Conceptual Design with ER Model

Lecture #2

What This Course Will Cover

- How to create and use a database
- How a database management system works

How to Create and Use a DB

- A company has
  - business people, support people (eg developers), customers
- Business people need DBs when the data is large
- Customers also need DBs when the data is large
- Accessing DBs
  - Customers use Web pages
  - Business people use Web pages or prompts
  - Developers mostly use prompts

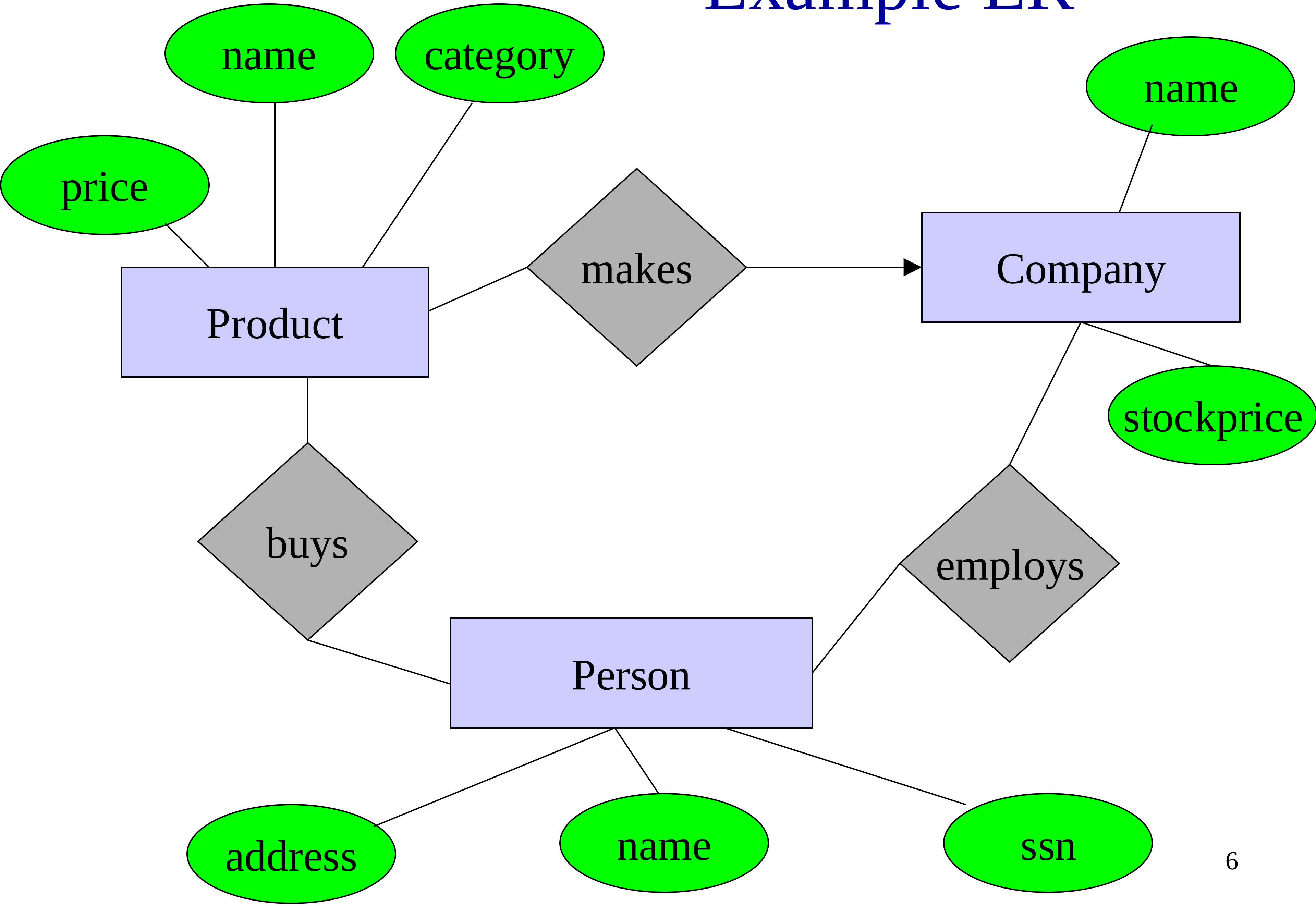
How to Create and Use a DB

- Business people tell developers what to do
- The conversation between these two people is tricky
- Developers need to know
  - The scope of a DB
  - All important constraints
    - That exist now, or may occur in the future
  - Other stuff
    - Such as how much data now, and may have in the future
- How to talk with business people about all these?

Steps in Building a DB Application

- Step 0: pick an application domain
- Step 1: conceptual design
  – discuss what to model in the application domain
  – need a modeling language to express what you want
  – ER model is the most popular such language
  – output: an ER diagram of the app. domain

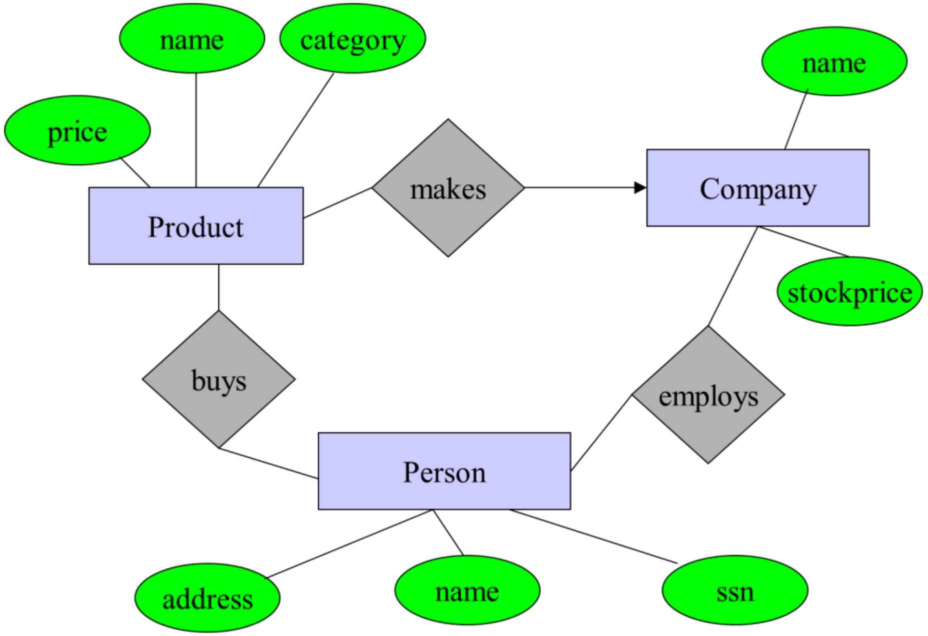
Example ER



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

Steps in Building a DB Application

- Step 2: pick a type of DBMS
  - relational DBMS is most popular and is our focus
- Step 3: translate ER design to a relational schema
  - use a set of rules to translate from ER to rel. schema
  - use a set of schema refinement rules to transform the above rel. schema into a good rel. schema
- At this point
  - you have a good relational schema on paper



- Product(name, category, price)
- Person(name, ssn, address)
- Company(name, stockprice)
- Buys(ssn, name)
- …

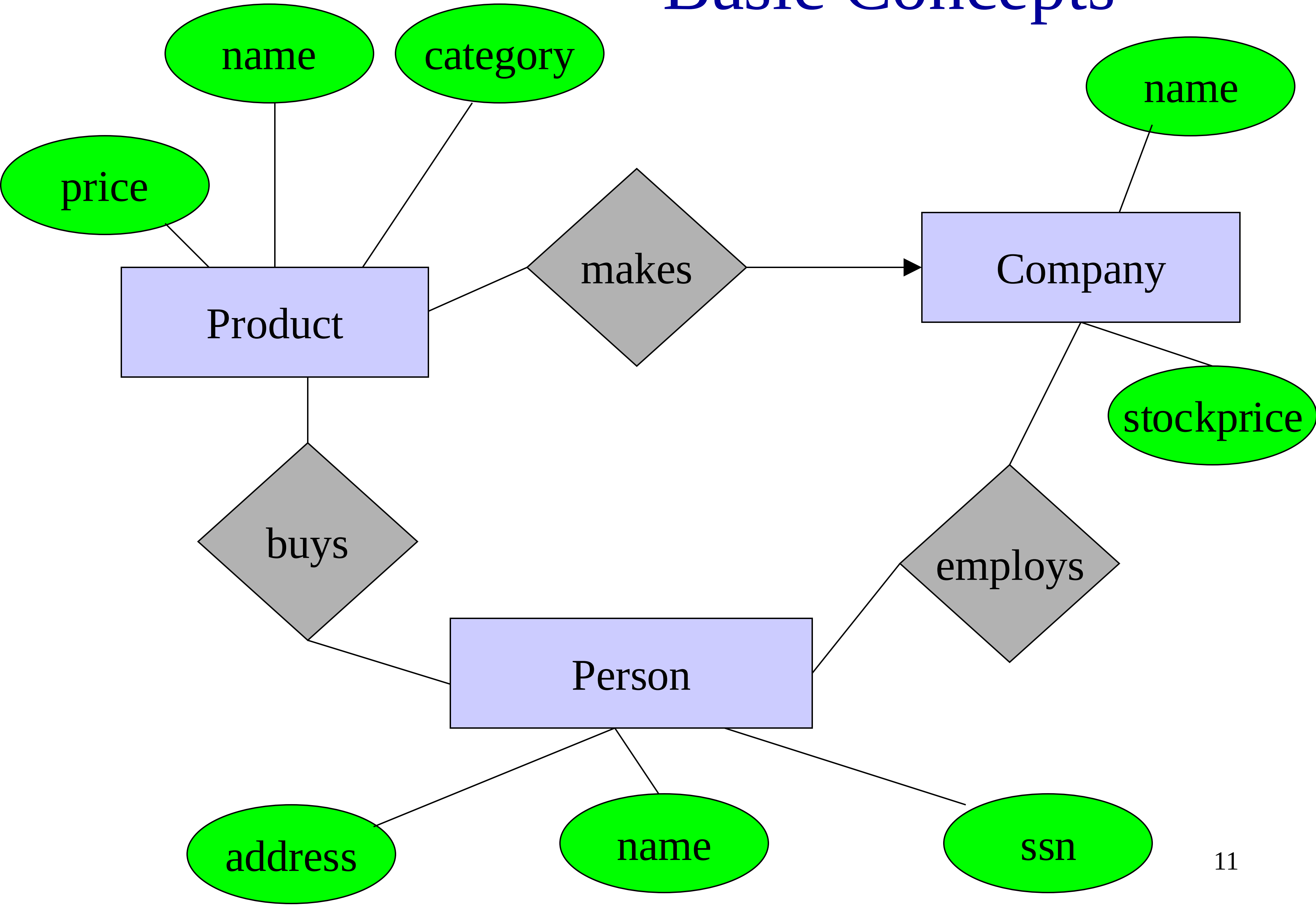
Steps in Building a DB Application

- Subsequent steps include
  - implement your relational DBMS using a "database programming language" called SQL
  - ordinary users cannot interact with the database directly
  - and the database also cannot do everything you want
  - hence write your application program in C++, Java, Python, etc to handle the interaction and take care of things that the database cannot do
- So, the first thing we should start with is to learn ER model ...

ER Model

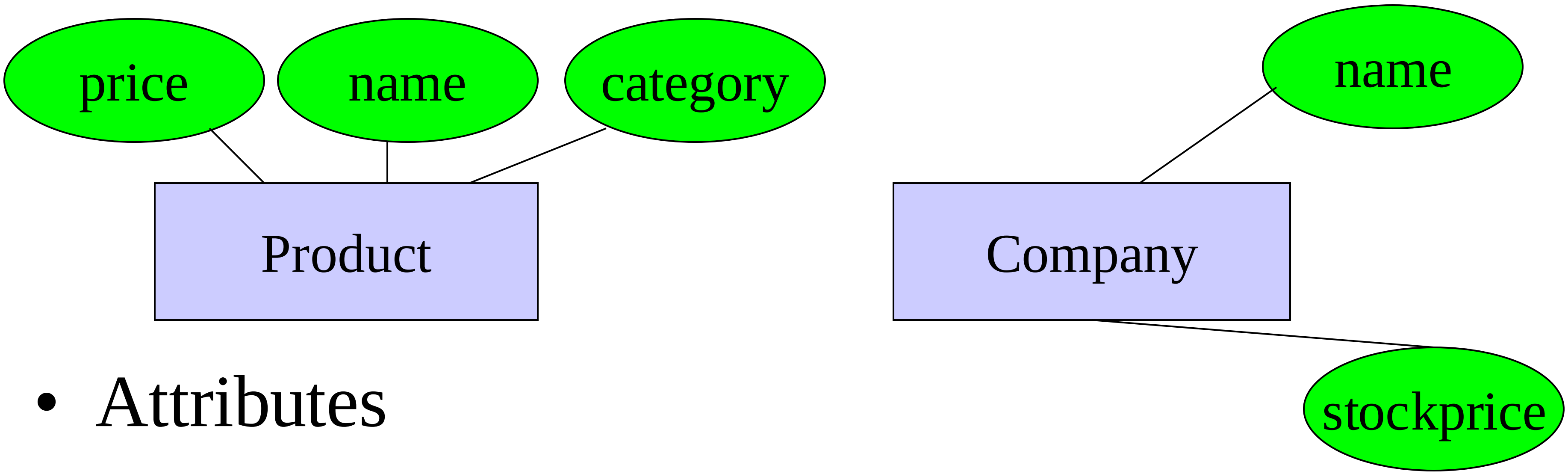
- Gives us a language to specify
  - what information the db must hold
  - what are the relationships among components of that information
- Proposed by Peter Chen in 1976
- What we will cover
  - basic stuff
  - constraints
  - weak entity sets
  - design principles

Basic Concepts



Entities and Attributes

- Entities
  - real-world objects distinguishable from other objects
  - described using a set of attributes



- Attributes
  - each has an atomic domain: string, integers, reals, etc.
- Entity set: a collection of similar entities

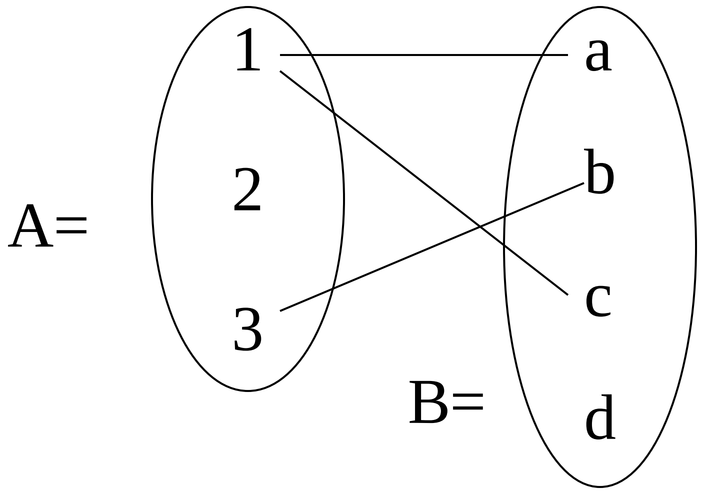
Discussion

- What does it mean to have “atomic attribute”?
- How to translate an entity set into a table?
- Why having atomic attributes?

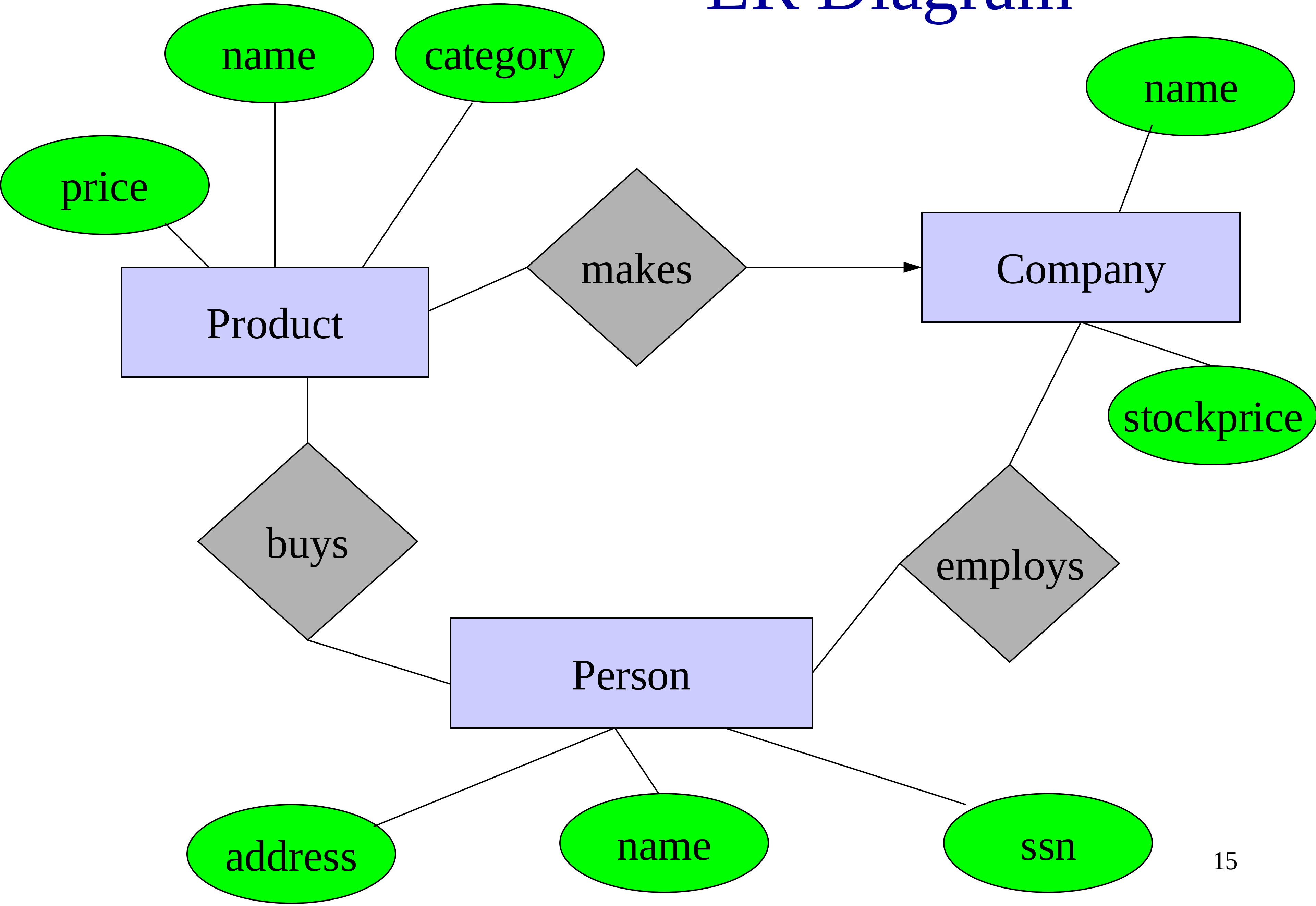
Relations

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

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



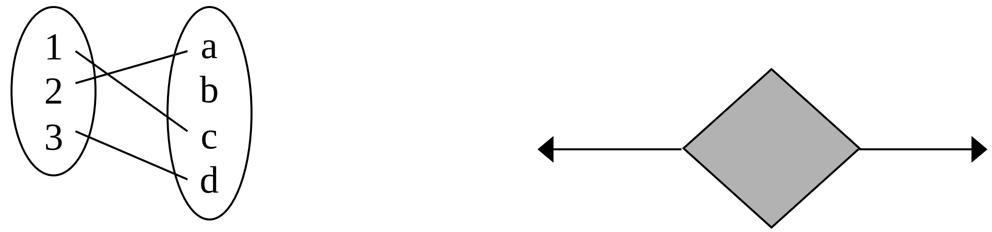
ER Diagram


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

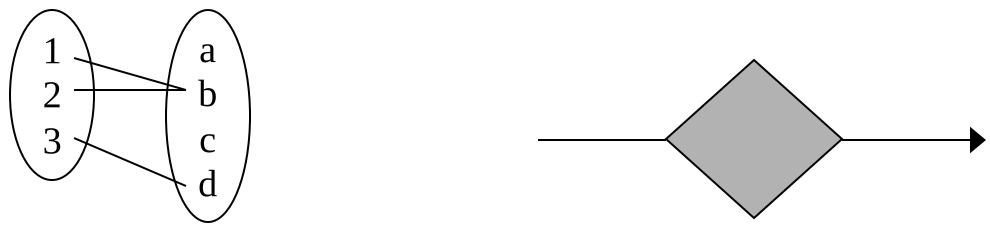
More about relationships ...

Types of Binary Relations

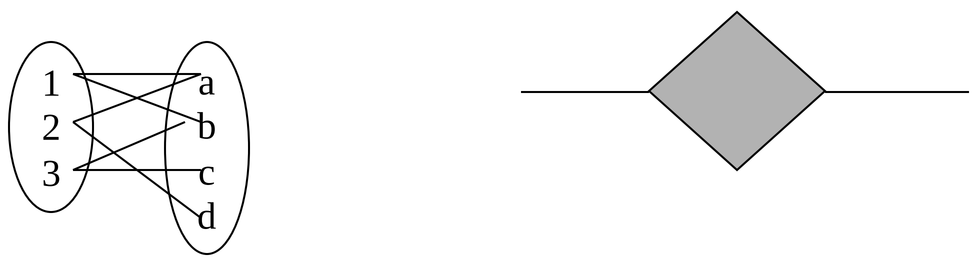
- one-one:

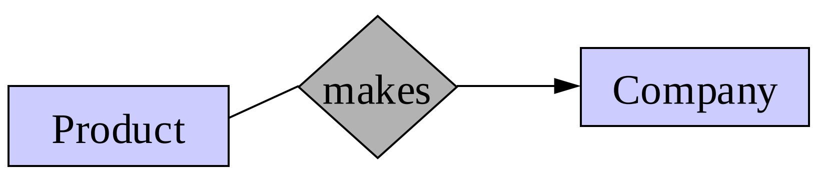


- many-one



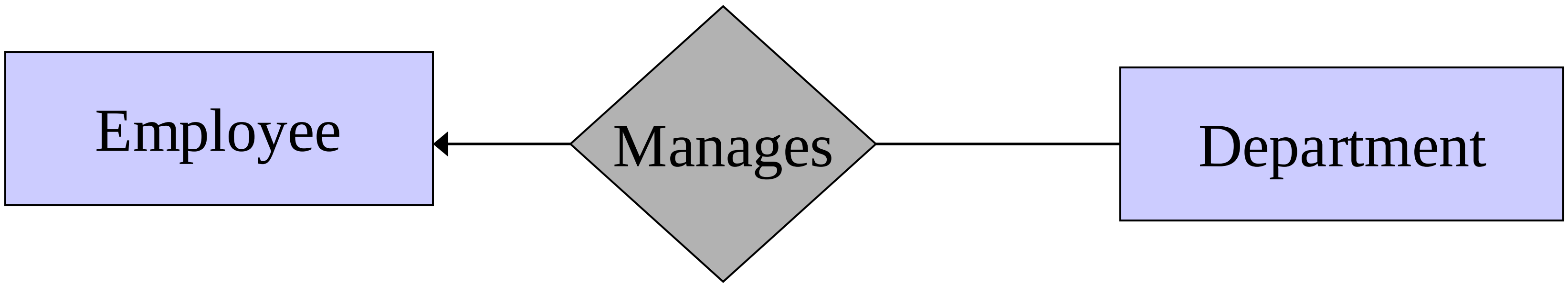
- many-many



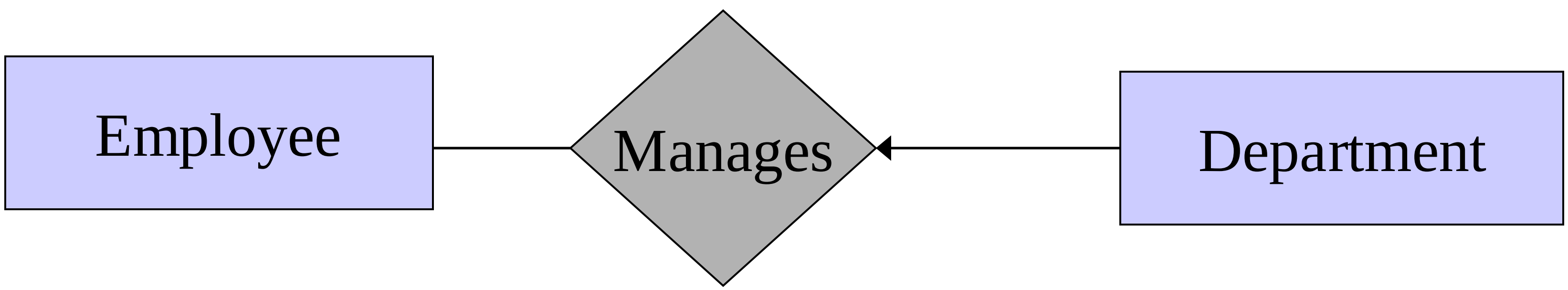


Important: Difference with the Book

- We will use



- Cow book use (see Page 33, 3rd edition)



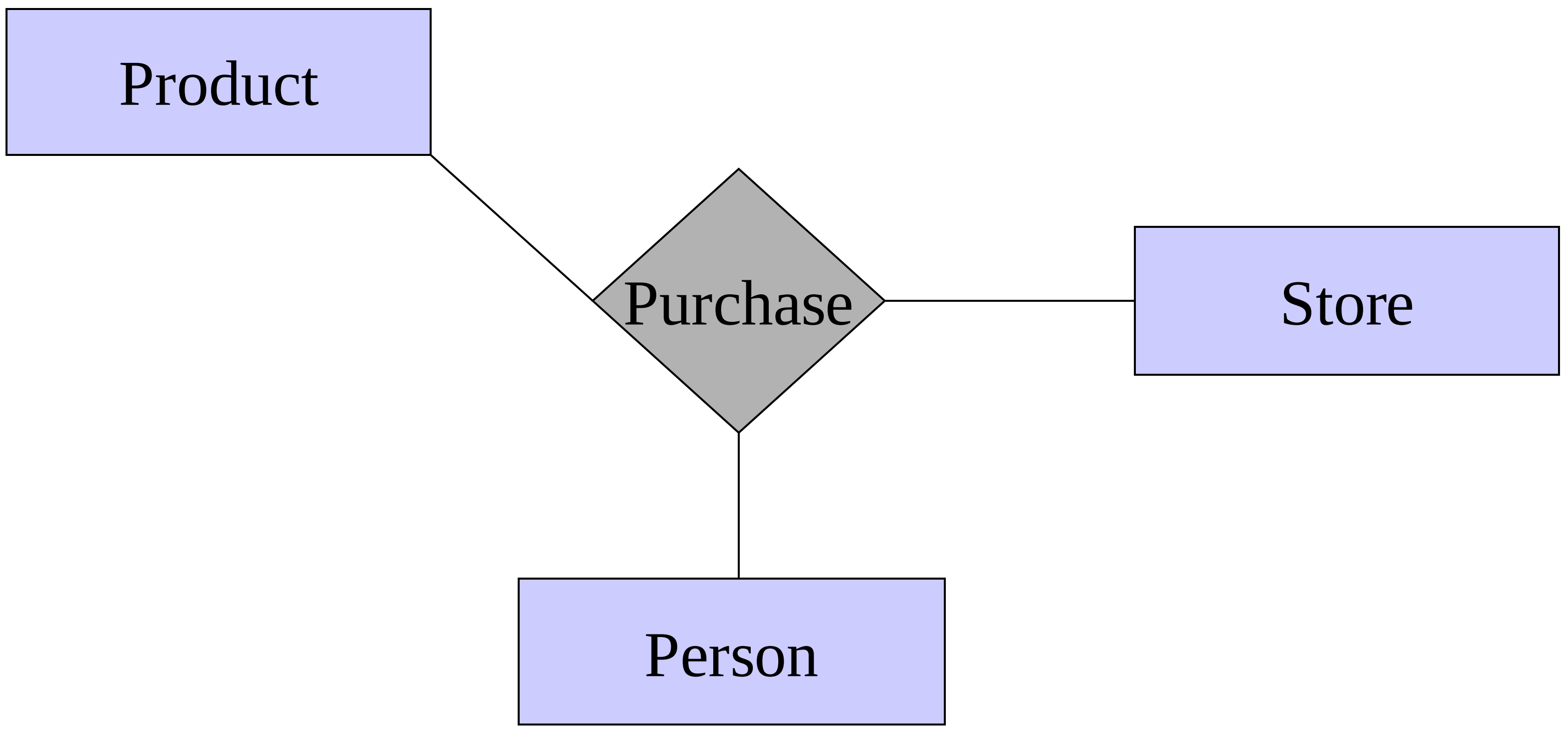
- You should use the notations in the lectures

Discussion

- one-one relation means
  - one entity in A is associated with AT MOST one entity in B, and vice versa
- how to translate these relations into tables?

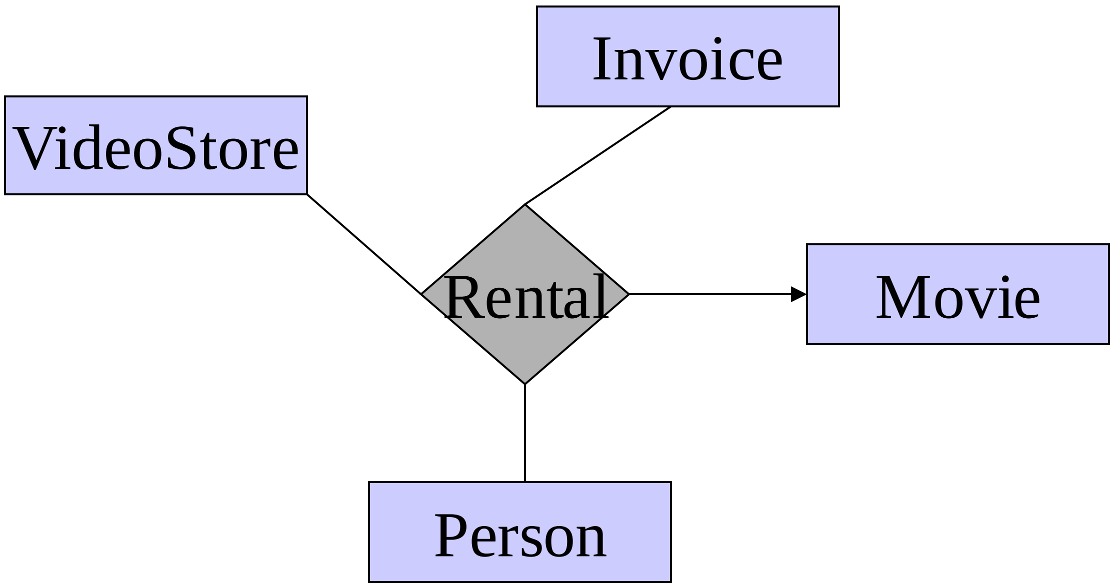
Multiway Relationships

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



Arrows in Multiway Relationships

Q: what does the arrow mean ?

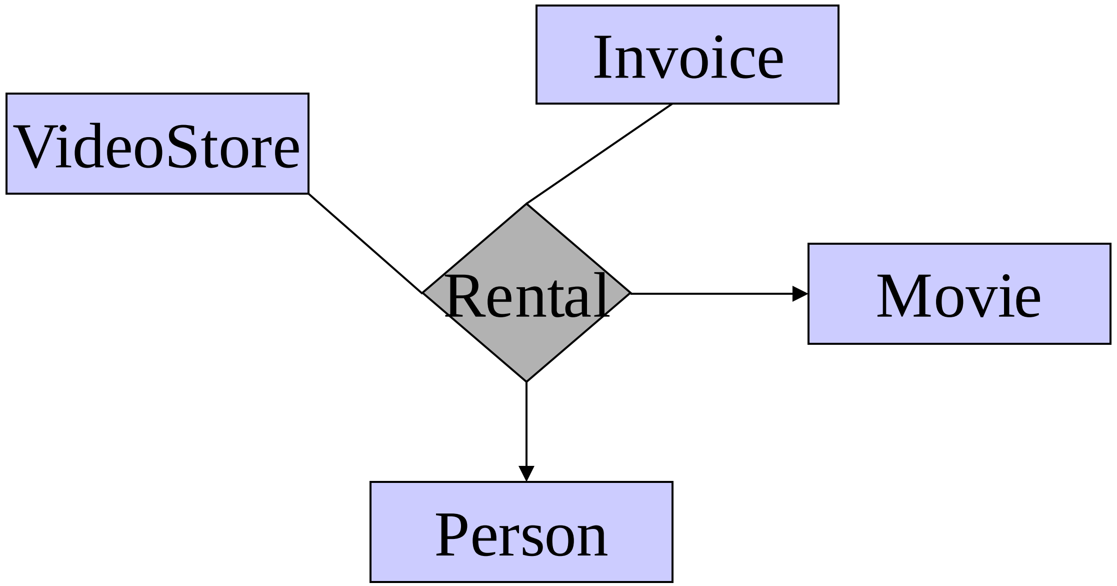


A: if I know the store, person, invoice, I know the movie too

will discuss in class what "know" means here

Arrows in Multiway Relationships

Q: what do these arrow mean ?



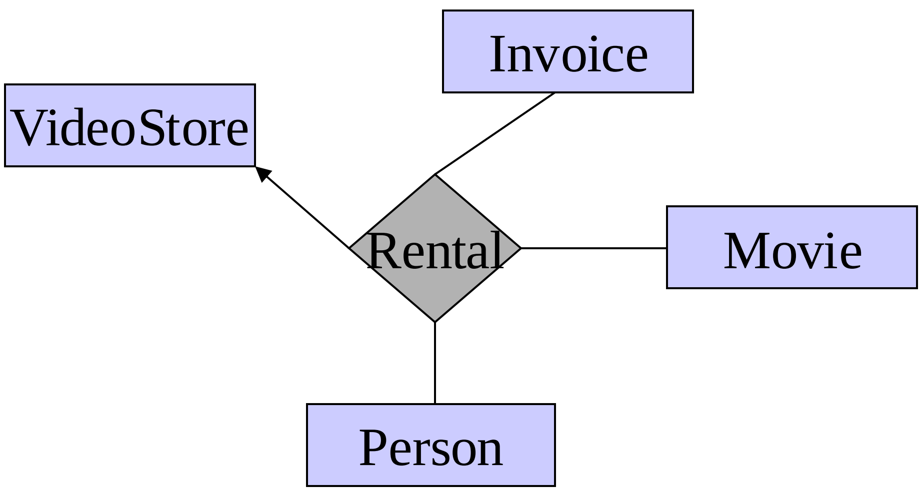
A: store, person, invoice determines movie
and store, invoice, movie determines person

will discuss in class what "determines" means

Arrows in Multiway Relationships

Q: how do I say: “invoice determines store” ?

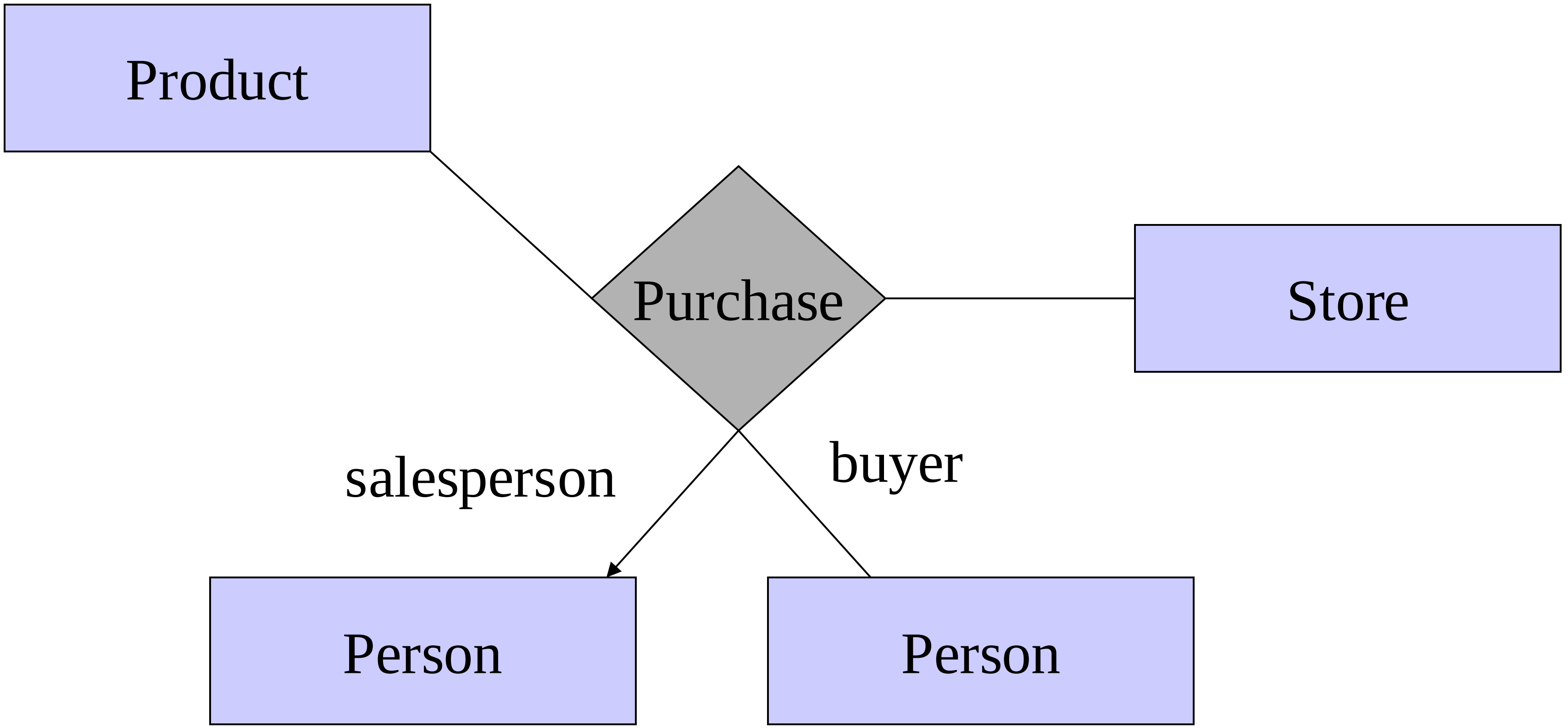
A: no good way; best approximation:



Q: Why is this incomplete ?

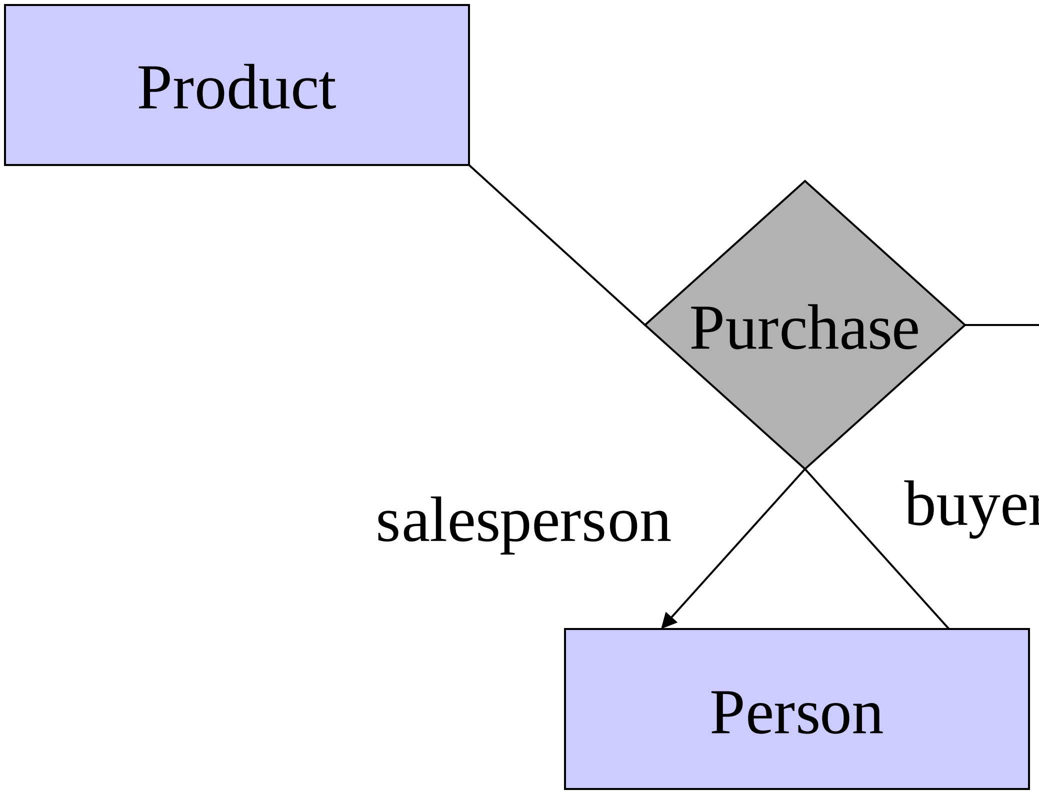
Roles in Relationships

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



Roles in Relationships

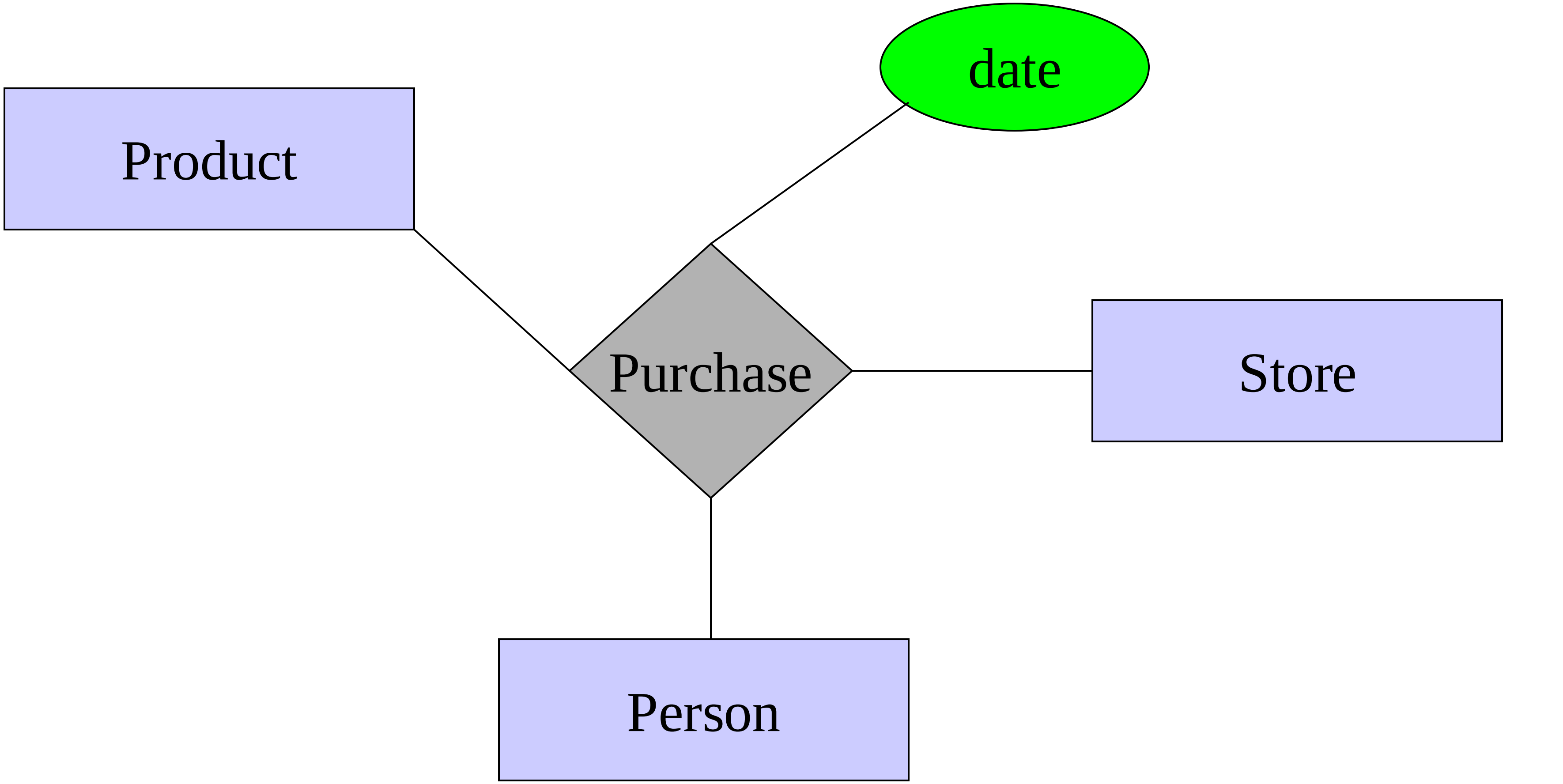
What if we need an entity set twice in one relationship?

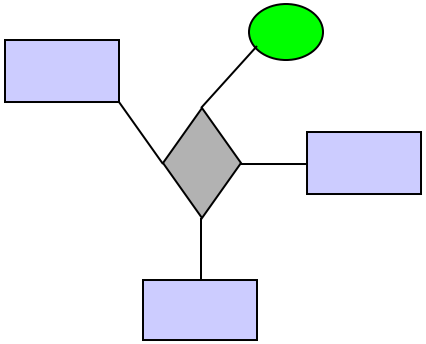


Another example: marriage

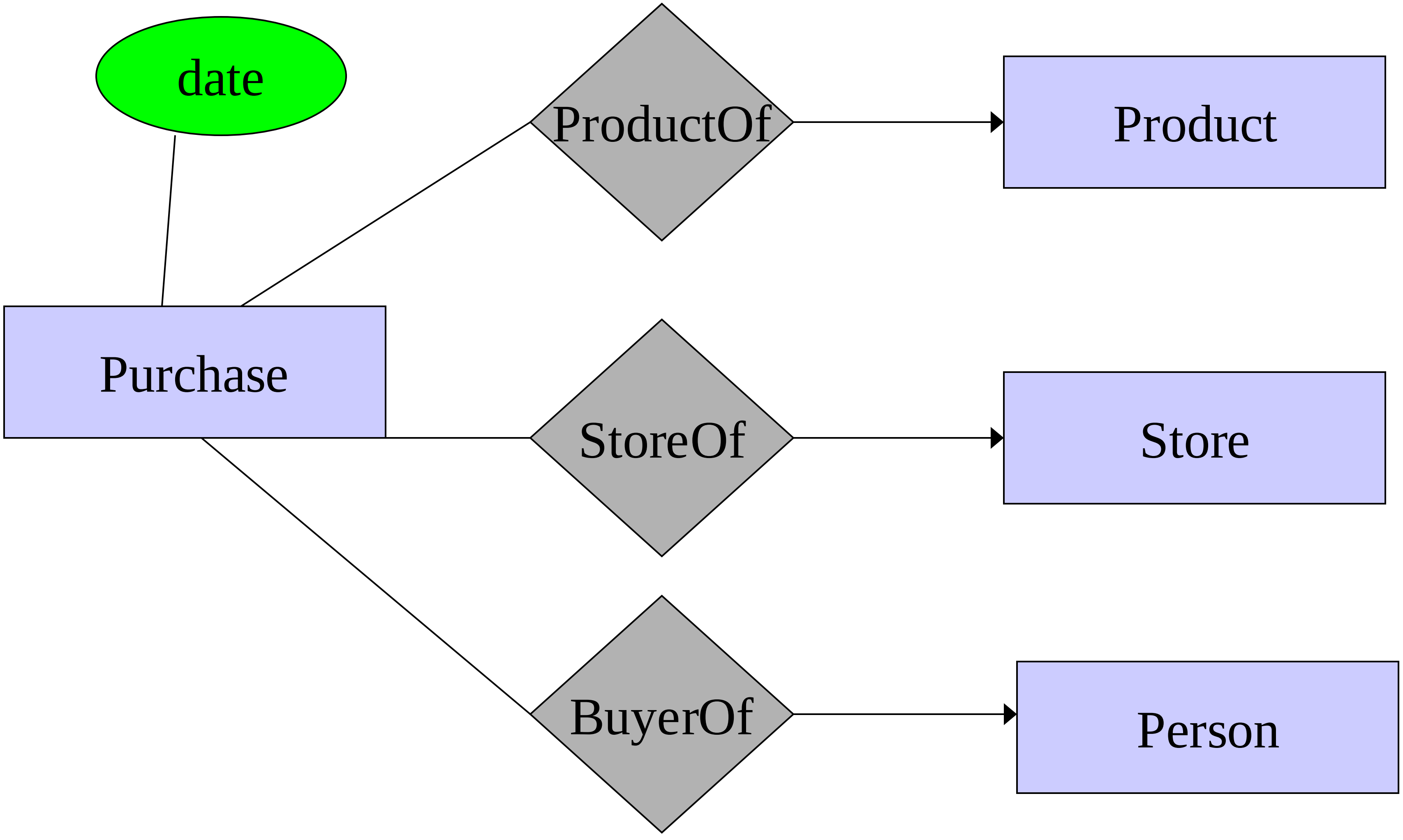
Discuss how to translate this to tables

Attributes on Relationships





Converting Multiway Relationships to Binary



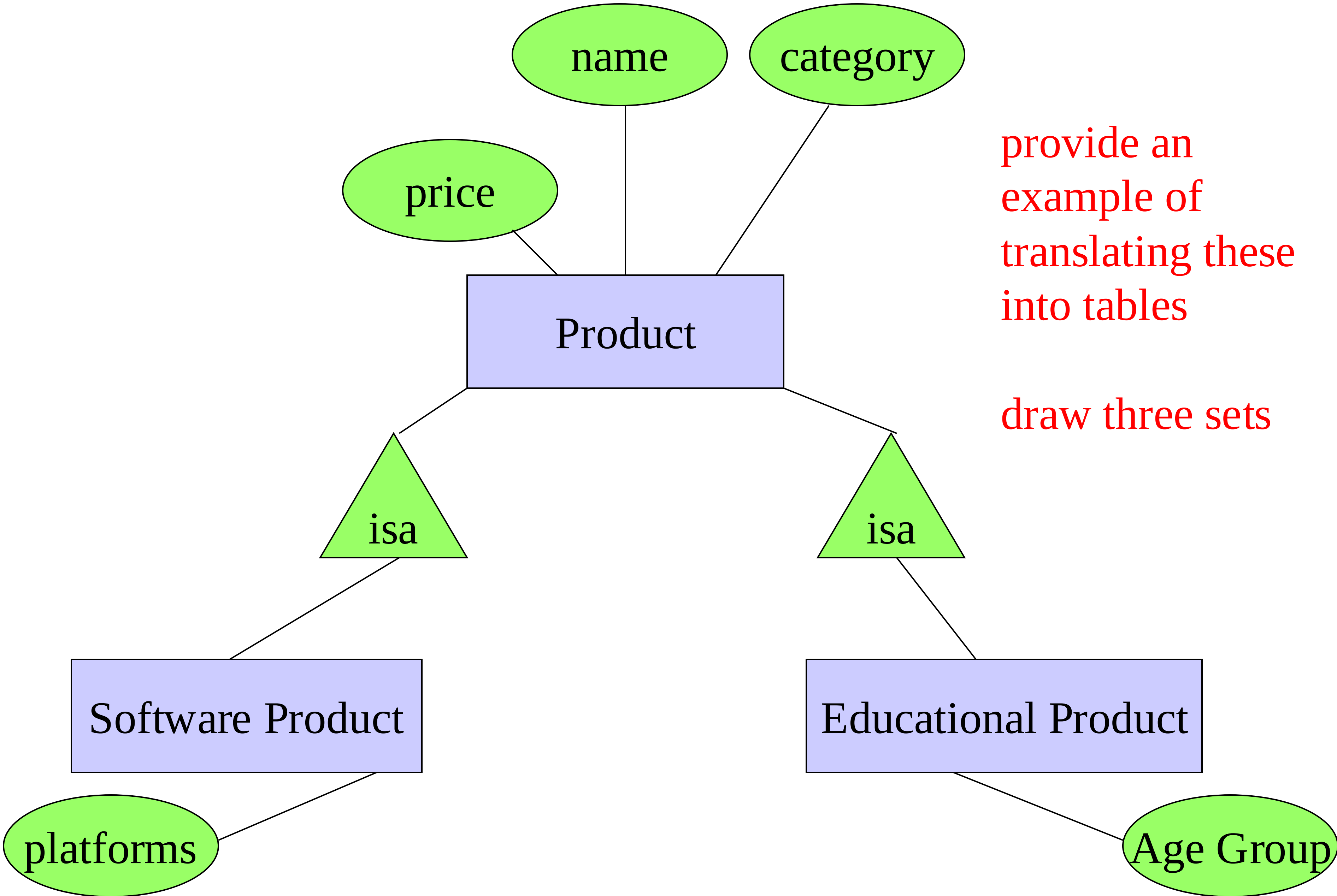
Discussion

- Give an example of tables to illustrate this conversion from multiway to binary relations
- Emphasize that
  - no clear boundary between entity set and relationship
  - something can be a relation, e.g., purchase
    - but we can revise the ER diagram to make it into an entity set

Relationships: Summary

- Modeled as a mathematical set
- Binary and multiway relationships
- Converting a multiway one into many binary ones
- Constraints on the degree of the relationship
  - many-one, one-one, many-many
  - limitations of arrows
- Attributes of relationships
  - useful to have

Subclasses in ER Diagrams



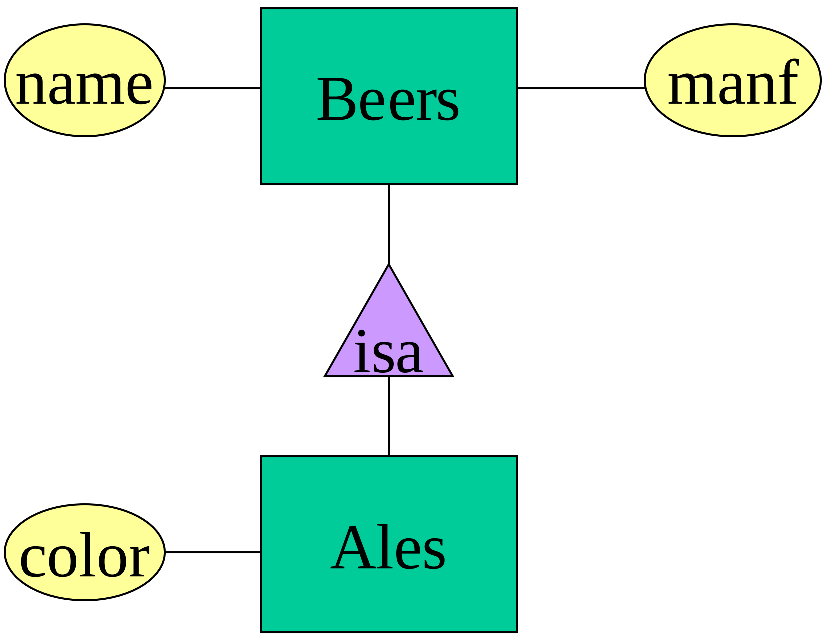
provide an example of translating these into tables

draw three sets

Subclasses

- Subclass = special case = fewer entities = more properties.
- Example: Ales are a kind of beer.
  - Not every beer is an ale, but some are.
  - Let us suppose that in addition to all the *properties* (attributes and relationships) of beers, ales also have the attribute *color*.

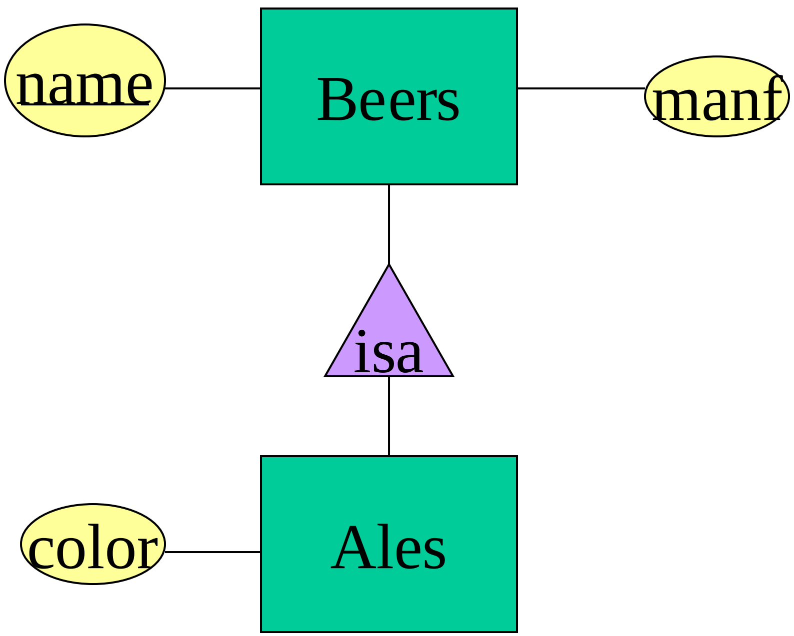
Example



Subclasses in ER Diagrams

- Assume subclasses form a tree.
  - I.e., no multiple inheritance.
- Isa triangles indicate the subclass relationship.
  - Point to the superclass.

Three Different Ways to Translate Class Hierarchies to Tables



Object-Oriented

name	manf
Bud	Anheuser-Busch

Beers

name	manf	color
Summerbrew	Pete's	dark

Ales

E/R Style

name	manf
Bud	Anheuser-Busch
Summerbrew	Pete's

Beers

name	color
Summerbrew	dark

Ales

Using Nulls

name	manf	color
Bud	Anheuser-Busch	NULL
Summerbrew	Pete's	dark

Beers

Comparison

name	manf
Bud	Anheuser-Busch

Beers

name	manf	color
Summerbrew	Pete's	dark

Ales

name	manf
Bud	Anheuser-Busch
Summerbrew	Pete's

Beers

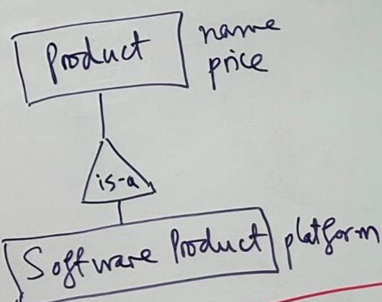
name	color
Summerbrew	dark

Ales

name	manf	color
Bud	Anheuser-Busch	NULL
Summerbrew	Pete's	dark

Beers

It's all
about trade-offs



object-oriented

Prods

name	price
A	2
B	6
C	4

SProds

name	price	platform
D	1	X
E	5	Y

ER style

Prods

name	price
A	2
B	6
C	4
D	1
E	5

SProds

name	platform
D	X
E	Y

using NULLs

AllProds

name	price	platform
A	2	NULL
B	6	NULL
C	4	NULL
D	1	X
E	5	Y

Constraints

- When you talk with the business person, you need to find out
  - all entities and relationships that person wants to model
- But also need to find out as many constraints as possible
  - e.g., this relationship is 1-1, or many-1, or 1-many
  - more examples of constraints in next slide

Modeling Constraints

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.

Domain constraints: peoples' ages are between 0 and 150.
General constraints: all others (at most 50 students enroll in a class)

Why Constraints are Important?

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

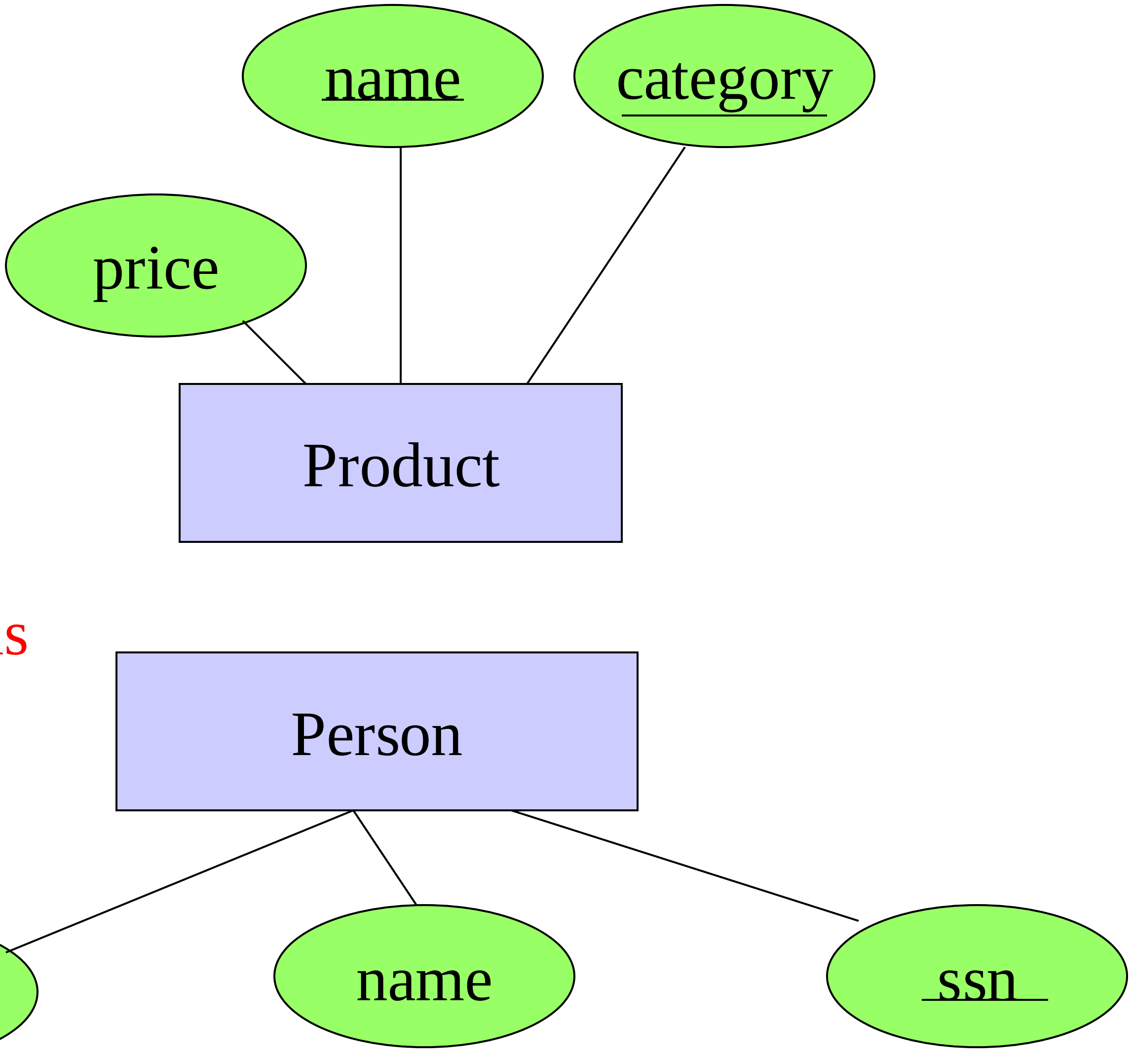
Specifying Constraints

- For the constraints you find out from the business person
  - specify them on the ER diagram
  - there may be special notations to represent some constraints
  - for other constraints, there are no special notations, you just write them down in English on the ER diagram

Keys in E/R Diagrams

Underline:

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

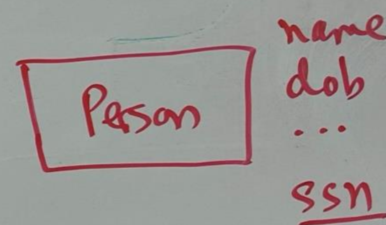


More about Keys

- Every entity set must have a key
  - why?
- 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
- Requirement for key in an isa hierarchy
  - not covered in this lecture

Person

name	DoB	address	phone	email	ssn
A	1	M	6	x	101
B	2	O	7	-	102
A	3	M	8	y	103
D	4	P	9	z	104
E	4	Q	10	u	105
A	3	N	11	-	106



name + DOB?

name + DOB + address?

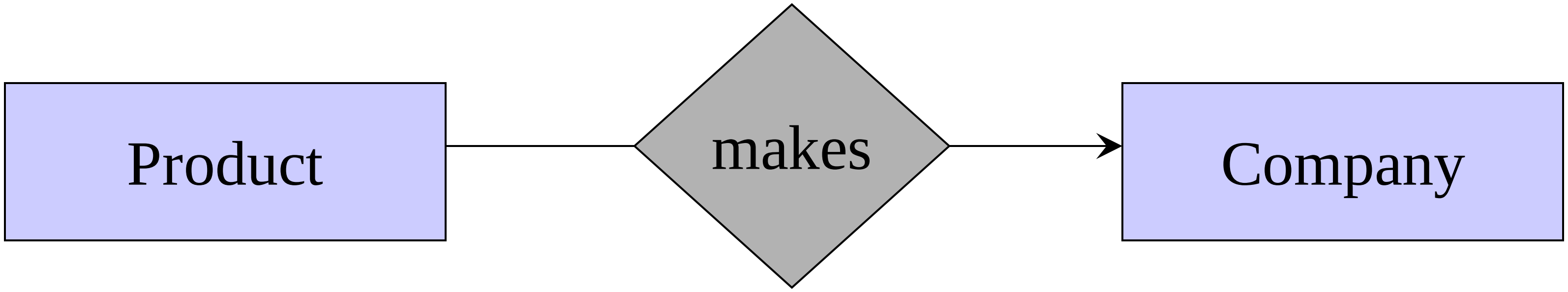
email?

ssn?

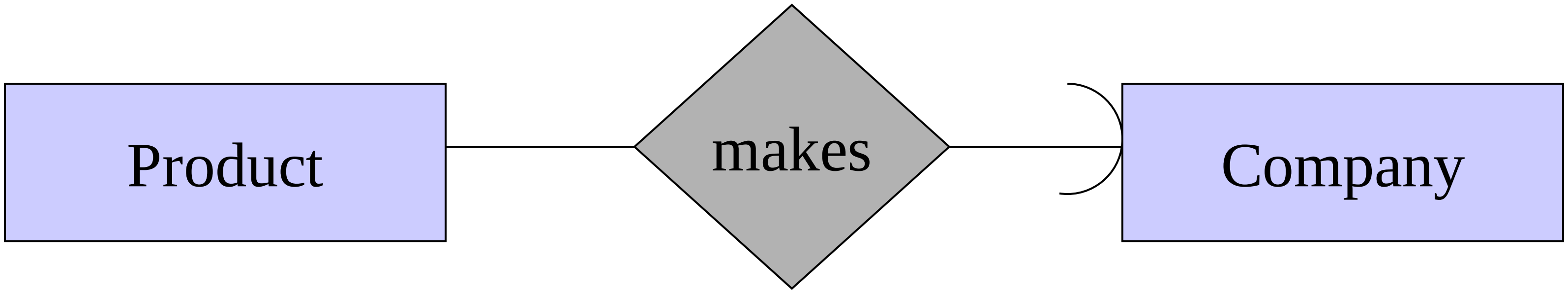
Single Value Constraint

- An attribute of an entity set or a relation is "atomic", that is, has a single value

A Constraint on Many-One Relationship (A Non-NULL Constraint)



Each product is “made” by at most one company.



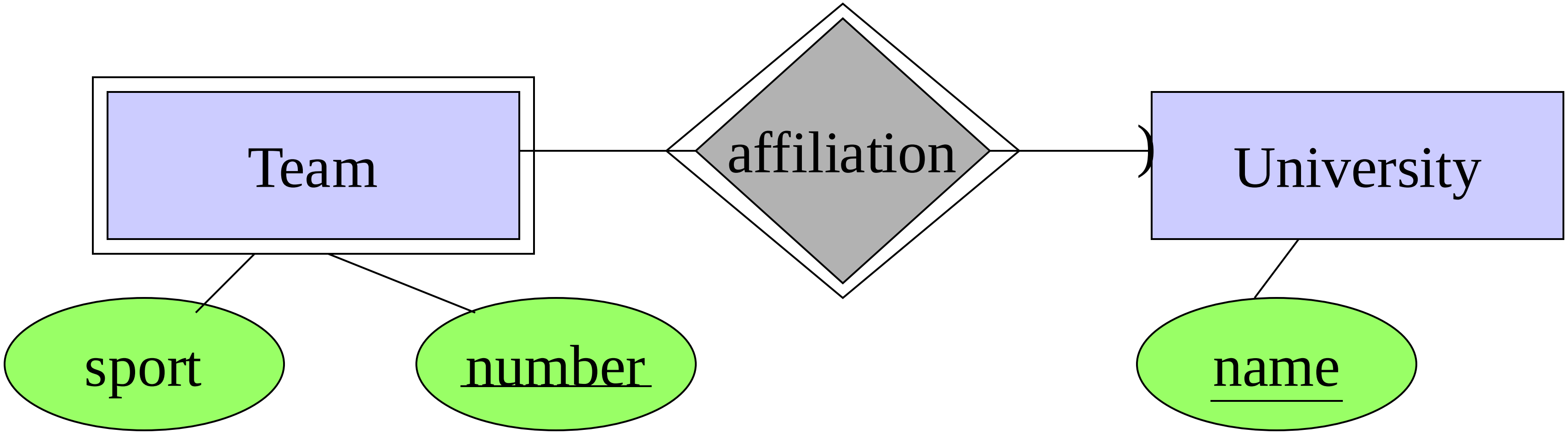
Each product is “made” by exactly one company.

Referential Integrity Constraint (Foreign Key Constraint)

- A constraint between two tables A and B
- Give an example: Students take Courses

Weak Entity Sets

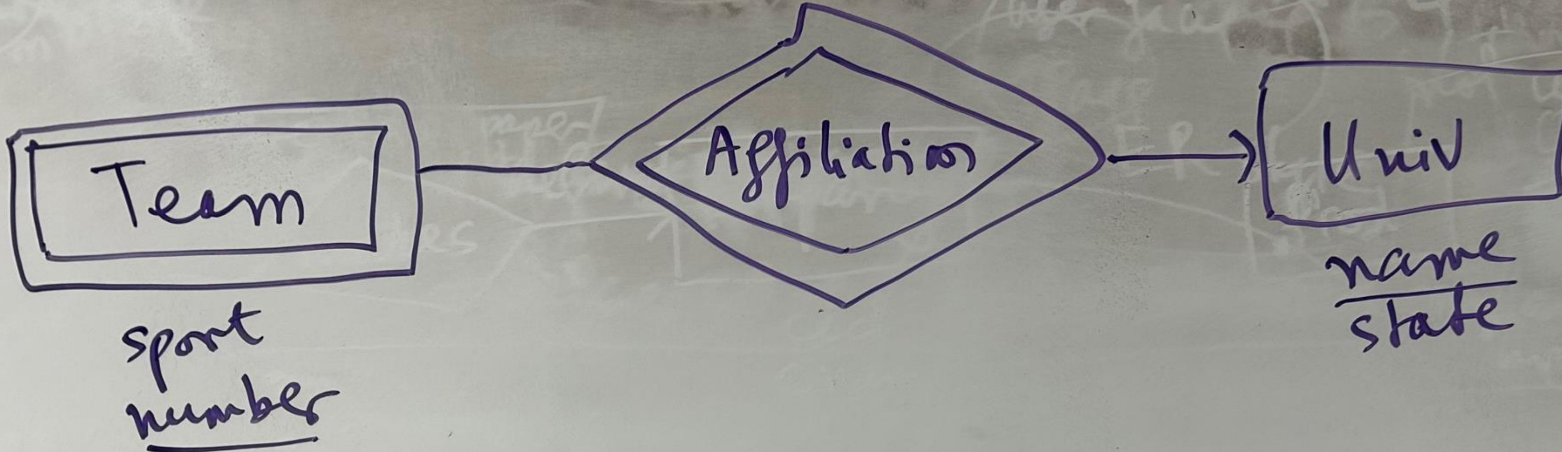
Entity sets are weak when their key attributes come from other classes to which they are related.



Weak Entity Sets

- Occasionally, entities of an entity set need "help" to identify them uniquely.

- Entity set *E* is said to be *weak* if in order to identify entities of *E* uniquely, we need to follow one or more many-one relationships from *E* and include the key of the related entities from the connected entity sets.



Teams

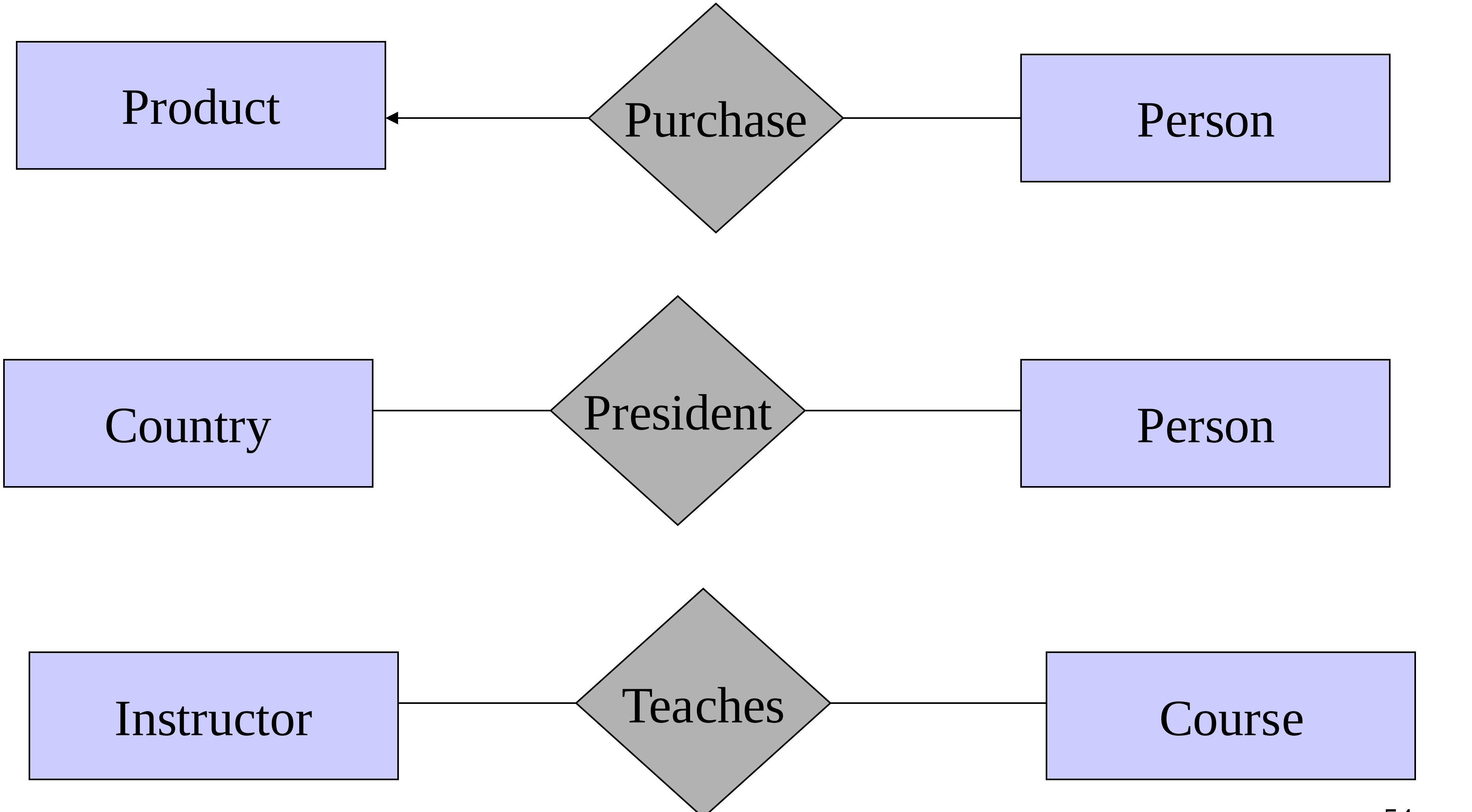
sport	number	uname
swim	1	A
run	2	A
swim	1	B
swim	2	B
run	3	B

Univs

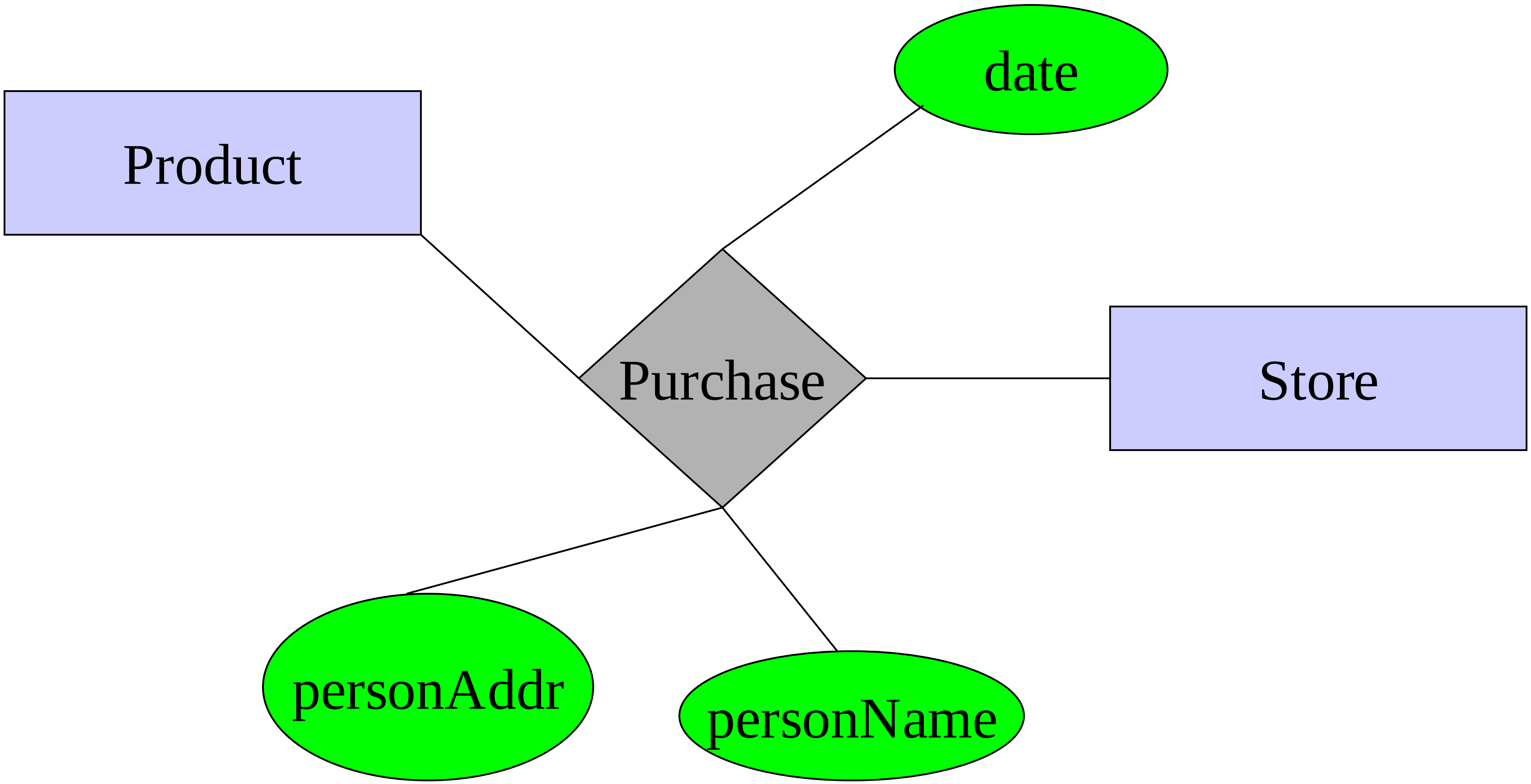
name	state
A	X
B	Y
C	Z
D	W

Now, about design techniques ...

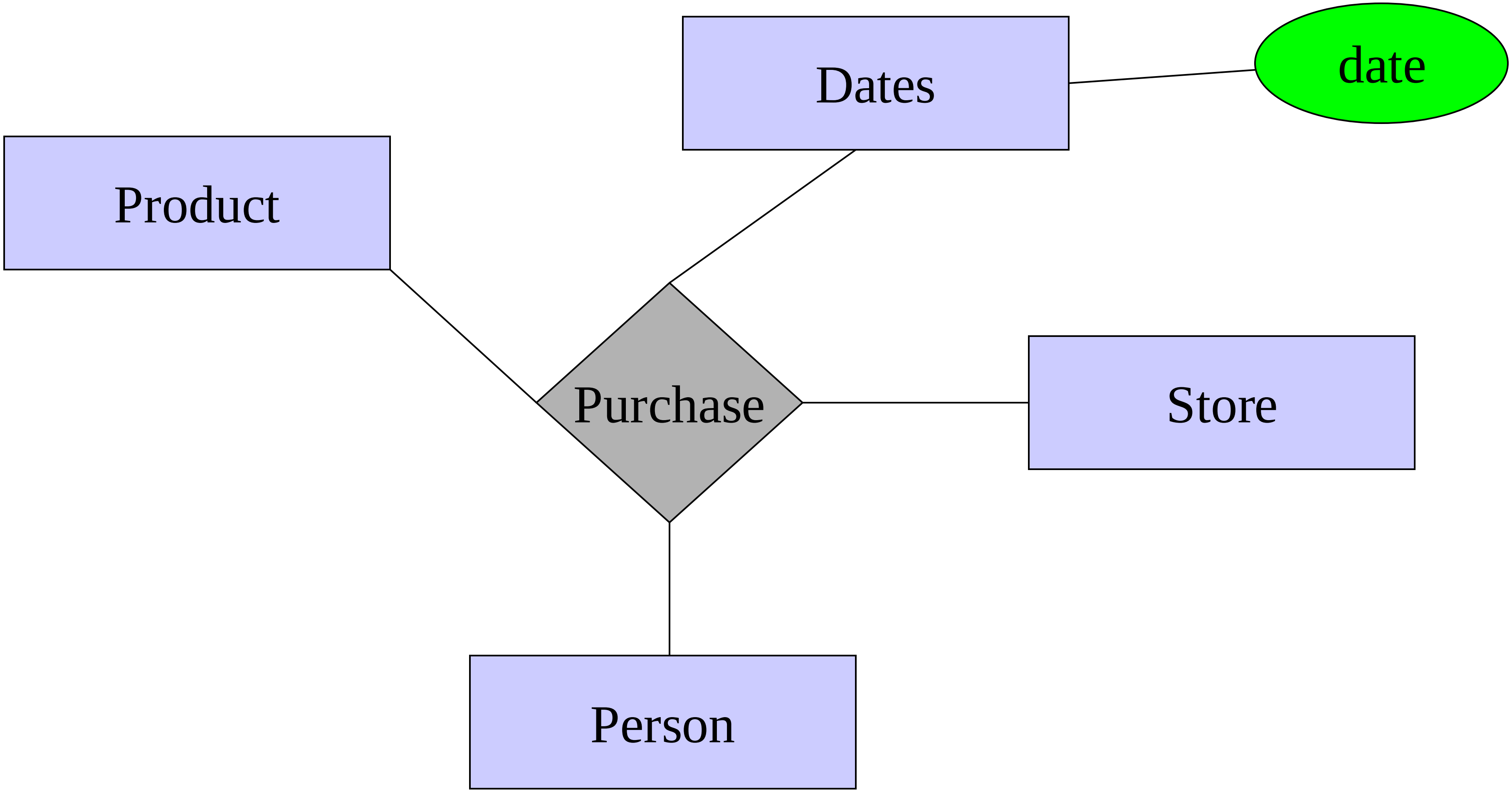
Design Principles 1: Be Faithful



Design Principles 2: Avoid Redundancy



Design Principles 3: KISS



- Read the book for more design principles

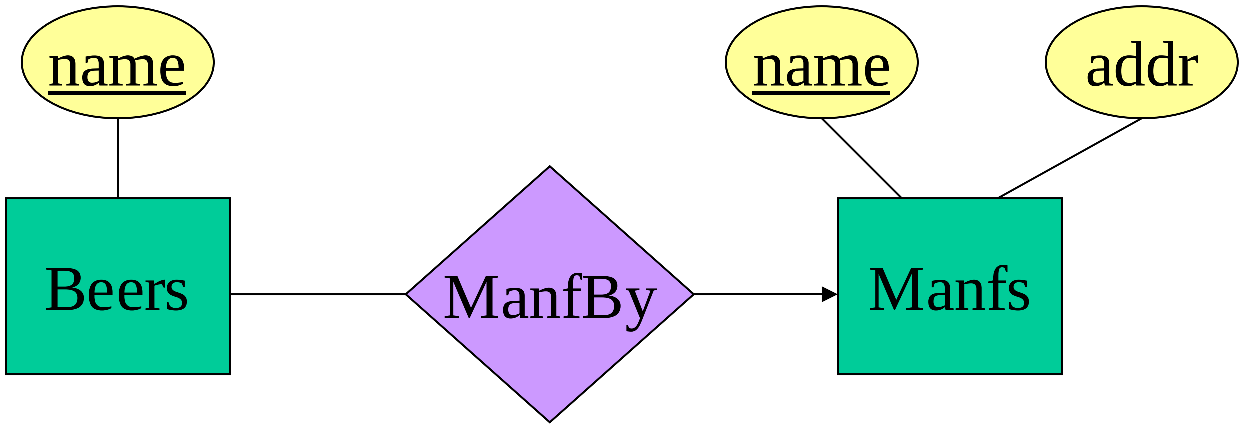
More on Design Techniques (from Ullman's slides)

1. Avoid redundancy.
2. Limit the use of weak entity sets.
3. Don’t use an entity set when an attribute will do.

Avoiding Redundancy

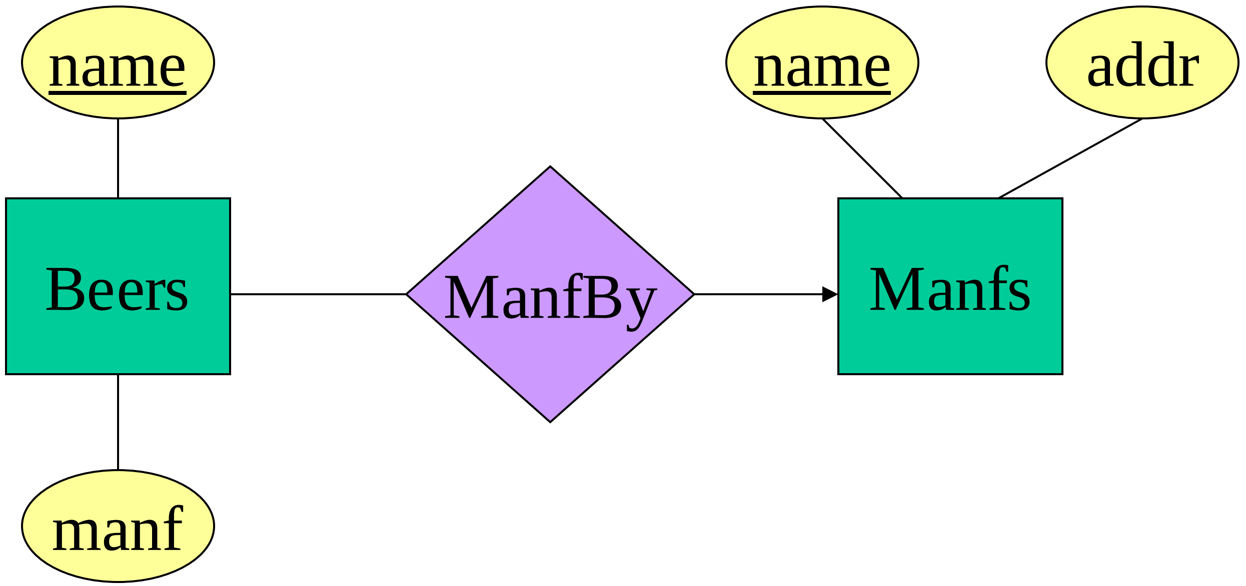
- Redundancy occurs when we say the same thing in two different ways.
- Redundancy wastes space and (more importantly) encourages inconsistency.
  - The two instances of the same fact may become inconsistent if we change one and forget to change the other, related version.

Example: Good



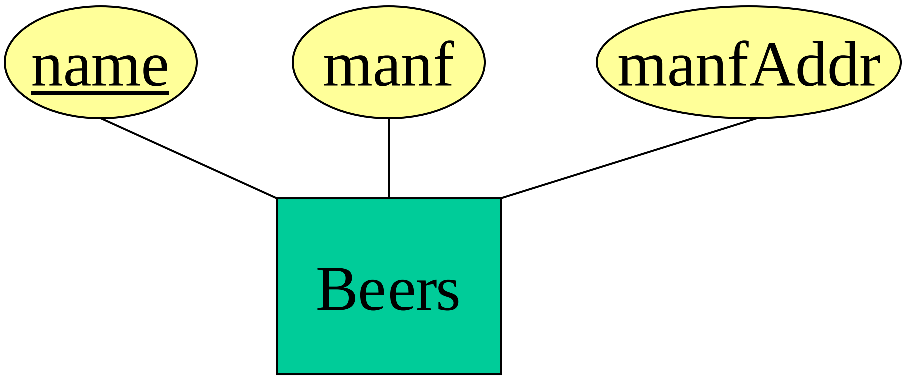
This design gives the address of each manufacturer exactly once.

Example: Bad



This design states the manufacturer of a beer twice: as an attribute and as a related entity.

Example: Bad



This design repeats the manufacturer's address once for each beer; loses the address if there are temporarily no beers for a manufacturer.

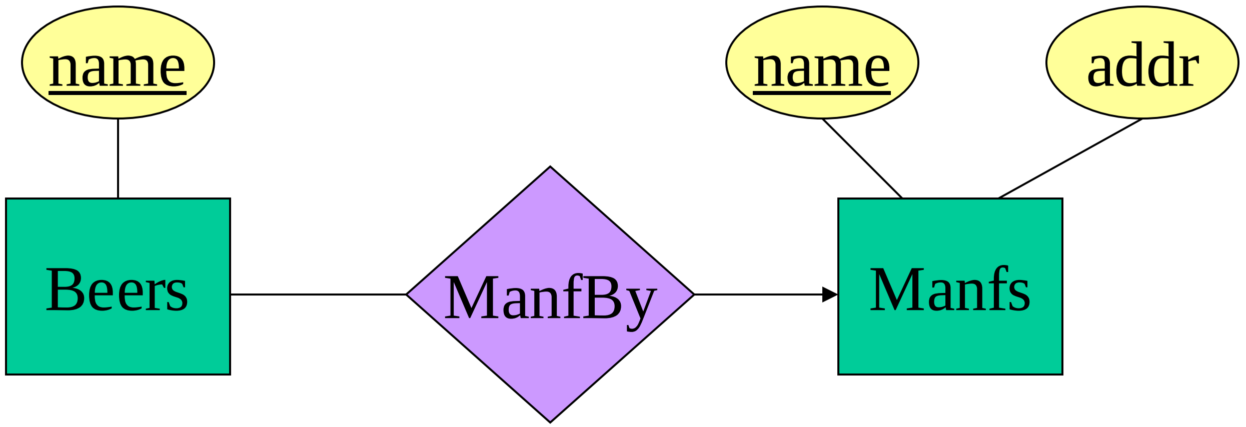
Entity Sets Versus Attributes

- 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 nonkey attribute.

    or

  - It is the “many” in a many-one or many-many relationship.

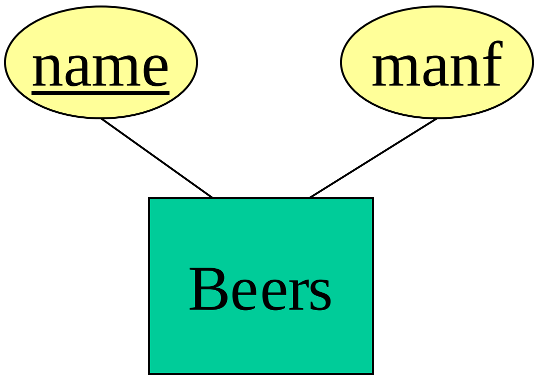
Example: Good



- *Manfs* deserves to be an entity set because of the nonkey attribute *addr*.

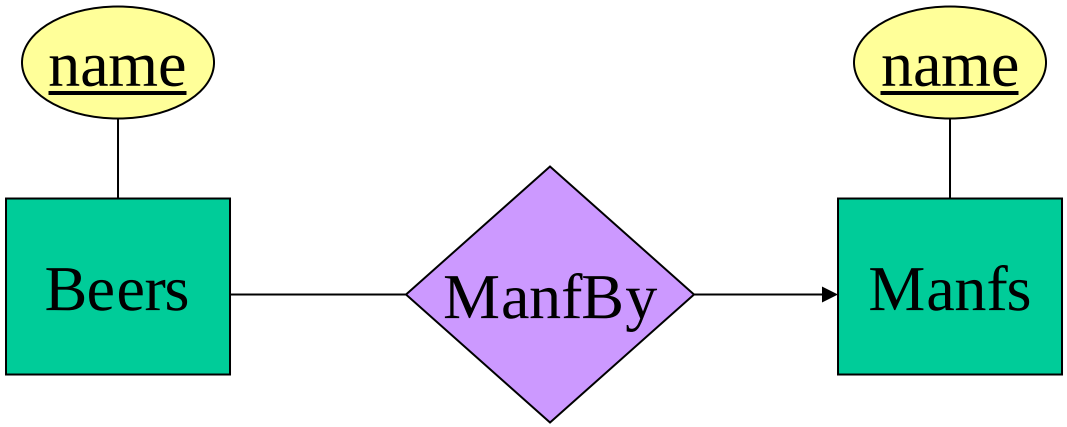
- *Beers* deserves to be an entity set because it is the "many" of the many-one relationship *ManfBy*.

Example: Good



There is no need to make the manufacturer an entity set, because we record nothing about manufacturers besides their name.

Example: Bad



Since the manufacturer is nothing but a name, and is not at the "many" end of any relationship, it should not be an entity set.

Don’t Overuse Weak Entity Sets

- Beginning 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 usually create unique ID’s for entity sets.
  - Examples include social-security numbers, automobile VIN’s etc.

When Do We Need Weak Entity Sets?

- The usual reason is that there is no global authority capable of creating unique ID's.
- Example: it is unlikely that there could be an agreement to assign unique player numbers across all football teams in the world.

ER Review

- Basic stuff
  - entity, attribute, entity set
  - relation: binary, multiway, converting from multiway
  - relationship roles, attributes on relationships
  - subclasses (is-a)
- Constraints
  - on relations
    - many-one, one-one, many-many
    - limitations of arrows
  - keys, single-valued, ref integrity, and more

ER Review

- Weak entity set
- Design principles
  - be faithful
  - avoid redundancy
  - KISS

Higher-Level Takeaways

- There are all kinds of languages in CS
  - programming languages, data languages
- Data languages can be used for multiple purposes
  - to talk with others (say business people): ER
  - to manipulate data: SQL, XML, JSON, graph data languages, etc.
- Data languages can be simple or complex
- There are always trade-offs
  - regarding understandability, ease of writing, expressive power, fast querying, efficient storage, etc.