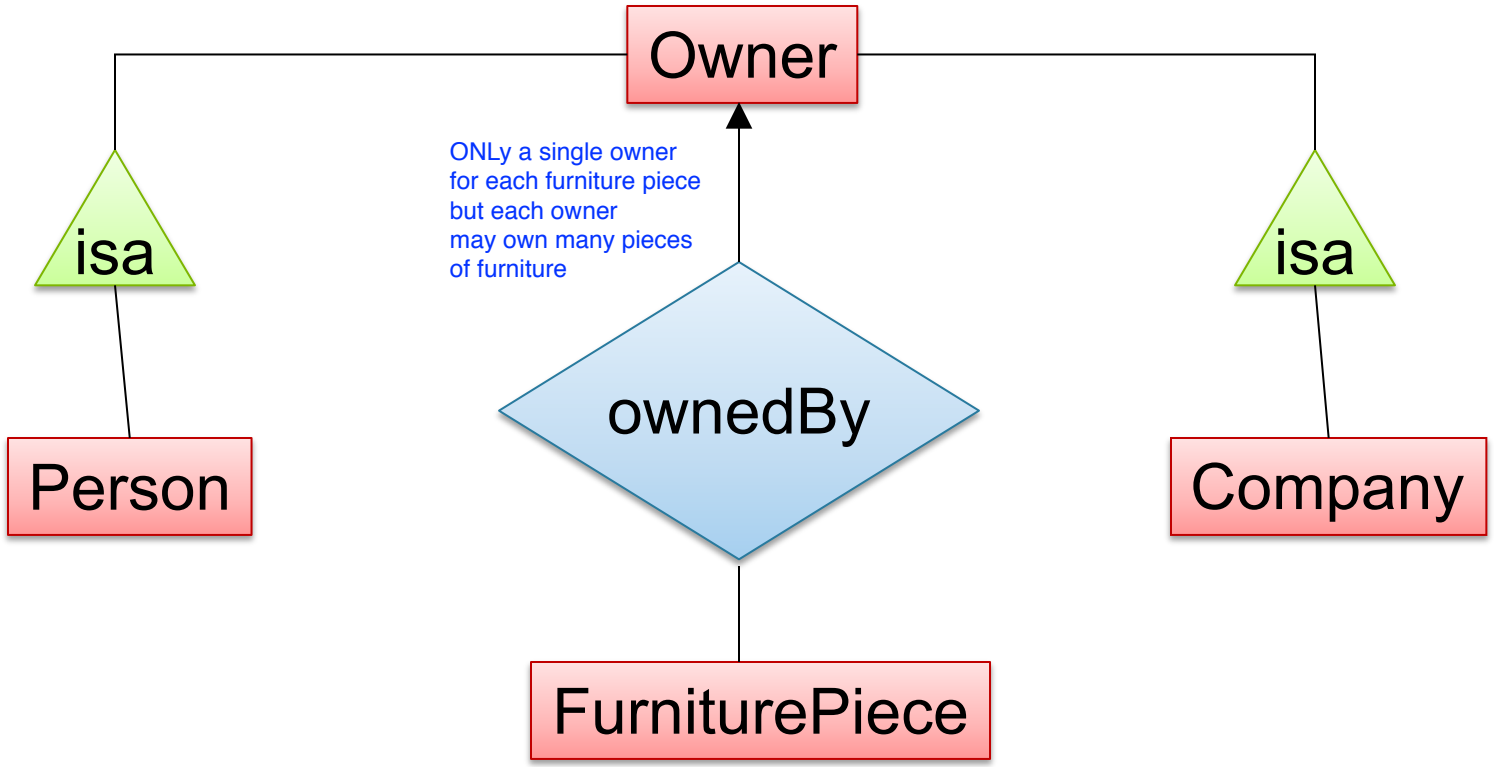
Say: each piece of furniture is owned either by a person or by a company

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



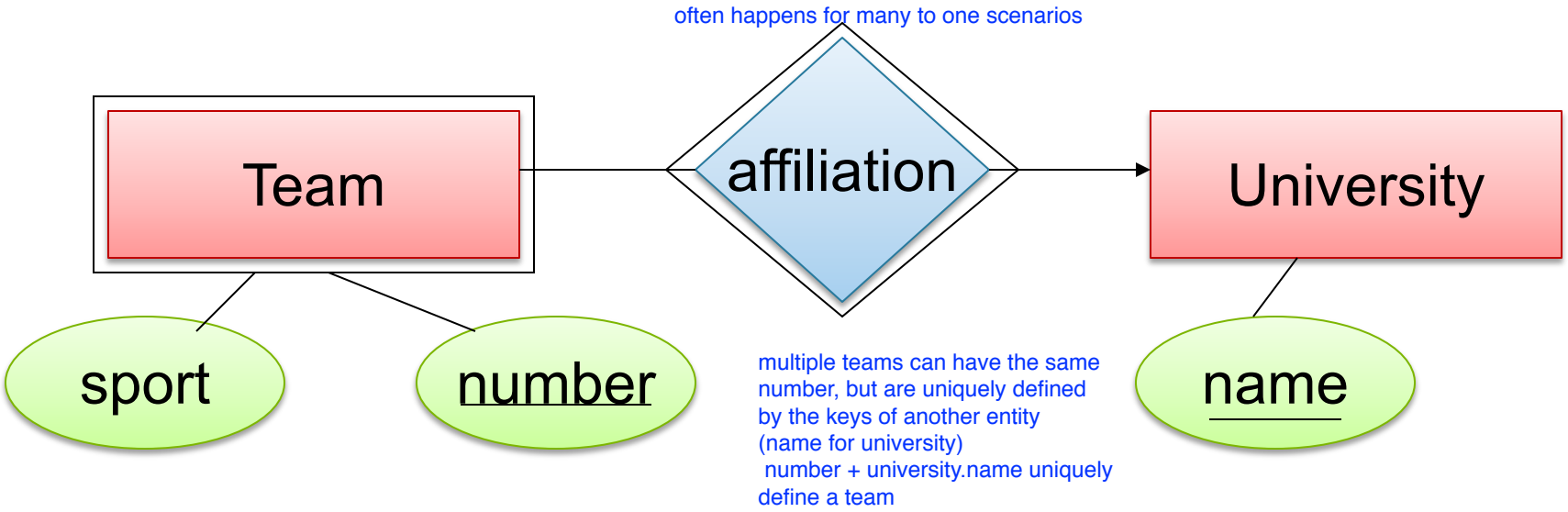
Modeling Union Types with Subclasses

Solution 2: better, more laborious



Weak Entity Sets

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



Team(sport, number, universityName)
University(name)

Introduction to Data Management
CSE 344

Integrity Constraints

Integrity Constraints Motivation

An integrity constraint is a condition specified on a database schema that restricts the data that can be stored in an instance of the database.

- ICs help prevent entry of incorrect information
- How? DBMS enforces integrity constraints
  - Allows only legal database instances (i.e., those that satisfy all constraints) to exist
  - Ensures that all necessary checks are always performed and avoids duplicating the verification logic in each application

Constraints in E/R Diagrams

Finding constraints is part of the modeling process.
Commonly used constraints:

Keys: social security number uniquely identifies a person.

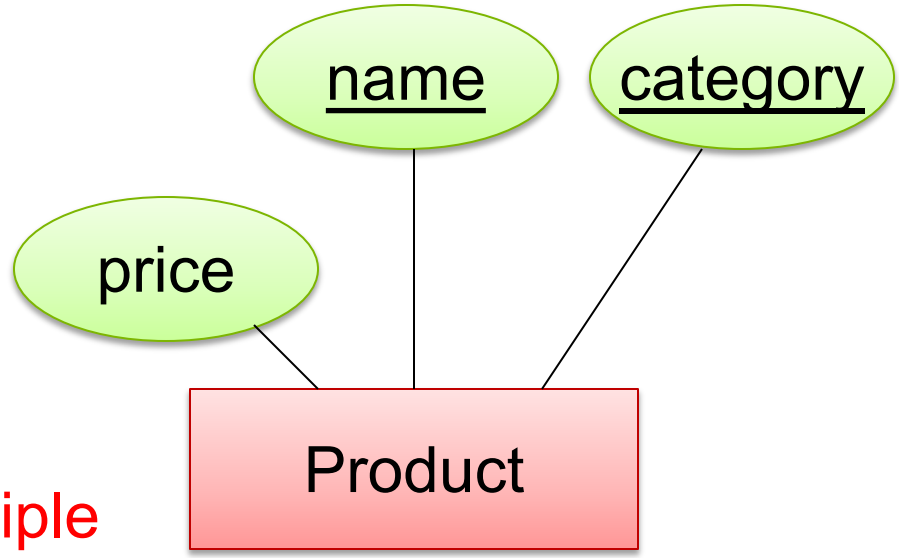
Single-value constraints: a person can have only one father.

Referential integrity constraints: if you work for a company, it must exist in the database.

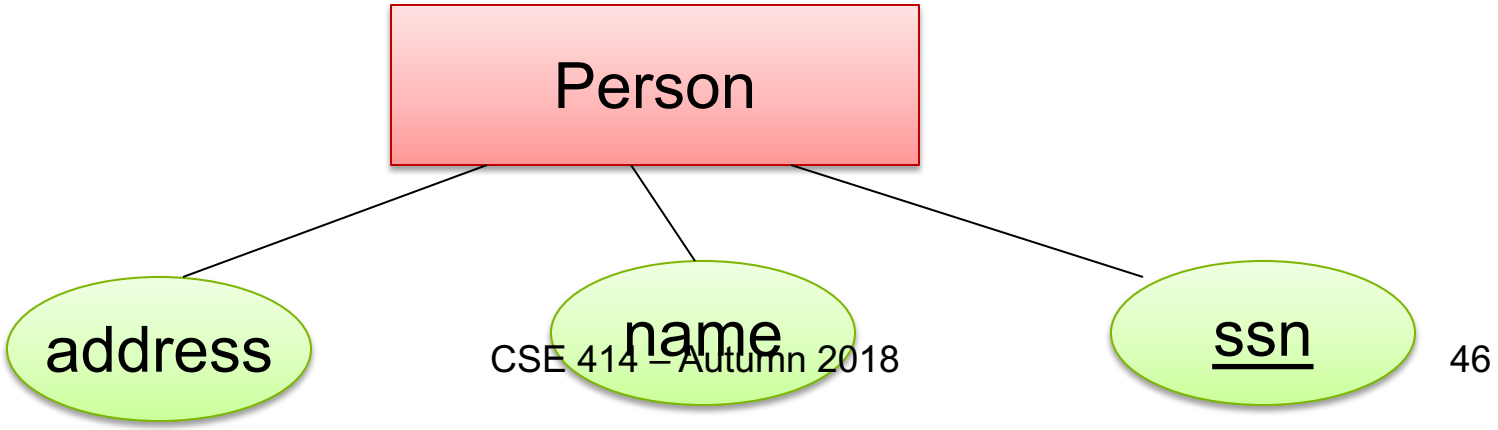
Other constraints: peoples' ages are between 0 and 150.

Keys in E/R Diagrams

Underline:



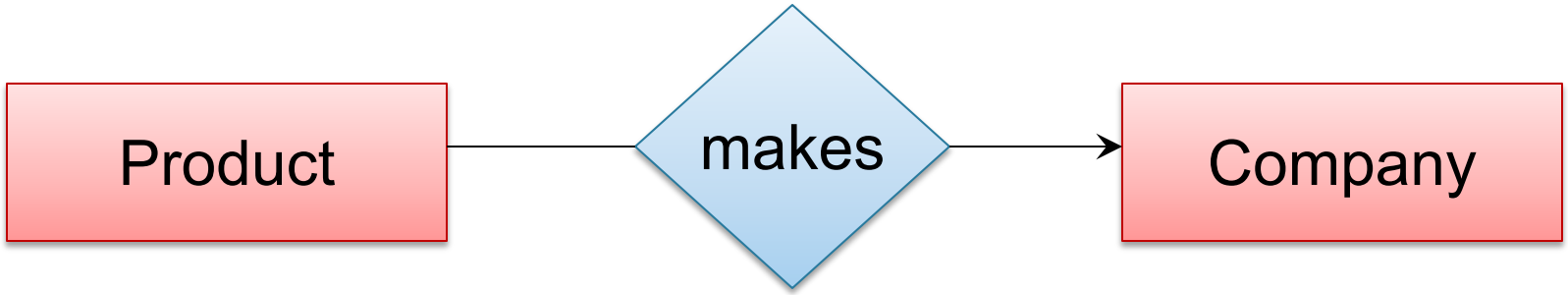
No formal way
to specify multiple
keys in E/R diagrams



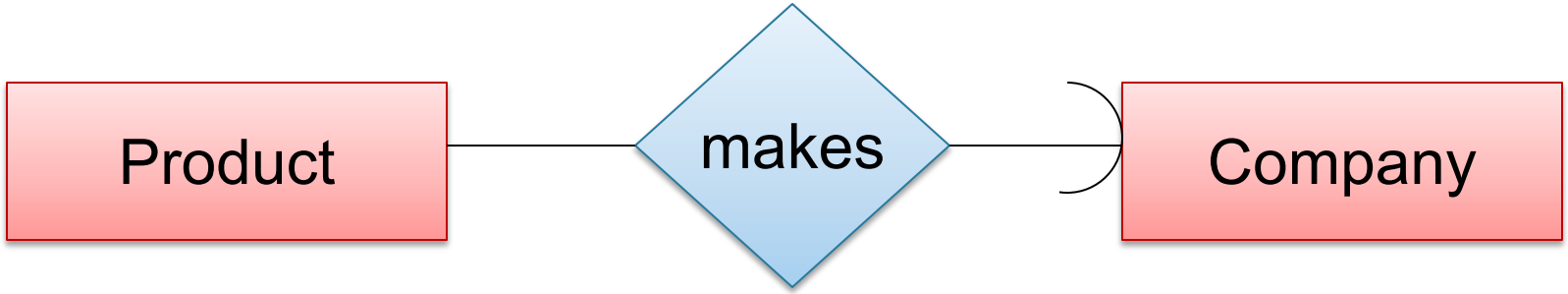
Single Value Constraints

vs.

Referential Integrity Constraints



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



Each product made by *exactly* one company.