Updated 10/20/2022

Topic 2: Database design
L12: ER modeling

Wolfgang Gatterbauer

CS3200 Database design (fa22)

https://northeastern-datalab.github.io/cs3200/fa22s3/

10/19/2022

Class warm-up

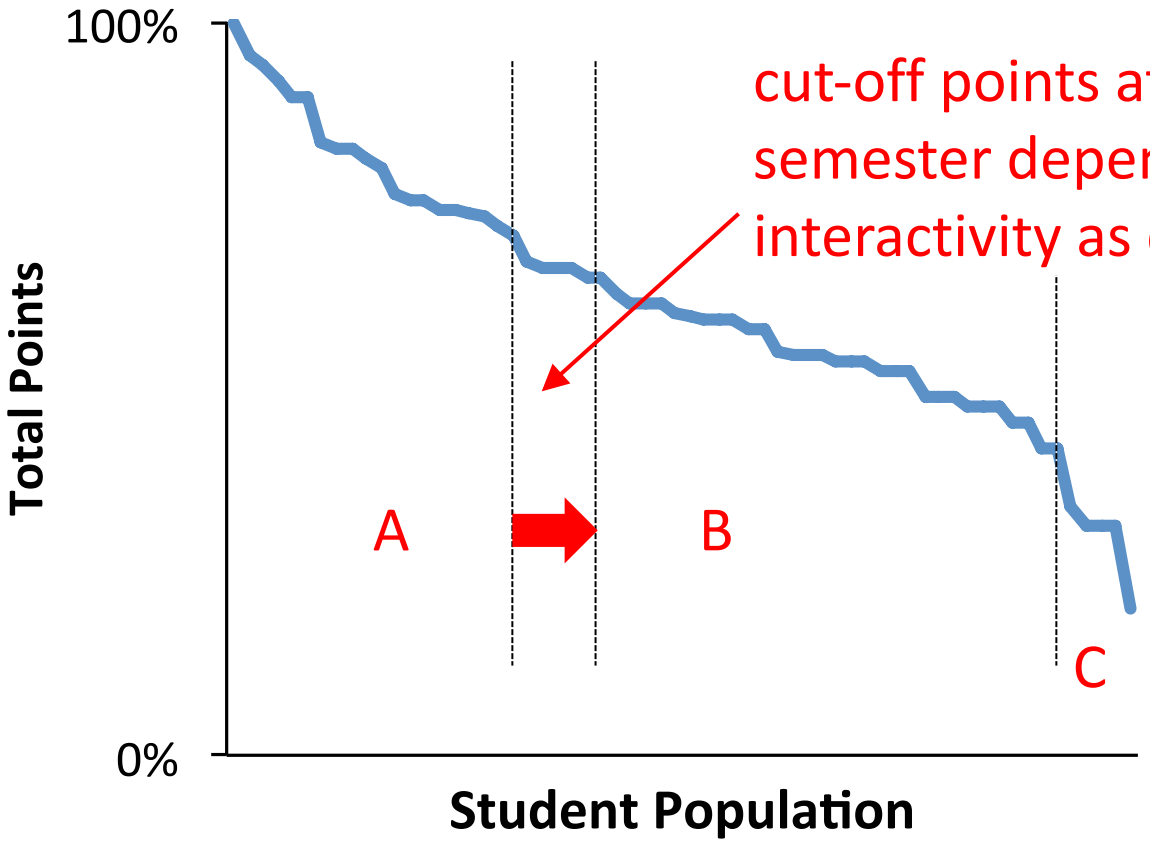
- Last class summary
- Please have paper and pen ready (or touchscreen, whatever works)
  - Please ask questions: many aspects of "database design" are not set in stone and in the "eye of the designer"
- Open Exam1 discussion:
  - I show you the preliminary distribution (remember Points vs Grades)
  - Was it fair and types of problems similar to problems seen in class and on HWs? Open-book vs closed book (open is more time constraint)
  - Please use our various options for feedback

- We continue with Database Design, hands-on

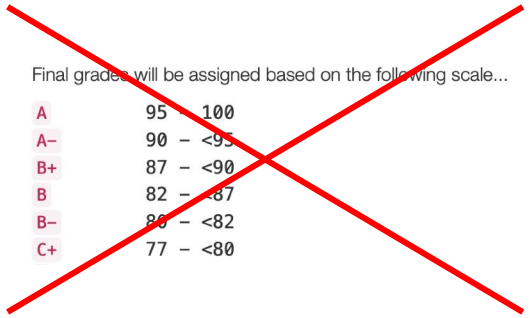
Grading Philosophy

Actual point distribution from a past final exam: long, but fair!

- no fixed percentages (e.g., top 30% get A)
- no fixed cut-offs (e.g., 80/100 points for A)



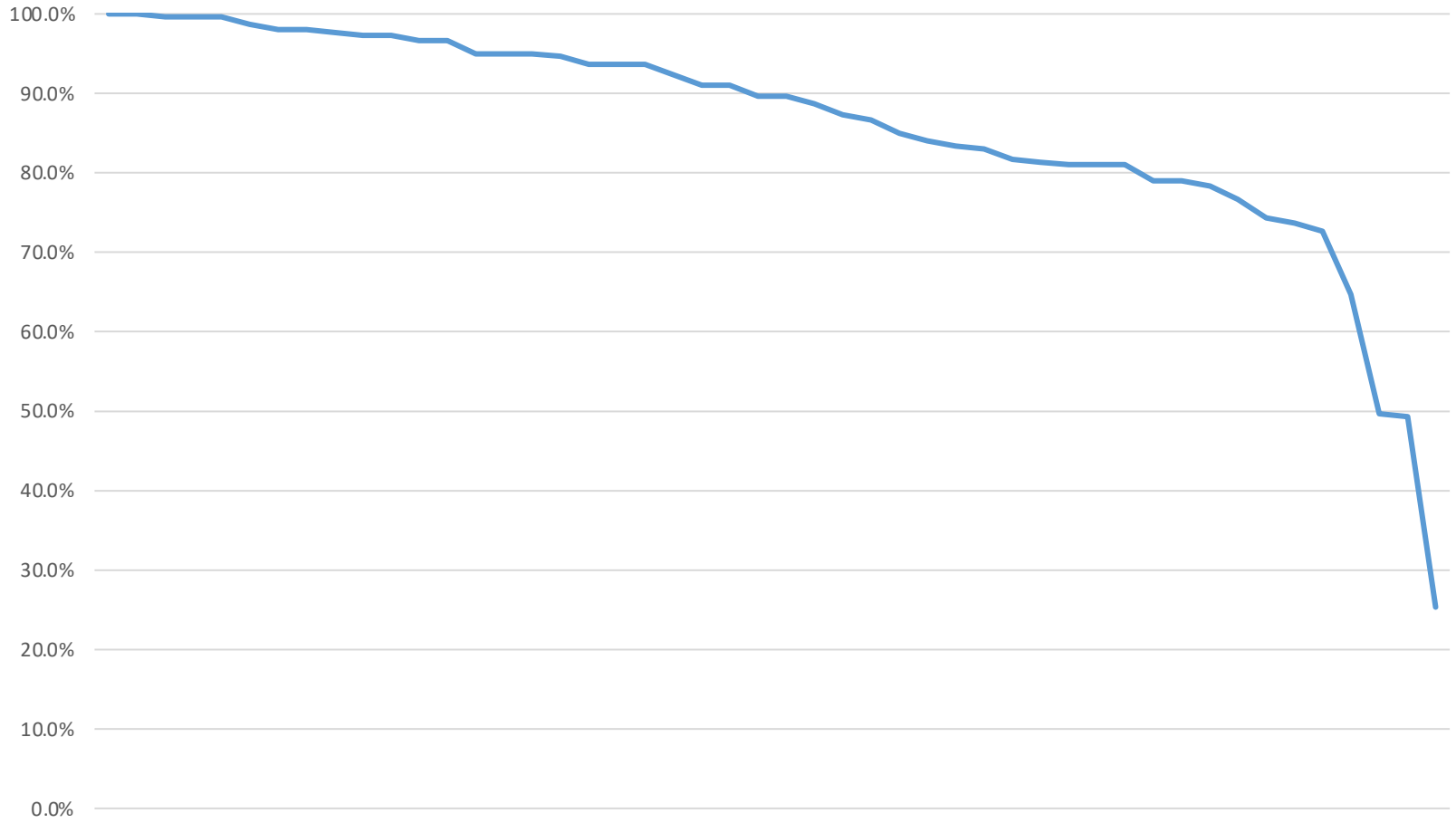
cut-off points at the end of the semester depend on overall class interactivity as compared to other years



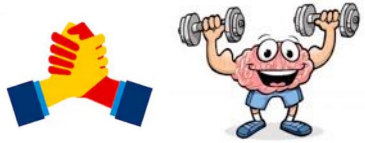
Contrast external grades vs. internal scores: Grade distributions are similar to other sections, but scores distributions are not

I will not disclose the actual cut-off points. Don't ask for an exception.

PRELIMINARY !



Example: modeling time-dependent data

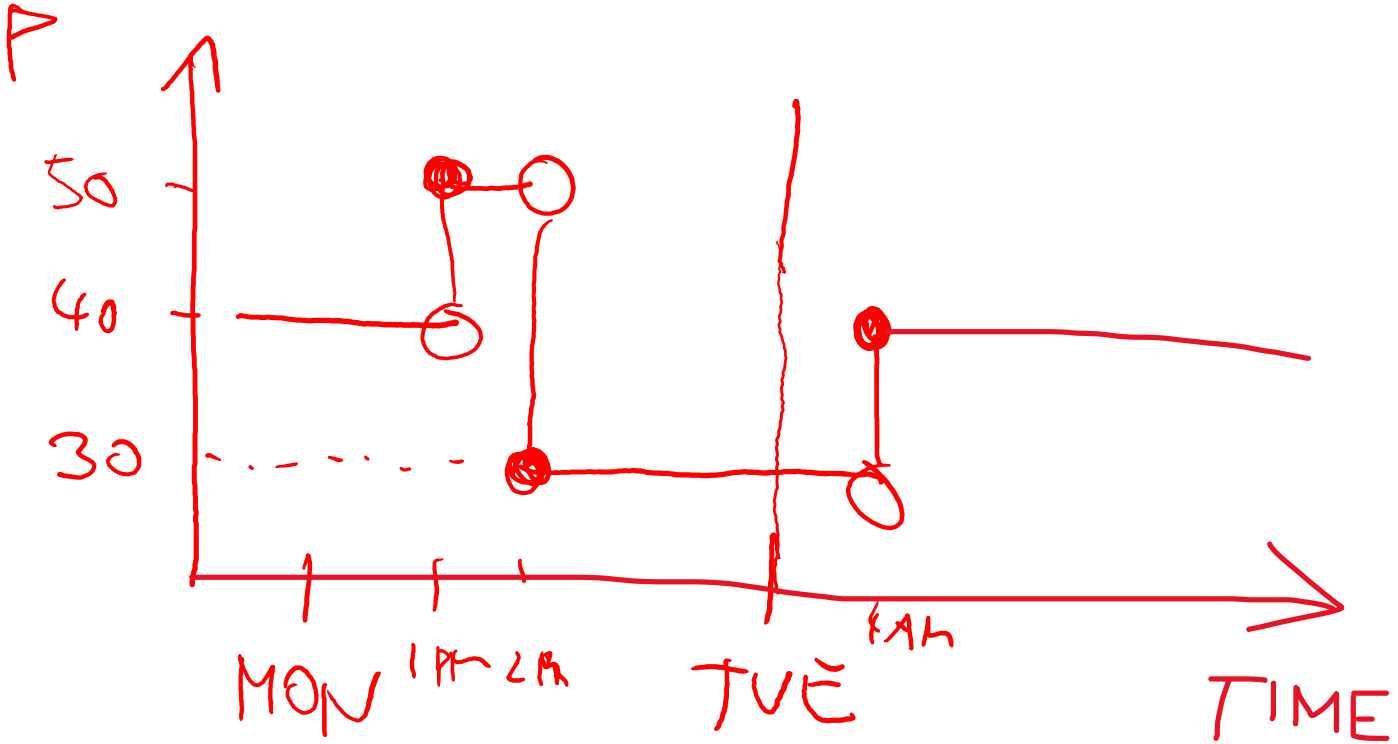


- Assume you have an entity "product"
- The price can change over time; you like to preserve the history of prices and the time period
- How to model that?

?

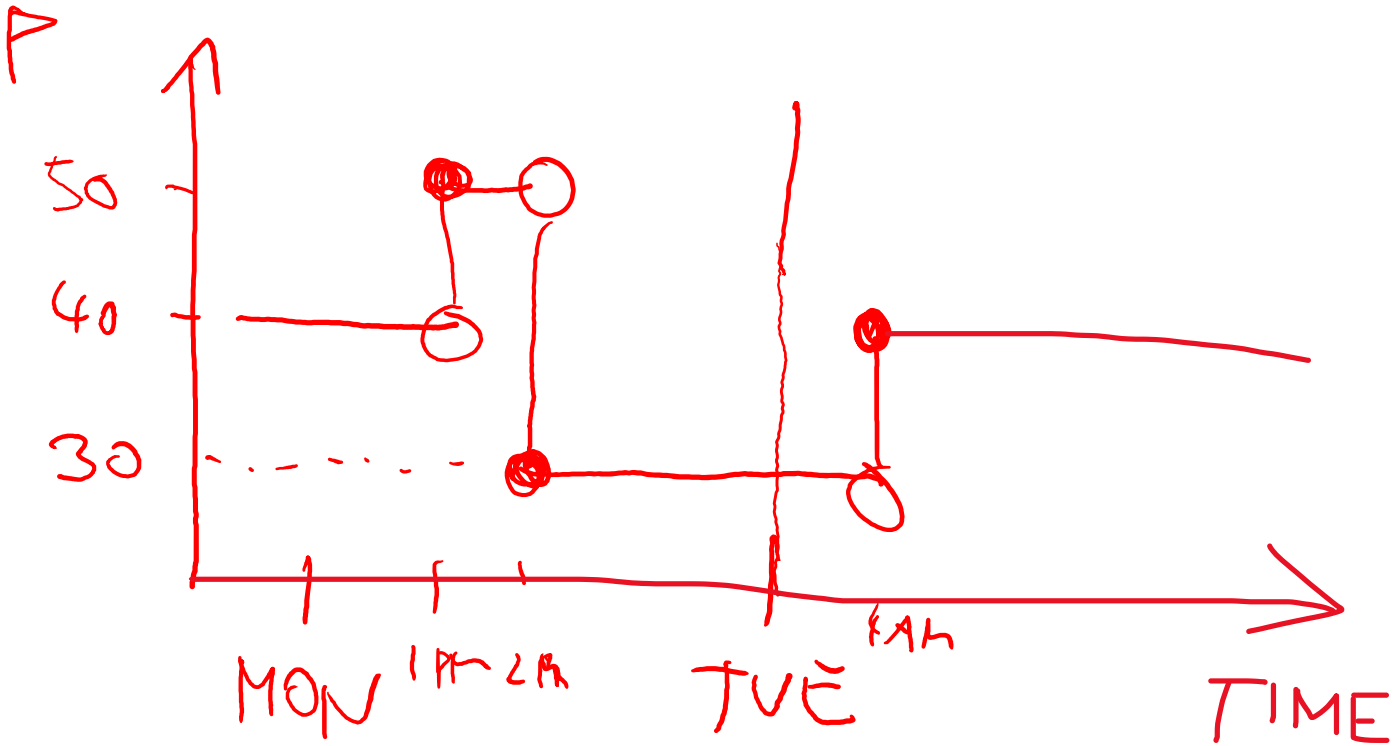
Example: modeling time-dependent data

- Assume you have an entity "product"
- The price can change over time; you like to preserve the history of prices and the time period
- How to model that?



Example: modeling time-dependent data

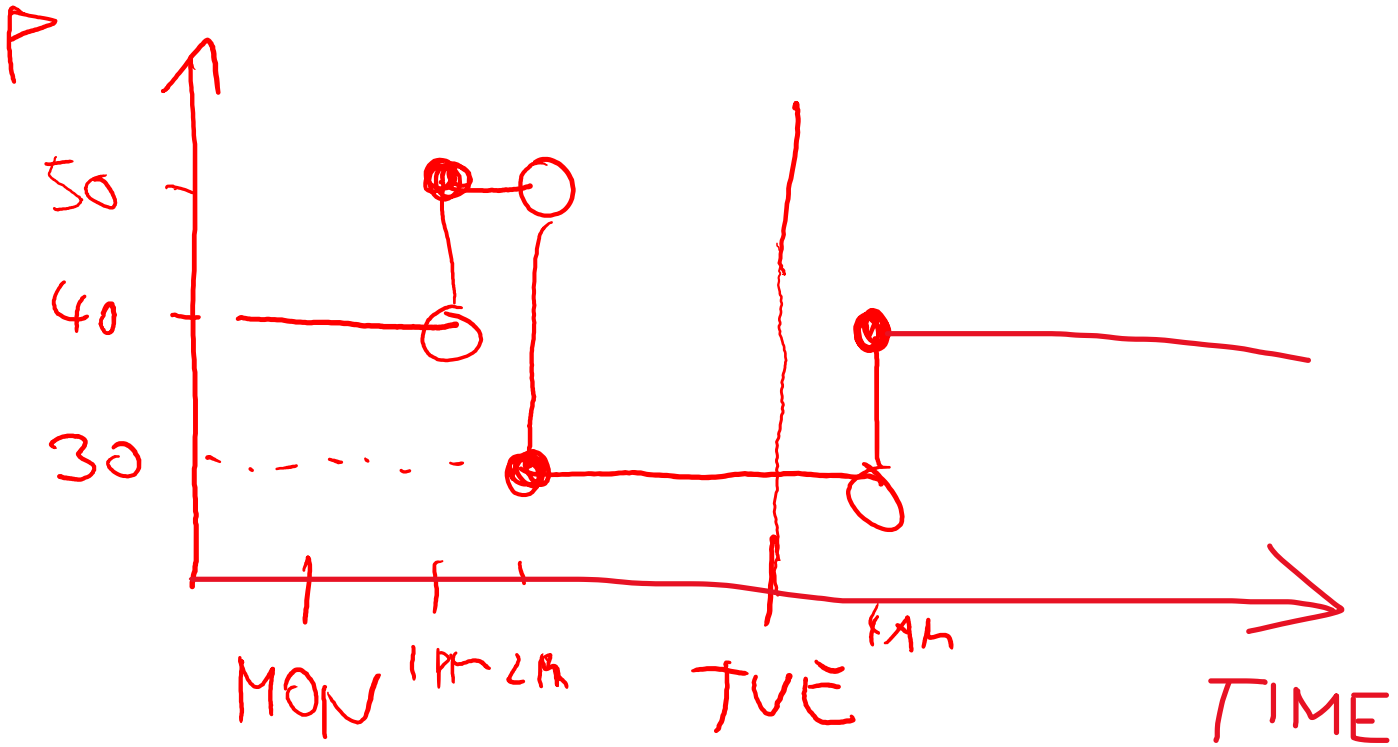
- Assume you have an entity "product"
- The price can change over time; you like to preserve the history of prices and the time period
- How to model that?



{ (P, S, E) }

Example: modeling time-dependent data

- Assume you have an entity "product"
- The price can change over time; you like to preserve the history of prices and the time period
- How to model that?



{ (P, S, [E]) }

40 SUN MON 1PM

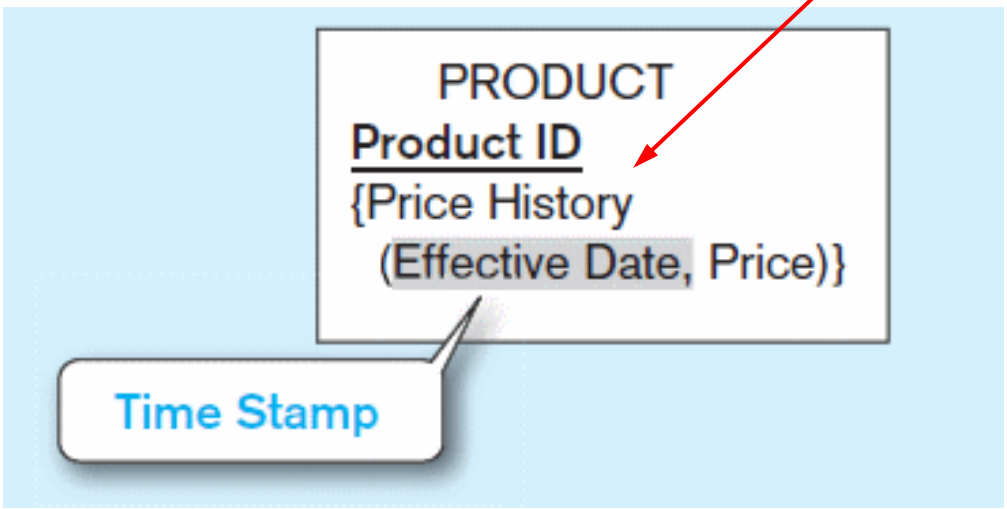
50 MON 1PM 2PM

30 2PM

Example: modeling time-dependent data

- Assume you have an entity "product"
- The price can change over time; you like to preserve the history of prices and the time period
- How to model that?

Usually also includes an extra attribute "current price"



Time-stamping is commonly done with a multi-valued and composite attribute (or associative entities: see later)

Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

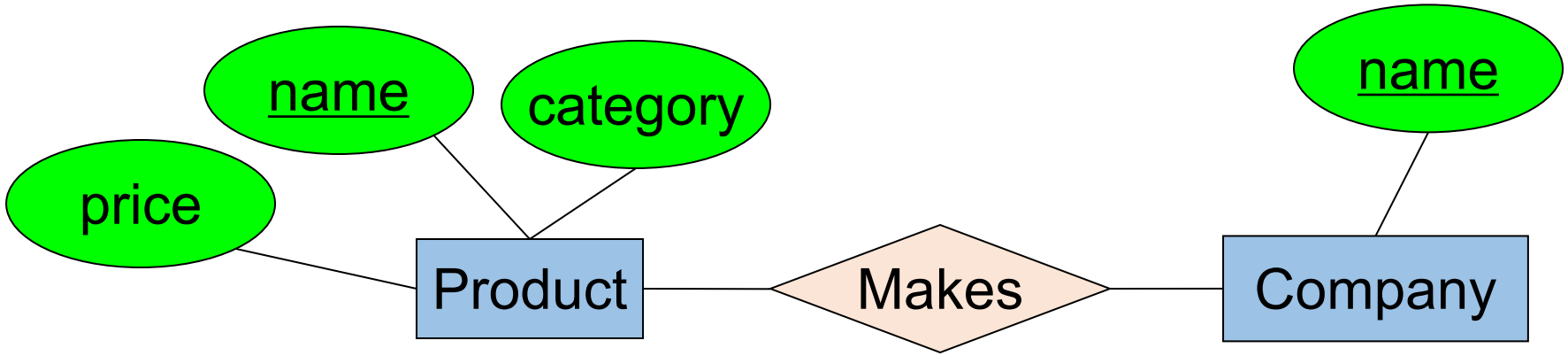
Relationships

The R in ER: Relationships

- A relationship is between two or more entities

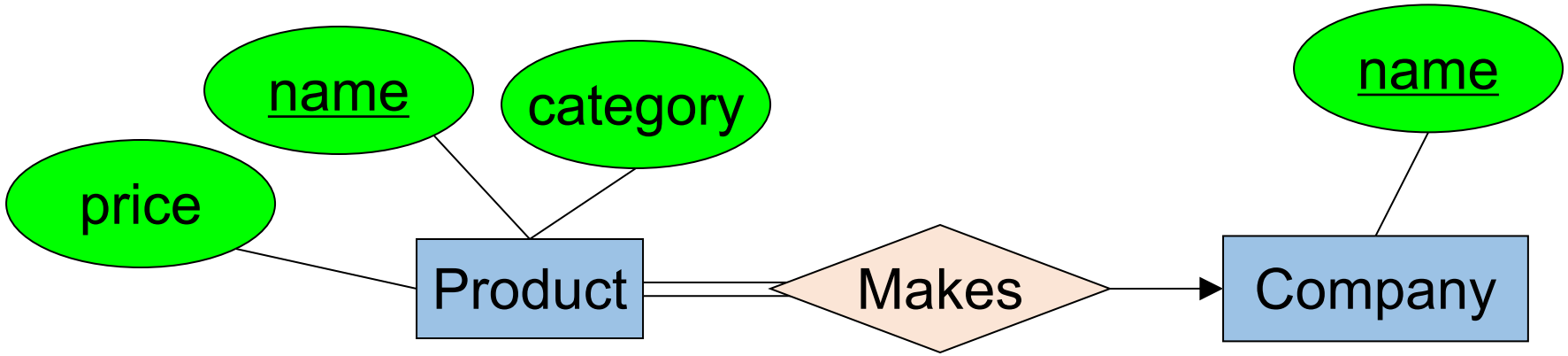
How to show that every product is made by exactly one company?
first in SDK arrow notation, then in Crow's foot notation

?



The R in ER: Relationships

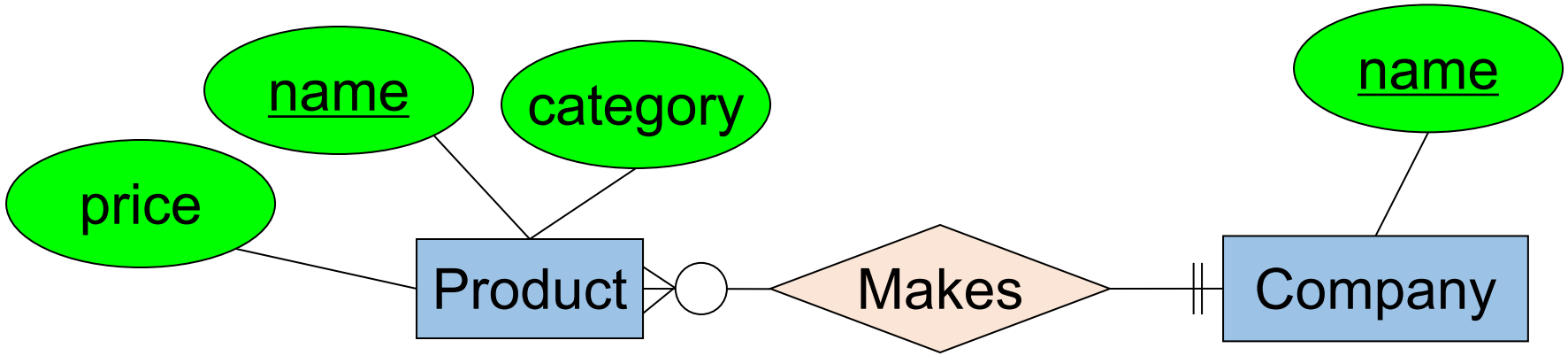
- A relationship is between two or more entities



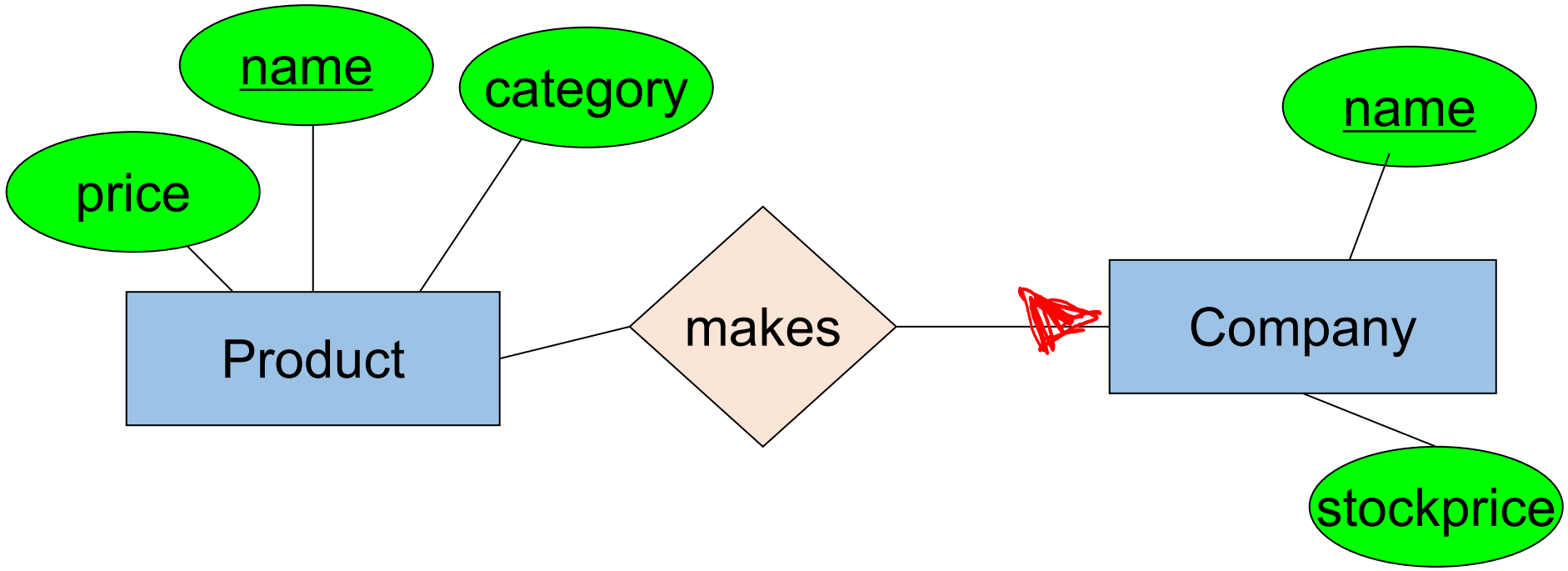
SDK arrow notation

The R in ER: Relationships

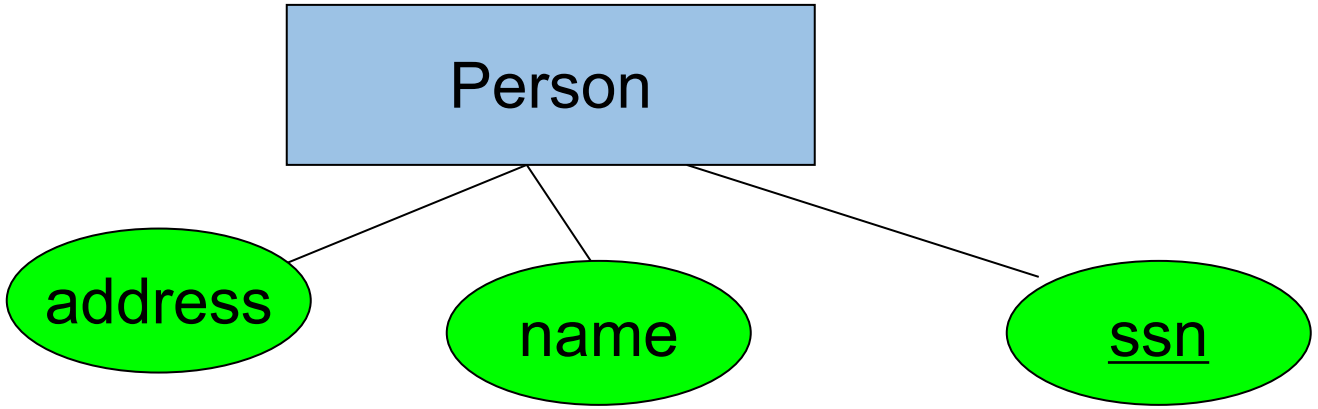
- A relationship is between two or more entities

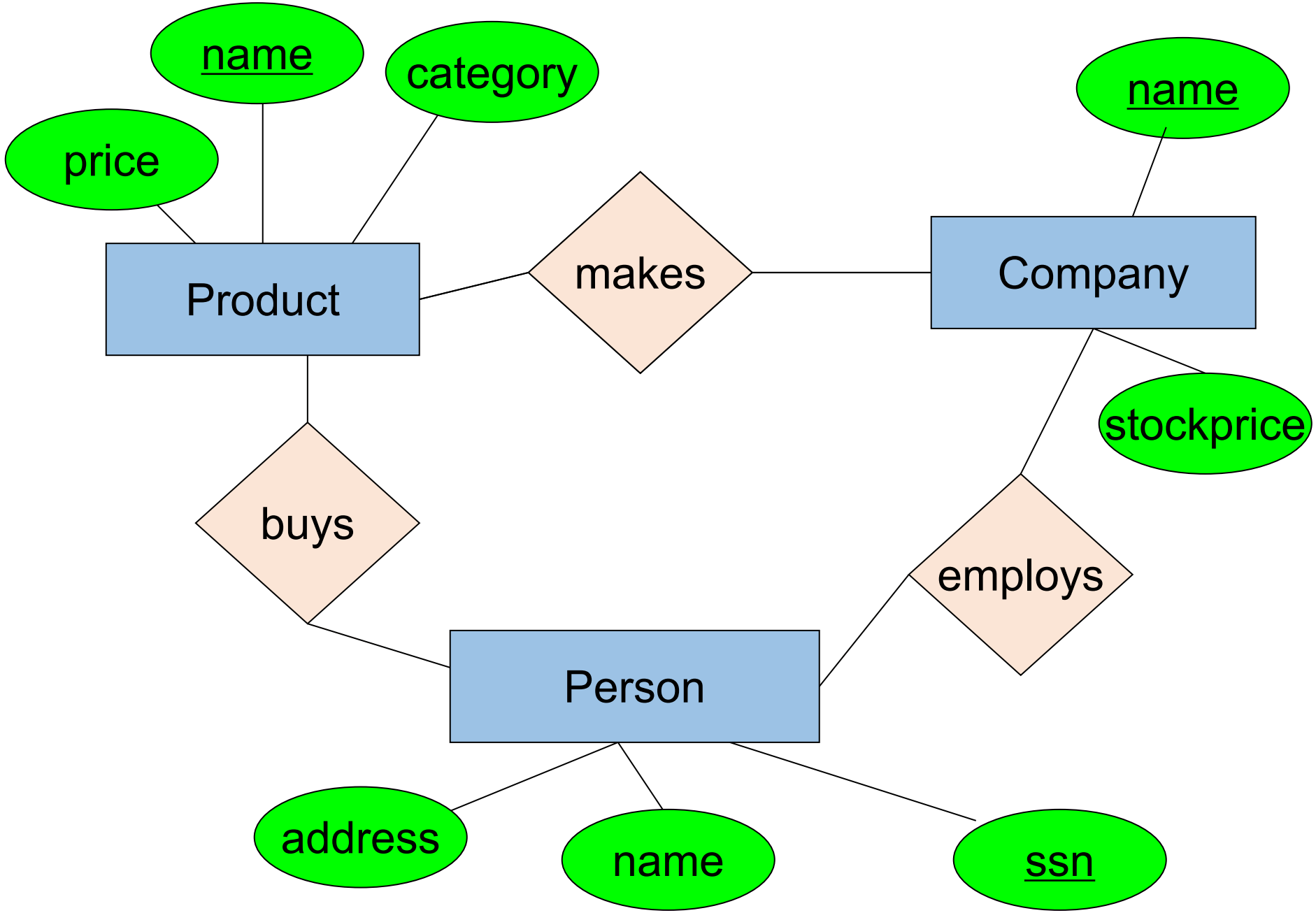


Crow's feet


name
category
price
Product
makes
name
Company
stockprice

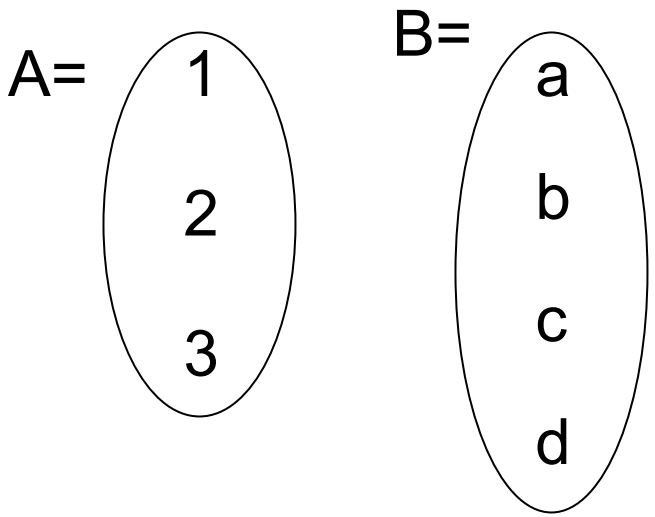
?


Person
address
name
ssn


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

What is a Relationship?

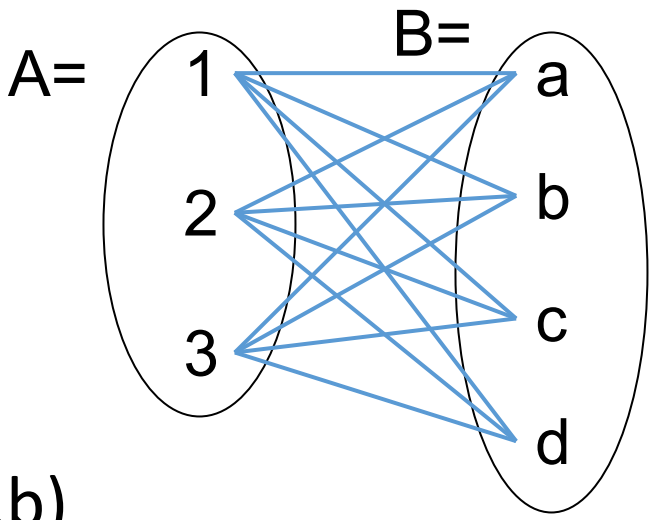
- A mathematical definition:
  - Let A, B be sets
    - A={1,2,3}, B={a,b,c,d}



what is the cross-product ?

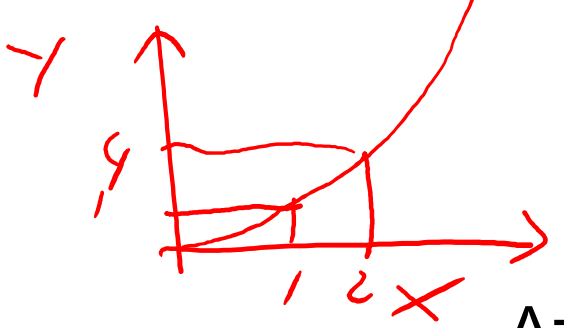
What is a Relationship?

- 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 ?

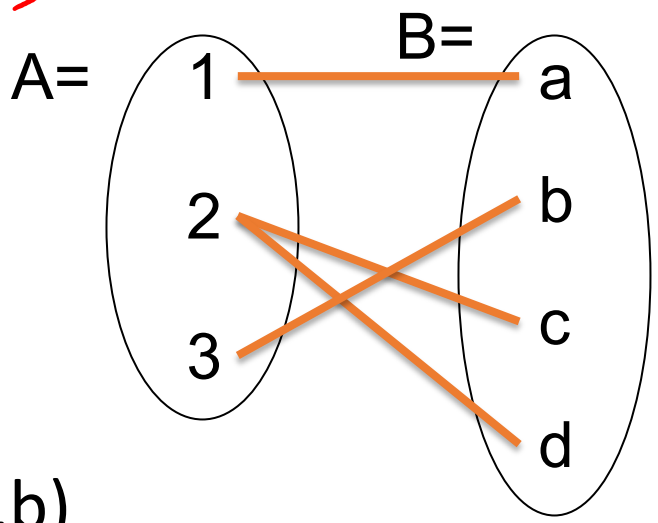
What is a Relationship?



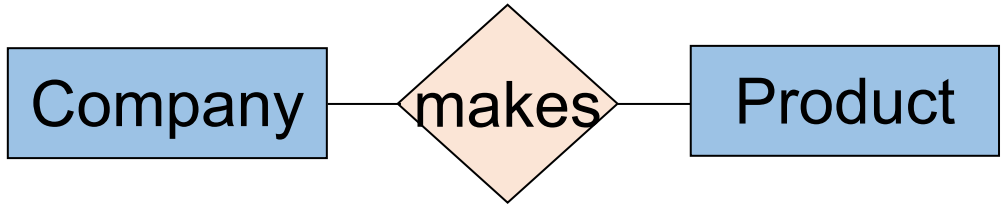
$f(x) = x^2 = y$

function vs "relation"

- 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)
    - $A \times B = \{(1,a), (1,b), (1,c), (1,d), (2,a), (2,b), (2,c), (2,d), (3,a), (3,b), (3,c), (3,d)\}$
  - We define a **relationship** to be a subset of A x B
    - R = {(1,a), (2,c), (2,d), (3,b)}



"Makes" is a relationship: it is a subset of the Product × company

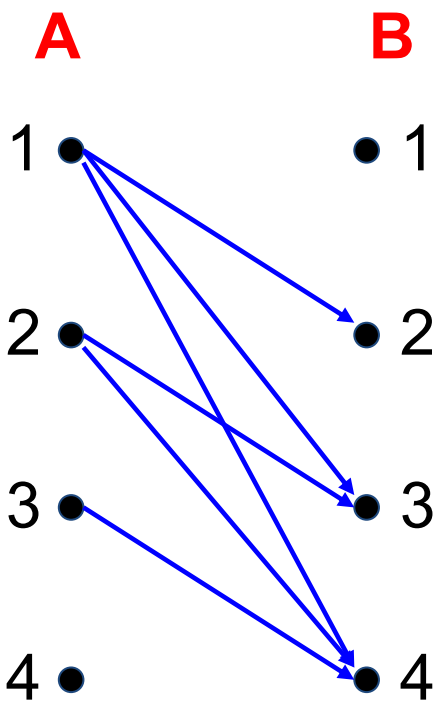


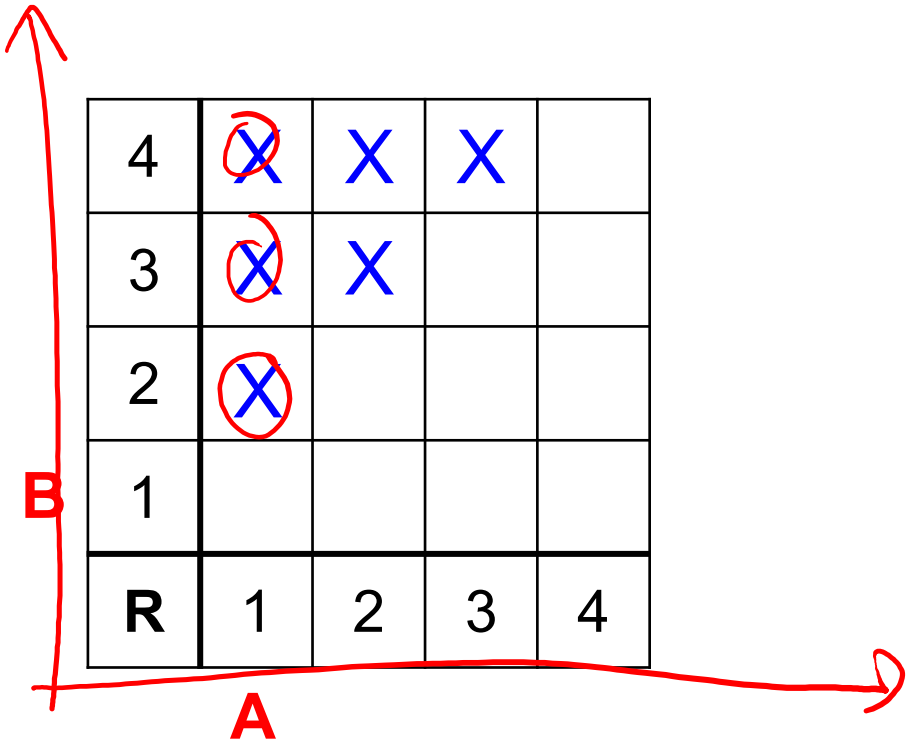
Relations

Definition: Let A and B be sets. A binary relation from A to B is a subset of A×B.

Example: R = {(1, 2), (1, 3), (1, 4), (2, 3), (2,4), (3,4)}

(R = {(a, b) | a < b} with a, b from A=B={1, 2, 3, 4})





Cp. to Def. of **Function**: For nonempty sets *A* and *B*, a function *f* from *A* to *B*, denoted *f:A→B*, is a relation from *A* to *B* in which every element of *A* appears exactly once as the first component of an ordered pair in the relation.

Another illustration of a Relationship:

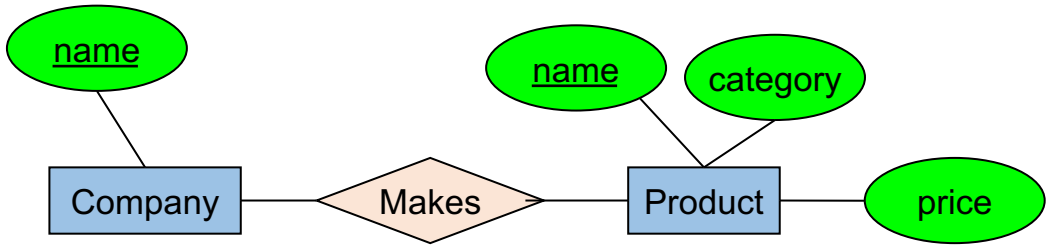
Company

name
GizmoWorks
GadgetCorp

Product

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

Cross-product ?



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

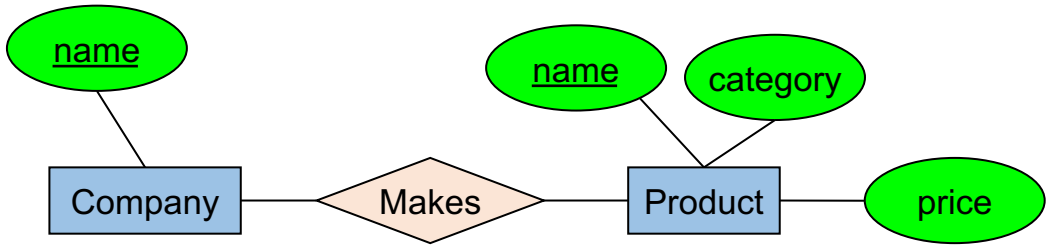
name
GizmoWorks
GadgetCorp

Product

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

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



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***

Relationship ?

What is a Relationship?

Company

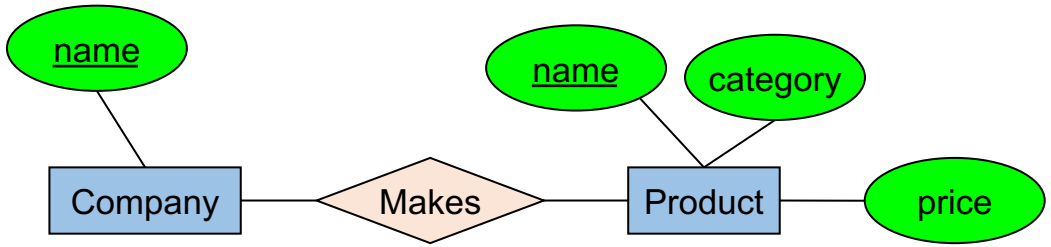
name
GizmoWorks
GadgetCorp

Product

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

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

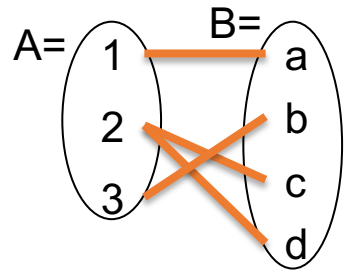


Makes

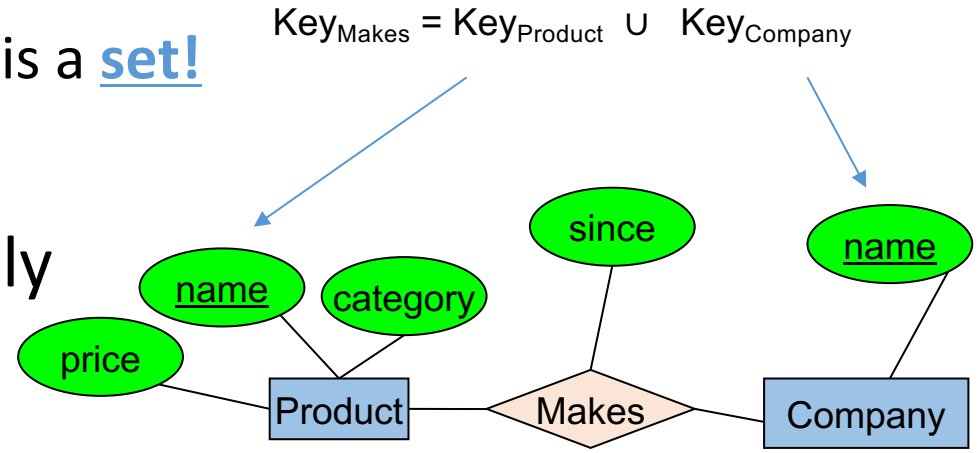
C.name	P.name
GizmoWorks	Gizmo
GizmoWorks	GizmoLite
GizmoWorks	Gadget

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?



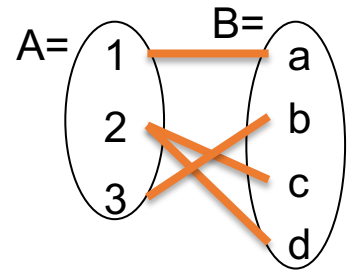
- In general, there can only be **one relationship (instance)** for every unique combination of entities
  - this follows from the definition of a relationship: it is a **set!**



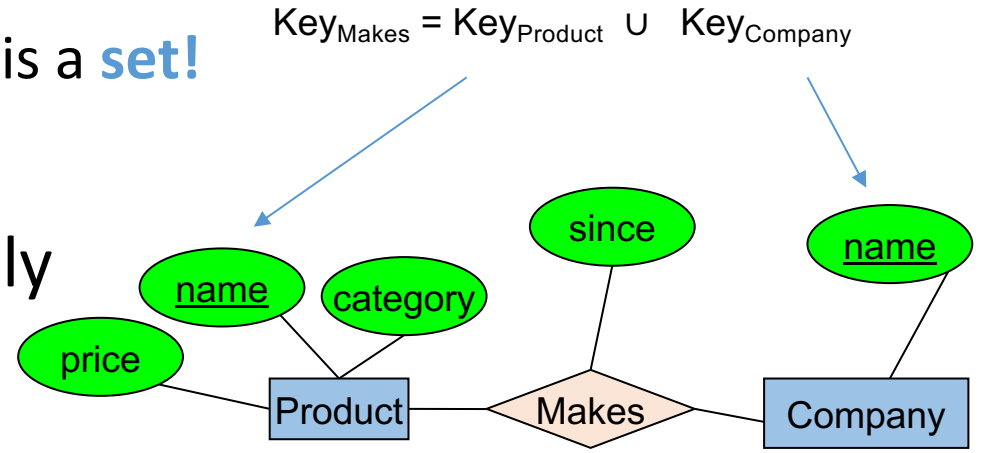
- This also means that the relationship is uniquely determined by the keys of its entities
  - Example: the "key" for Makes (to right) is {Product.name, Company.name}

- Exception: ?

What is a Relationship?



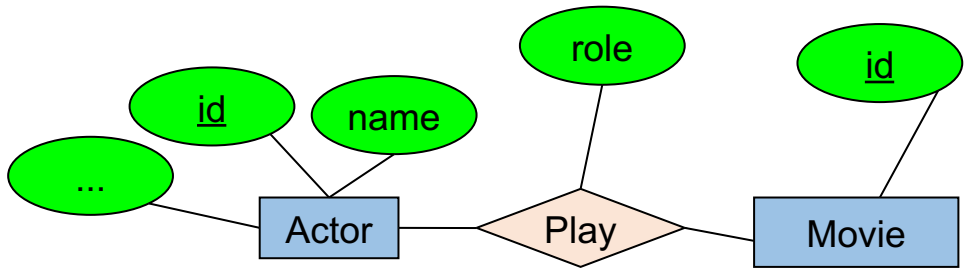
- In general, there can only be **one relationship (instance)** for every unique combination of entities
  - this follows from the definition of a relationship: it is a **set!**



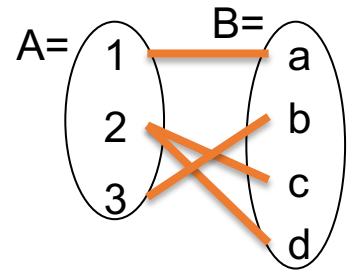
- This also means that the relationship is uniquely determined by the keys of its entities
  - Example: the "key" for Makes (to right) is {Product.name, Company.name}

- Exception: certain attributed relationships

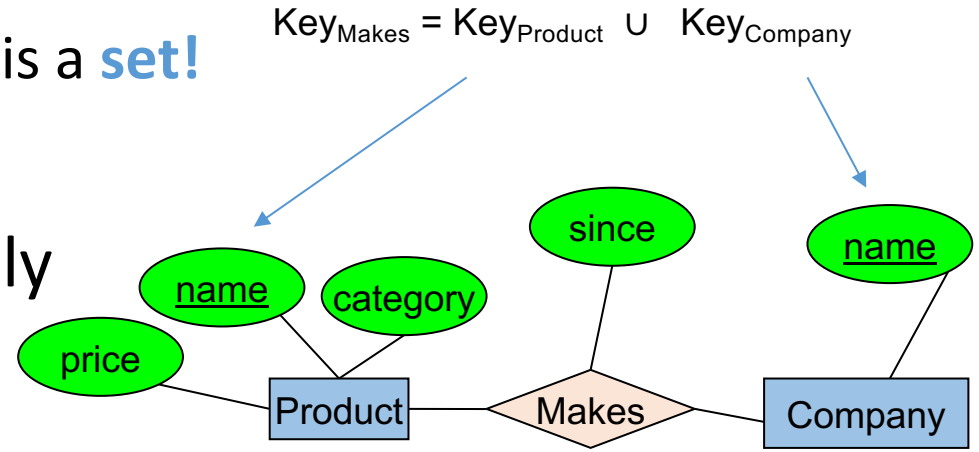
?



What is a Relationship?

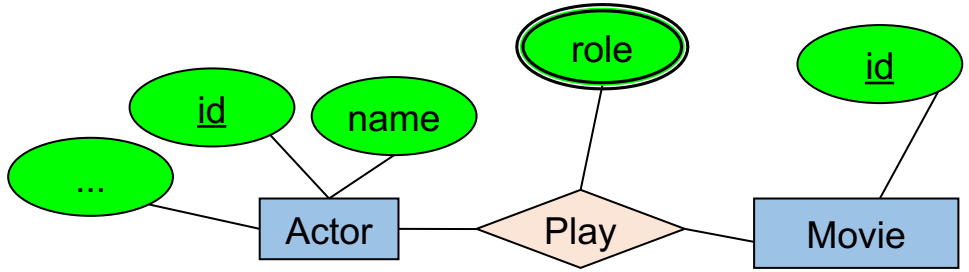


- In general, there can only be **one relationship (instance)** for every unique combination of entities
  - this follows from the definition of a relationship: it is a **set!**



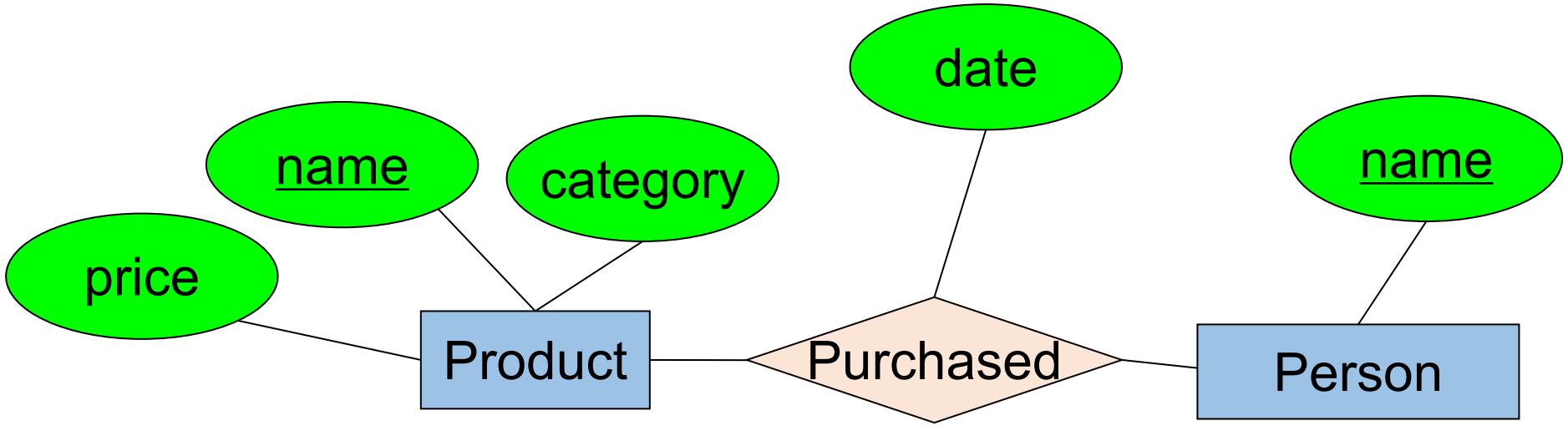
- This also means that the relationship is uniquely determined by the keys of its entities
  - Example: the "key" for Makes (to right) is {Product.name, Company.name}

- Exception: **multi-valued attributes** on relationships



Decision: Relationship vs. Entity?

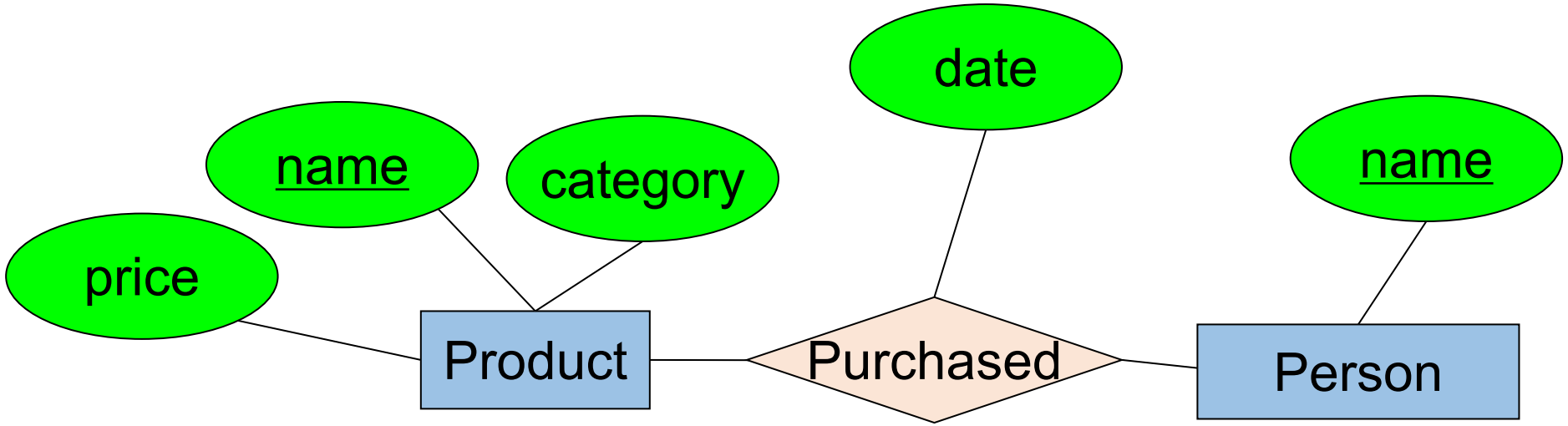
- Q: What does this say?



?

Decision: Relationship vs. Entity?

- Q: What does this say?



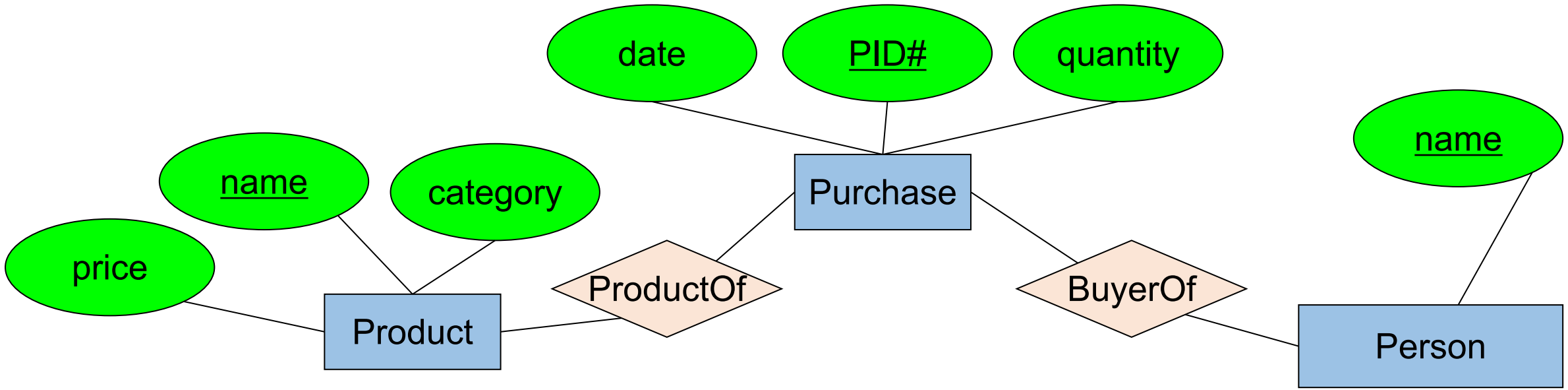
- A: A person can only buy a specific product once (on one date)

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

Decision: Relationship vs. Entity?

Can we improve this further ?

- What about this way?



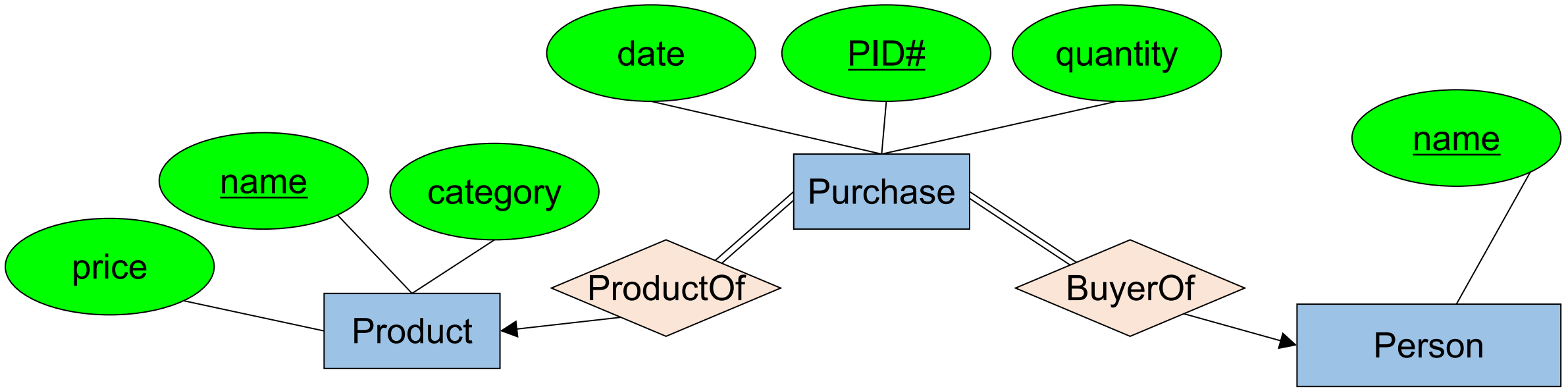
- Now we can have multiple purchases per product, person pair!

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

Decision: Relationship vs. Entity?

Can we improve this further ?

- What about this way?

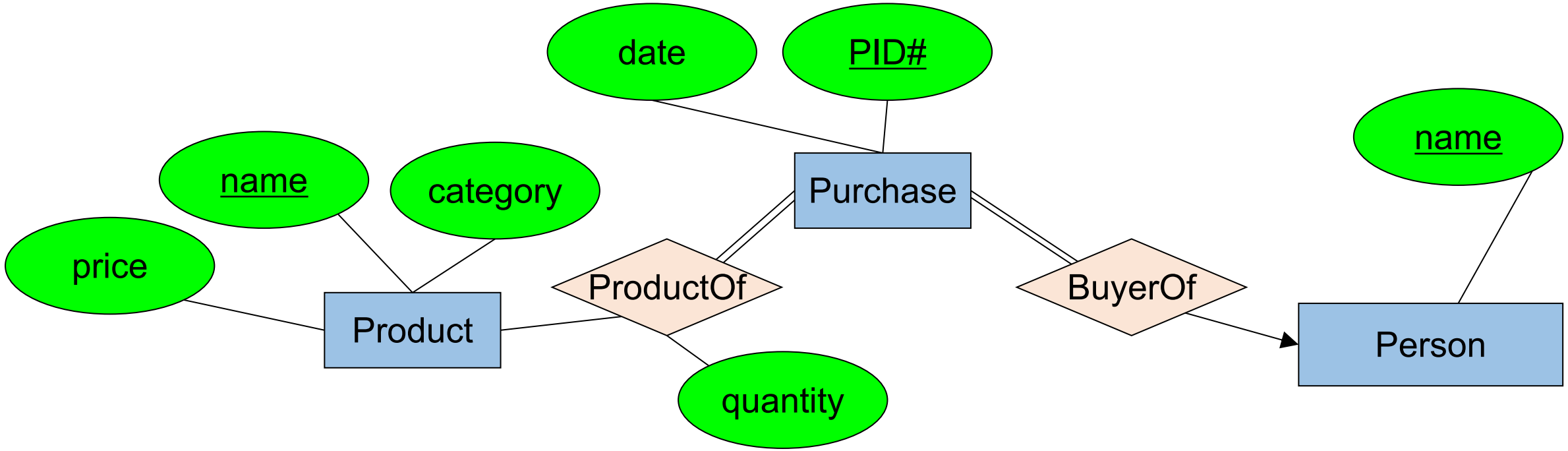


- Now we can have multiple purchases per product, person pair!

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

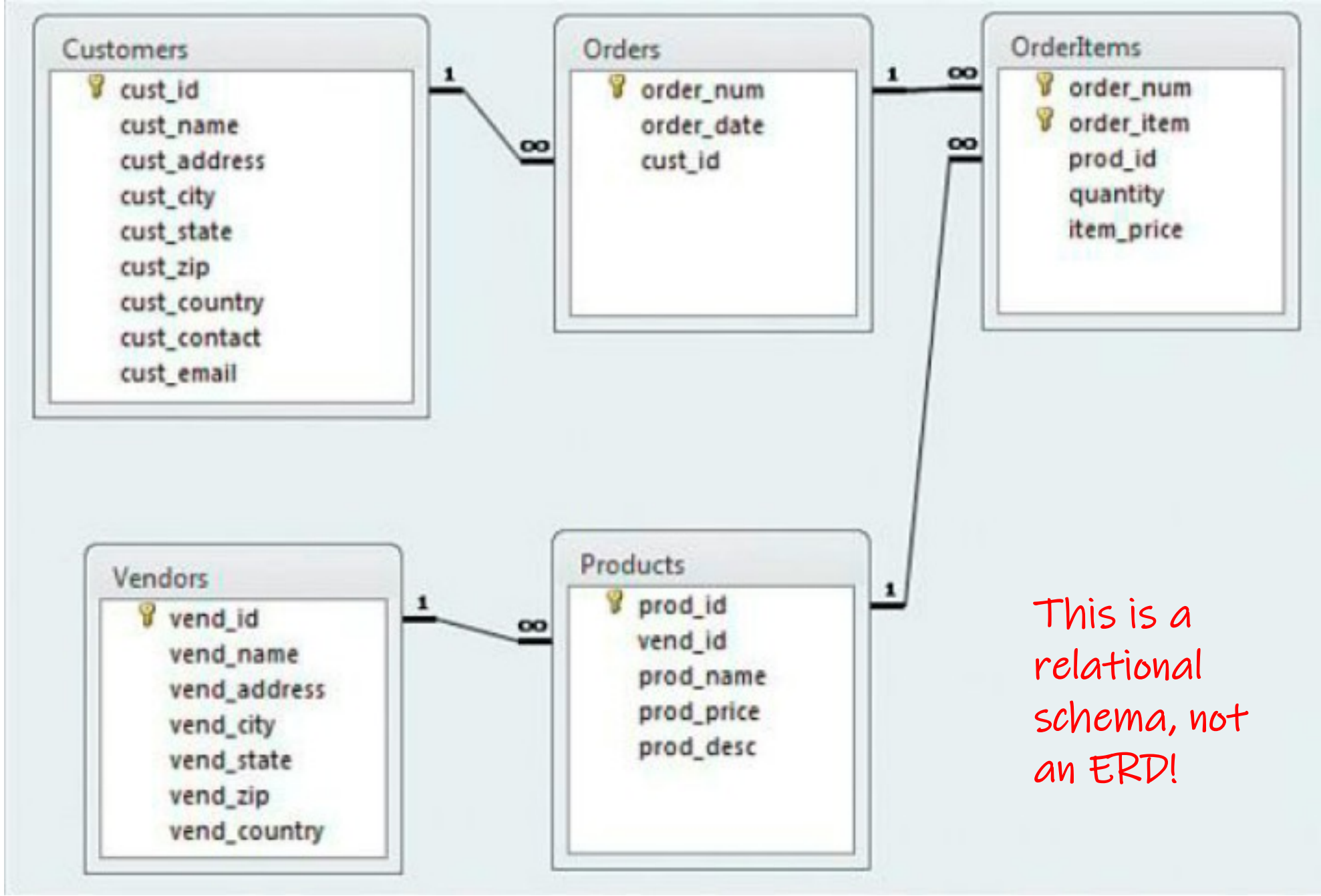
Decision: Relationship vs. Entity?

- What about this way?



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

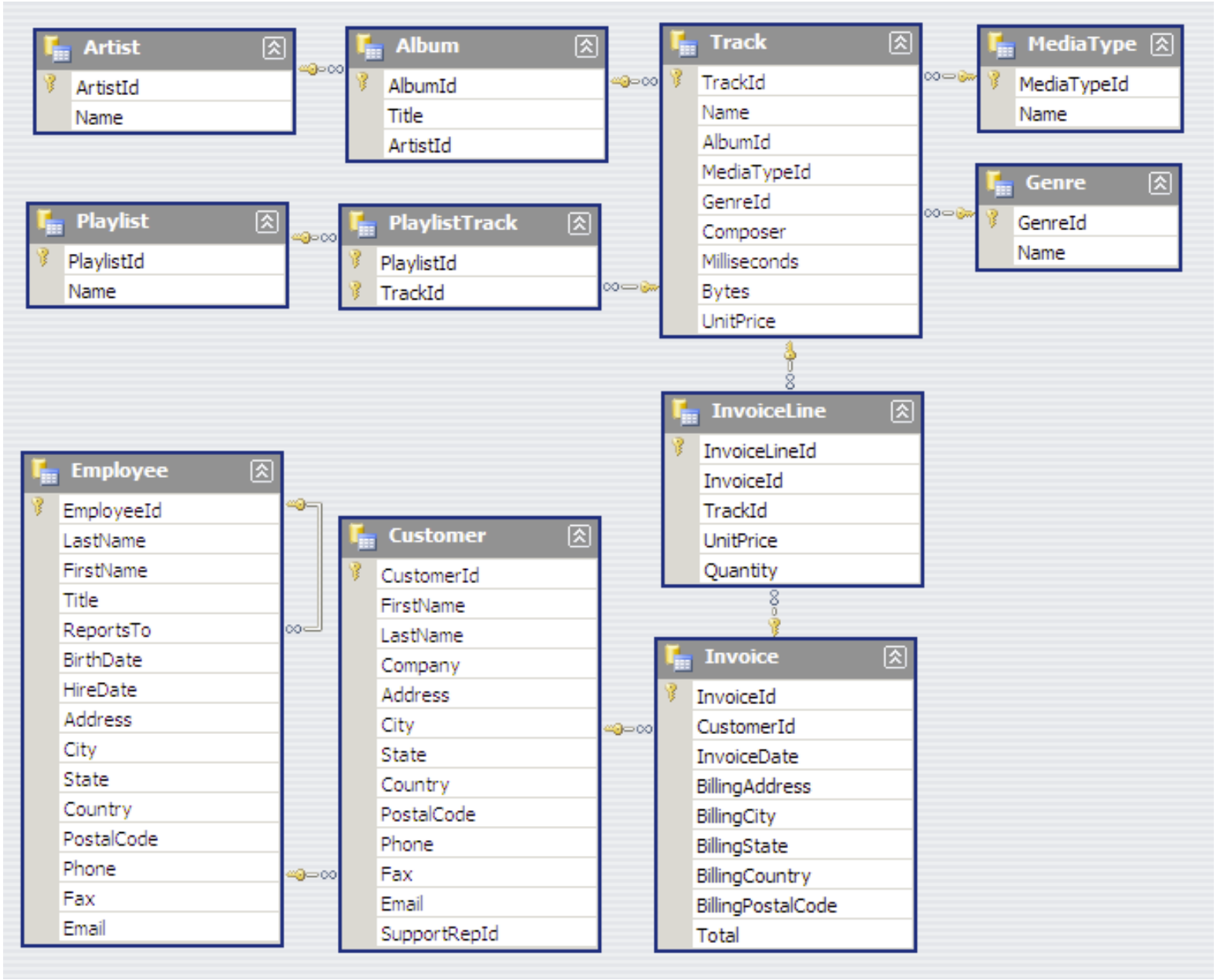
We can always use **a new entity** instead of a relationship. For example, to permit multiple instances of each entity combination!



This is a relational schema, not an ERD!

Source: SAMS Teach yourself SQL in 10min. Forta. 5th ed. 2019. Figure A.1. Sample tables relationship diagram
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

319 Chinook database

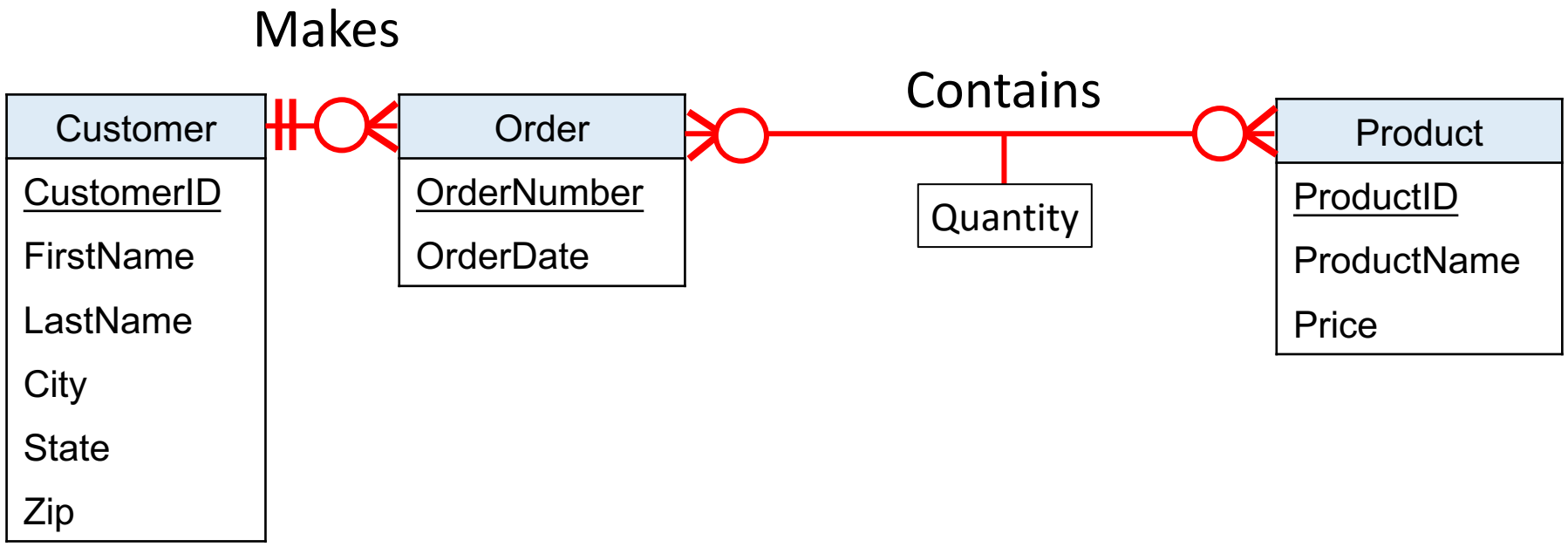


Notice that join conditions are not shown correctly here. You may have to look at the actual schema definition in SQL ☺

This is a relational schema, not an ERD!

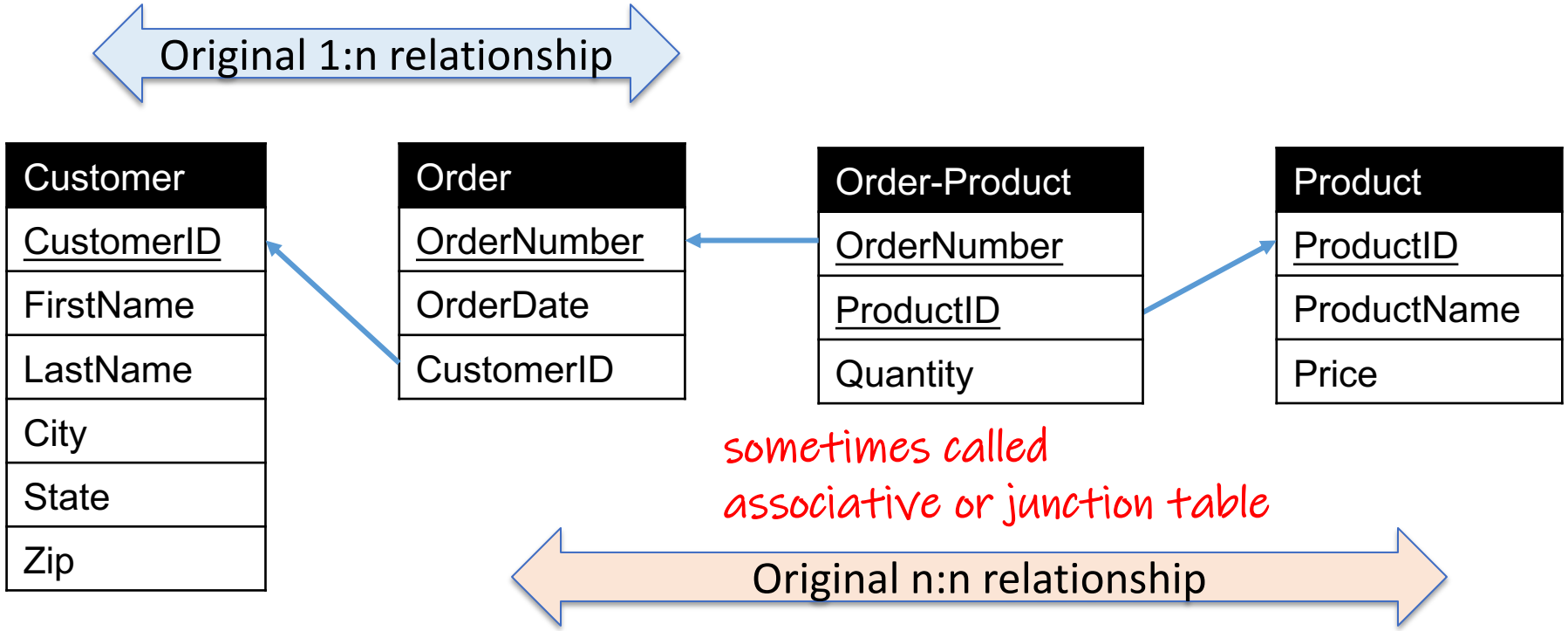
Source: https://chinookdatabase.codeplex.com/
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

From ERD to tables (Crow-foot notation)



What do we do?

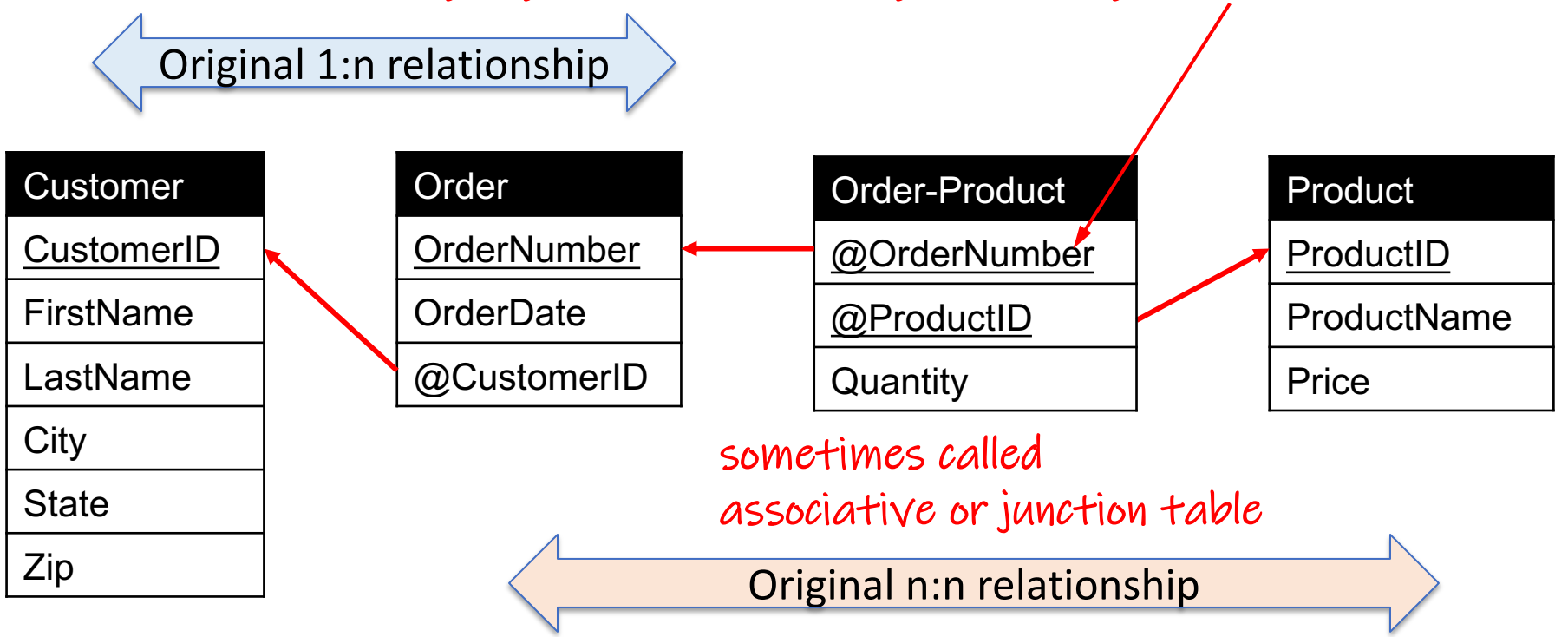
Relational schema



- Order-Product is a decomposed many-to-many relationship
  - Order-Product has a 1:n relationship with Order and Product
  - Now an order can have multiple products, and a product can be associated with multiple orders

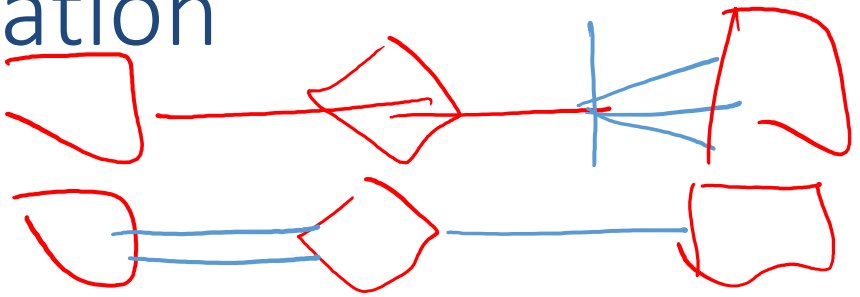
Relational schema

Attributes that are part of some Foreign key are sometimes highlighted by an at sign (@), e.g. **@OrderNumber**

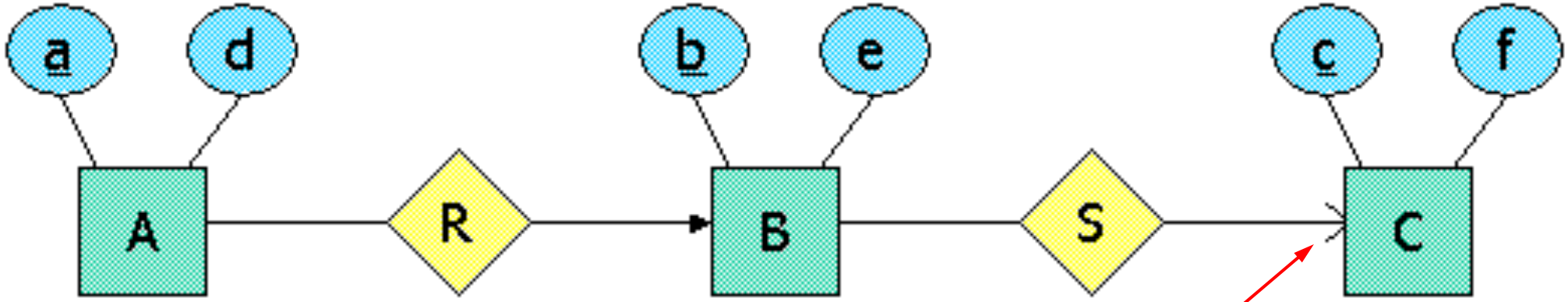


- Order-Product is a decomposed many-to-many relationship
  - Order-Product has a 1:n relationship with Order and Product
  - Now an order can have multiple products, and a product can be associated with multiple orders

Question ID 501: Stanford notation

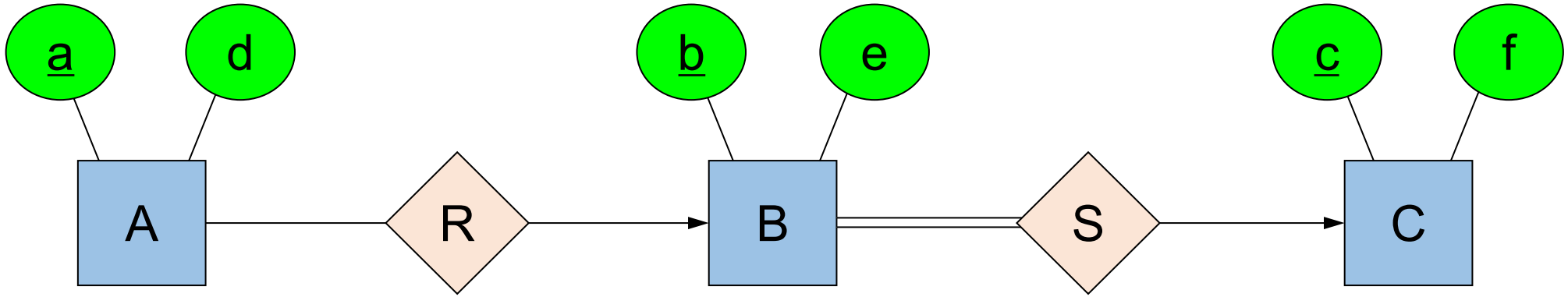


not possible w/ Stanford notation

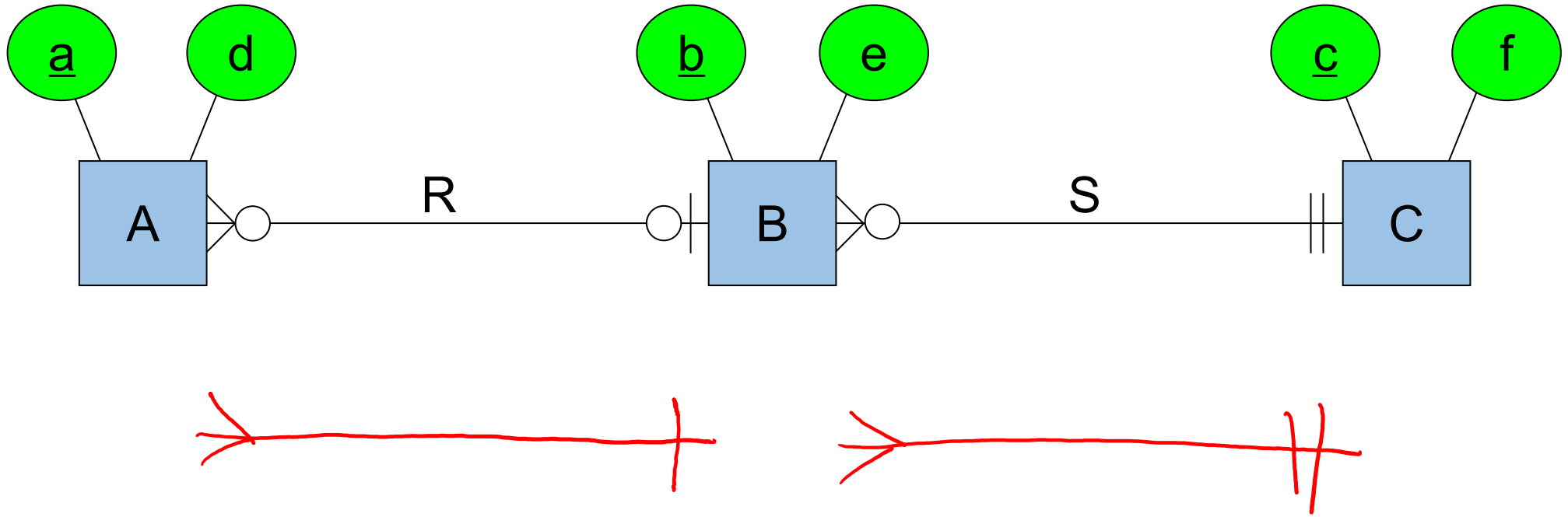


Referential integrity: mandtory participation, one

Question ID 501: SDK Silberschatz+ notation



Question ID 501: Crow foot notation



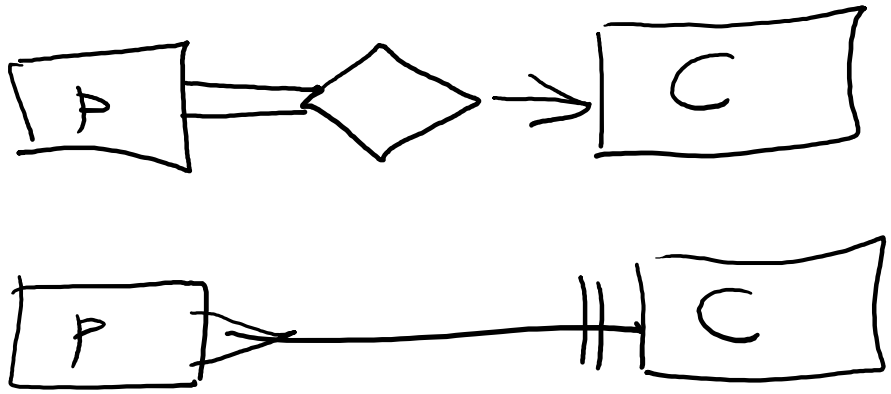
Back to ERDs: Relationships (a variety of notations)

Constraints in ER Diagrams

- Finding constraints is part of the modeling process.
- Commonly used constraints:
  - Keys: Implicit constraints on uniqueness of entities
    - Ex: An SSN uniquely identifies a person
  - **Cardinality constraints** (single-value / uniqueness constraint, **max** value)
    - Ex: a product can have only one company producing it
  - **Participation constraints** (Referential integrity constraints): Referenced entities must exist in the database (**min** value: mandatory or optional)
    - Ex: if you work for a company, it must exist in the database
  - Other constraints:
    - Ex: peoples' ages are between 0 and 150

Overview: 3 important concepts for relationships

- **Cardinality (max):** "arity", number of entity instances that participate (~mainly max) (also uniqueness constraints)

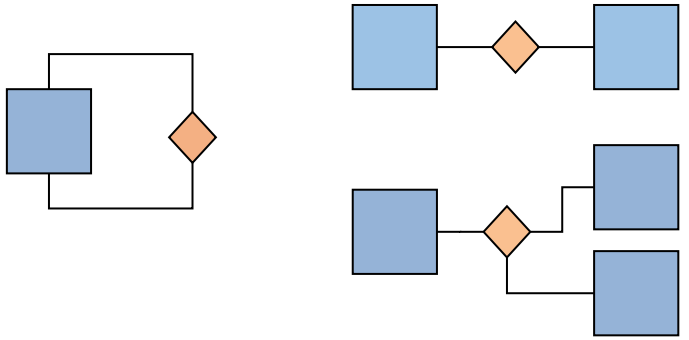


- **Participation constraints (min)**: mandatory or optional (equivalent to minimum cardinality 0 or 1), also referential integrity

What is here notation for cardinality / participation?

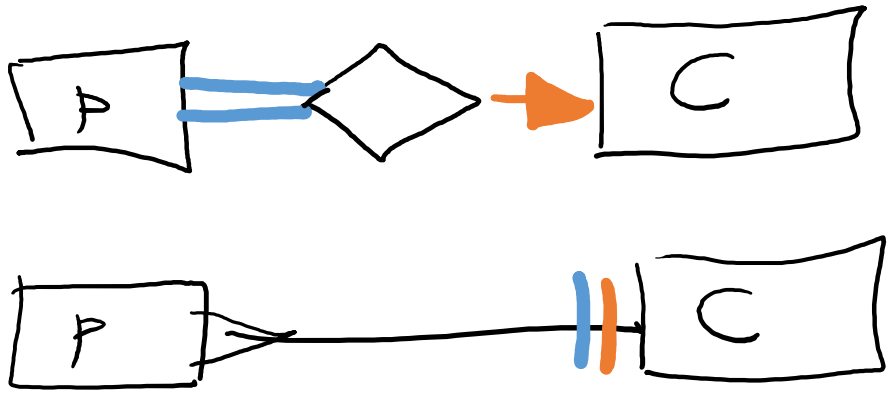
?

- **Degree**: number of entity types that participate

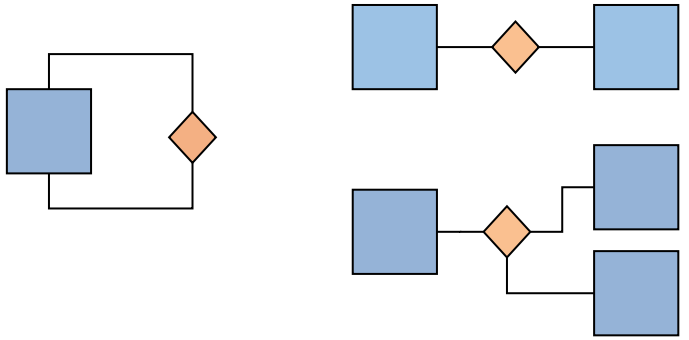


Overview: 3 important concepts for relationships

- **Cardinality (max):** "arity", number of entity instances that participate (~mainly max) (also uniqueness constraints)



- **Participation constraints (min)**: mandatory or optional (equivalent to minimum cardinality 0 or 1), also referential integrity



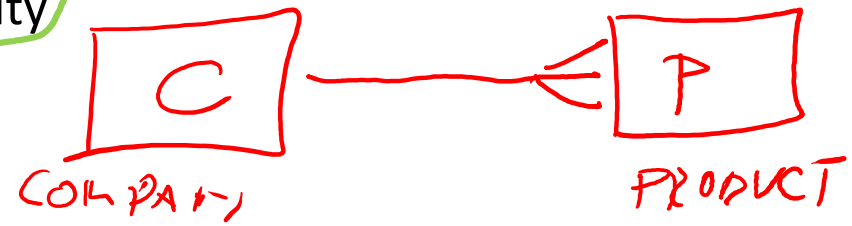
- **Degree**: number of entity types that participate

Random question: Why "relational model"?

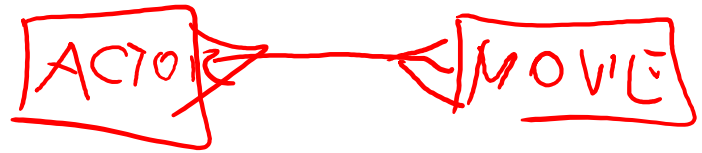
1. Cardinality (multiplicity) of relationships

- Defines the number of entity instances that participate in it. Also called type of relationship (or "connectivity" at times).

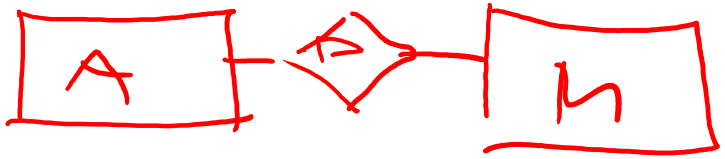
- One-to-One
  - Each entity in the relationship will have exactly one related entity



- One-to-Many
  - An entity on one side of the relationship can have many related entities, but an entity on the other side will have a maximum of one related entity



- Many-to-Many
  - Entities on both sides of the relationship can have many related entities on the other side

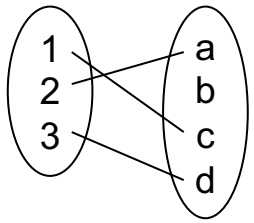


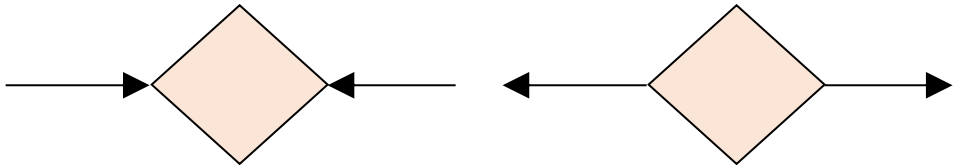
This notation is also called "crow's foot notation"

1. Cardinality (multiplicity) of E/R relationships

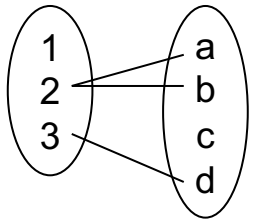
Crow's Foot **Arrow notation**

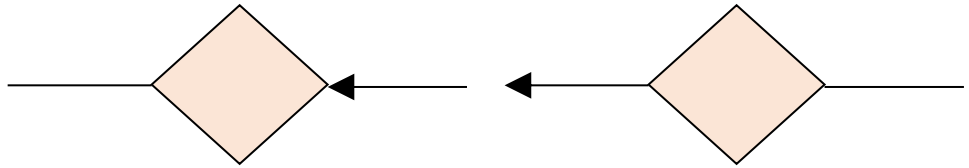
One-to-one:



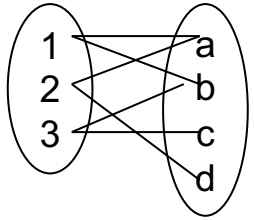


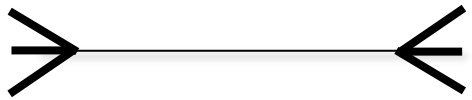
One-to-many:

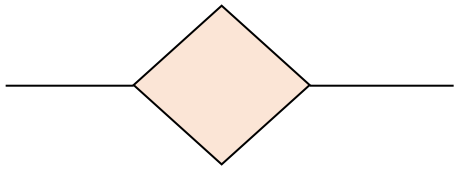




Many-to-many:







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

2. Participation constraints

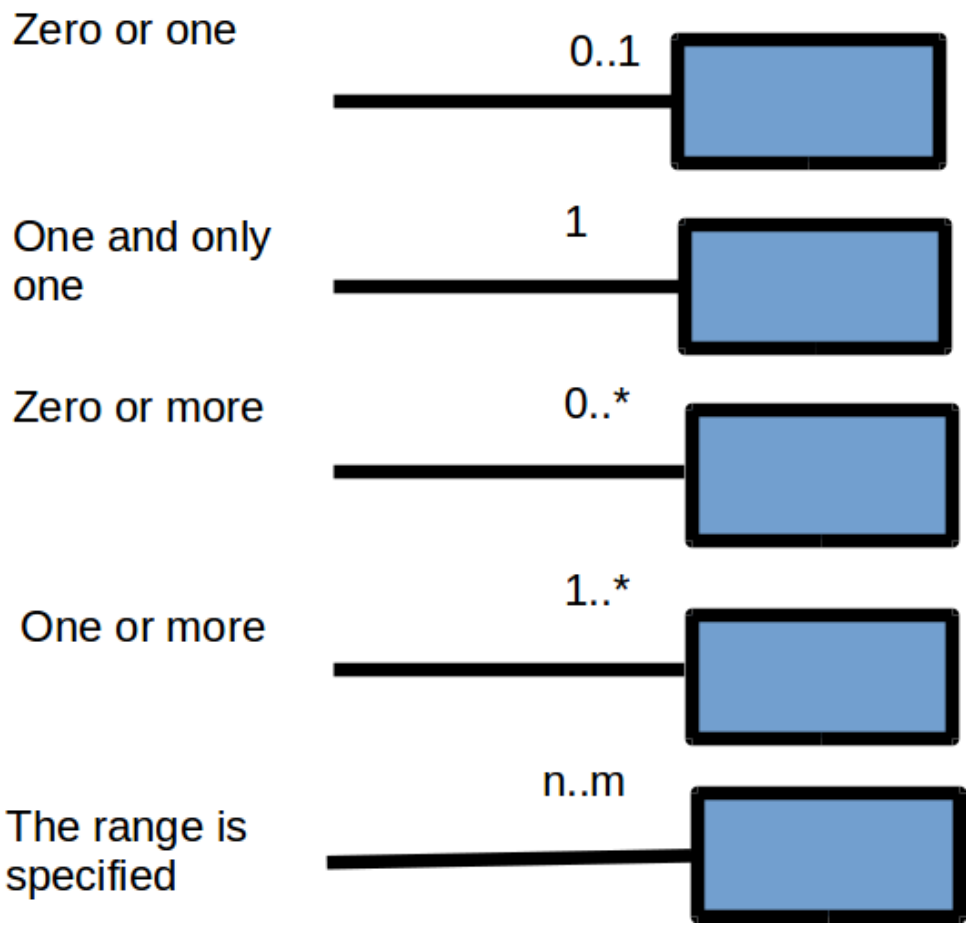
- The number of instances of one entity that can or must be associated with each instance of another entity
  - Minimum: if zero, then optional,
    if one or more, then mandatory (or "total")
  - Maximum

  - **Mandatory** one 
  - Mandatory many 
  - **Optional** one 
  - Optional many 

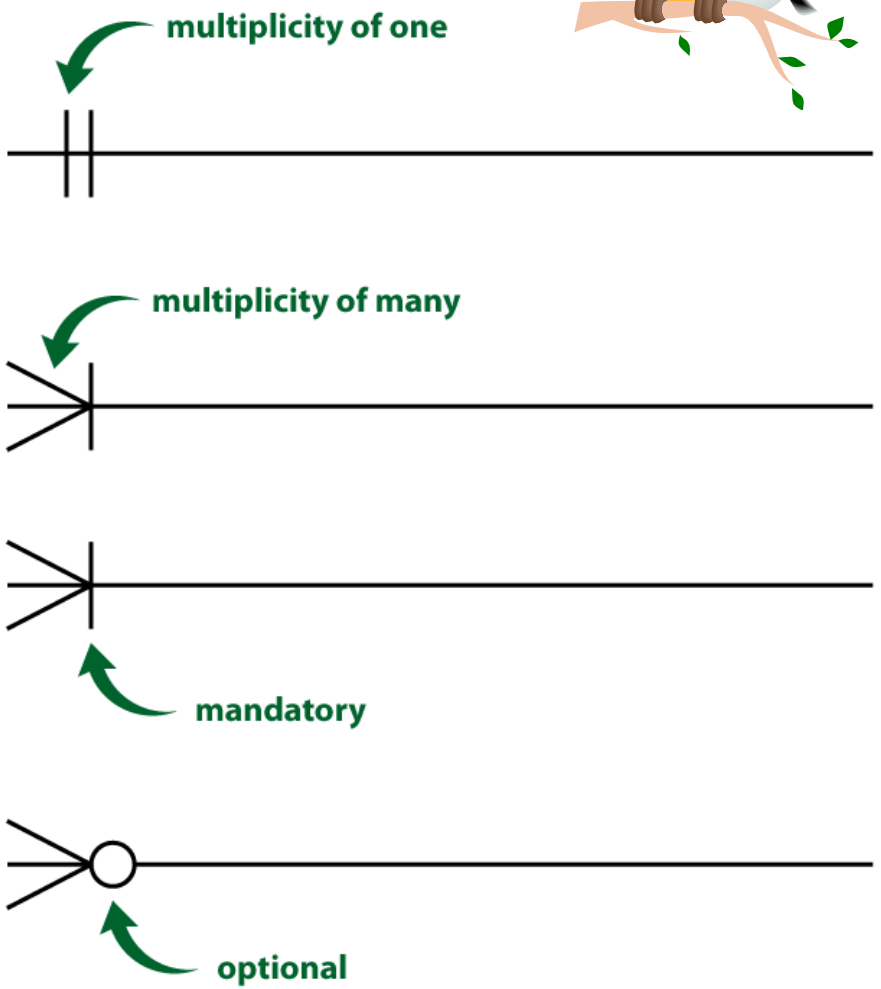
Whether a relation is mandatory or optional is also sometimes called "participation constraints"

Relationships with specified cardinalities and participation

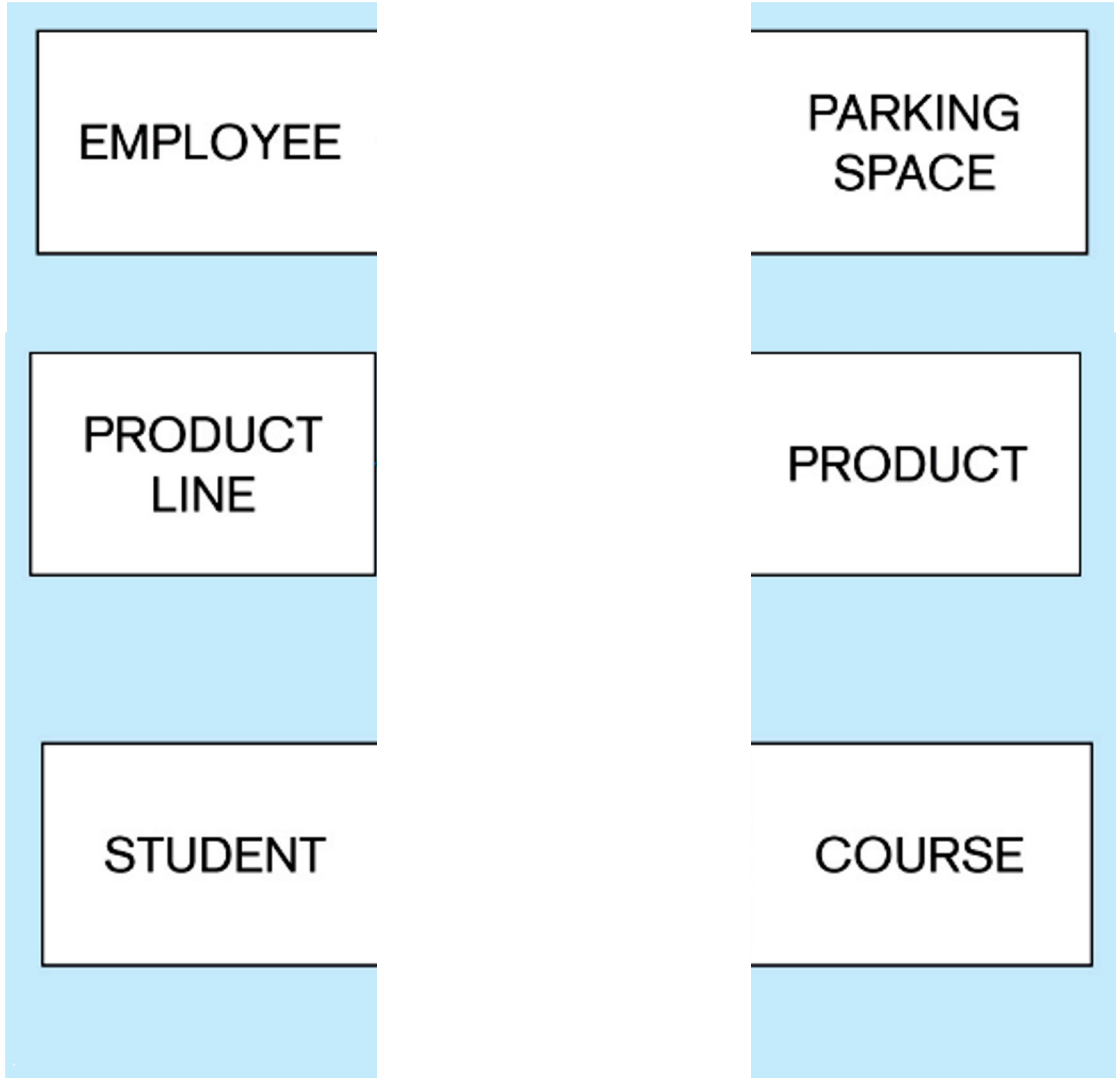
UML notation



Crow's Foot



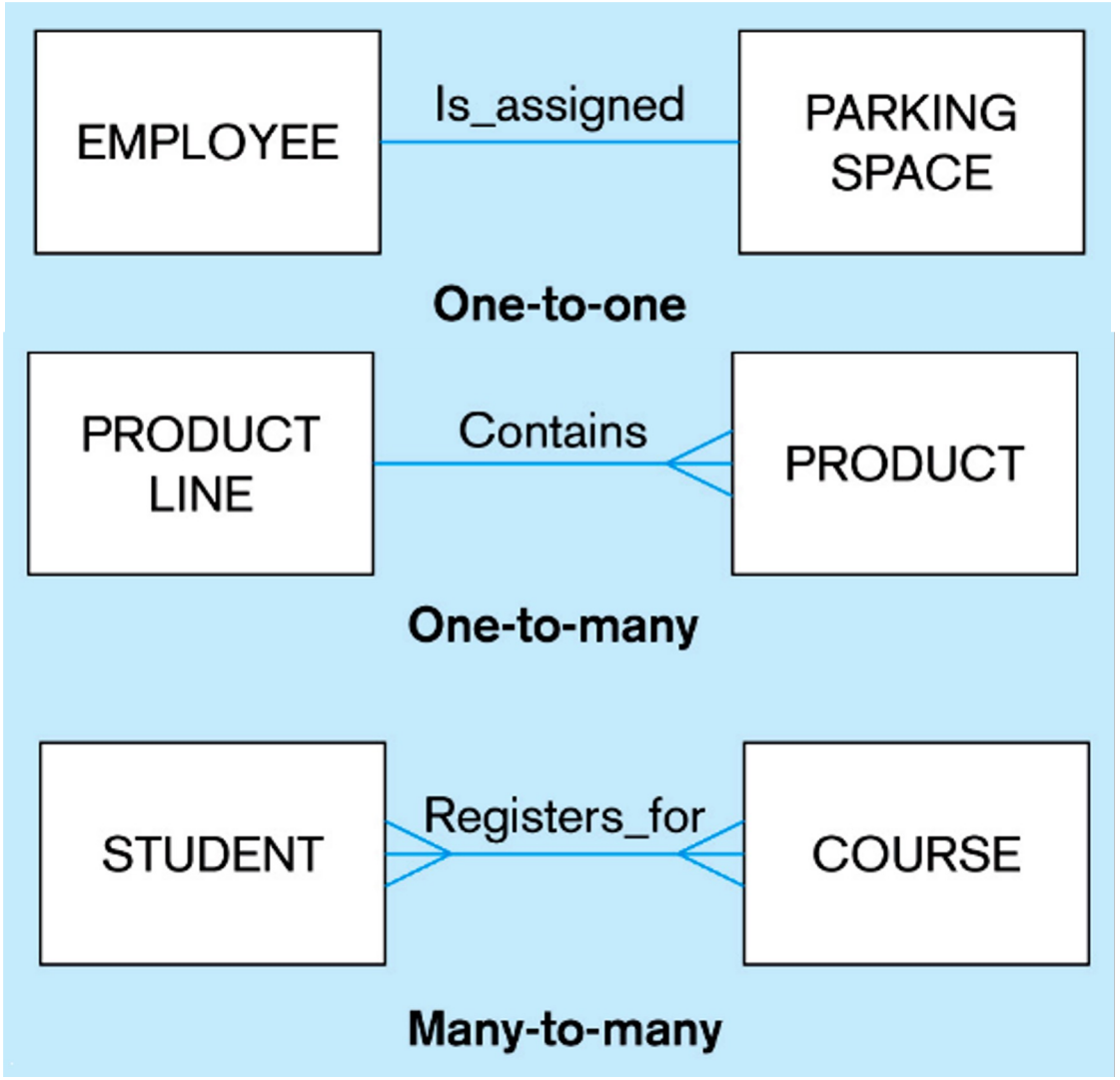
Example: Binary Relationships



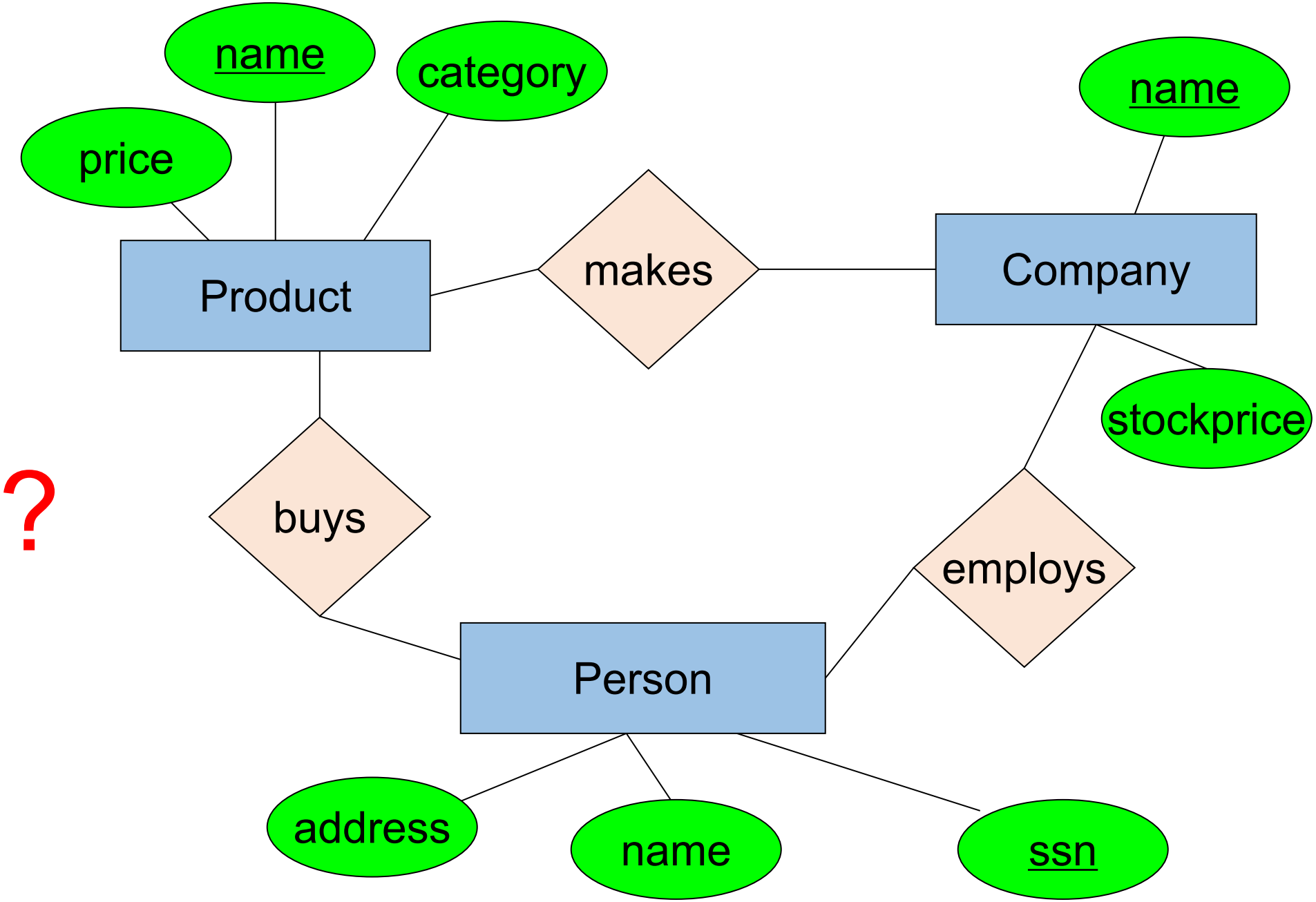
Cardinalities ?

Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

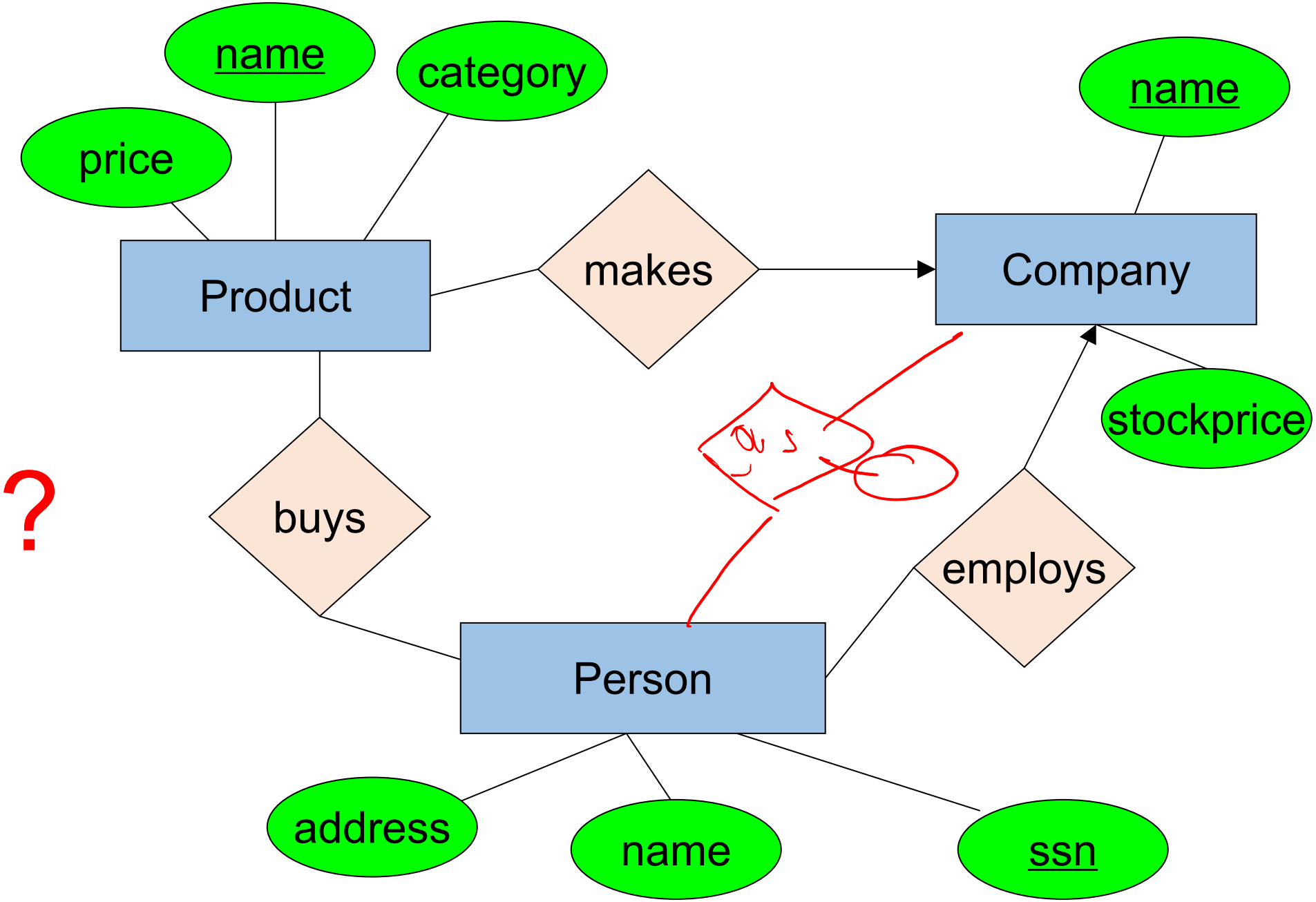
Example: Binary Relationships



Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/



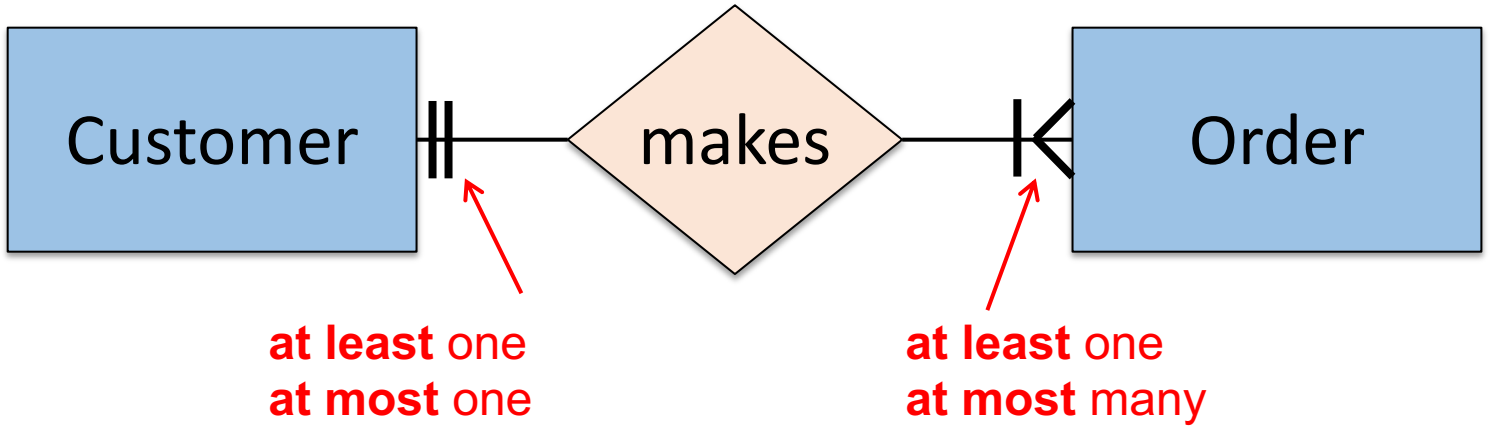
What does this say ?



What does this say ?

1./2. Cardinality + participation constraints

- Defines the rules of the association between entities

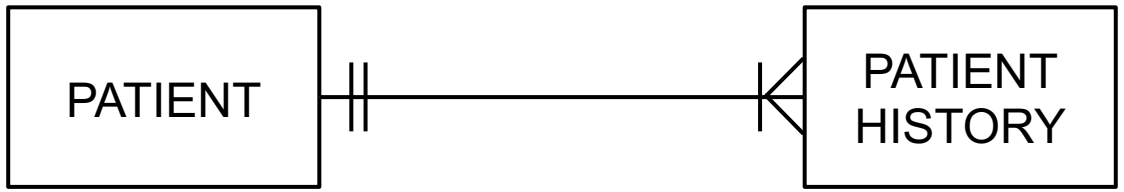


This is a one-to-many relationship: One customer can have many orders.

The symbols closest to the Order entity represents "one, or many", whereas an order has "one and only one" Customer.

Ex: cardinality + participation constraints (1/2)

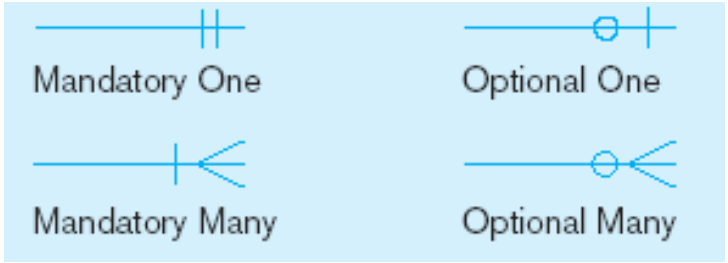
Please interpret the constraints
in the following ER diagram:



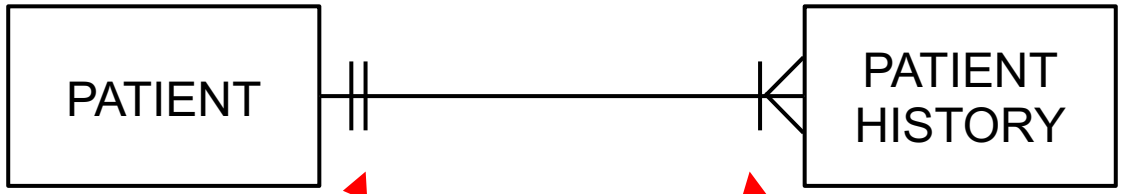
Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010. Fig. 2.17
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

Ex: cardinality + participation constraints (1/2)

Cardinalities + Participation



Please interpret the constraints in the following ER diagram:



Alice — Visit 1

Bob — Visit 2, Visit 3

Charlie — Visit 4

A patient history is recorded for one and only one patient

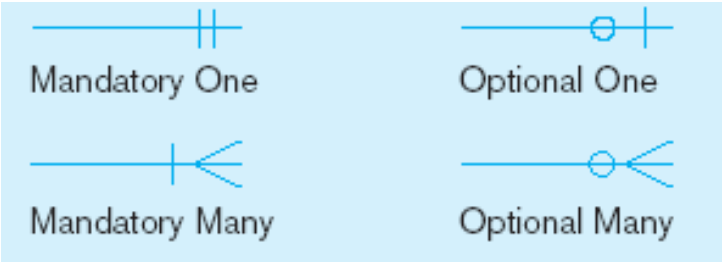
A patient must have recorded at least one history, and can have many

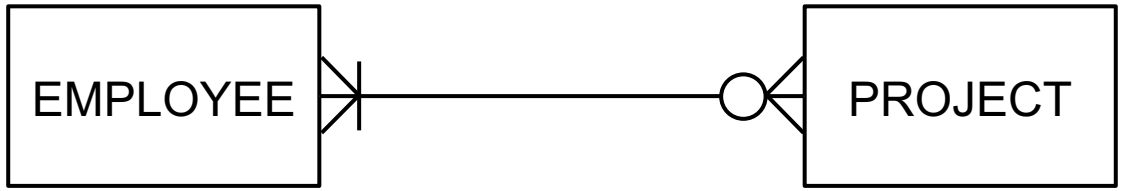
Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010. Fig. 2.17
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

Ex: cardinality + participation constraints (2/2)

Please interpret the constraints in the following ER diagram:

Cardinalities + Participation



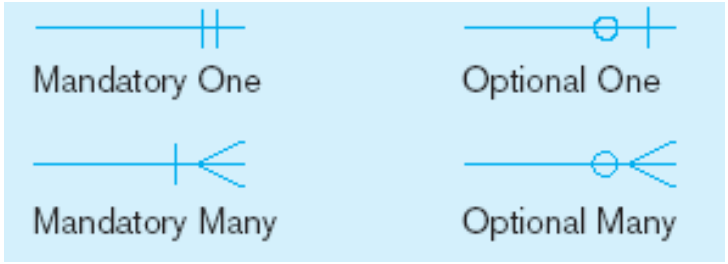


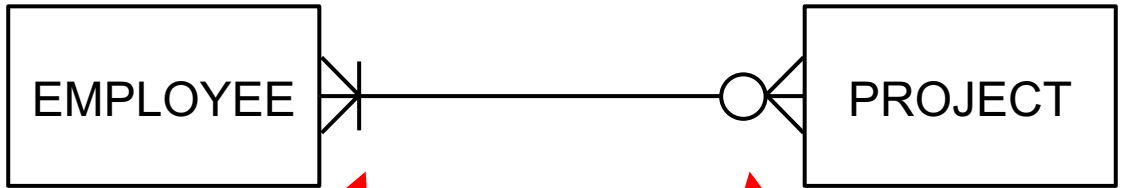
Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010. Fig. 2.17
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

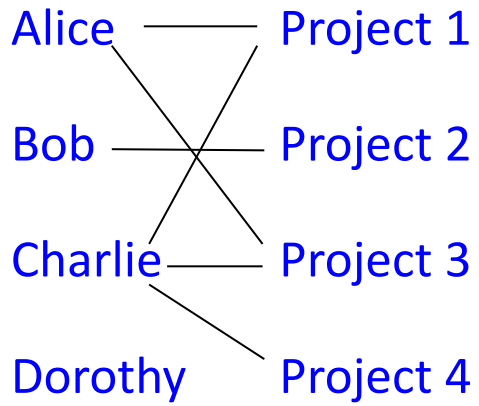
Ex: cardinality + participation constraints (2/2)

Please interpret the constraints in the following ER diagram:

Cardinalities + Participation





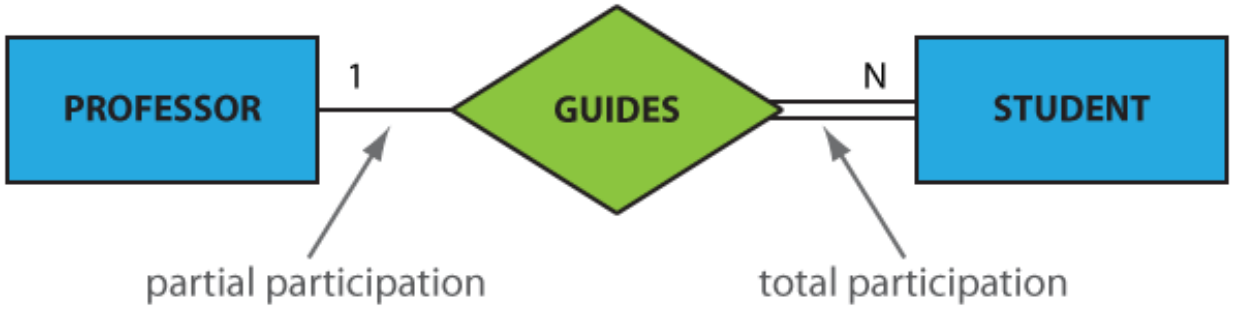


A project must be assigned to at least one employee, and may be assigned to many

An employee can be assigned to any number of projects, or may not be assigned to any at all

Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010. Fig. 2.17
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

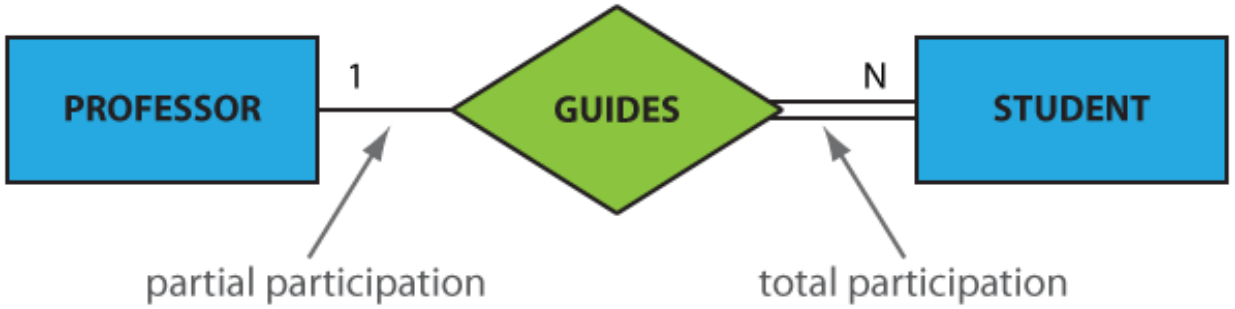
Participation constraints (min) in [Elmasri+'15] notation



What does this say ?

Source: http://www.vertabelo.com/blog/technical-articles/chen-erd-notation
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

Participation constraints (min) in [Elmasri+'15] notation



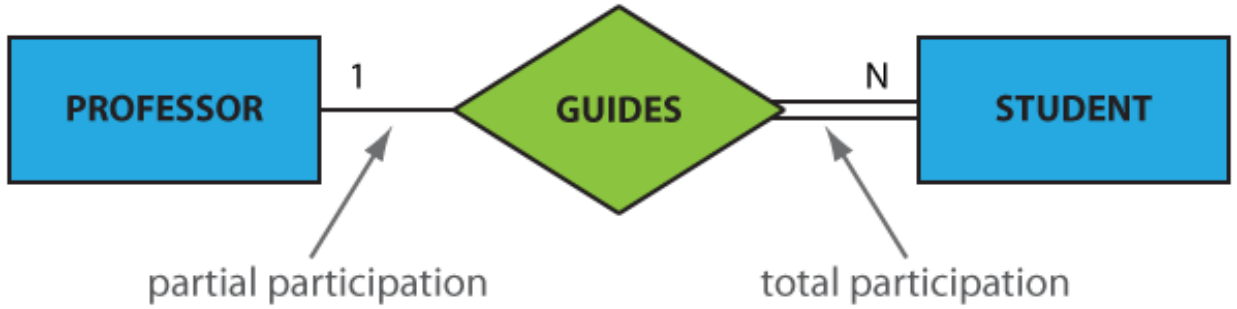
total or (mandatory participation) means that every entity in the set is involved in the relationships

Each student must be guided by one professor.
A professor can (but does not have to) guide one or more students.

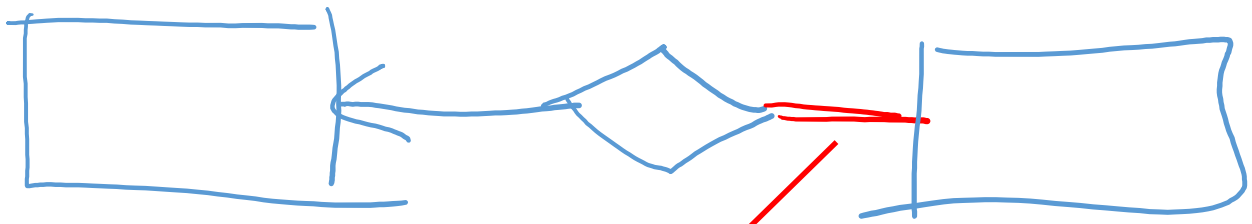
Transform those into crow's foot! ?

Source: http://www.vertabelo.com/blog/technical-articles/chen-erd-notation
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

Participation constraints (min) in [Elmasri+'15] notation

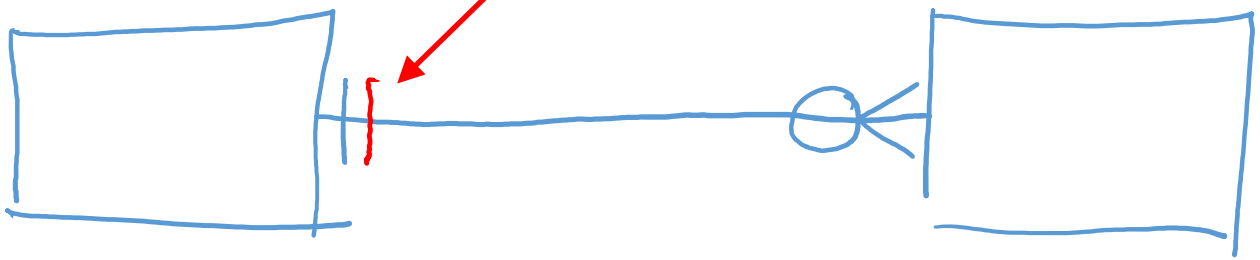


total or (mandatory participation) means that every entity in the set is involved in the relationships



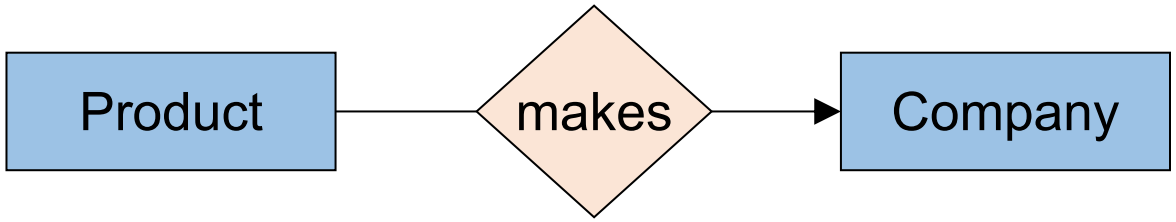
Each student must be guided by one professor.

A professor can (but does not have to) guide one or more students.



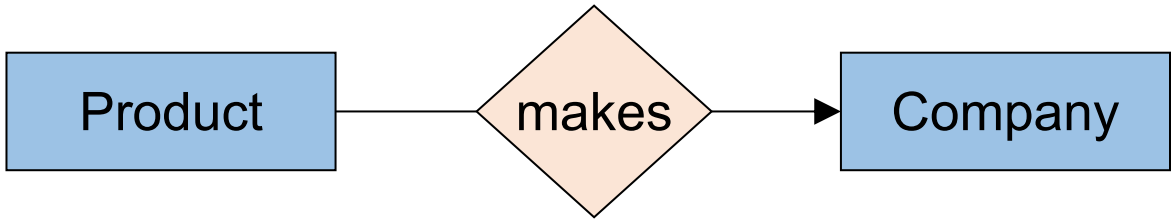
Source: http://www.vertabelo.com/blog/technical-articles/chen-erd-notation

[Stanford book'08] & Gradiance notation



?

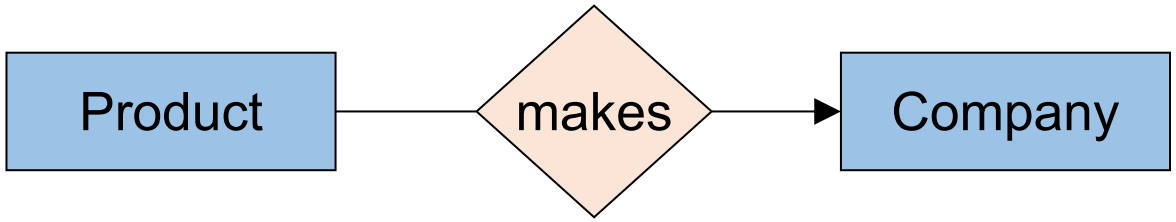
[Stanford book'08] & Gradiance notation



Transform into "crow foot notation" and SDK [Silberschatz'20] if different ?

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

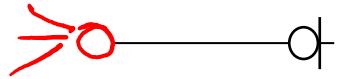
[Stanford book'08] & Gradiance notation



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

Transform into "crow foot notation" and SDK [Silberschatz'20] if different

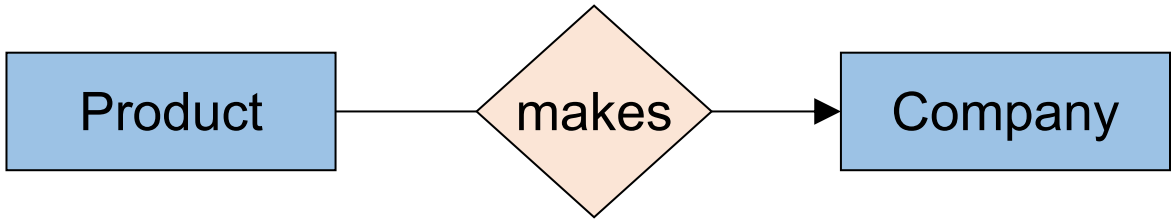
identical ☺

Crow's foot: 

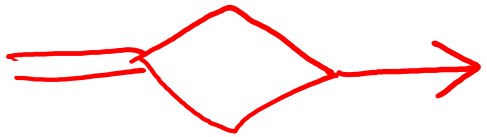
?

Each product made by exactly one company.

[Stanford book'08] & Gradiance notation

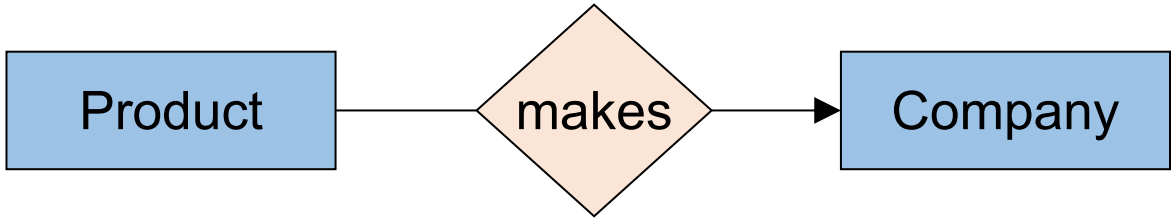


Each product made by at most one company.
Some products made by no company

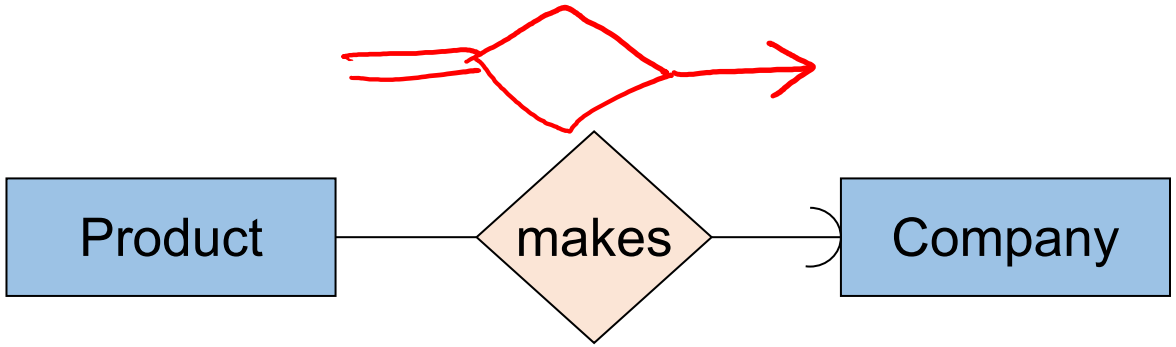


Each product made by <u>exactly one</u> company.

[Stanford book'08] & Gradiance notation



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



Each product made by <u>exactly one</u> company.

Gradiance / Stanford arrows:
<u>total participation</u> (min 1) can only be shown when also <u>uniqueness constraint</u> (max 1) holds!

Participation constraints in Stanford arrow notation

1. What does this say?
2. Write in SDK and Crow's foot notation?

?

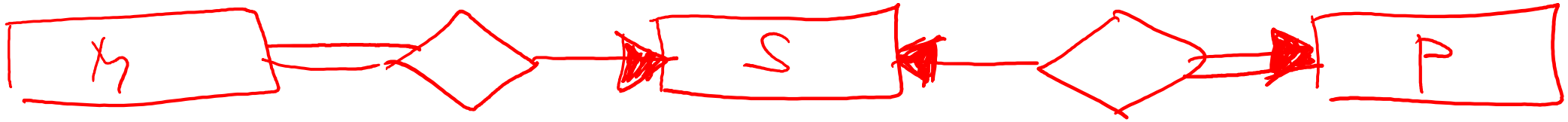
Example taken from: Database Systems -- the complete book, Garcia-Molina, Ullman, Widom, 2nd international ed, 2014. Section 4, Figure 18
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

Participation constraints in Stanford arrow notation

A movie is owned by exactly one studio.
A studio can have an unconstrained number of movies

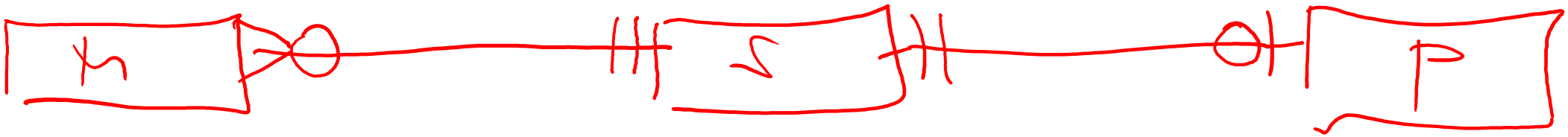
A studio can have at most one president.
Each president must run exactly one studio (that exists in the studio entity set)

Participation constraints in Stanford arrow notation



A movie is owned by exactly one studio.
A studio can have an unconstrained number of movies

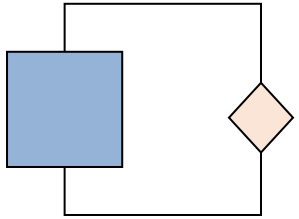
A studio can have at most one president.
Each president must run <u>exactly one</u> studio (that exists in the studio entity set)



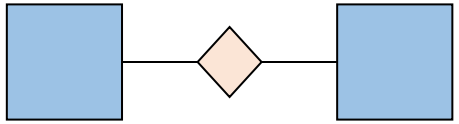
3. Degree of Relationship

- Defines the number of entity types that participate in it

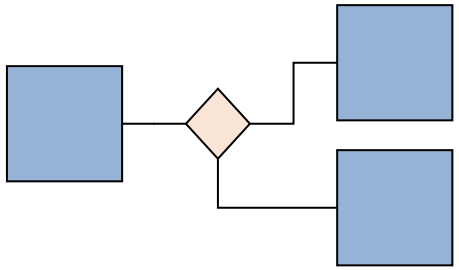
Unary Relationship
- One entity type related to itself



Binary Relationship
- Two entity types related to each other



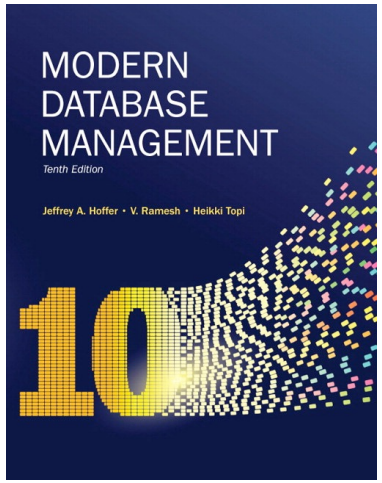
Ternary Relationship
- Three entity types related to each other



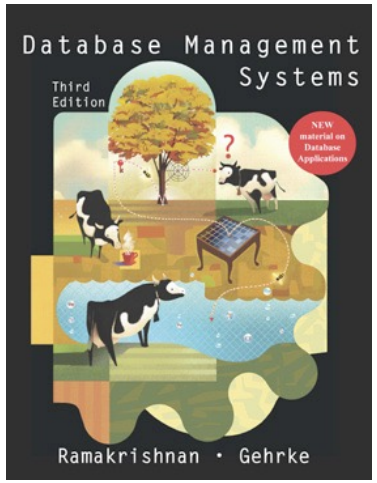
- It is possible, but unusual, to have relationship types of larger than 3 entities

Various notations: the big overview Part 1: binary relationships

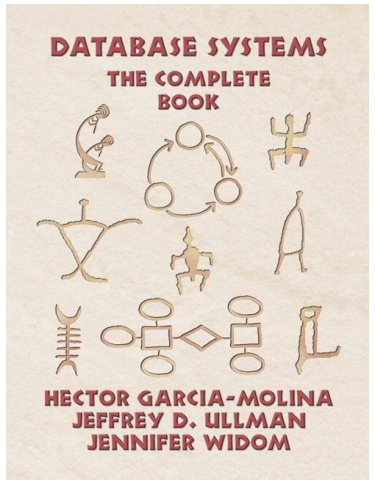
Different sources, different notations



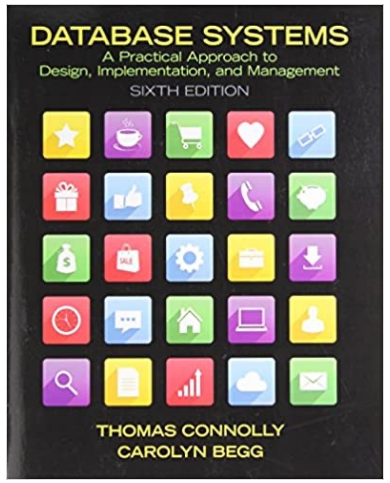
[Hoffer+'10]



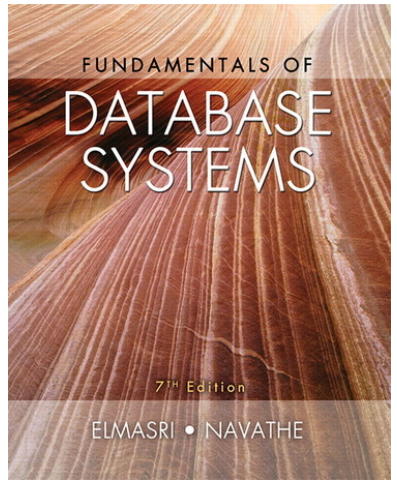
[Cow book'03]



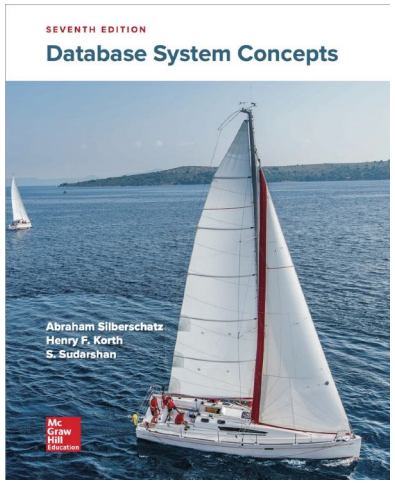
[Stanford book'08]



[Connolly+'15]



[Elmasri+'15]



[Silberschatz+'20]

Crow foot

SDK arrows

[Hoffer+'10]: Hoffer, Ramesh, Topi. Modern Database Management, 10th ed, 2010. https://www.pearson.com/us/higher-education/product/Hoffer-Modern-Database-Management-10th-Edition/9780136088394.html

[Cow book'03]: Ramakrishnan, Gehrke, Database Management Systems, 3rd ed, 2003. http://pages.cs.wisc.edu/~dbbook/

[Stanford book'08]: Garcia-Molina, Ullman, Widom. Database Systems: The Complete Book, 2nd ed, 2008. http://infolab.stanford.edu/~ullman/dscb.html

[Connolly+'15]: Connolly, Begg. Database systems: A practical approach to design, implementation, and management, 6th ed, 2015. https://www.pearson.com/us/higher-education/program/Connolly-Database-Systems-A-Practical-Approach-to-Design-Implementation-and-Management-6th-Edition/PGM116956.html

[Elmasri+'15]: Elmasri, Navathe. Fundamentals of Database Systems, 7th ed, 2015. https://www.pearson.com/us/higher-education/program/Elmasri-Fundamentals-of-Database-Systems-7th-Edition/PGM189052.html

[Silberschatz+'20]: Silberschatz, Korth, Sudarshan. Database system concepts, 7th ed, 2020. https://www.db-book.com/db7

Various Notations for binary one-to-many relationships

Every student is advised by 0-1 instructor.
An instructor may advise 0, 1 or more students.

Every product is manufactured by exactly 1 company.
A company may manufacture 0, 1 or more products

SDK [Silberschatz+'20]

?

[Stanford book'03]
also used by Gradiance

Crow's foot
[Hoffer+'10]

?

UML
[Connolly+'15]

[Elmasri+'15]

?

(min-max)
[Elmasri+'15]

?

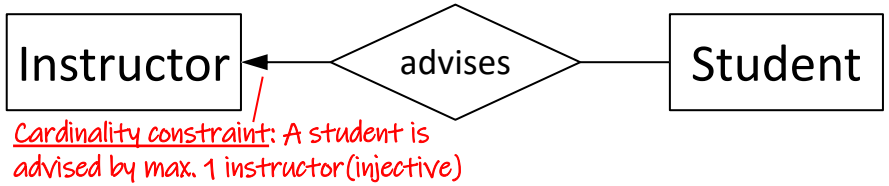
[Cow book'03]

Various Notations for binary one-to-many relationships

Every student is advised by 0-1 instructor.
An instructor may advise 0, 1 or more students.

Every product is manufactured by exactly 1 company.
A company may manufacture 0, 1 or more products

SDK [Silberschatz+'20]



[Stanford book'03]
also used by Gradiance

Crow's foot
[Hoffer+'10]

UML
[Connolly+'15]

?

[Elmasri+'15]

(min-max)
[Elmasri+'15]

?

[Cow book'03]

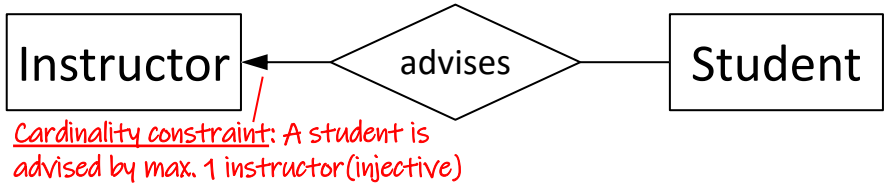
?

Various Notations for binary one-to-many relationships

Every student is advised by 0-1 instructor.
An instructor may advise 0, 1 or more students.

Every product is manufactured by exactly 1 company.
A company may manufacture 0, 1 or more products

SDK [Silberschatz+'20]

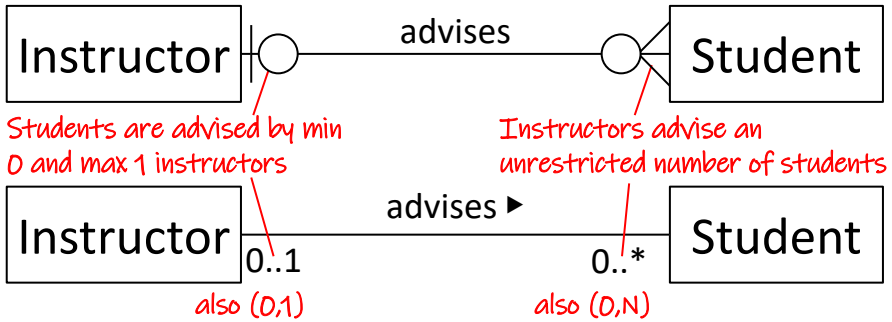


[Stanford book'03]
also used by Gradiance

Crow's foot
[Hoffer+'10]

UML
[Connolly+'15]

look across notation



[Elmasri+'15]

look across for cardinality, same-side for participation

(min-max)
[Elmasri+'15]

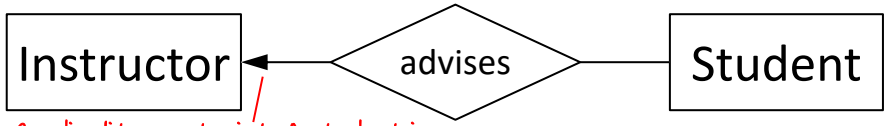
[Cow book'03]

Various Notations for binary one-to-many relationships

Every student is advised by 0-1 instructor.
An instructor may advise 0, 1 or more students.

Every product is manufactured by exactly 1 company.
A company may manufacture 0, 1 or more products

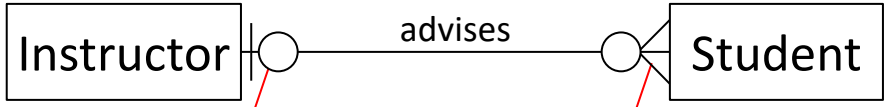
SDK [Silberschatz+'20]



Cardinality constraint: A student is advised by max. 1 instructor(injective)

[Stanford book'03]
also used by Gradiance

Crow's foot
[Hoffer+'10]

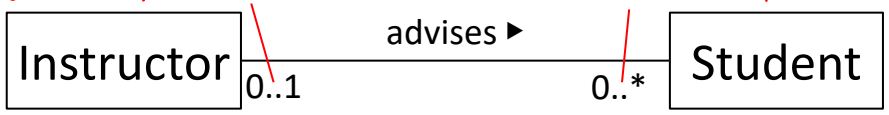


look across notation

Students are advised by min 0 and max 1 instructors

Instructors advise an unrestricted number of students

UML
[Connolly+'15]



also (0,1)

also (0,N)

look across for cardinality, same-side for participation

[Elmasri+'15]

Avoid (min-max) !!!

same-side notation

(min-max)
[Elmasri+'15]

strongly discouraged since it is the exact opposite of the more commonly used crow's foot look across notation

[Cow book'03]

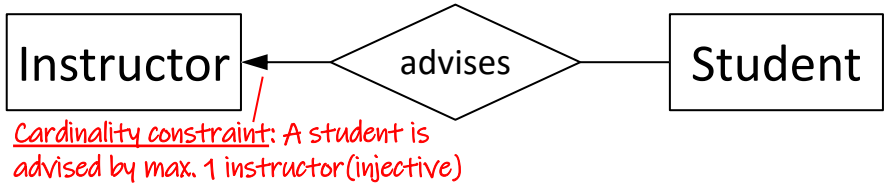
?

Various Notations for binary one-to-many relationships

Every student is advised by 0-1 instructor.
An instructor may advise 0, 1 or more students.

Every product is manufactured by exactly 1 company.
A company may manufacture 0, 1 or more products

SDK [Silberschatz+'20]

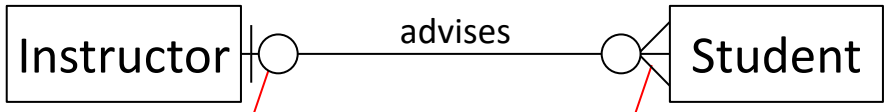


?

[Stanford book'03]
also used by Gradiance

Crow's foot
[Hoffer+'10]

look across notation



UML
[Connolly+'15]

?

[Elmasri+'15]

look across for cardinality, same-side for participation

Avoid (min-max) !!!

(min-max)
[Elmasri+'15]

same-side notation

strongly discouraged since it is the exact opposite of the more commonly used crow's foot look across notation

?

[Cow book'03]

?

Various Notations for binary one-to-many relationships

Every student is advised by 0-1 instructor.
An instructor may advise 0, 1 or more students.

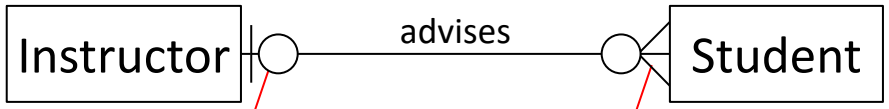
SDK [Silberschatz+'20]

Cardinality constraint: A student is advised by max. 1 instructor(injective)

[Stanford book'03]
also used by Gradiance

Crow's foot
[Hoffer+'10]

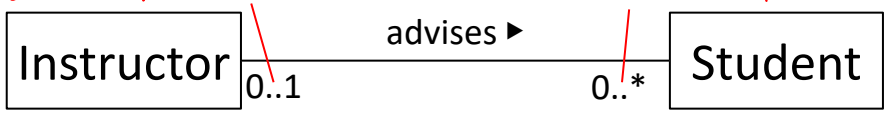
look across notation



Students are advised by min 0 and max 1 instructors

Instructors advise an unrestricted number of students

UML
[Connolly+'15]



also (0,1)

also (0,N)

[Elmasri+'15]

look across for cardinality, same-side for participation

Avoid (min-max) !!!

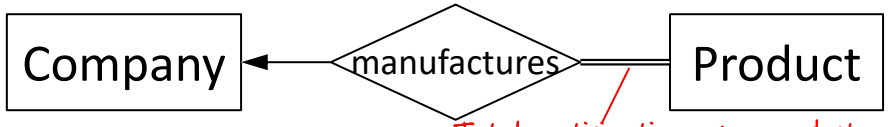
(min-max)
[Elmasri+'15]

same-side notation

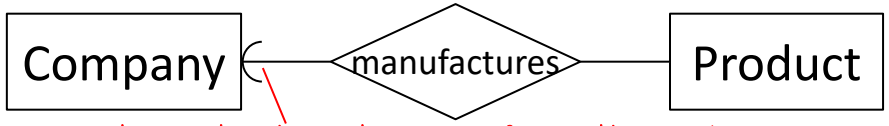
strongly discouraged since it is the exact opposite of the more commonly used crow's foot look across notation

[Cow book'03]

Every product is manufactured by exactly 1 company.
A company may manufacture 0, 1 or more products



Total participation: every product needs to be manufactured



Every product needs to be product is manufactured by exactly 1 company. (referential integrity constraint).

?

?

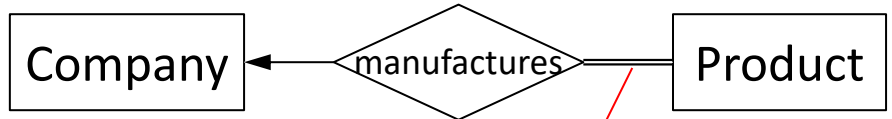
?

Various Notations for binary one-to-many relationships

Every student is advised by 0-1 instructor.
An instructor may advise 0, 1 or more students.

Every product is manufactured by exactly 1 company.
A company may manufacture 0, 1 or more products

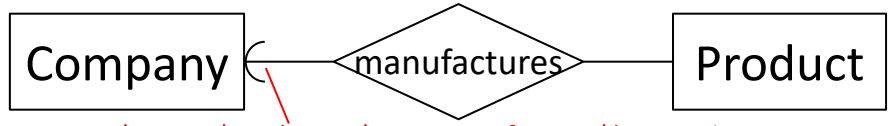
SDK [Silberschatz+'20]



Cardinality constraint: A student is advised by max. 1 instructor(injective)

Total participation: every product needs to be manufactured

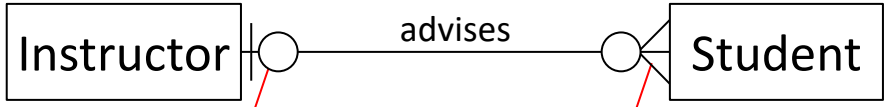
[Stanford book'03]
also used by Gradiance

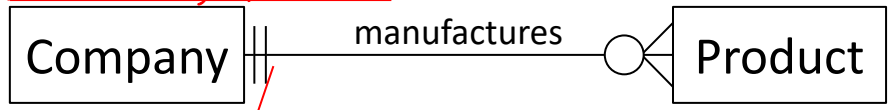


Every product needs to be product is manufactured by exactly 1 company. (referential integrity constraint).

Crow's foot
[Hoffer+'10]

look across notation



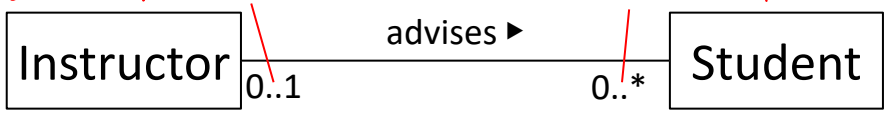


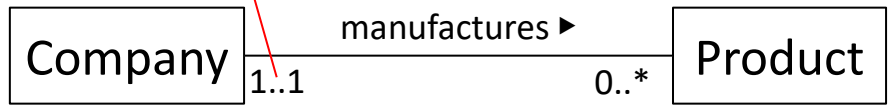
Students are advised by min 0 and max 1 instructors

Instructors advise an unrestricted number of students

Products are mapped to min 1 and max 1 companies

UML
[Connolly+'15]





also (0,1)

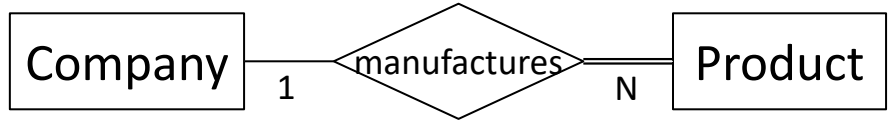
also (0,N)

also (1,1)

also: sometimes participation not shown

[Elmasri+'15]

look across for cardinality, same-side for participation



Avoid (min-max) !!!

(min-max)
[Elmasri+'15]

same-side notation

strongly discouraged since it is the exact opposite of the more commonly used crow's foot look across notation

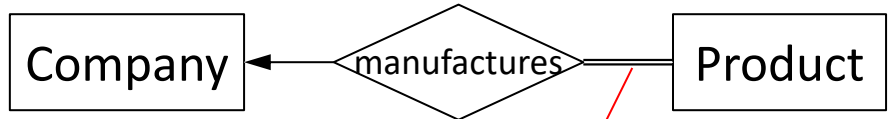
[Cow book'03]

Various Notations for binary one-to-many relationships

Every student is advised by 0-1 instructor.
An instructor may advise 0, 1 or more students.

Every product is manufactured by exactly 1 company.
A company may manufacture 0, 1 or more products

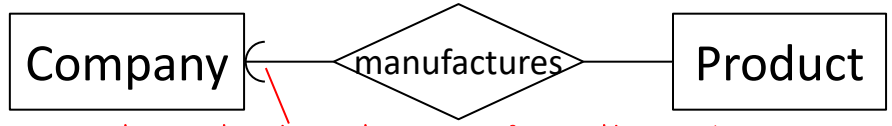
SDK [Silberschatz+'20]



Cardinality constraint: A student is advised by max. 1 instructor(injective)

Total participation: every product needs to be manufactured

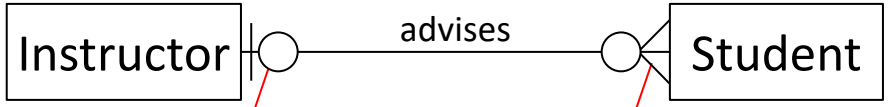
[Stanford book'03]
also used by Gradiance

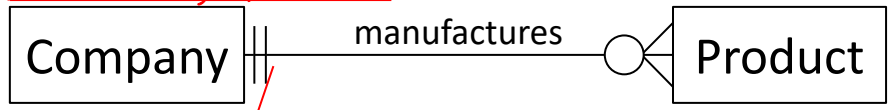


Every product needs to be product is manufactured by exactly 1 company. (referential integrity constraint).

look across notation

Crow's foot
[Hoffer+'10]



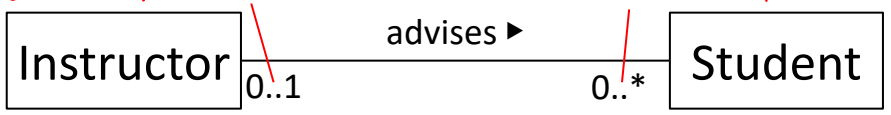


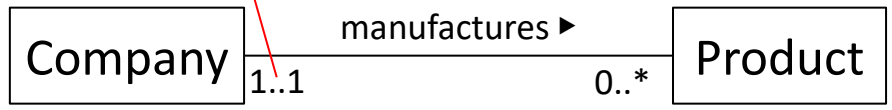
Students are advised by min 0 and max 1 instructors

Instructors advise an unrestricted number of students

Products are mapped to min 1 and max 1 companies

UML
[Connolly+'15]





also (0,1)

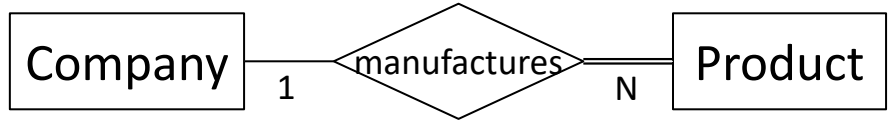
also (0,N)

also (1,1)

also: sometimes participation not shown

look across for cardinality, same-side for participation

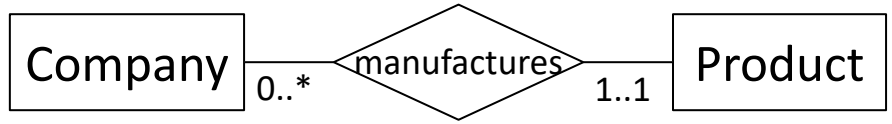
[Elmasri+'15]



Avoid (min-max) !!!

same-side notation

(min-max)
[Elmasri+'15]



strongly discouraged since it is the exact opposite of the more commonly used crow's foot look across notation

?

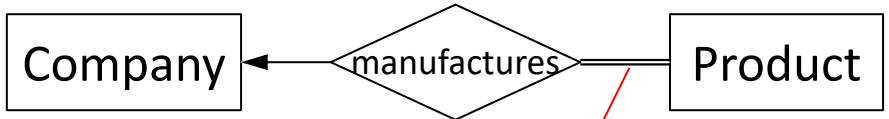
[Cow book'03]

Various Notations for binary one-to-many relationships

Every student is advised by 0-1 instructor.
An instructor may advise 0, 1 or more students.

Every product is manufactured by exactly 1 company.
A company may manufacture 0, 1 or more products

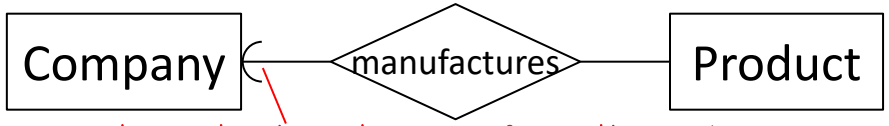
SDK [Silberschatz+'20]



Cardinality constraint: A student is advised by max. 1 instructor(injective)

Total participation: every product needs to be manufactured

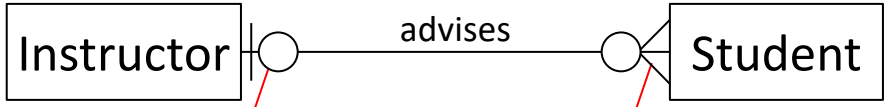
[Stanford book'03]
also used by Gradiance

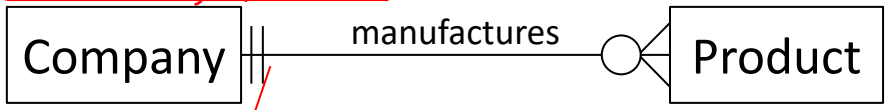


Every product needs to be product is manufactured by exactly 1 company. (referential integrity constraint).

look across notation

Crow's foot
[Hoffer+'10]



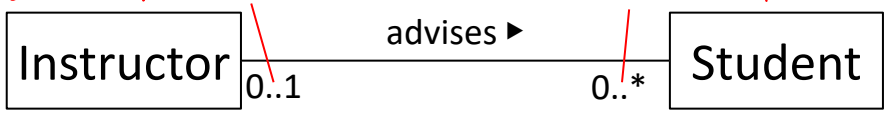


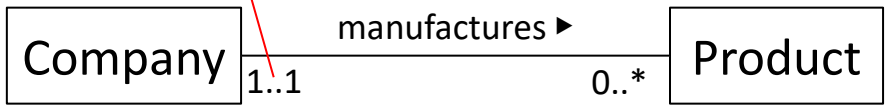
Students are advised by min 0 and max 1 instructors

Instructors advise an unrestricted number of students

Products are mapped to min 1 and max 1 companies

UML
[Connolly+'15]





also (0,1)

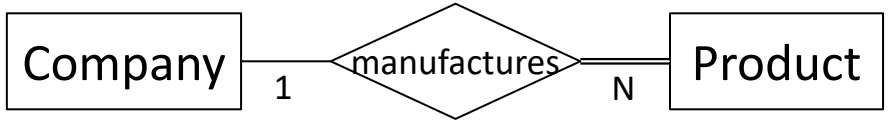
also (0,N)

also (1,1)

also: sometimes participation not shown

look across for cardinality, same-side for participation

[Elmasri+'15]



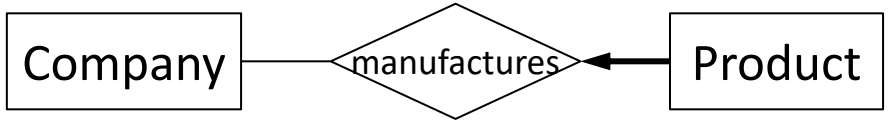
Avoid (min-max) !!!

same-side notation

(min-max)
[Elmasri+'15]

strongly discouraged since it is the exact opposite of the more commonly used crow's foot look across notation

[Cow book'03]



Various Notations for binary one-to-many relationships

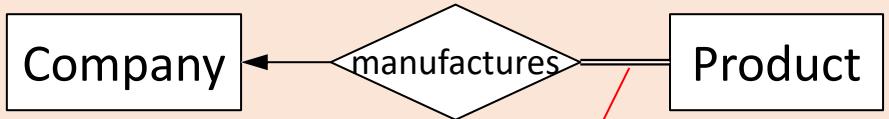
if you are overwhelmed by the different notations, just use the one from our textbook

Every student is advised by maximum 1 instructor.
An instructor may advise 0, 1 or more students.

Every product is manufactured by exactly 1 company.
A company may manufacture 0, 1 or more products.

SDK [Silberschatz+'20]

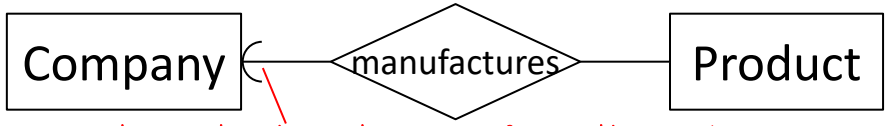
Cardinality constraint: A student is advised by max. 1 instructor(injective)



Total participation: every product needs to be manufactured

[Stanford book'03]

also used by Gradiance



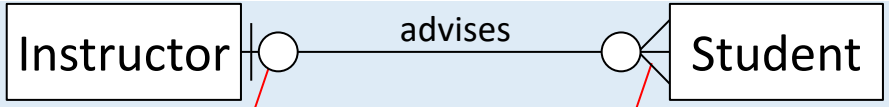
Every product needs to be product is manufactured by exactly 1 company. (referential integrity constraint).

Crow's foot

[Hoffer+'10]

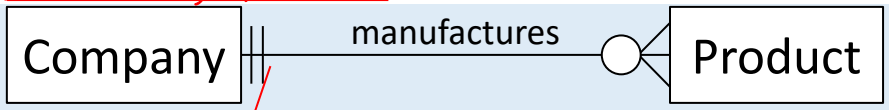
Most often used in practice

look across notation



Students are advised by min 0 and max 1 instructors

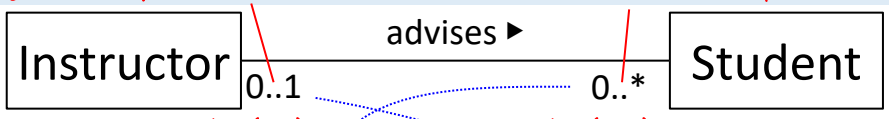
Instructors advise an unrestricted number of students



Products are mapped to min 1 and max 1 companies

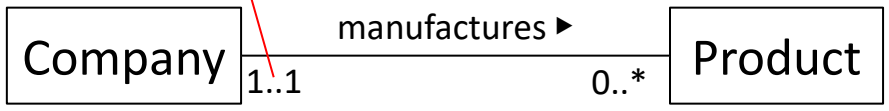
UML

[Connolly+'15]



also (0,1)

also (0,N)

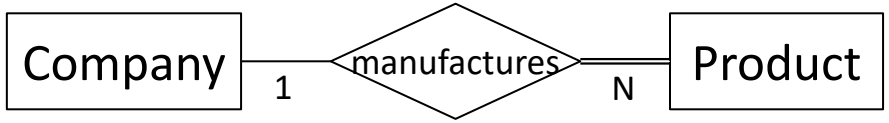


also (1,1)

also: sometimes participation not shown

[Elmasri+'15]

look across for cardinality, same-side for participation



(min-max)

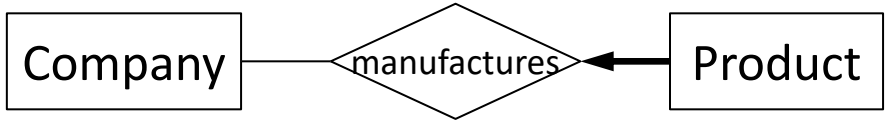
[Elmasri+'15]

Avoid (min-max) !!!

same-side notation

strongly discouraged since it is the exact opposite of the more commonly used crow's foot look across notation

[Cow book'03]

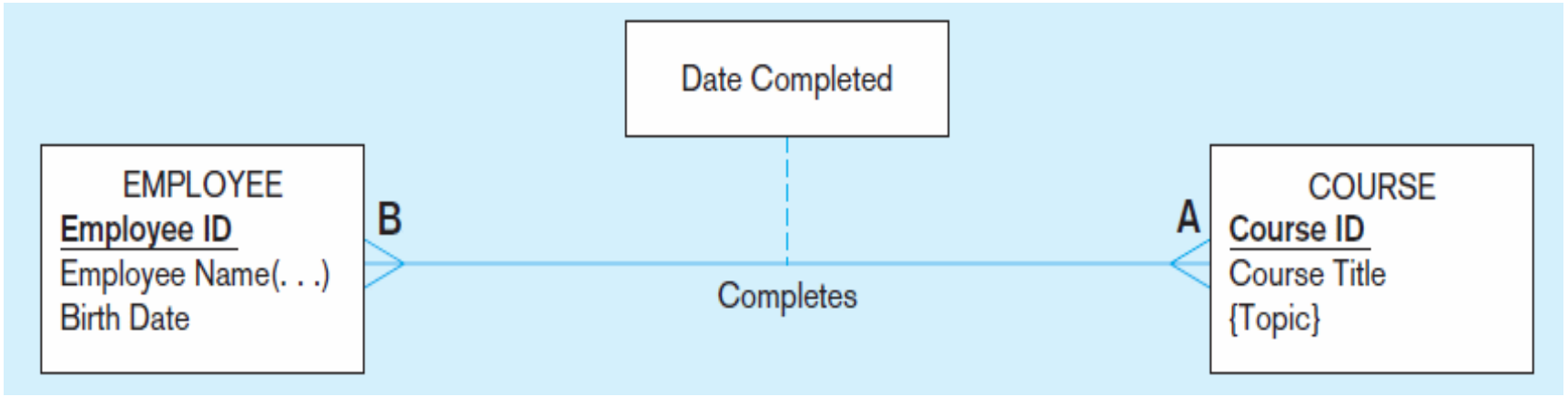


Associative Entities (specific to crow's foot & UML notation)

Associative Entities

- It seems like a relationship: it links entities together
- Yet it seems like an entity: it has attributes
- When should a relationship with attributes instead be an associative entity?
  - All relationships for the associative entity should be many (cardinalities)
  - The associative entity could have meaning independent of the other entities
  - The associative entity preferably has a unique identifier (but not necessarily), and should also have other attributes
  - Ternary relationships should be converted to associative entities
  - (The associative entity may participate in other relationships other than the entities of the associated relationship)

Example: Associative Entity

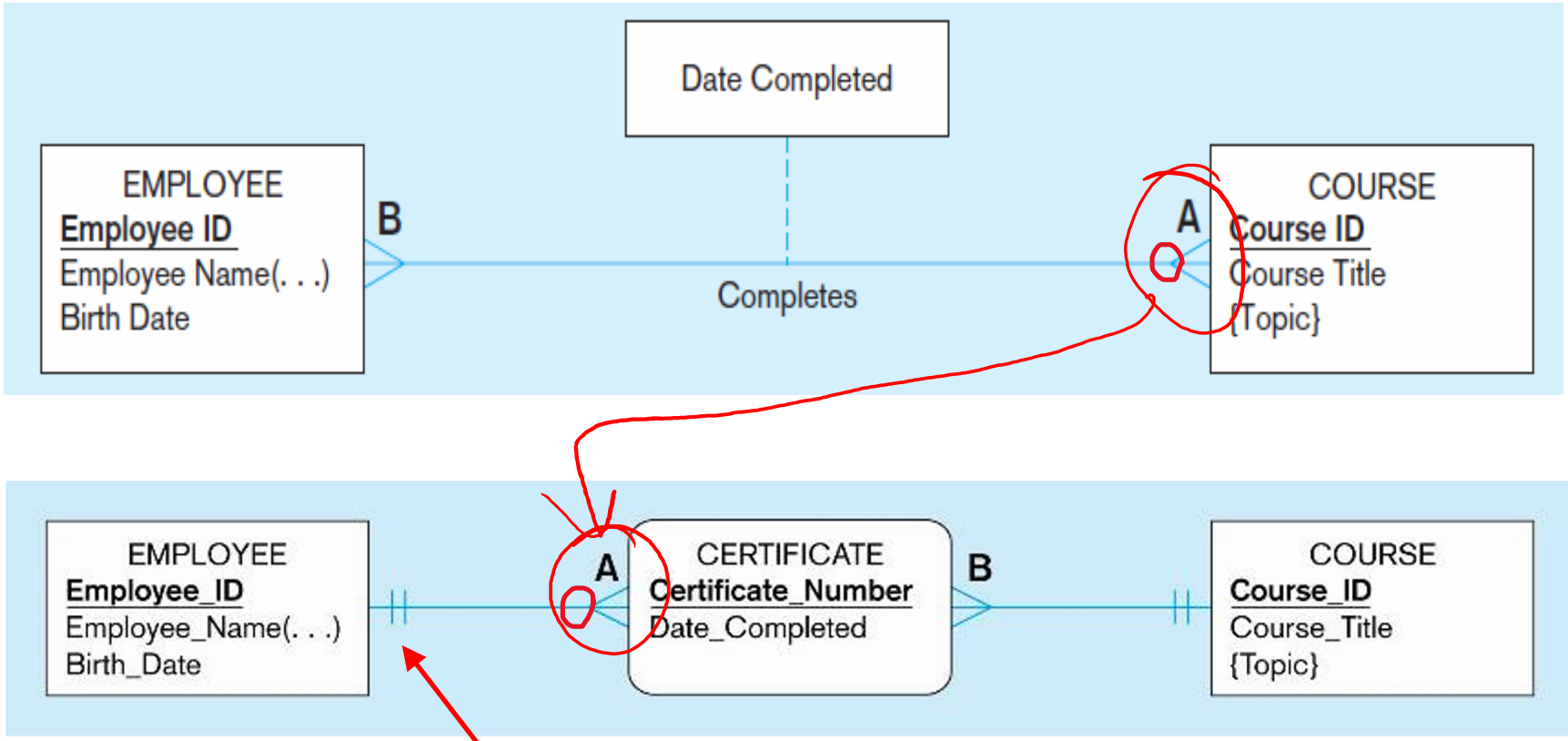


Associative entity can be represented as rectangle with rounded corners ([Hoffer+'10] notation)

?

Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

Example: Associative Entity



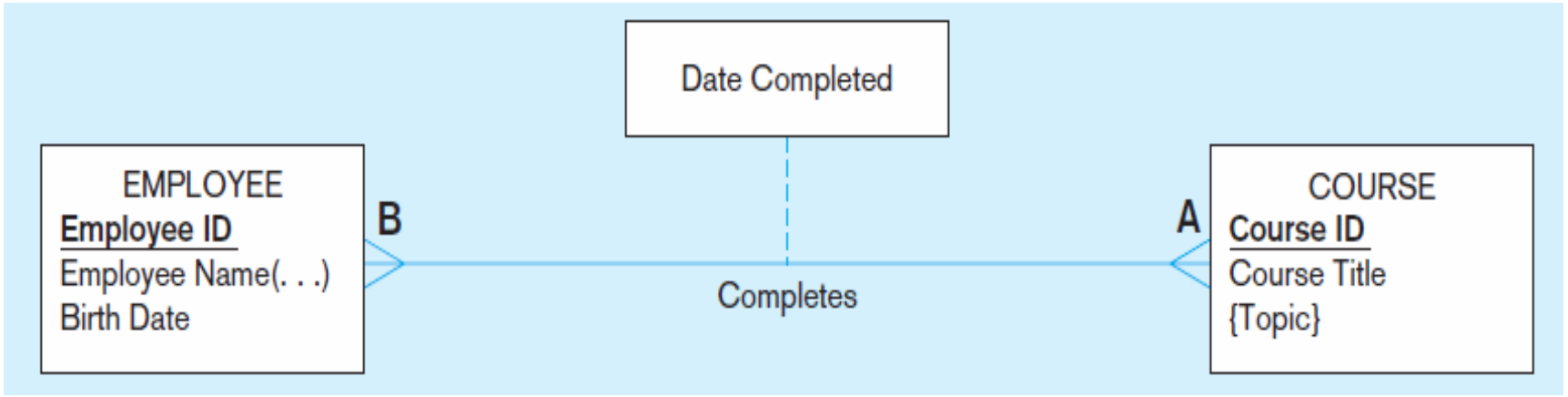
Associative entity can be represented as rectangle with rounded corners ([Hoffer+'10] notation)

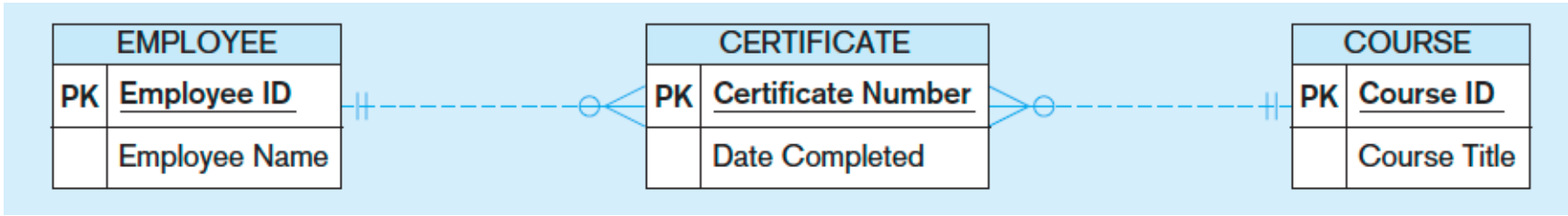
Notice that the many-to-many cardinality symbols face toward the associative entity and not toward the other entities

Where does the "mandatory" participation constraint come from? ?

Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

Example: Associative Entity





Associative entity can be represented as rectangle with rounded corners ([Hoffer+'10] notation)

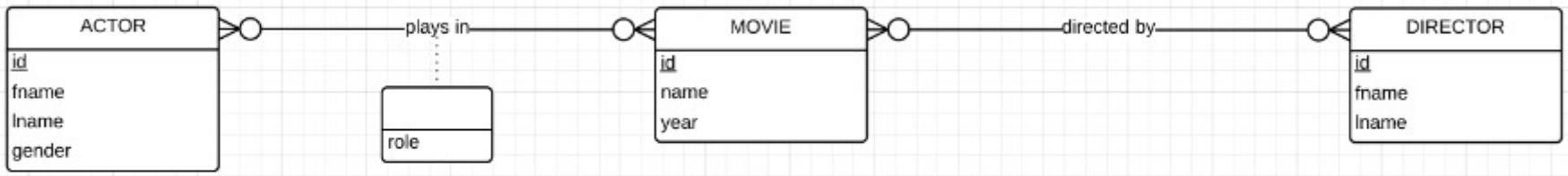
Notice that the many-to-many cardinality symbols face toward the associative entity and not toward the other entities

Microsoft Visio notation.
Remember: PKs yes, but no FKs in ERDs!

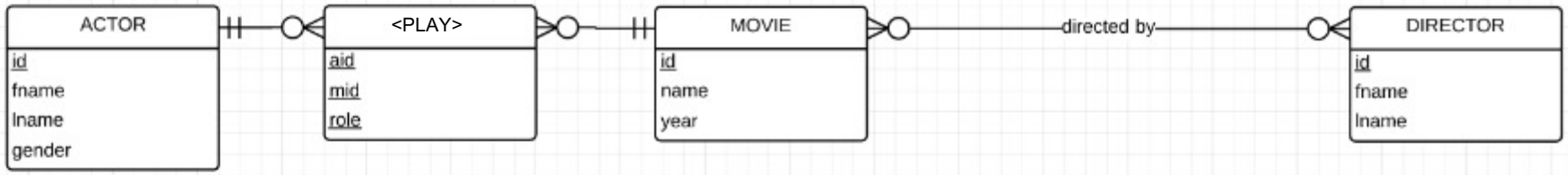
Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

Play table in our IMDB movie database

ER diagram: don't forget identifiers, but no FKs



ER diagram: PLAY as associative entity can be justified



Relational schema: don't forget PKs and FKs