Entity-Relationship Model

CS 564- Spring 2018

WHAT IS THIS LECTURE ABOUT

E/R Model:

- entity sets, attribute
- relation: binary, multi-way
- relationship roles, attributes on relationships
- subclasses (ISA)
- weak entity sets
- constraints
- design principles
- E/R to Relational Model

HOW TO BUILD A DB APPLICATION

I THINK WE SHOULD BUILD AN SQL DATABASE.
UH-OH
DOES HE UNDERSTAND WHAT HE SAID OR IS IT SOMETHING HE SAW IN A TRADE MAGAZINE AD?
WHAT COLOR DO YOU WANT THAT DATABASE?
I THINK MAUVE HAS THE MOST RAM.
E-mail: SCOTTADAMS@AOL.COM
S. Adams
© 1995 United Feature Syndicate, Inc.(NYC)

HOW TO BUILD A DB APPLICATION

- Pick an application
- Figure out what to model (ER model)
  - Output: ER diagram
- Transform the ER diagram to a relational schema
- Refine the relational schema (normalization)
- Now ready to implement the schema and load the data!

RUNNING EXAMPLE

We want to store information about:

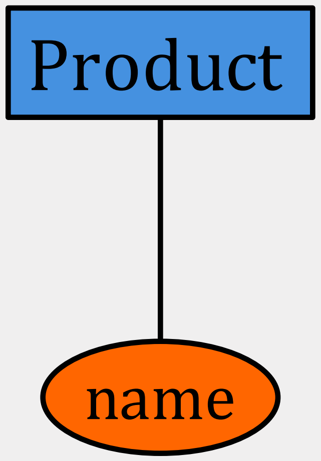
- companies and employees
  - Each **company** has a name, an address, …
  - Each company has a list of **employees**
- products manufactured by these companies
  - Each **product** has a name, a description, …

E/R MODEL

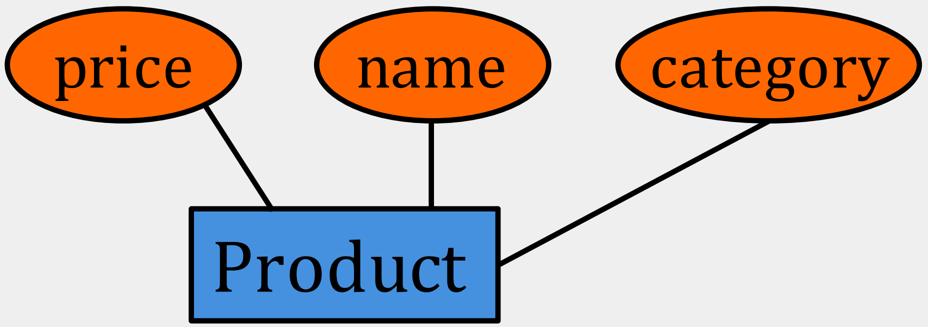
- Gives us a **<u>visual language</u>** to specify
  - what information the DB must hold
  - what are the relationships among components of that information
- Proposed by Peter Chen in 1976
- What we will cover:
  1. basic stuff: entities, attributes, relationships
  2. constraints
  3. weak entity sets
  4. design principles

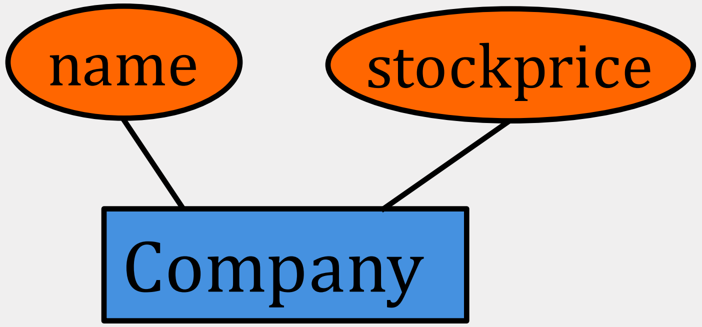
ENTITIES & ATTRIBUTES

- **Entity**
  - an object distinguishable from other object
- **Entity set**
  - a collection of similar entities
  - represented by rectangles
  - described using a set of attributes
- **Attribute**
  - represented by ovals attached to an entity set

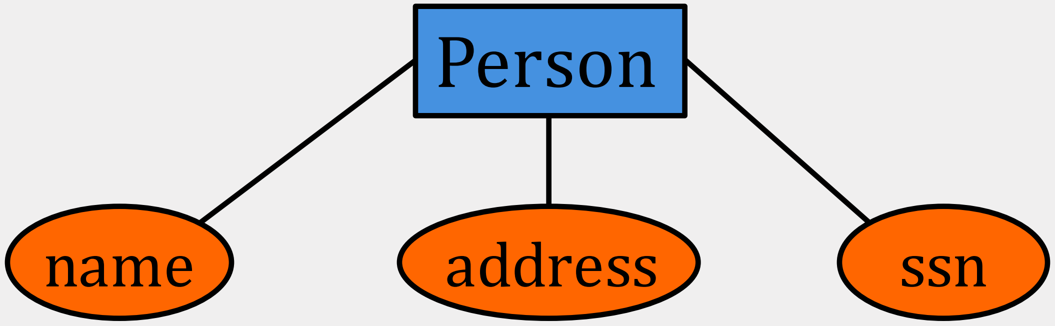


ENTITIES & ATTRIBUTES



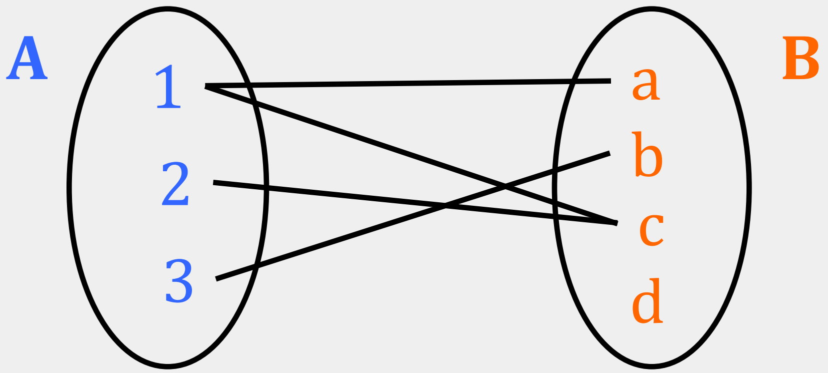


Entities are not explicitly represented in E/R diagrams!

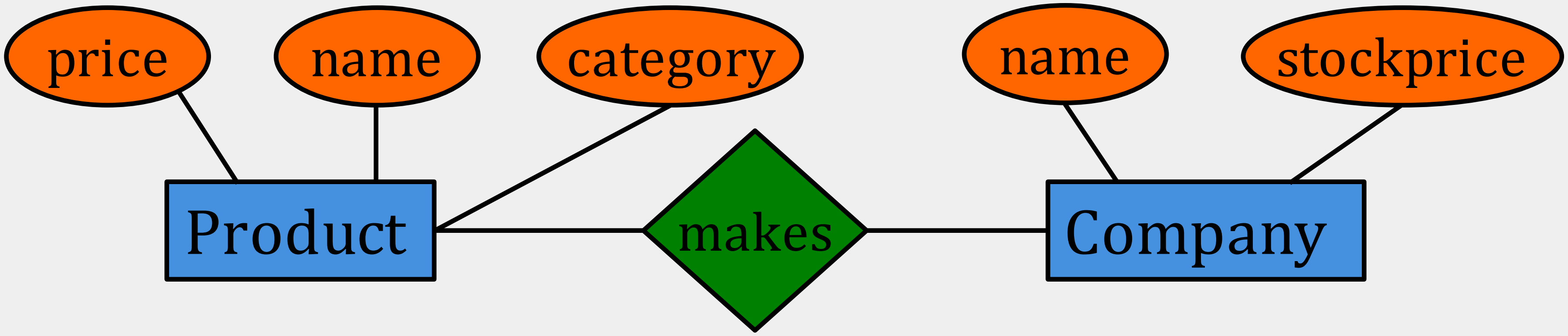


RELATIONS

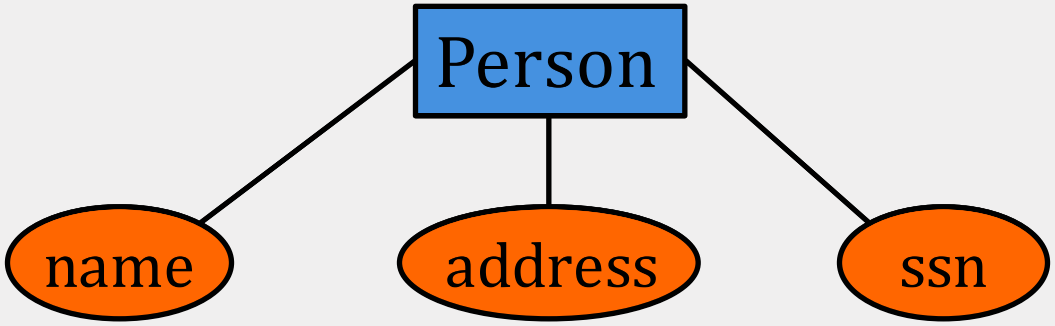
- A mathematical definition:
  - if **A**, **B** are sets, then a **relation** **R** is a subset of **A** x **B**
- Example
  - **A** = {1, 2, 3}, **B** = {a, b, c, d}
  - **R** = {(1, a), (1, c), (2, c), (3, b)}



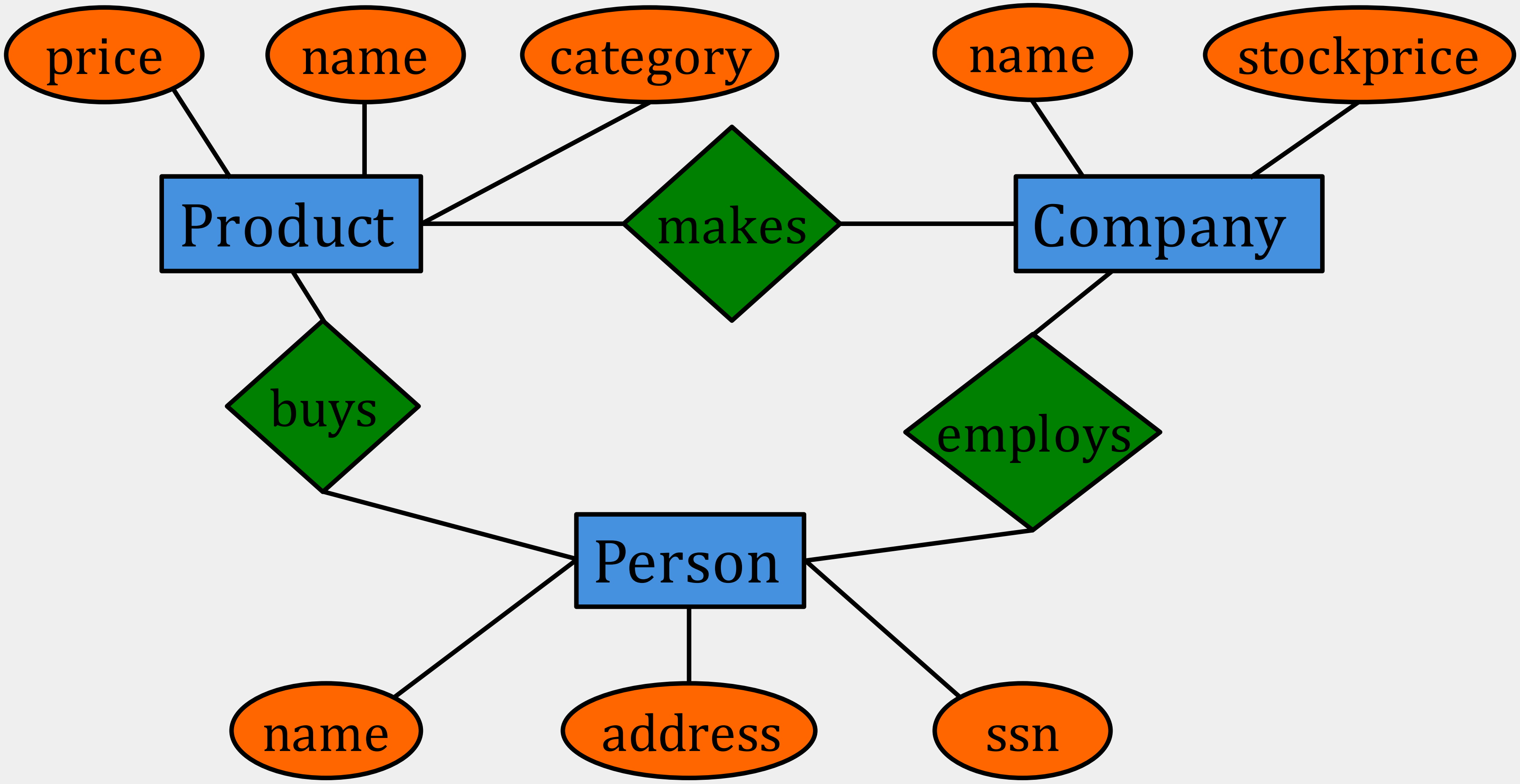
RELATIONSHIPS



makes is a subset of **Product** x **Company**

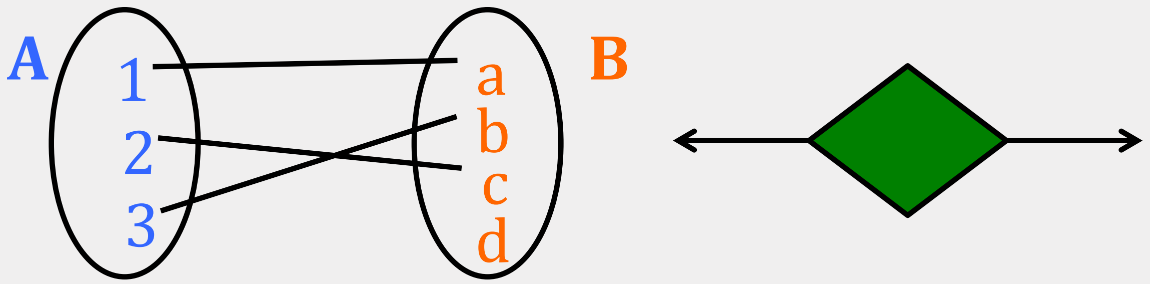


RELATIONSHIPS

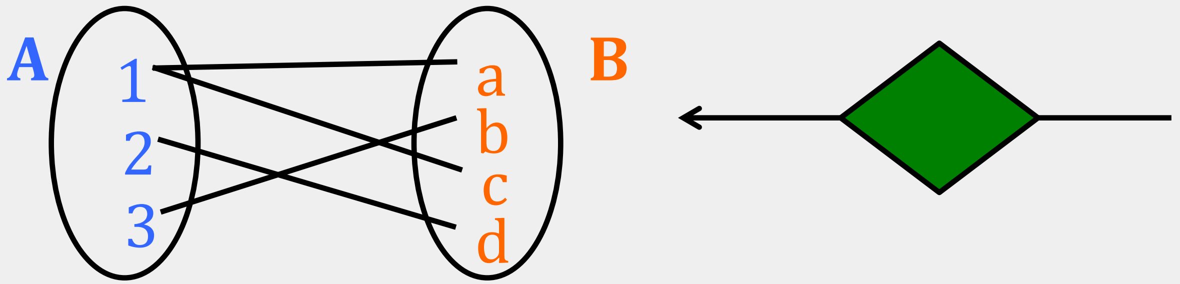


MULTIPLICITY OF RELATIONSHIPS

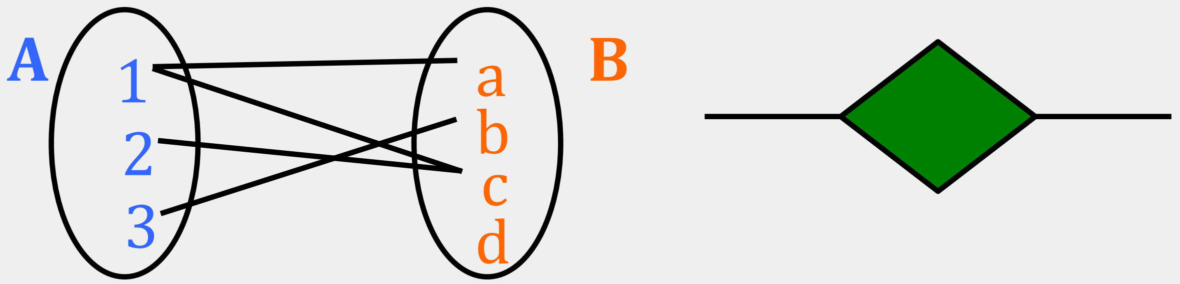
- one-one



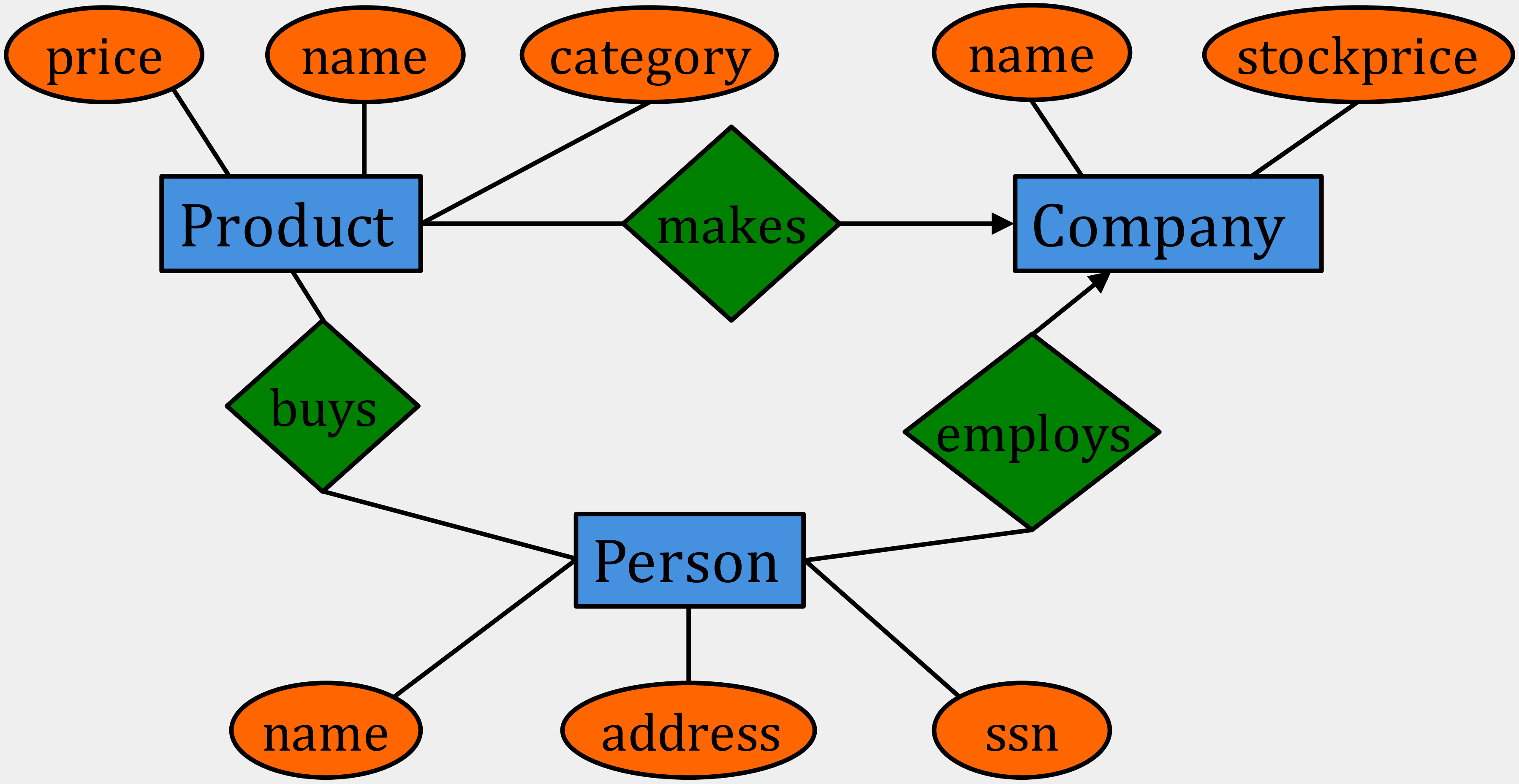
- many-one



- many-many

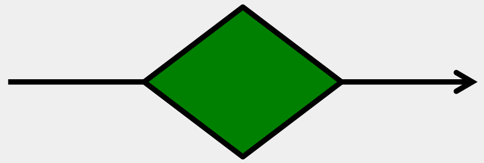


MULTIPLICITY OF RELATIONSHIPS

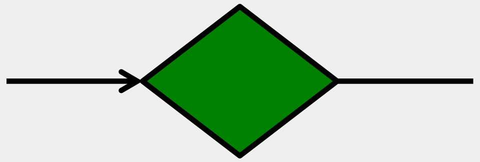


NOTATION DIFFERENCE

- We use:



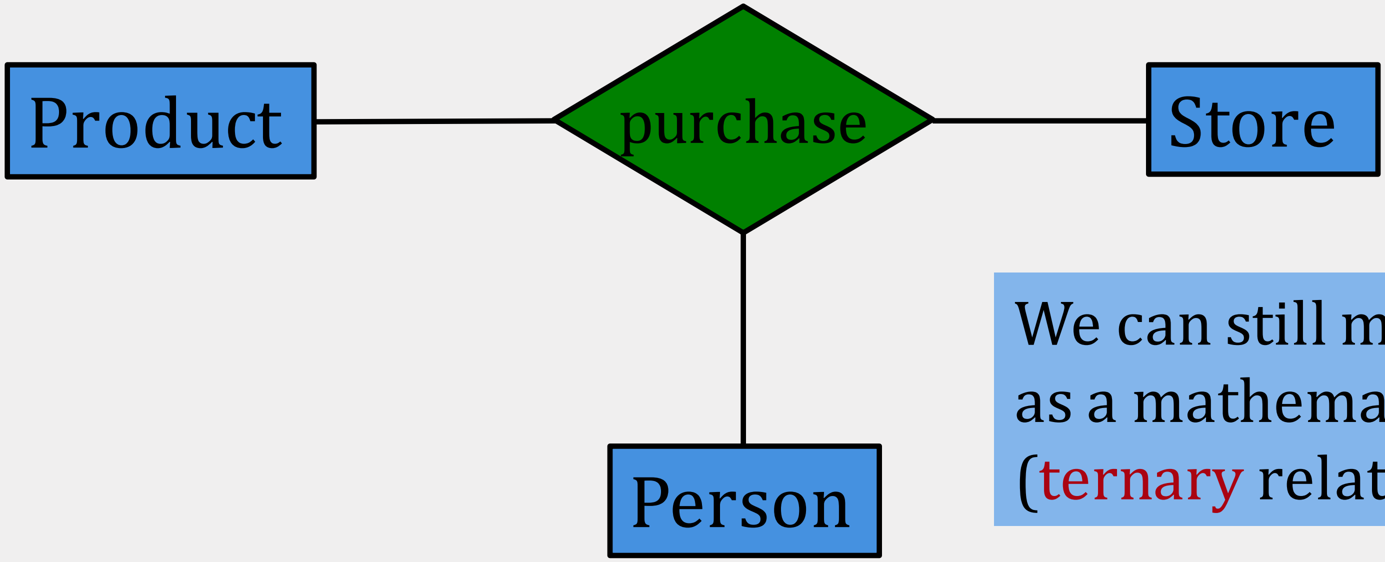
- The cow book uses (page 33):



You should use the notation in the slides!

MULTI-WAY RELATIONSHIPS

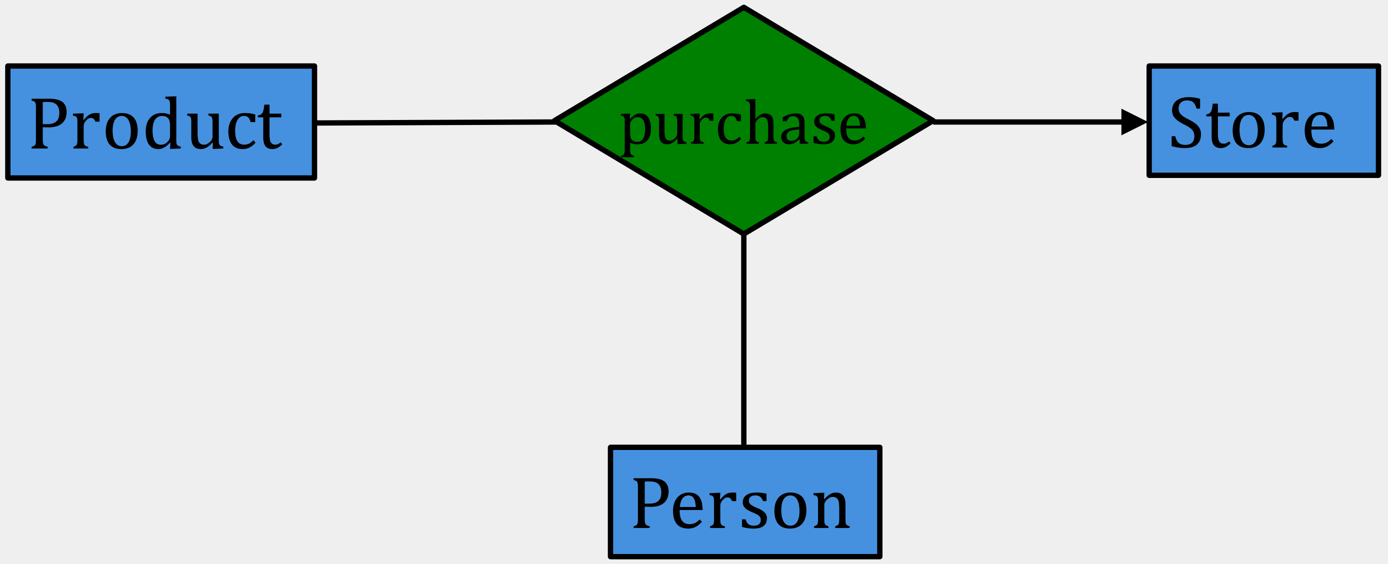
How do we model a **purchase** relation between **buyers**, **products** and **stores**?



We can still model this as a mathematical set (ternary relation)

ARROWS IN MULTI-WAY RELATIONSHIPS

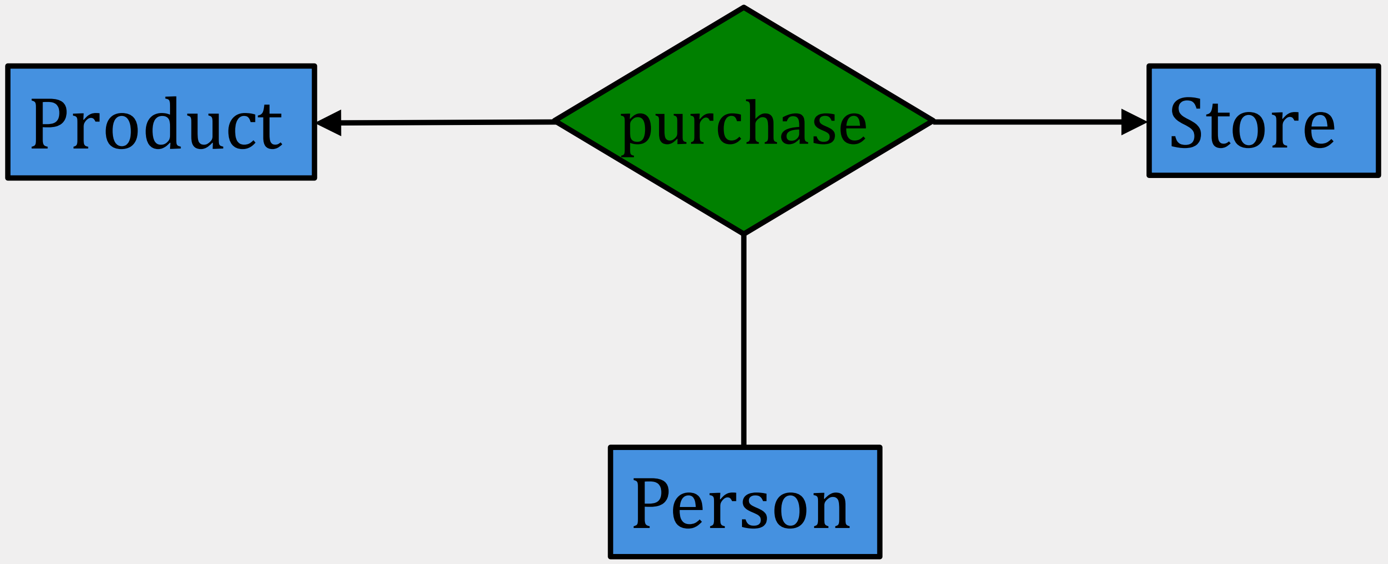
What does the arrow mean here?



A given **person** can **purchase** a given **product** from at most one **store**!

ARROWS IN MULTI-WAY RELATIONSHIPS

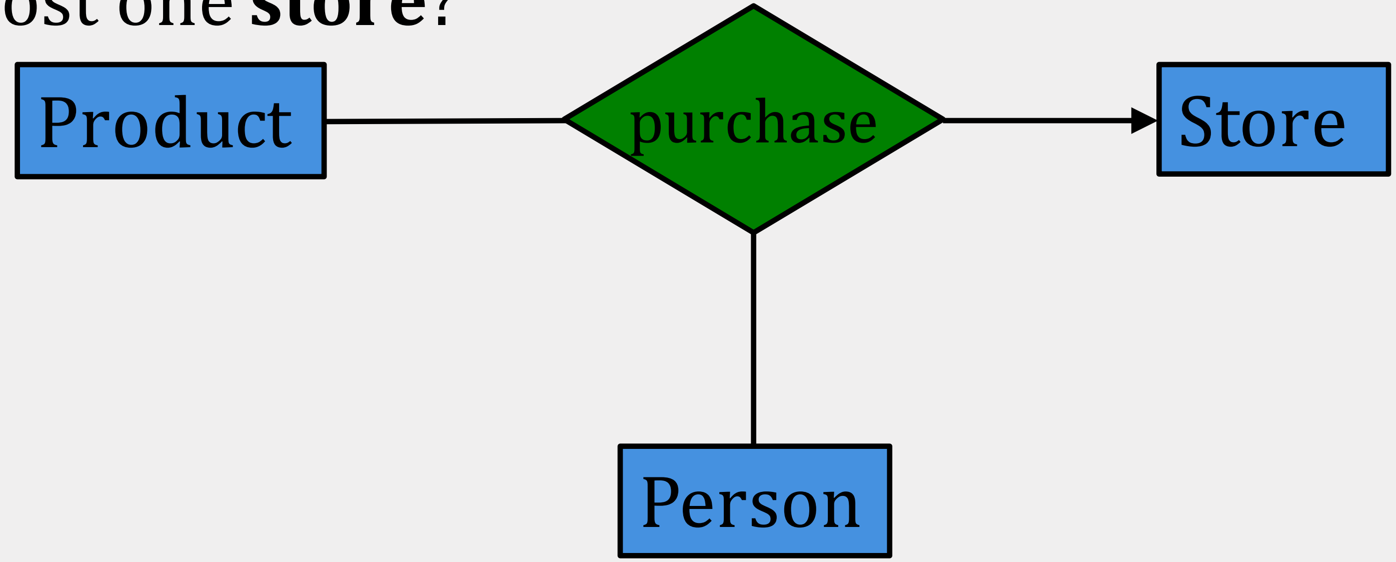
What about here?



A given **person** can **purchase** a given **product** from at most one **store** *AND* a given **store** sells to a given **person** at most one **product**

ARROWS IN MULTI-WAY RELATIONSHIPS

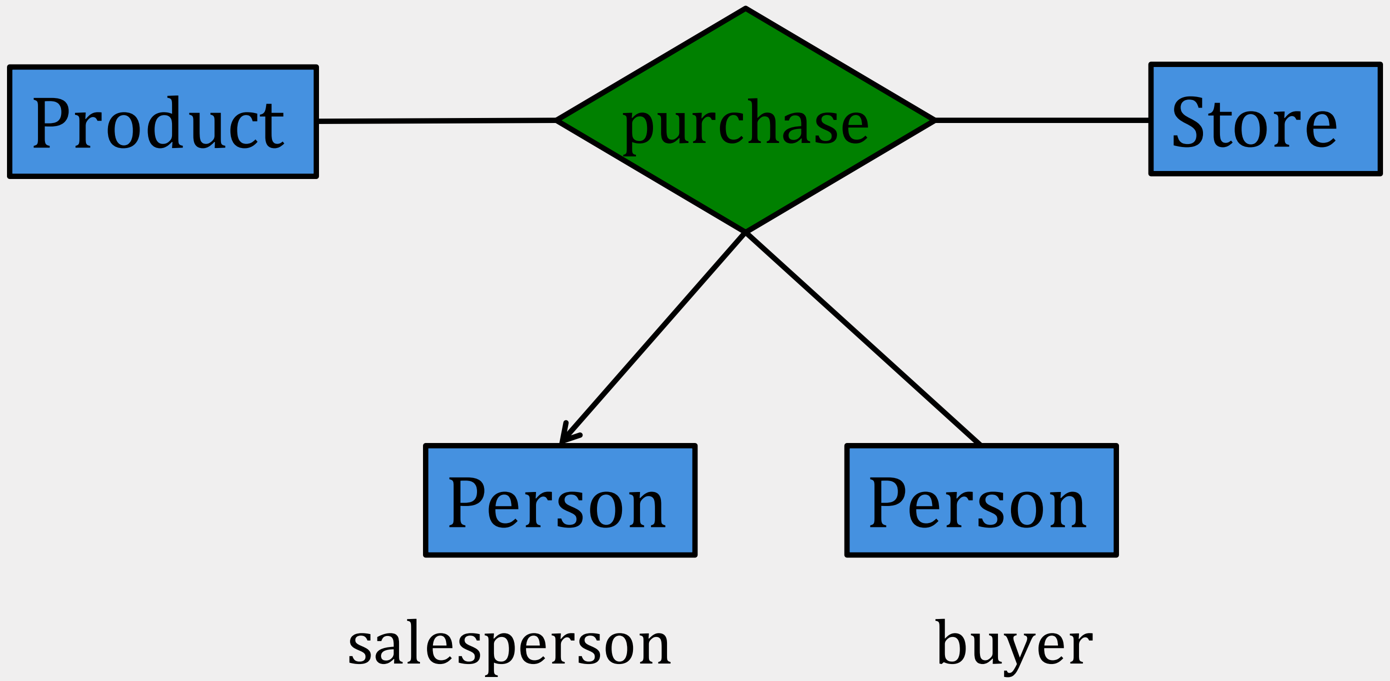
How can we say that a given **person** buys from at most one **store**?



Not possible, we can only approximate!

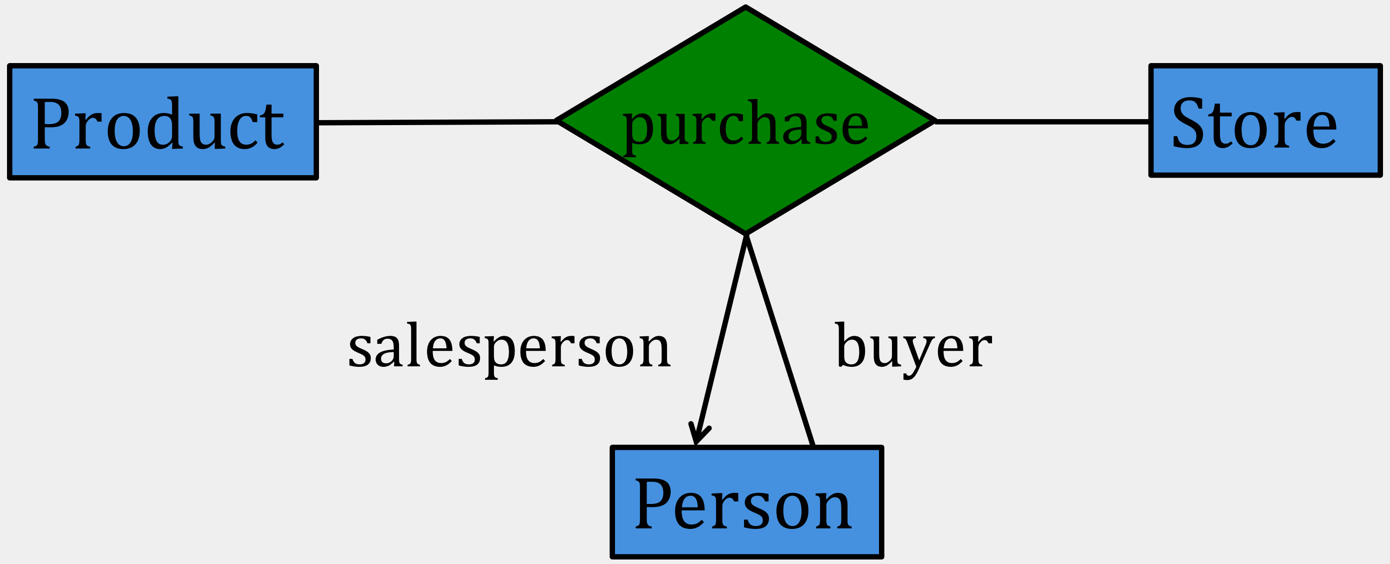
ROLES IN RELATIONSHIPS

What if we need an entity set twice in a relationship?

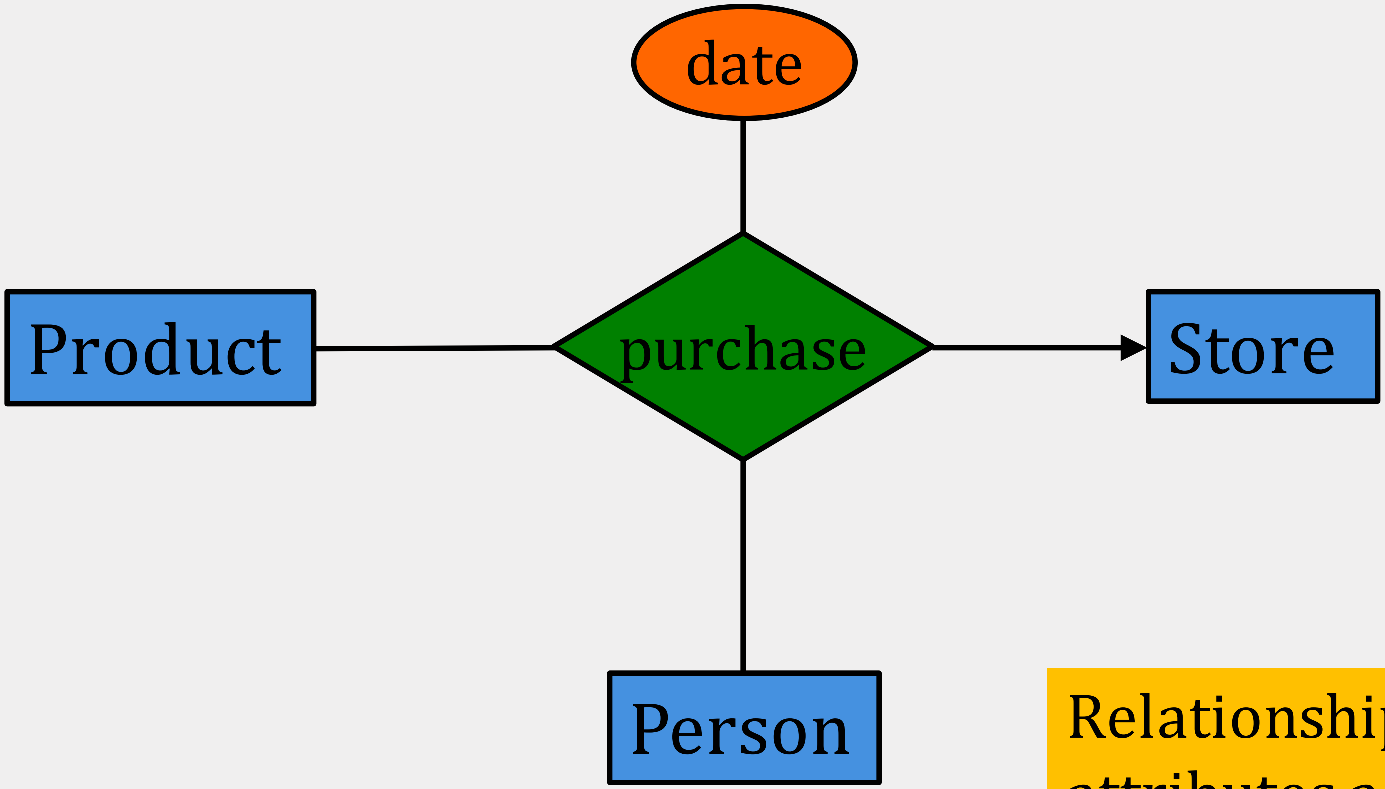


ROLES IN RELATIONSHIPS

- Label the edges to indicate the roles
- Collapse the two entity sets into one

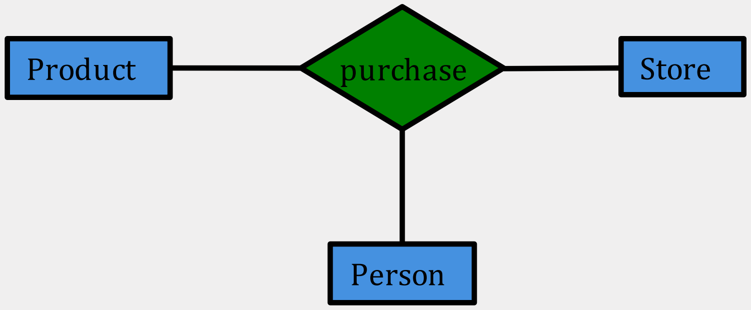


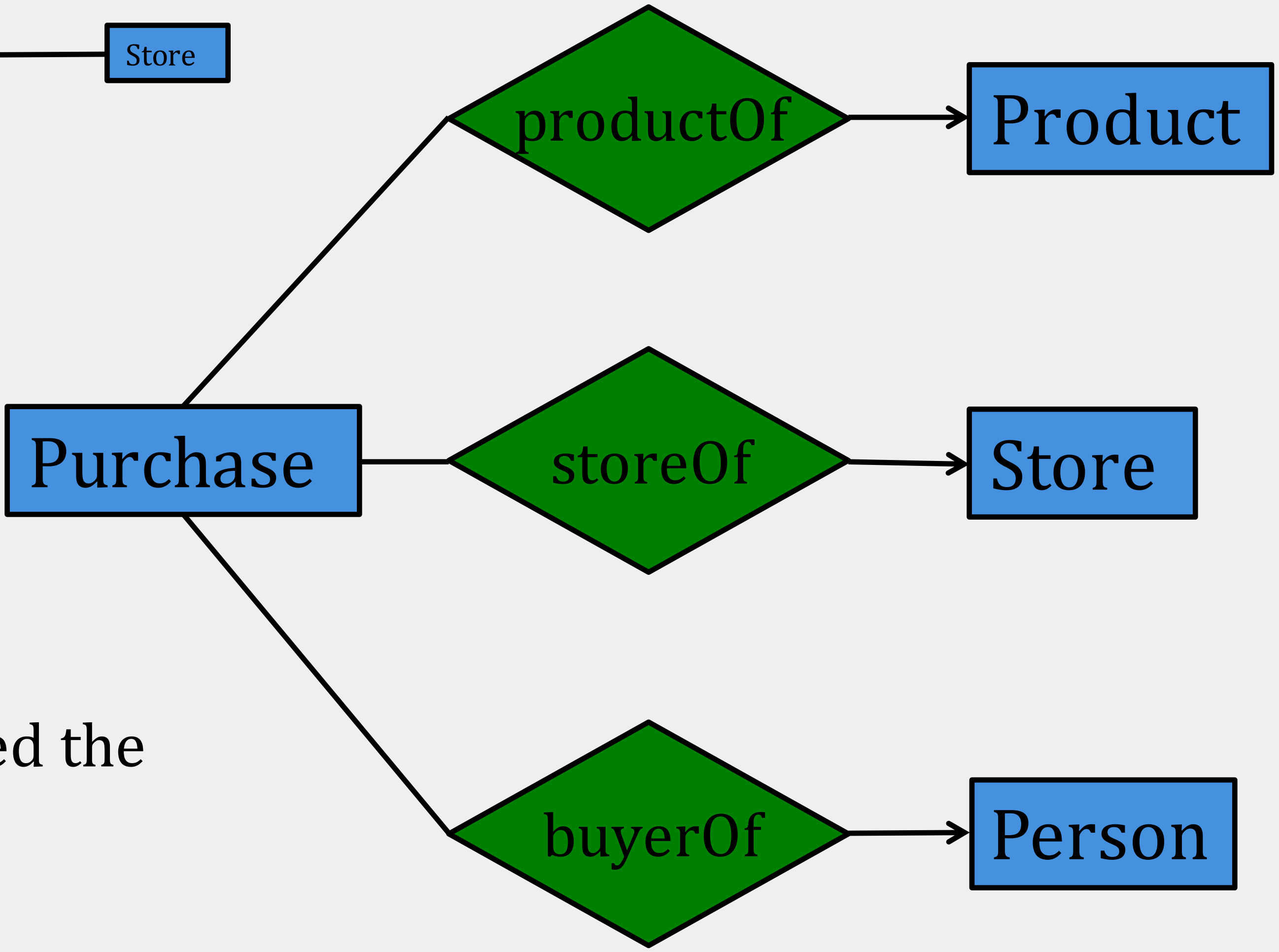
ATTRIBUTES IN RELATIONSHIPS



Relationships have attributes as well!

MULTI-WAY TO BINARY RELATIONSHIPS





Why do we need the arrows?

RELATIONSHIPS: RECAP

- Modeled as a mathematical set
- Binary and multi-way relationships
- Converting a multi-way one into many binary ones
- Constraints on the degree of the relationship
  - many-one, one-one, many-many
  - limitations of arrows
- Attributes of relationships
  - not necessary, but useful!

E/R: Additional Features

SUBCLASSES

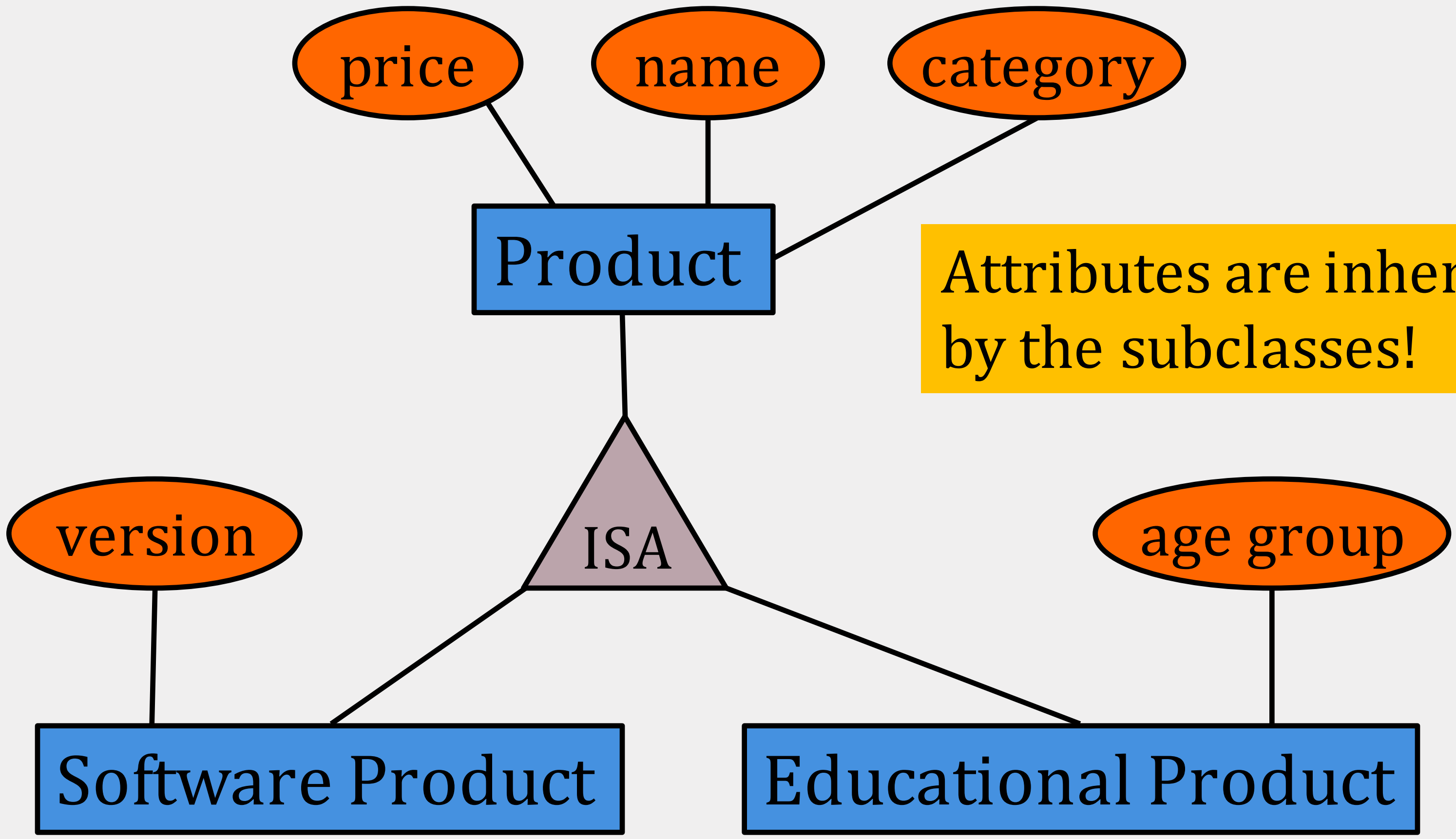
<u>subclass</u> = specialized case

= fewer entities

= more properties

- Example: Products
  - Software products
  - Educational products

SUBCLASSES



Attributes are inherited by the subclasses!

CONSTRAINTS

constraint := an assertion about the database that must be true at all times

- part of the database schema
- central in database design

When creating the ER diagram, you need to find as many constraints as possible!

TYPES OF CONSTRAINTS

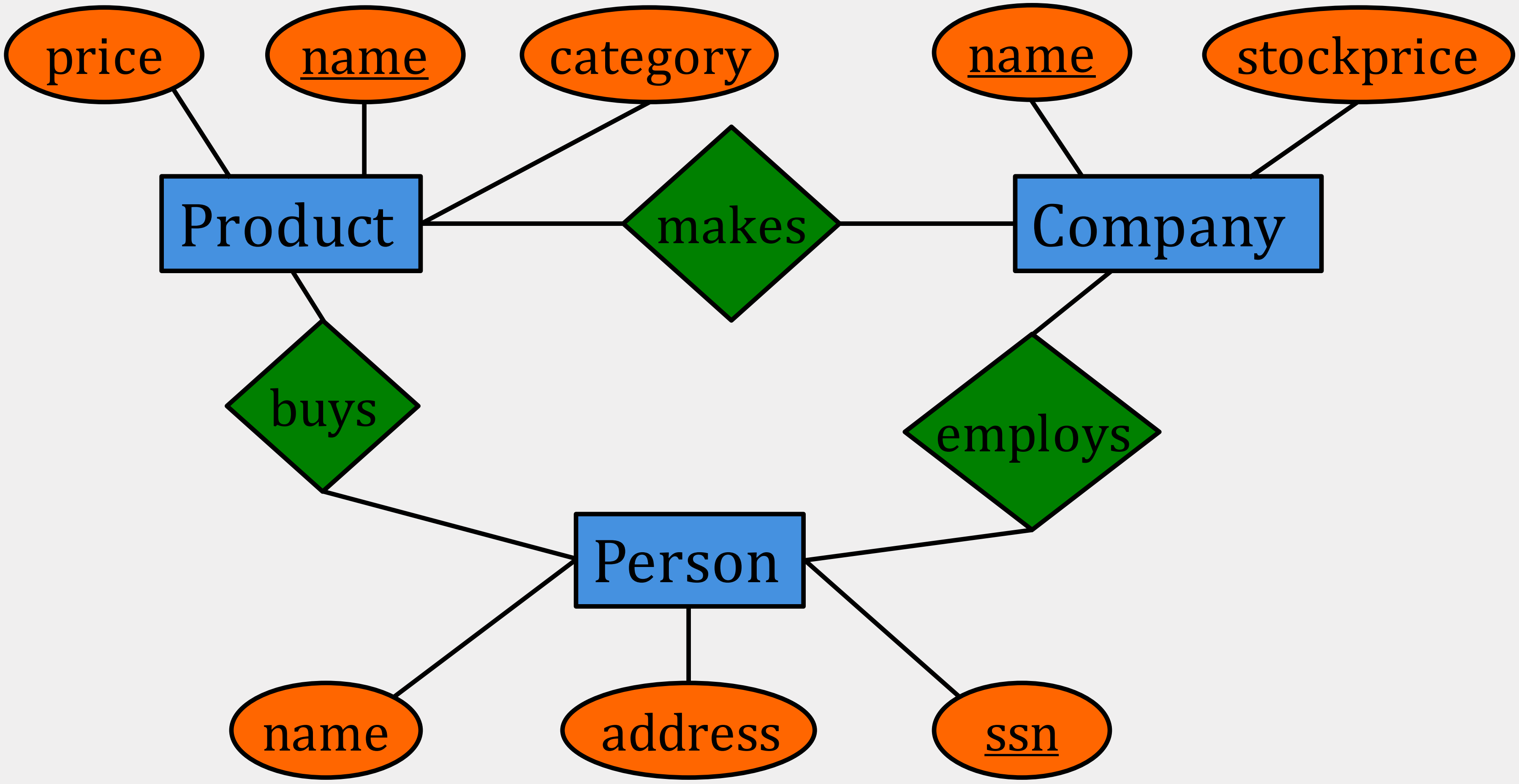
- **keys**: SSN uniquely identifies a person
- **single-value**: a person can have only one father
- **referential integrity**: if you work for a company, it must exist in the database
- **domain**: age is between 0 and 150

- **other**: e.g. at most 80 students enroll in a class

WHY DO WE NEED CONSTRAINTS?

- Give more semantics to the data
  - help us better understand it
- Prevent wrong data entry
- Allow us to refer to entities (e.g. using keys)
- Enable efficient storage and data lookup

KEY CONSTRAINTS



KEY CONSTRAINTS

Every entity set must have a key!

- A key can consist of more than one attribute
- There can be more than one key for an entity set
  - one key will be designated as primary key
- No formal way to specify multiple keys in an ER diagram

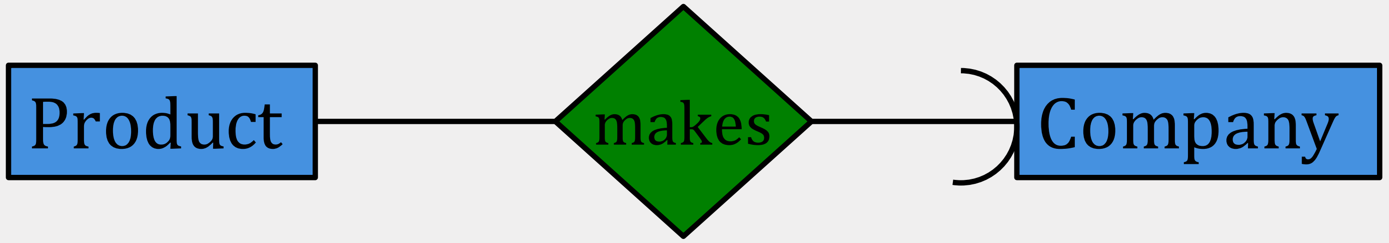
SINGLE-VALUE CONSTRAINTS

An entity may have **at most one value** for a given attribute or relationship

- an attribute of an entity set has a single value
- a many-one relation implies a single value constraint

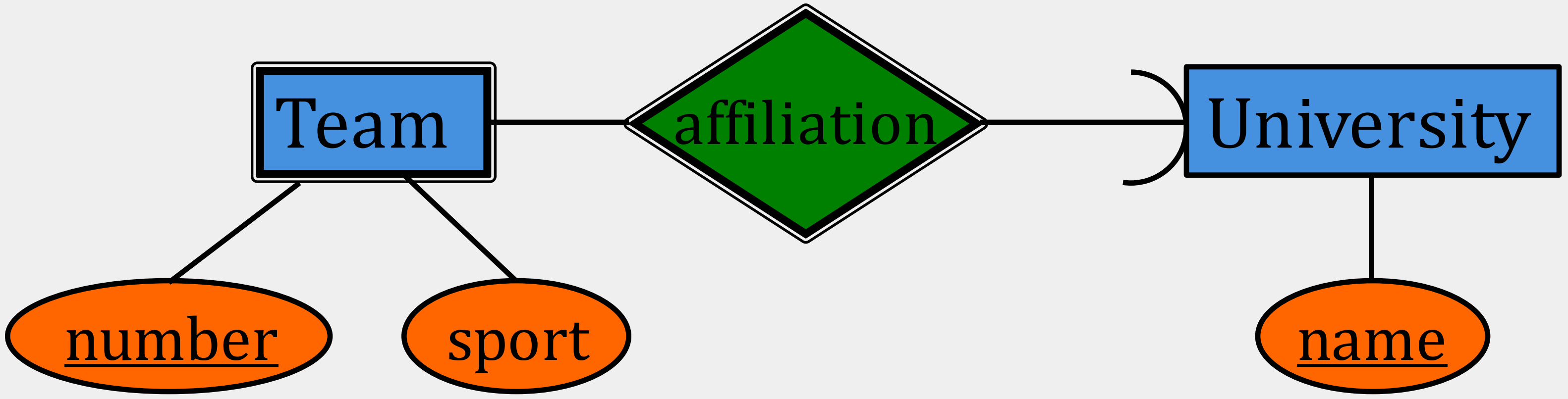
REFERENTIAL INTEGRITY CONSTRAINT

A relationship has **one value** and the value must exist



WEAK ENTITY SETS

Entity sets are **weak** when their key attributes come from other classes to which they are related

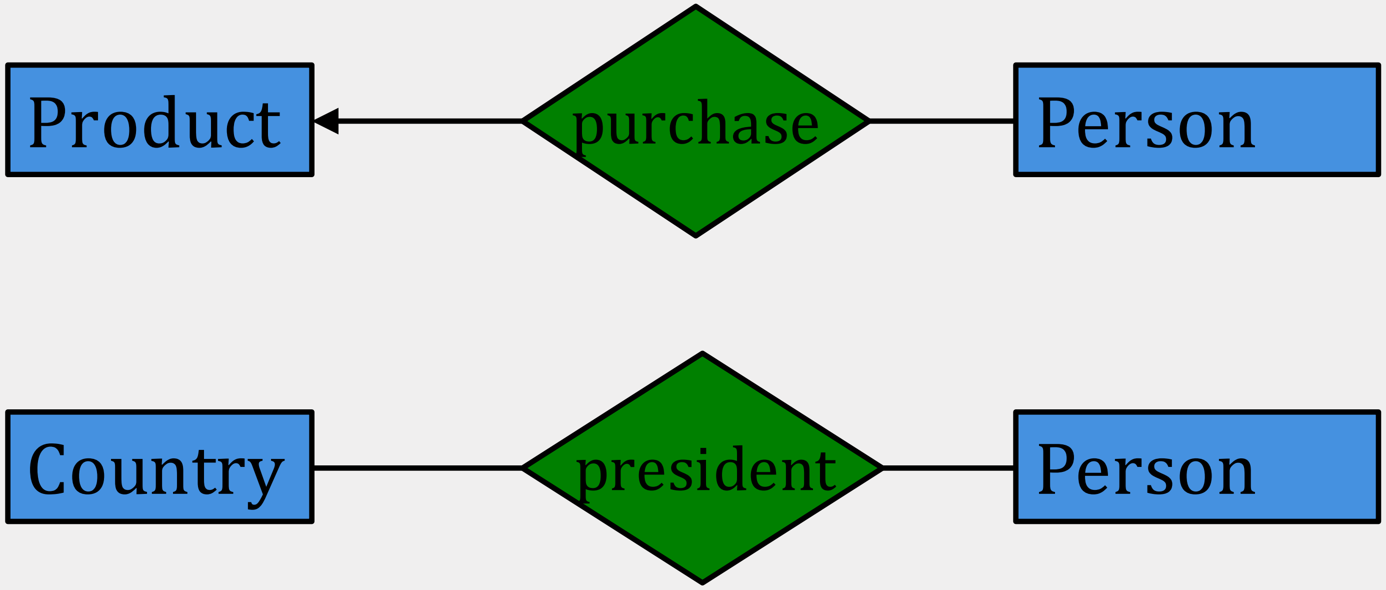


entities of an entity set need "help" to identify them uniquely!

E/R: Design Principles

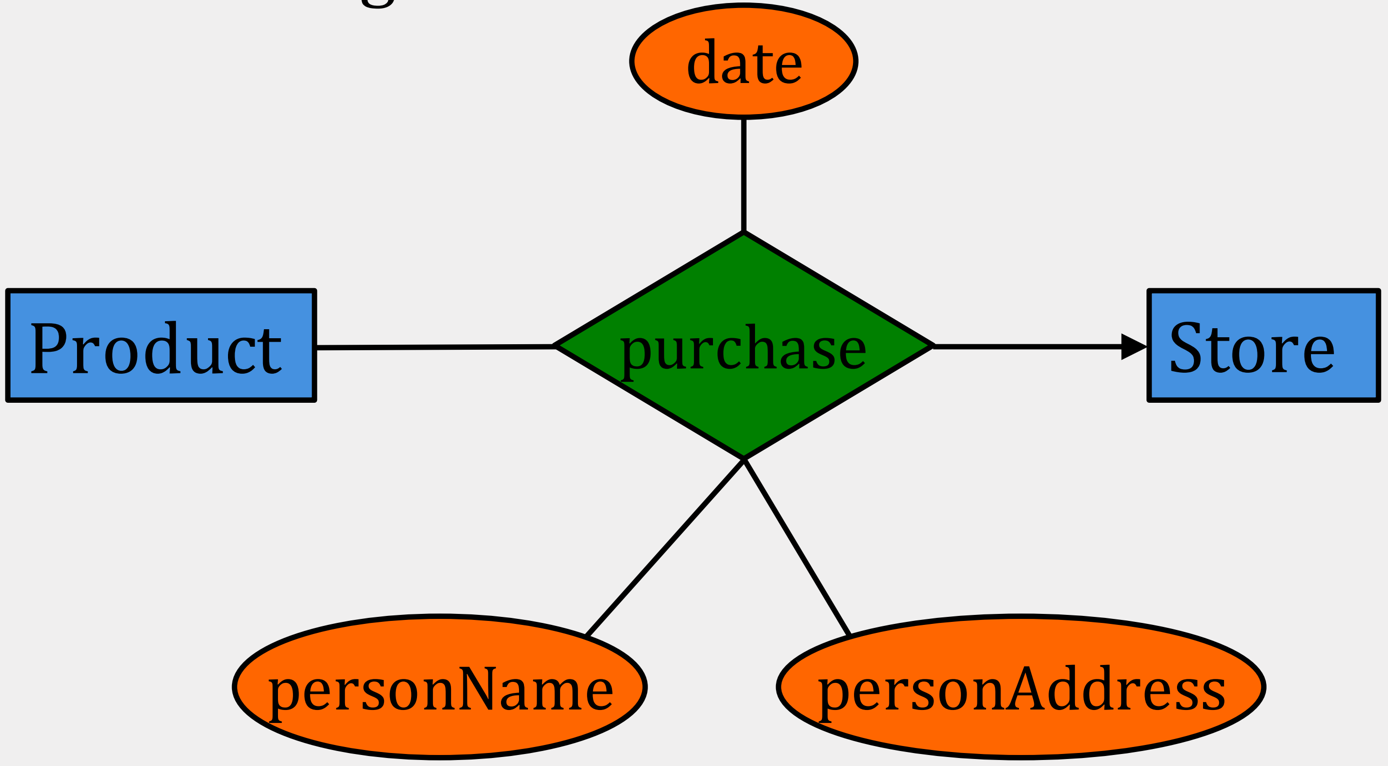
1. BE FAITHFUL TO THE APP!

What is wrong here?



2. AVOID REDUNDANCY!

What is wrong here?

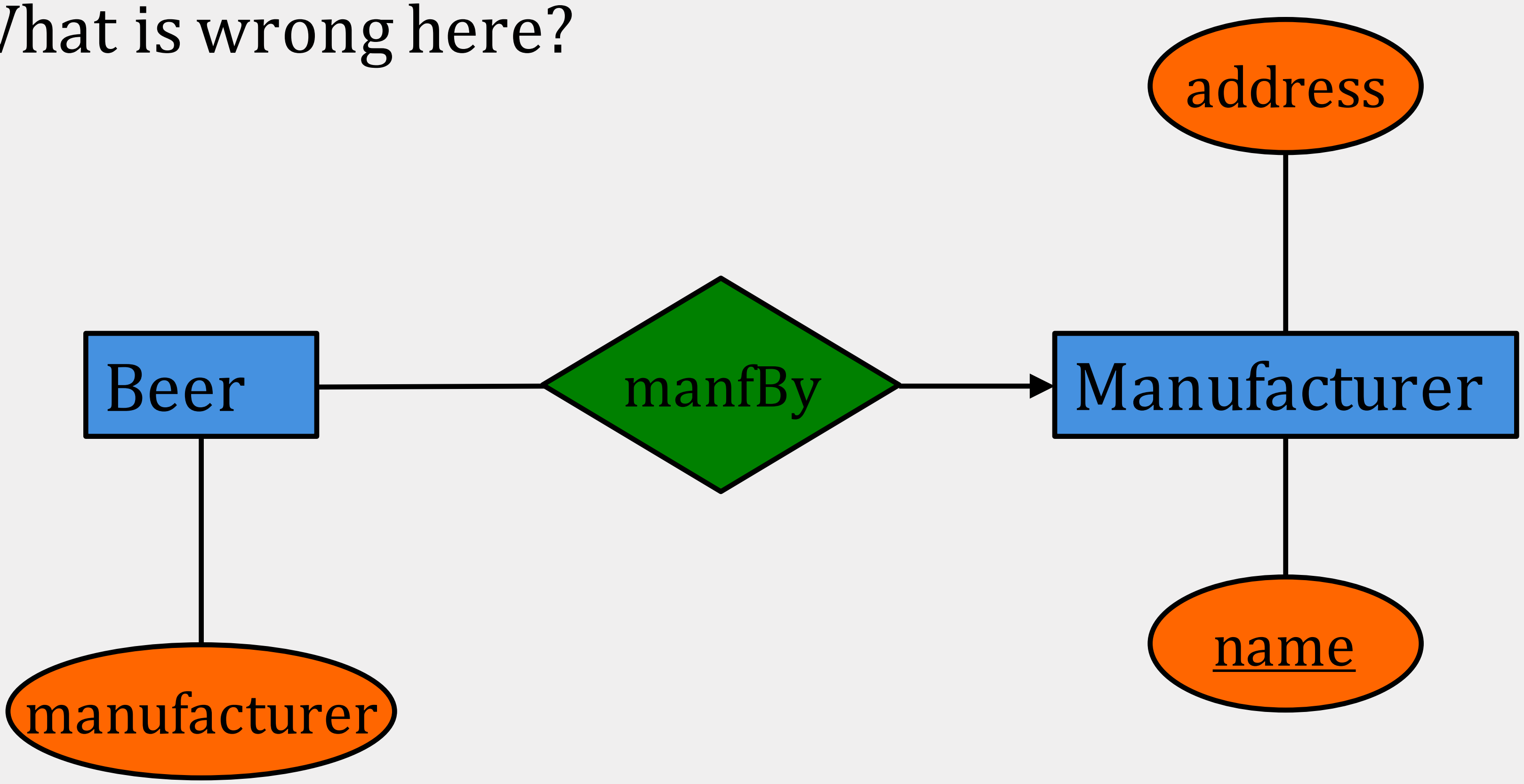


2. AVOID REDUNDANCY!

- Redundancy occurs when we say the same thing in two different ways
- Redundancy wastes space and encourages inconsistency
  - The two instances of the same fact may become inconsistent if we change one and forget to change the other

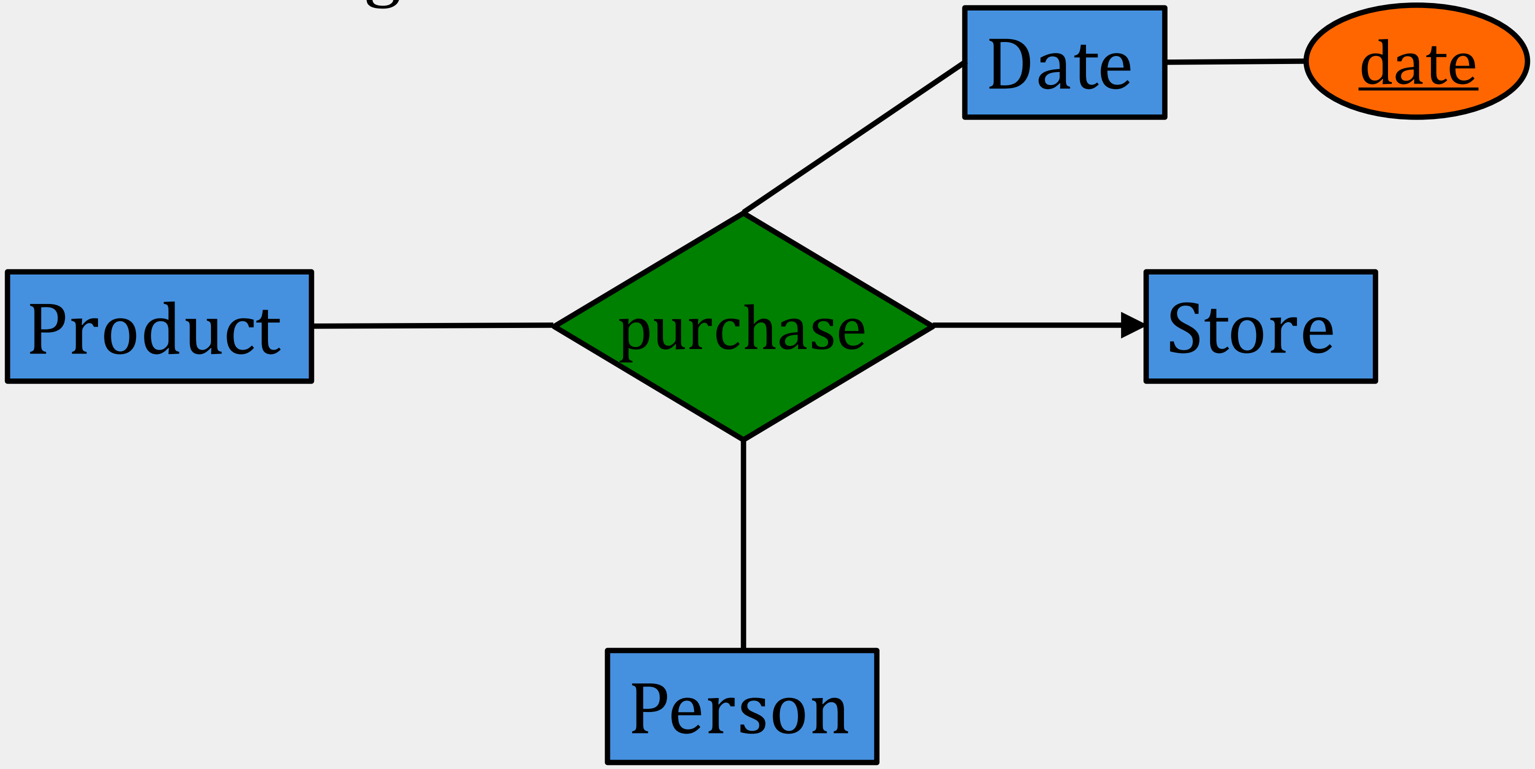
2. AVOID REDUNDANCY!

What is wrong here?



3. KEEP IT SIMPLE!

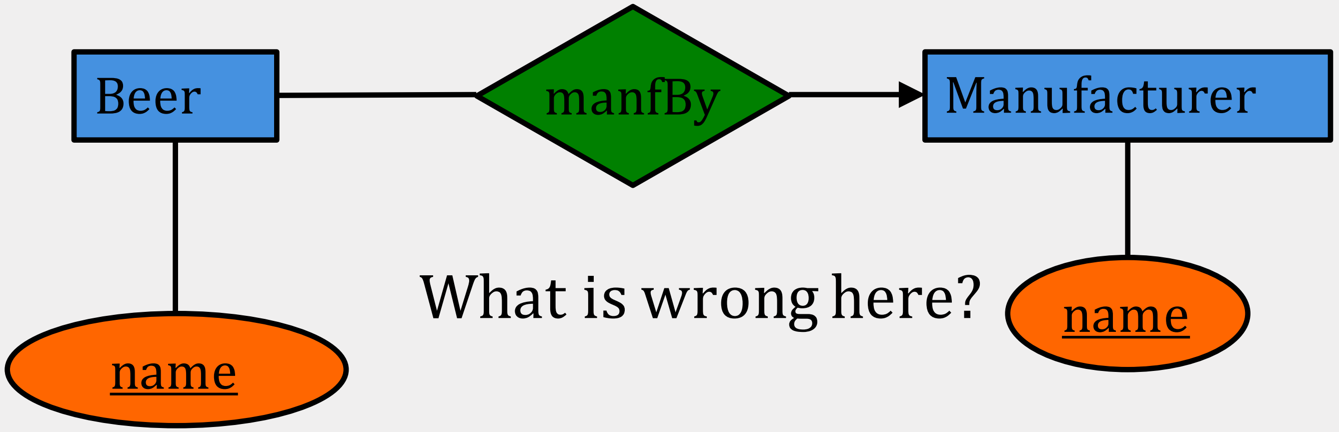
What is wrong here?



4. ATTRIBUTES OVER ENTITY SETS

An entity set should satisfy at least one of the following conditions

- it is more than the name of something; it has at least one non-key attribute
- it is the "many" in a many-one or many-many relationship



What is wrong here?

5. DON'T OVERUSE WEAK ENTITY SETS

- Beginner database designers often doubt that anything could be a key by itself
  - They make all entity sets weak, supported by all other entity sets to which they are linked
- In reality, we create unique IDs for entity sets
  - Examples: SSN, ISBN, …

E/R to Relational Model

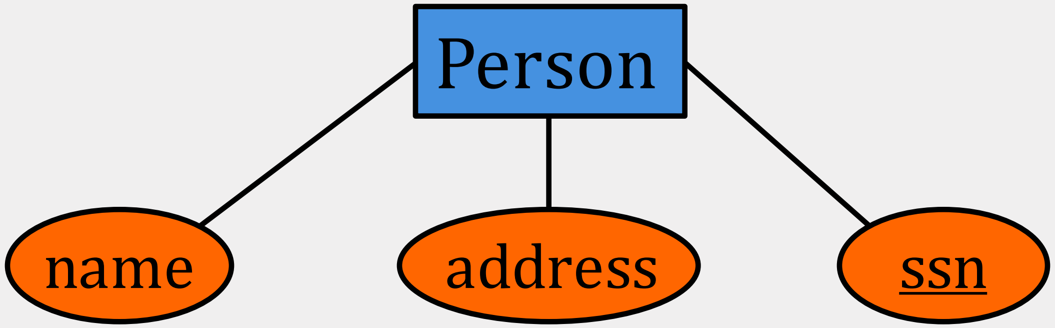
ER MODEL VS RELATIONAL MODEL

- **ER model**
  - many concepts: entities, relations, attributes, etc.
  - well-suited for capturing the app requirements
  - not well-suited for computer implementation
- **Relational model**
  - has just a single concept: relation
  - world is represented with a collection of tables
  - well-suited for efficient manipulations on computers

TRANSLATION

- Basic cases:
  - entity set **E** -- > relation with attributes of **E**
  - relationship **R** -- > relation with attributes being keys of related entity sets + attributes of **R**
- Special cases:
  - combining two relations
  - weak entity sets
  - is-a relationships

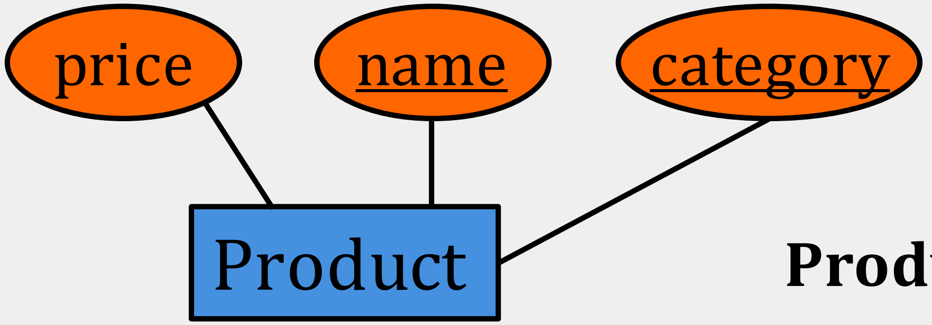
ENTITY SET TO RELATION



Person(<u>ssn,</u> name, address)

```
CREATE TABLE Person (ssn CHAR(11) PRIMARY KEY,
                     name CHAR(40),
                     address CHAR(50))
```

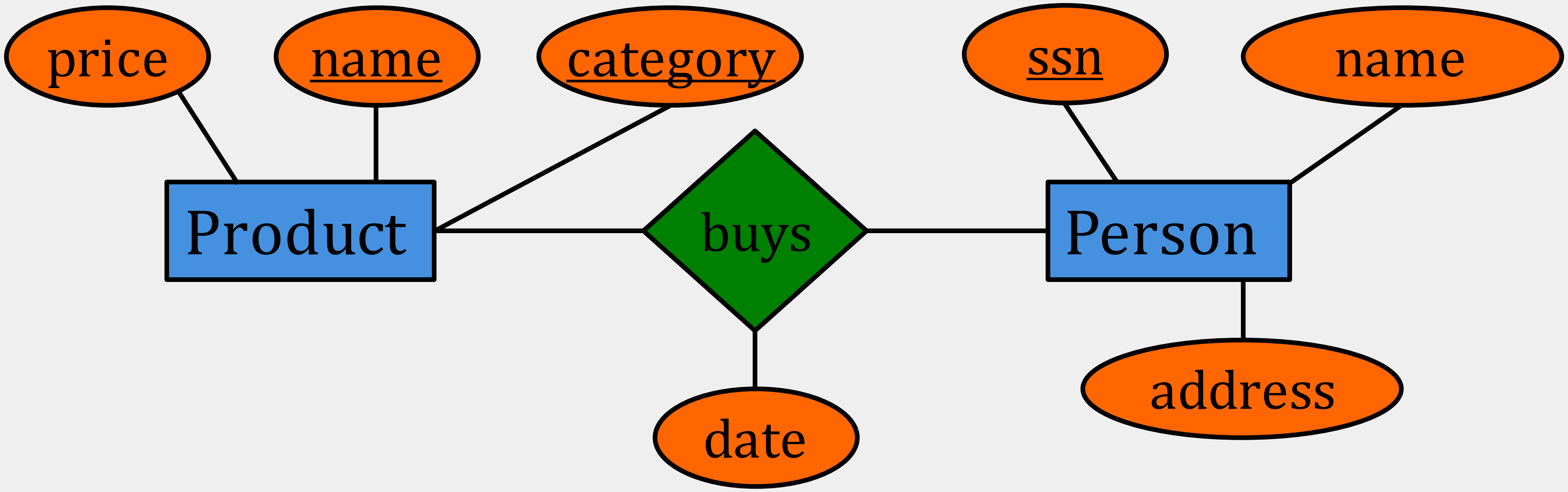
ENTITY SET TO RELATION



Product(<u>name, category</u>, price)

```
CREATE TABLE Product (name CHAR(40),
                      category CHAR(20),
                      price REAL,
                     PRIMARY KEY(name, category))
```

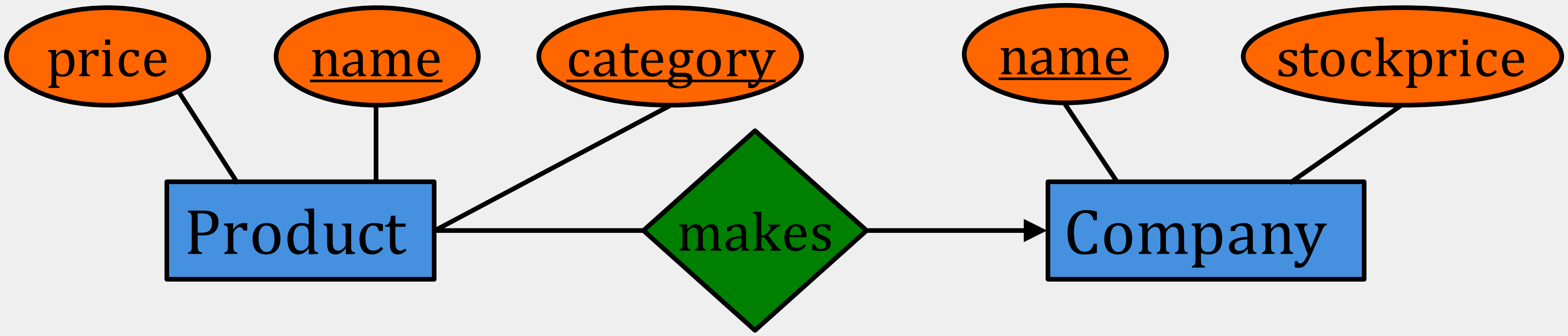
RELATIONSHIP TO RELATION



Product(name, category, price)
Person(ssn, name, address)

Buys(prodname, prodcategory, ssn, date)

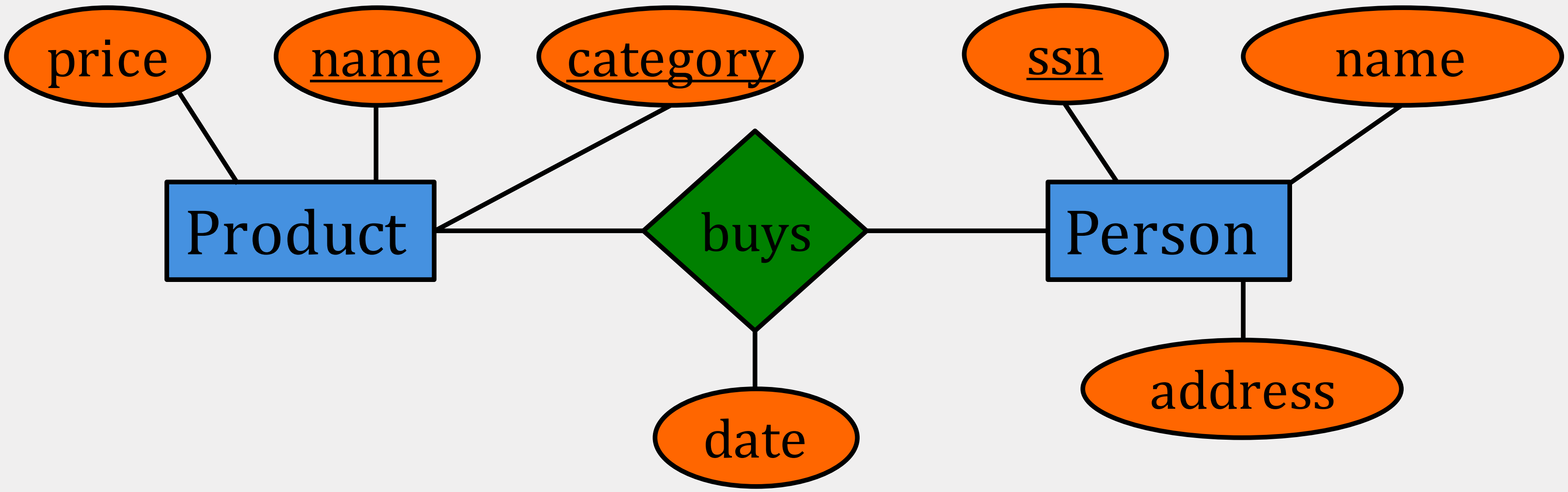
MANY-ONE RELATIONSHIPS



No need for a **Makes** relation; instead modify **Product**:

Product(<u>name, category</u>, price, company_name)
Company(<u>name</u>, stockprice)

MANY-MANY RELATIONS



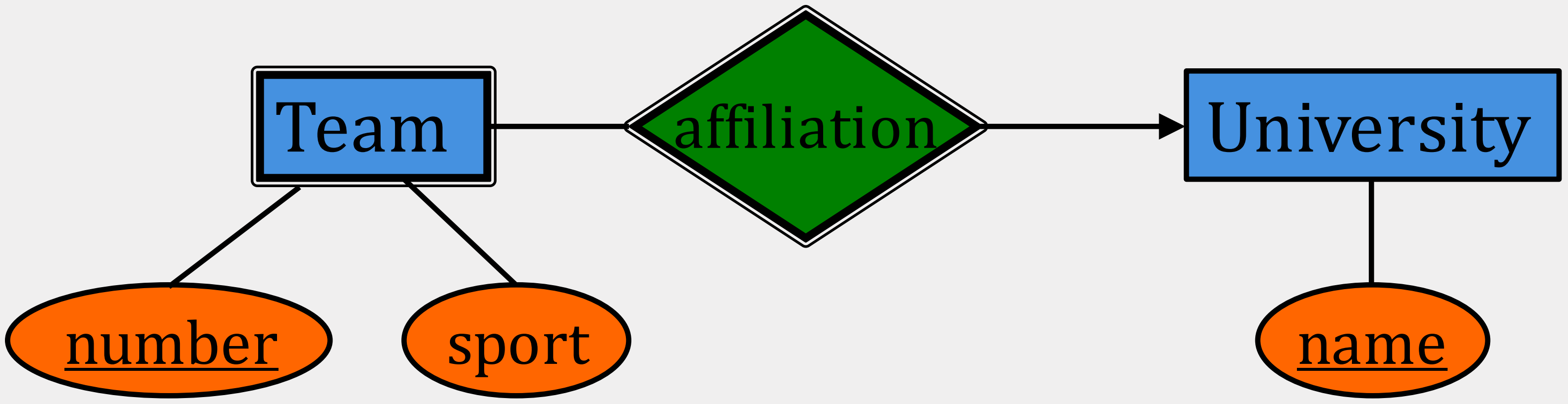
Product(name, category, price, ssn)

What is wrong here?

RELATIONSHIP TO RELATION: SQL

```
CREATE TABLE Buys
   (prodname CHAR(40),
    prodcategory CHAR(20),
    ssn CHAR(11),
    date DATE,
    PRIMARY KEY(prodname,prodcategory,ssn)
    FOREIGN KEY (ssn)
        REFERENCES Person,
    FOREIGN KEY (prodname, prodcategory)
        REFERENCES Product(name, category))
```

WEAK ENTITY SETS



Team(<u>number, affiliated-university</u>, sport)

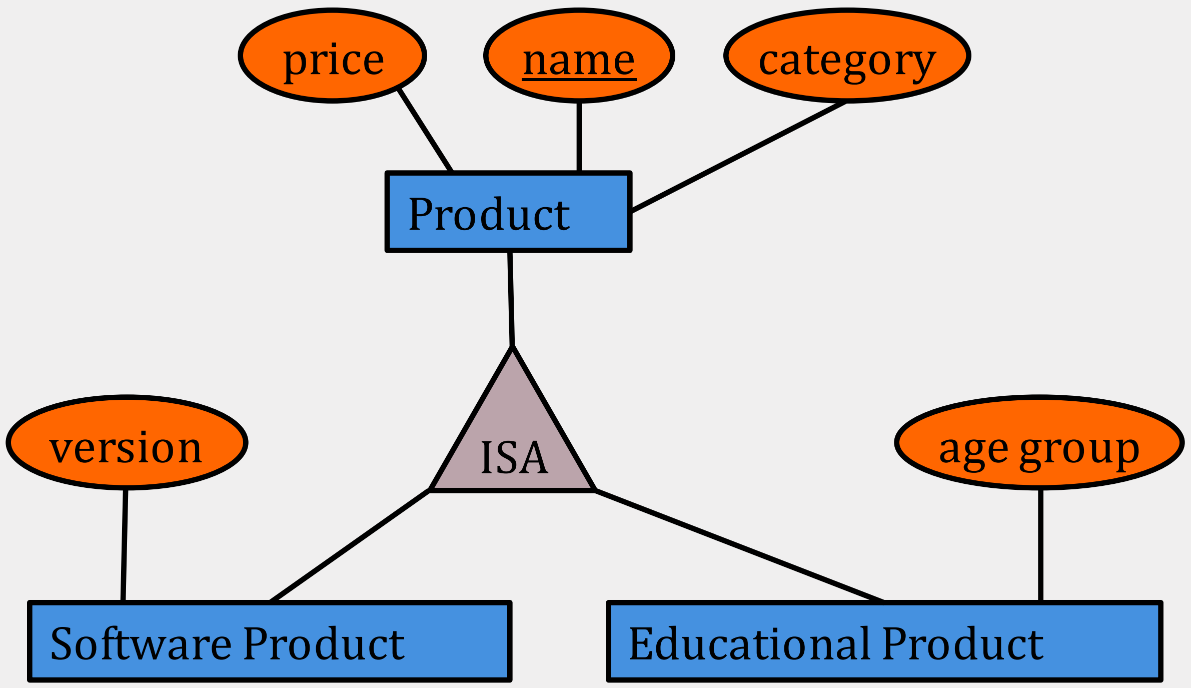
- **Affiliation** does not need a separate relation!
- Attribute '**name**' needed as part of the key

WEAK ENTITY SETS

- The relation for a weak entity set must include:
  - attributes for its complete key (including those in other entity sets)
  - its own, non-key attributes

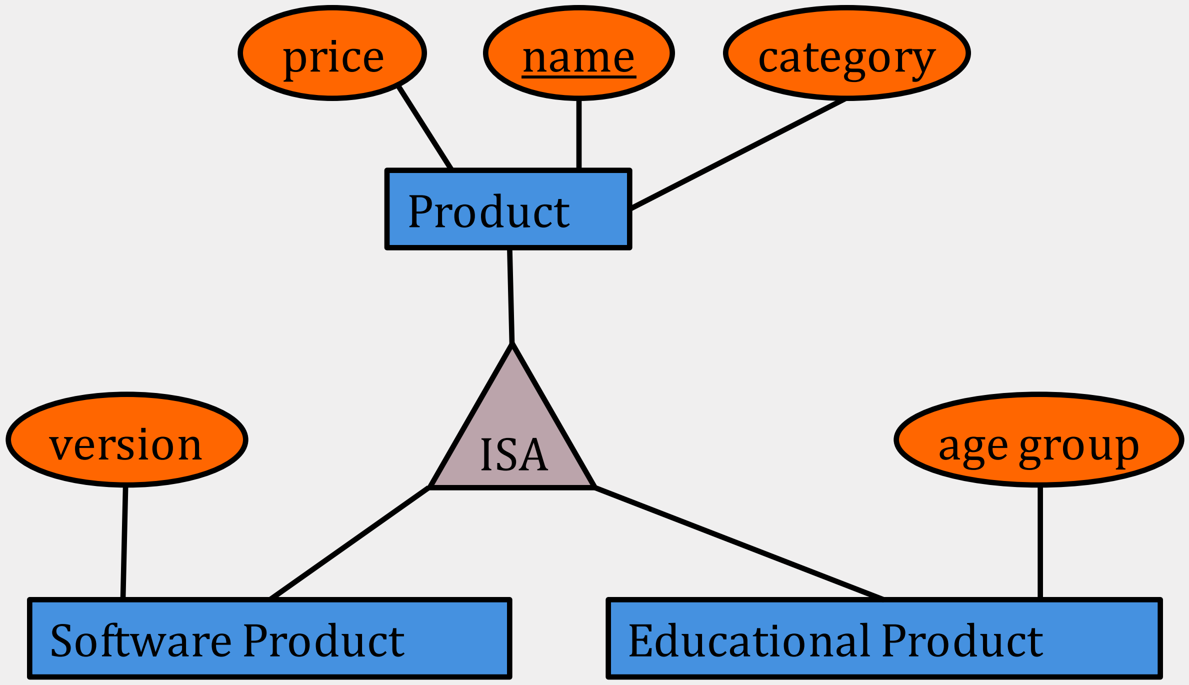
- A supporting (double-diamond) relationship is redundant and produces no relation

SUBCLASSES: OPTION 1



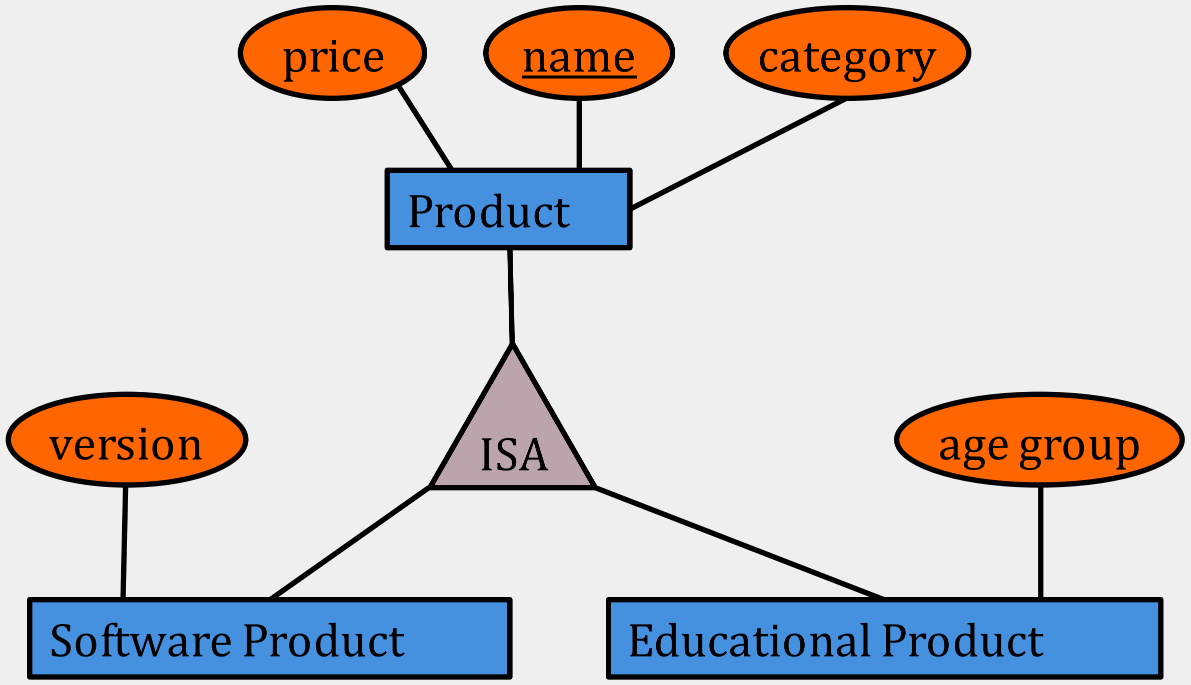
- **Product**(<u>name,</u> category, price)
- **SoftwareProduct**(<u>name</u>, category, price, version)
- **EducationalProduct**(<u>name</u>, category, price, age-group)

SUBCLASSES: OPTION 2



- **Product**(<u>name</u>, category, price)
- **SoftwareProduct**(<u>name</u>, version)
- **EducationalProduct**(<u>name</u>, age-group)

SUBCLASSES: OPTION 3



- **Product**(<u>name</u>, category, price, version, age-group)
- Use NULL to denote that the attribute makes no sense for a specific tuple

SUBCLASSES RECAP

Three approaches:

1. create a relation for each class with all its attributes
2. create one relation for each subclass with only the key attribute(s) and attributes attached to it
3. create one relation; entities have null in attributes that do not belong to them