Updated 10/31/2022

Topic 2: Database design
L15: ER & Relational modeling

Wolfgang Gatterbauer

CS3200 Database design (fa22)

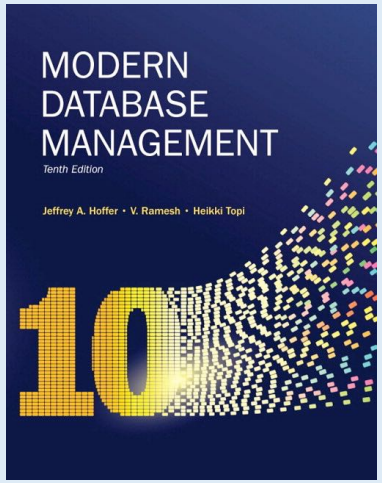
https://northeastern-datalab.github.io/cs3200/fa22s3/

10/31/2022

Class warm-up

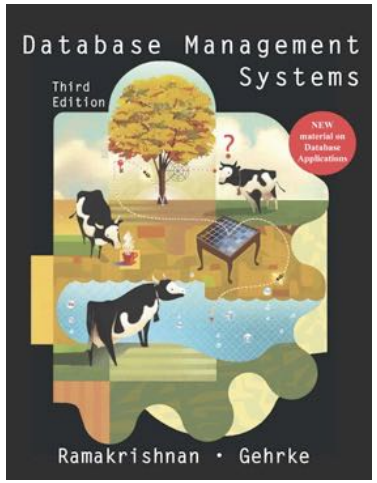
- Last class summary (incl. University example)
- WED: Zoom meeting
  - please use your webcams (it helps me and thus also helps you)
  - slides will be posted WED on Piazza, I am unavailable through SUN
- Project phase 1: see Piazza post, my feedback by TUE
- Exam2: WED next week:
  - SQL part: to be submitted on Gradescope
  - Design: paper & pen: to be handed in afterwards and we bulk-upload
  - normalization may or may not be part of exam (to be decided based on our progress). But we continue with normal forms after the exam.
- Thanks for feedback: more in-class text to ERD next time (need to prepare)
- We continue with Database Design, also hands-on

Different sources, different notations

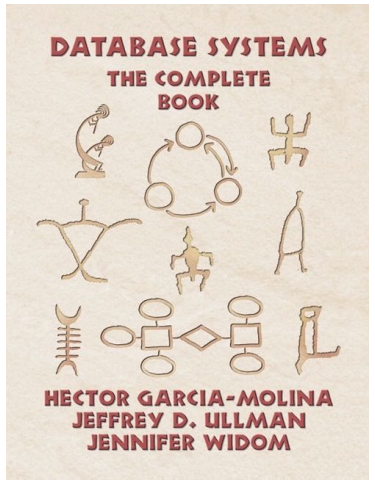


[Hoffer+'10]

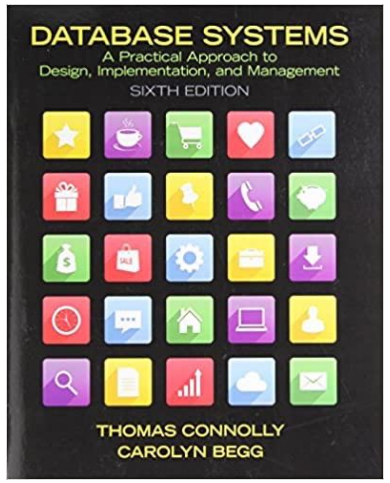
Crow's foot



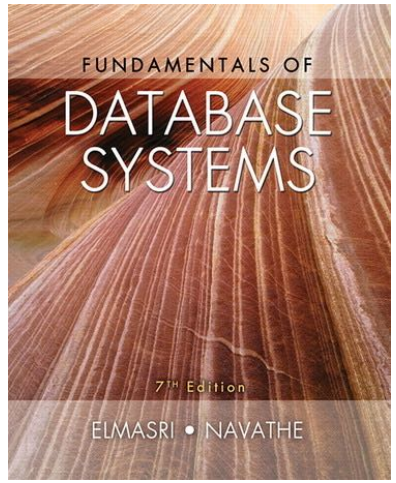
[Cow book'03]



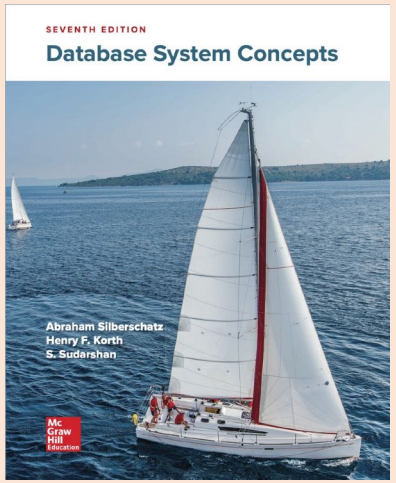
[Stanford book'08]



[Connolly+'15]



[Elmasri+'15]



[Silberschatz+'20]

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

Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

Notations for binary one-to-many relationships

10/24/2022

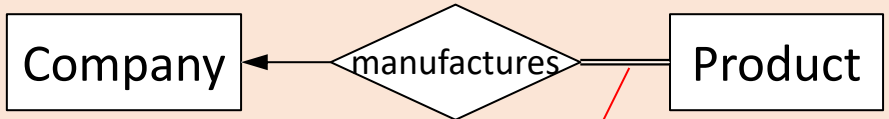
if you are overwhelmed by the different notations, just use the one from our textbook

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]

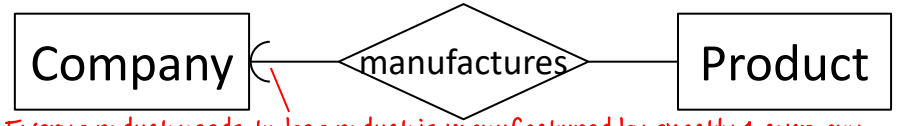
Cardinality constraint: A student is advised by max. 1 instructor(injective)



Total participation: every product needs to be manufactured

[Stanford book'03]

also used by Gradiance



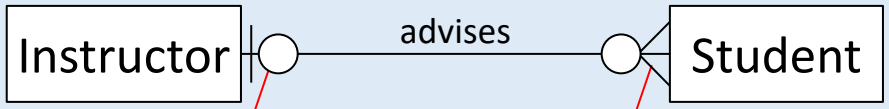
Every product needs to be product is manufactured by exactly 1 company. (referential integrity constraint).

Crow's foot

[Hoffer+'10]

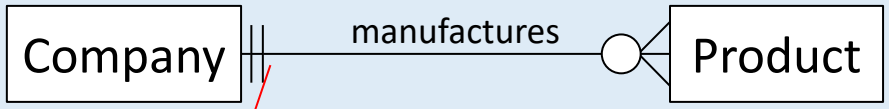
Most often used in practice

look across notation



Students are advised by min 0 and max 1 instructors

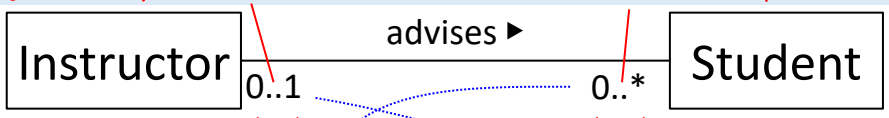
Instructors advise an unrestricted number of students



Products are mapped to min 1 and max 1 companies

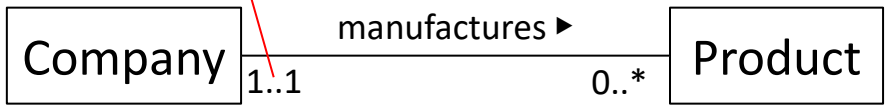
UML

[Connolly+'15]



also (0,1)

also (0,N)

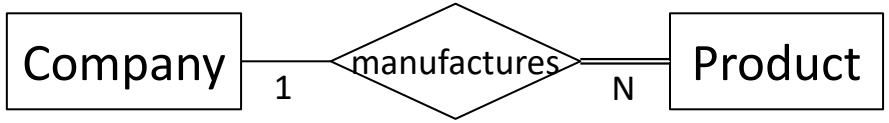


also (1,1)

also: sometimes participation not shown

[Elmasri+'15]

look across for cardinality, same-side for participation



(min-max)

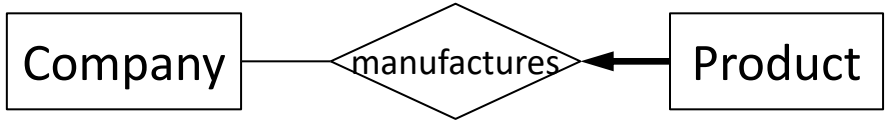
[Elmasri+'15]

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]



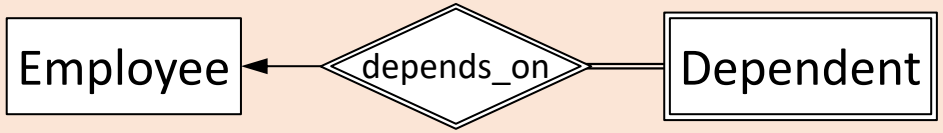
Notations for weak entities

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

A course may have 0, 1 or more sections

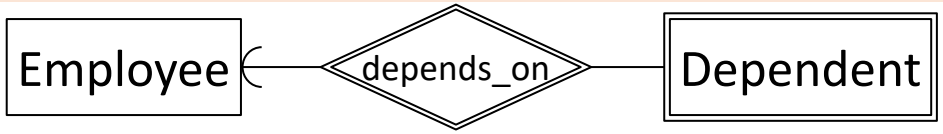
Every product is manufactured by exactly 1 company.
A company may manufacture 0, 1 or more products.

[Silberschatz+'20]



Total participation: every product needs to be manufactured (surjective)

[Stanford book'03]
also by Gradiance

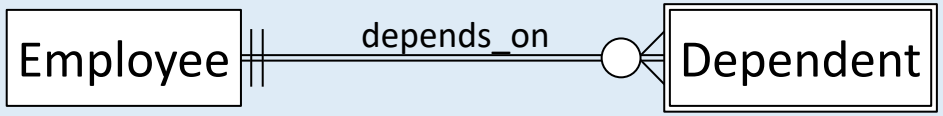


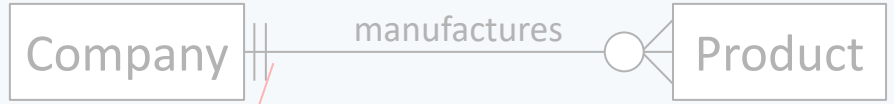
Every product needs to be product is manufactured by exactly 1 company. (referential integrity constraint).

Crow's foot
[Hoffer+'10]

Most often used in practice

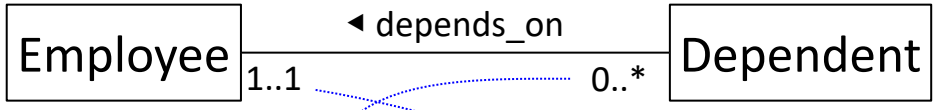
look across notation





Products are mapped to min 1 and max 1 companies

UML
[Connolly+'15]

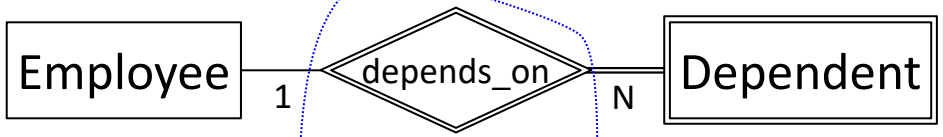


also (1,1)

also: sometimes participation not shown

[Elmasri+'15]

look across for cardinality, same-side for participation



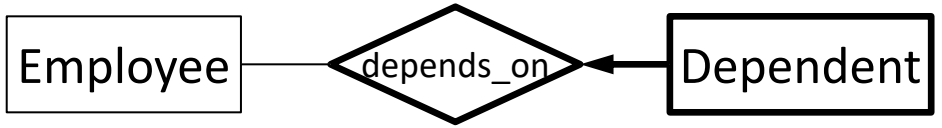
Avoid (min-max) !!!

(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]



Notations for entities and their attributes

Chen "bubble" variants

"Boxed" variants

[Silberschatz+'20]

[Stanford book'03]
also used by Gradiance

Crow's foot
[Hoffer+'10]

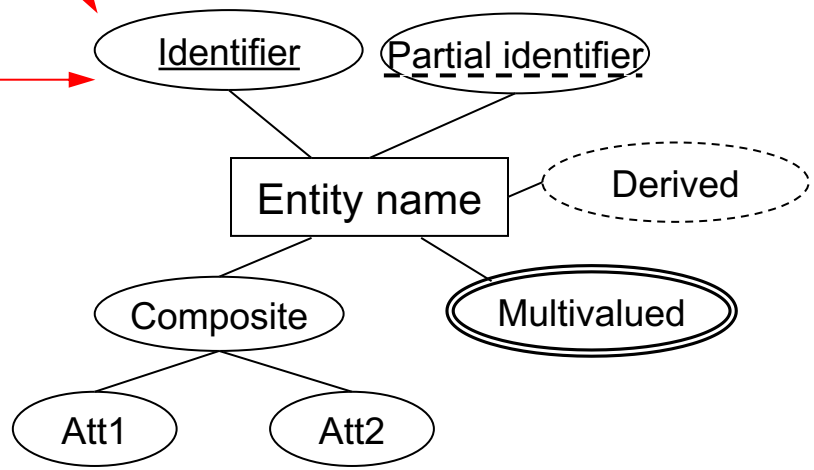
UML
[Connolly+'15]

[Elmasri+'15]

[Cow book'03]

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

no visual notation for optional attributes (vs. mandatory, which will later become "NOT NULL" in the relational schema)



entity name
identifier
partial identifier
derived ()
{multivalued}
composite
att1
att2

ENTITY NAME
Identifier
Partial identifier
Optional
[Derived]
{Multivalued}
Composite(, ,)

Entity name
identifier {PK}
optional [0..1]
/derived
multivalued [1..*]
composite
Att1
Att2

sometimes {id}

unified concept for mandatory, optional, multivalued

no notion of weak or strong entity in UML notation

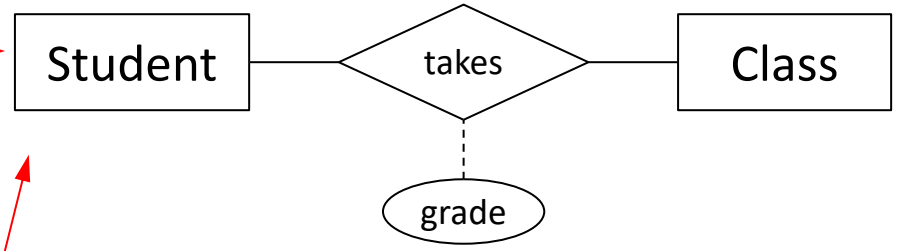
In practice you will likely see some variant of all these three variants

Notations for attributes on binary relationships

"Bubble variants"

[Silberschatz+'20]

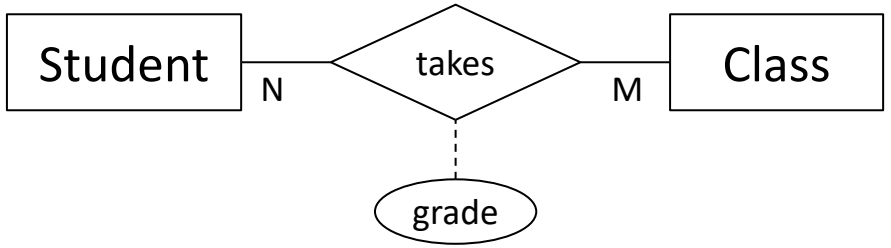
[Stanford book'03]
also used by Gradiance



Crow's foot
[Hoffer+'10]

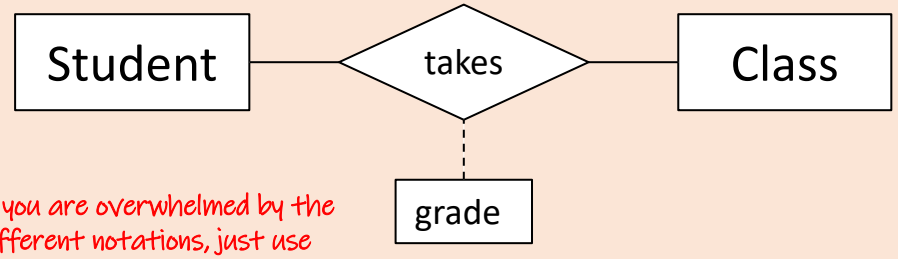
UML
[Connolly+'15]

[Elmasri+'15]

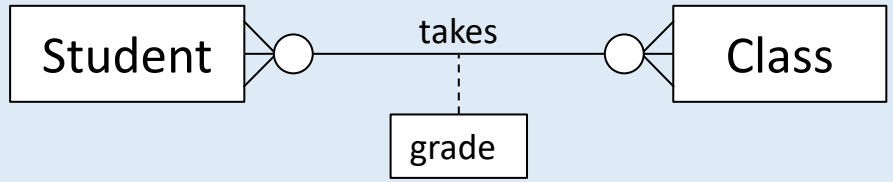


[Cow book'03]

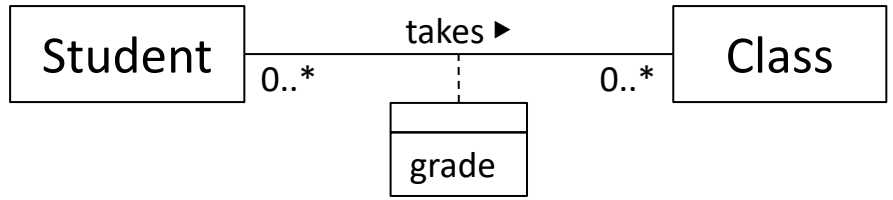
"Box variants"



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



In practice you will likely see some variant of all these three variants



Notations for specialization ("ISA relationship")

Participation (or covering) constraint
(optional=partial | mandatory=total)

Partial-overlapping optional

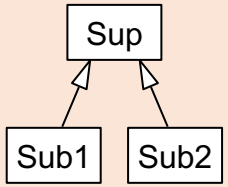
Partial-disjoint

Total-overlapping mandatory

Total-disjoint

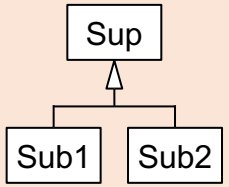
[Silberschatz+'20]

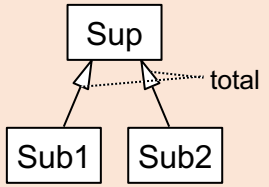
if you are overwhelmed by the different notations, just use the one from our textbook

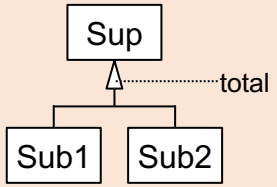


Super entity (more generalized, higher-level)

Sub entity (more specialized, lower-level)



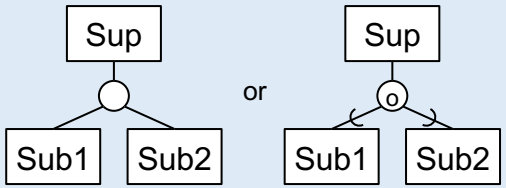


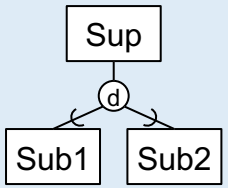


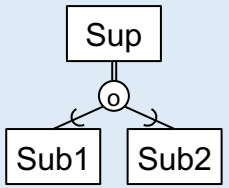
[Elmasri+'15]

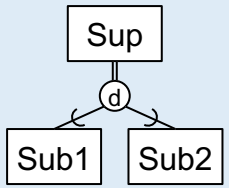
Crow's foot

[Hoffer+'10]





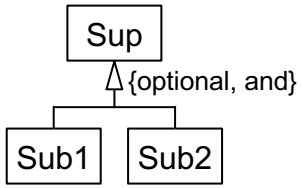


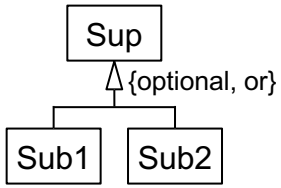


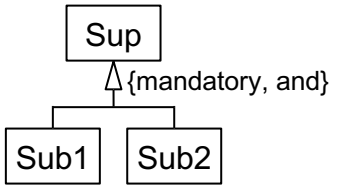
UML

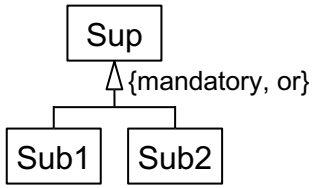
[Connolly+'15]

Overlap (or disjoint) constraints
(or=disjoint | and=overlapping)





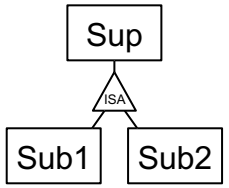




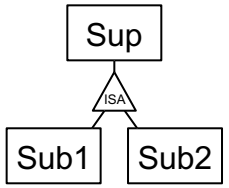
[Stanford book'03]

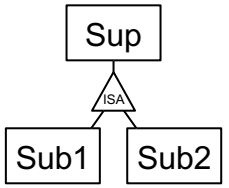
also by Gradiance

[Cow book'03]

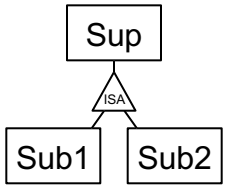


Sub1 overlaps Sub2





Sub1 overlaps Sub2
Sub1 and Sub2 cover Sup



Sub1 and Sub2 cover Sup

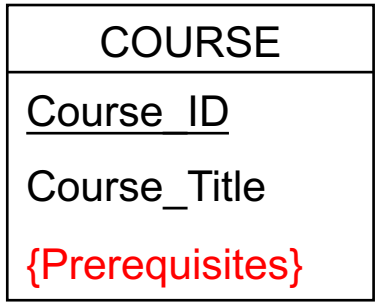
More Practice

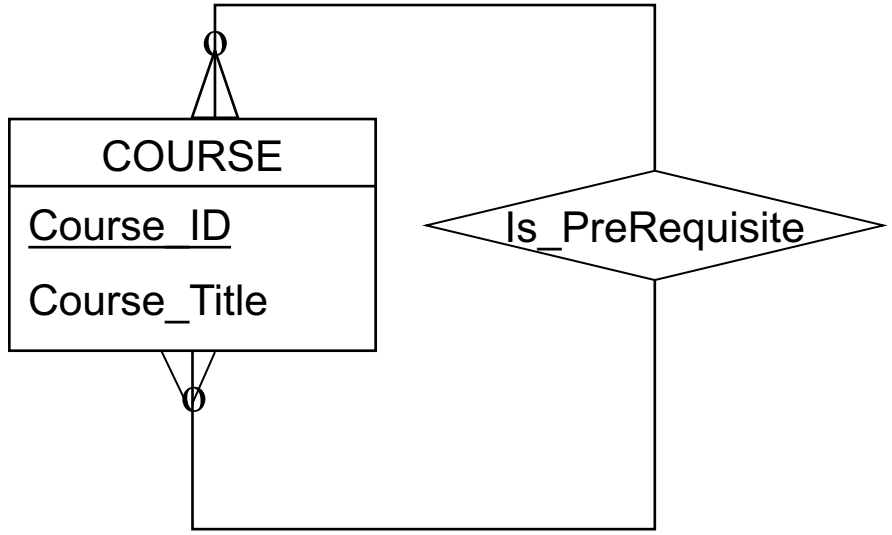
1. Multivalued attributes represented as relationships

COURSE
Course_ID
Course_Title
{Prerequisites}

?

1. Multivalued attributes represented as relationships





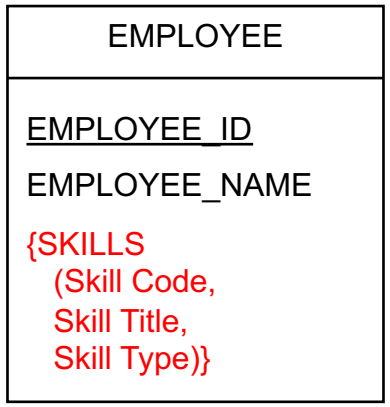
2. Multivalued attributes can be represented as entities

EMPLOYEE
EMPLOYEE_ID EMPLOYEE_NAME {SKILLS (Skill Code, Skill Title, Skill Type)}

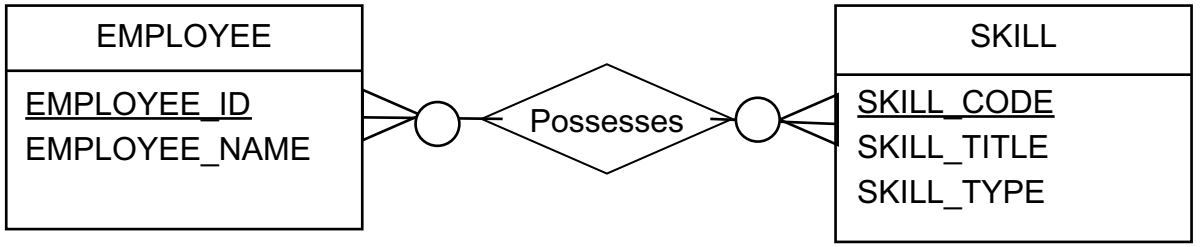
composite

?

2. Multivalued attributes can be represented as entities



composite

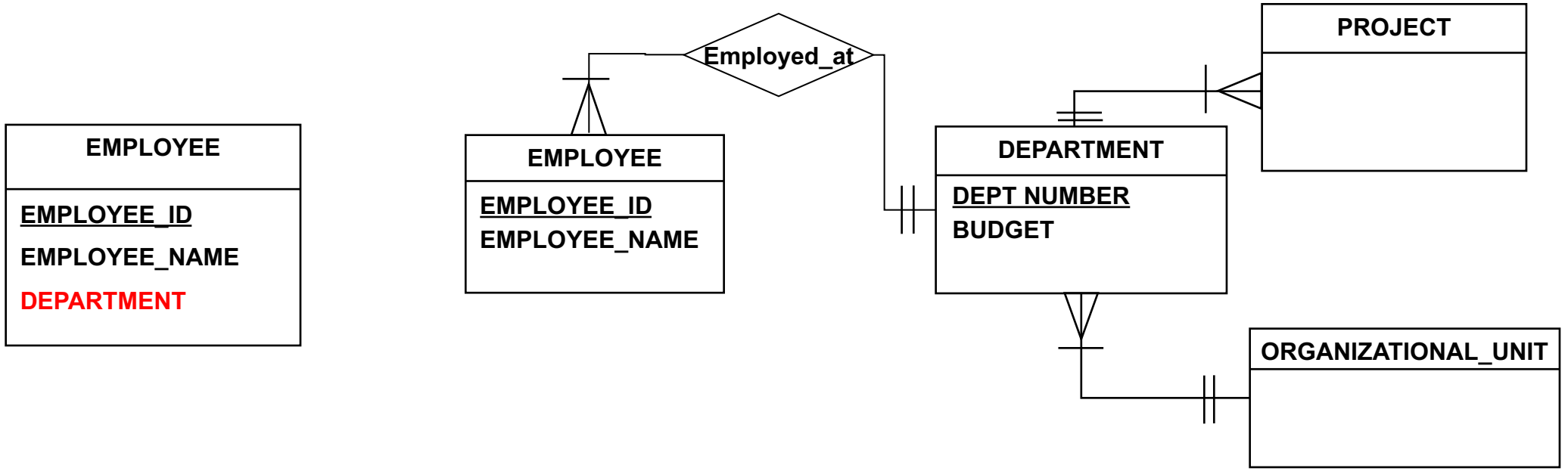


3. Attribute vs.

?

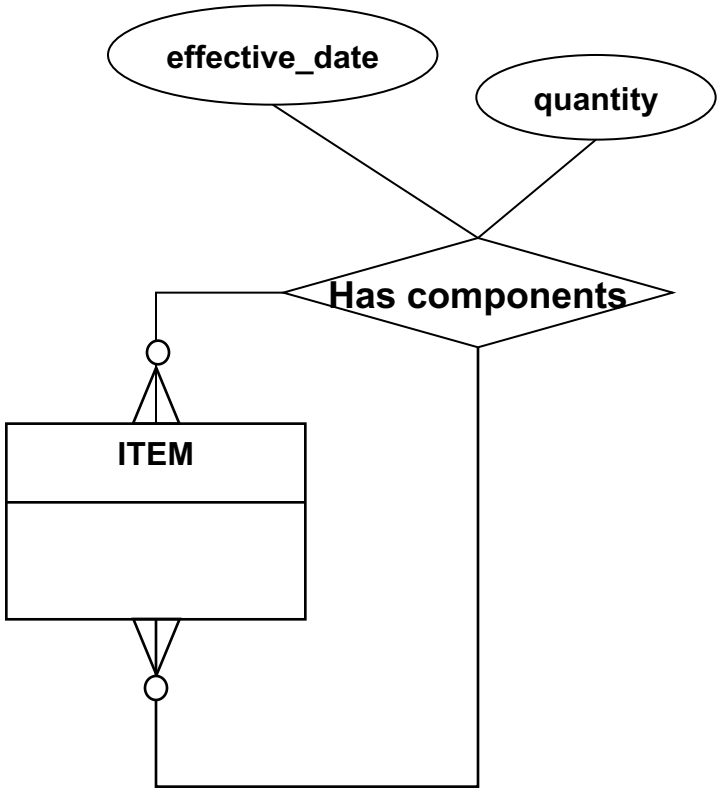
EMPLOYEE
EMPLOYEE_ID
EMPLOYEE_NAME
DEPARTMENT

3. Attribute vs. Multiple Entities



Bill-of-materials (BOM) structure

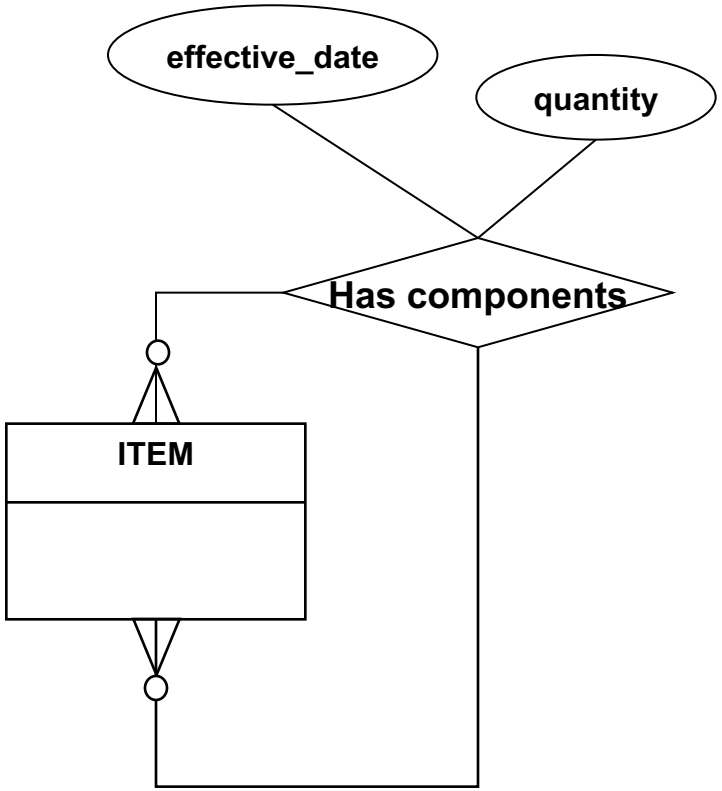
Relationship



?

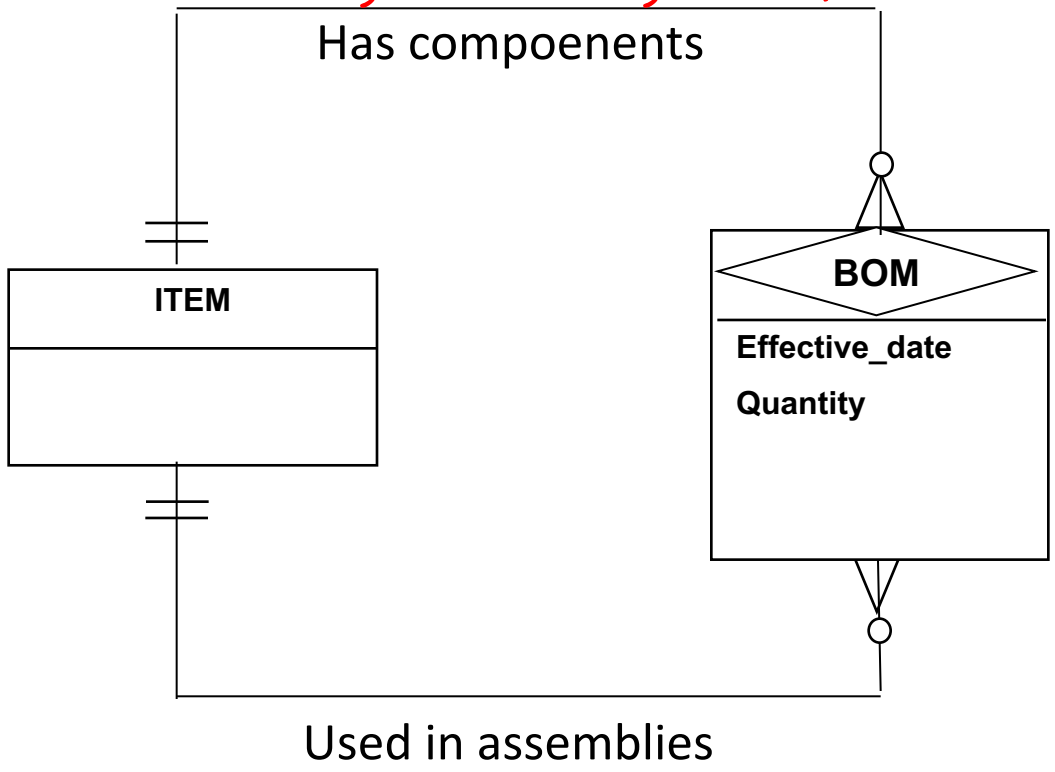
Bill-of-materials (BOM) structure

Relationship



Associative entity

possible but I would not recommend; unless want to preserve history:
- weak entity with partial identifier
- or strong with surrogate keys

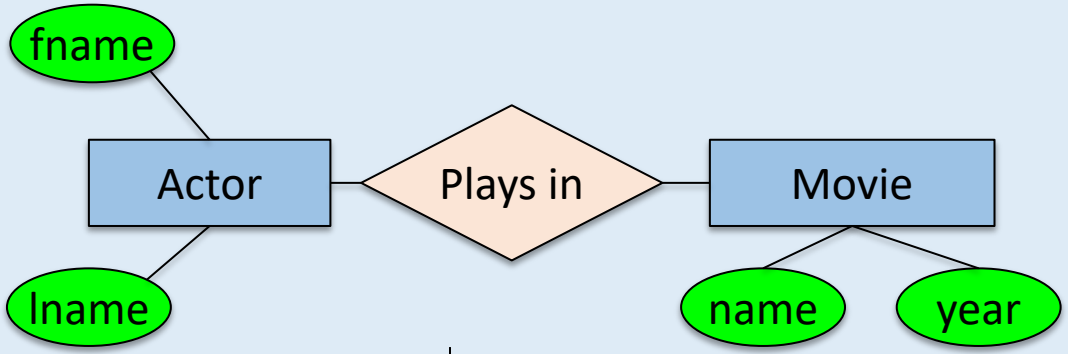


Quick overview: "Relational modeling" From ERDs to Relations

Data modeling and Database Design Process

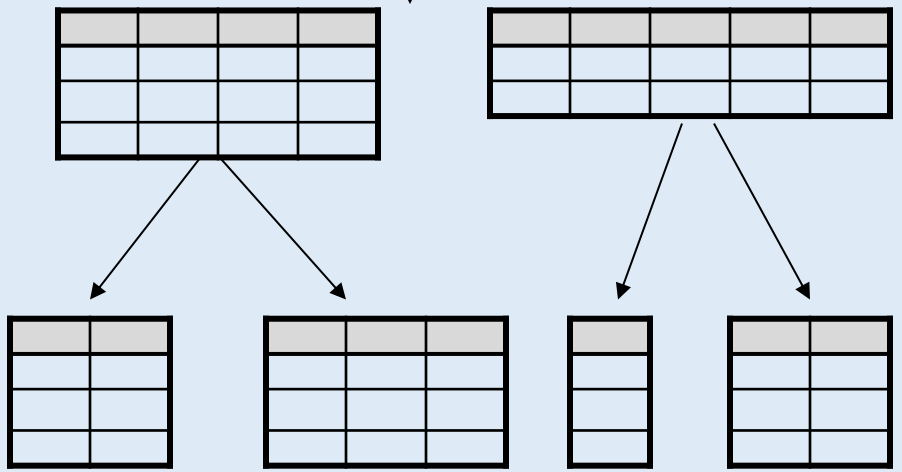
1. ER Diagram

Conceptual Model:
("technology independent")
describe main data items



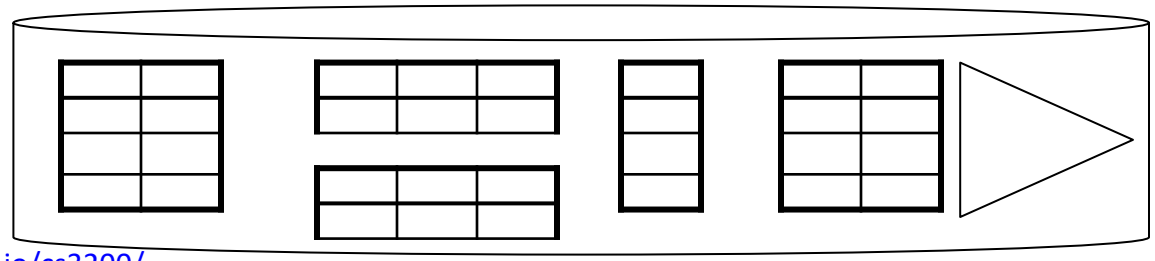
2. Relational Database Design

Logical Model
("for relational databases"):
Tables, Constraints
Functional Dependencies
Normalization:
Eliminates anomalies



3. Database Implementation

Physical Model
Physical storage details
Result: Physical Schema

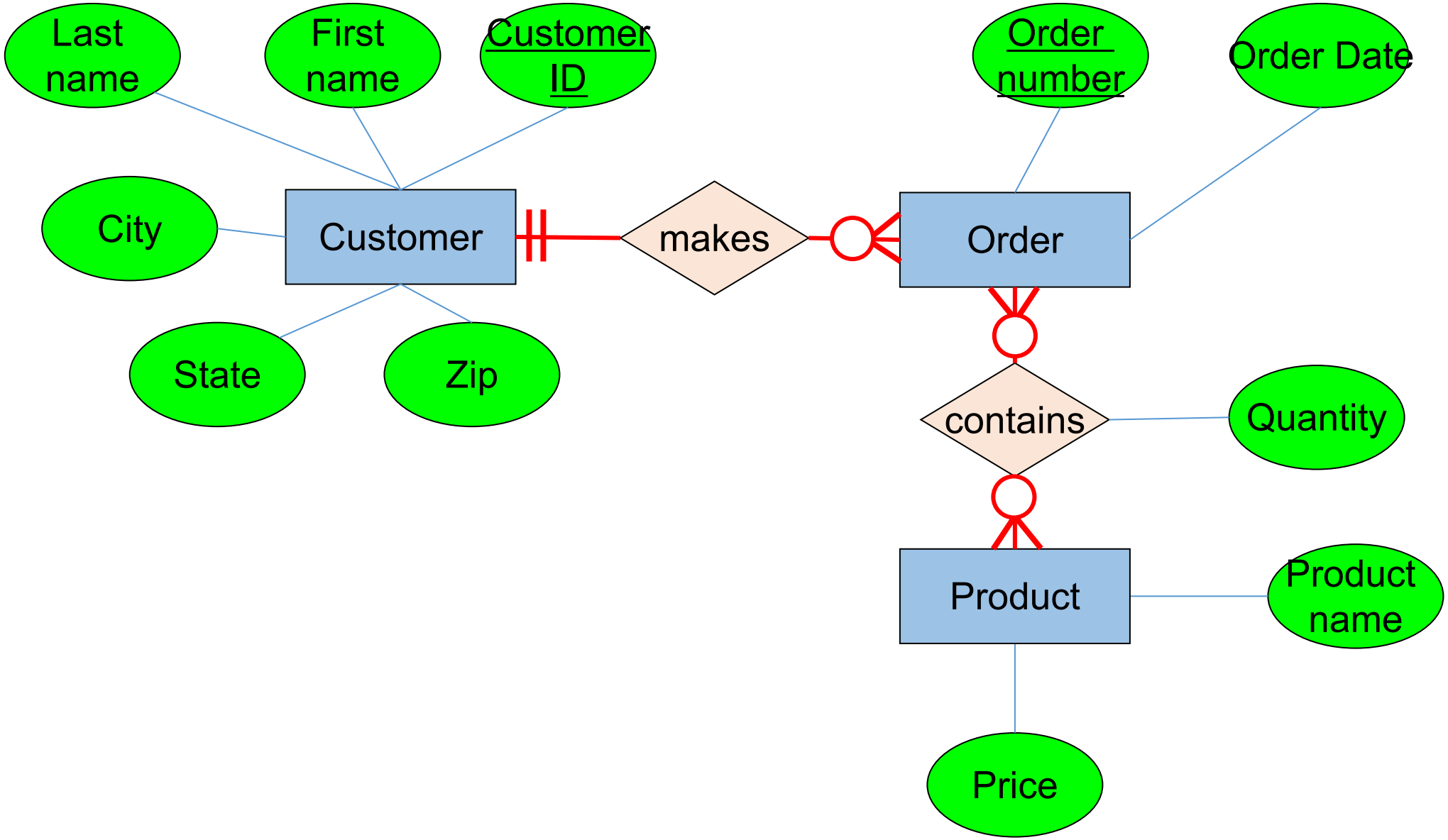


From ER Diagrams to Relational Schema

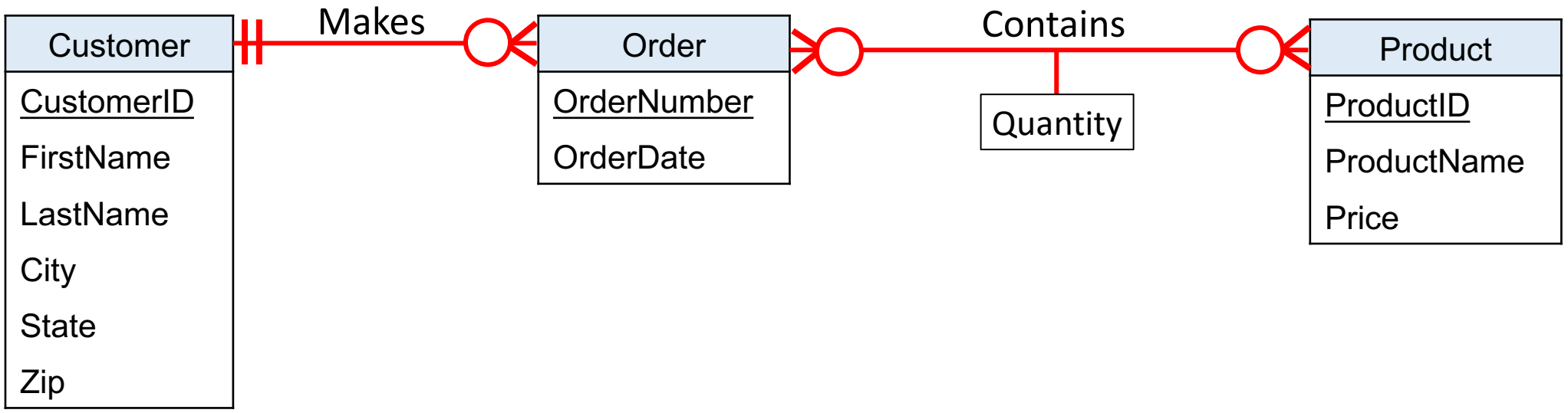
- Key concept
  - Entity sets become relations, Relationships can become relations (tables in RDBMS)
  - Tables are connected with foreign key constraints (~ 1:many relationships!)

- A database schema
  - A map of the tables and fields (attributes) in the database
  - This is what is implemented in the database management system
  - Part of the “design” process

From ERD to tables (Chen or "bubble" notation)



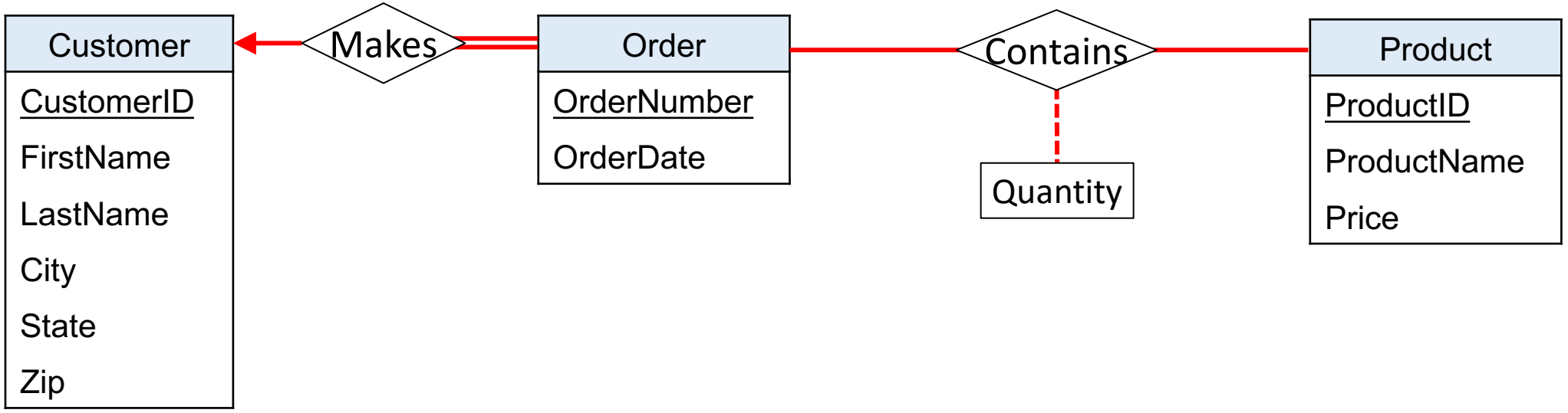
From ERD to tables (Crow-foot or "block" notation)



How to translate into our SDK textbook notation?

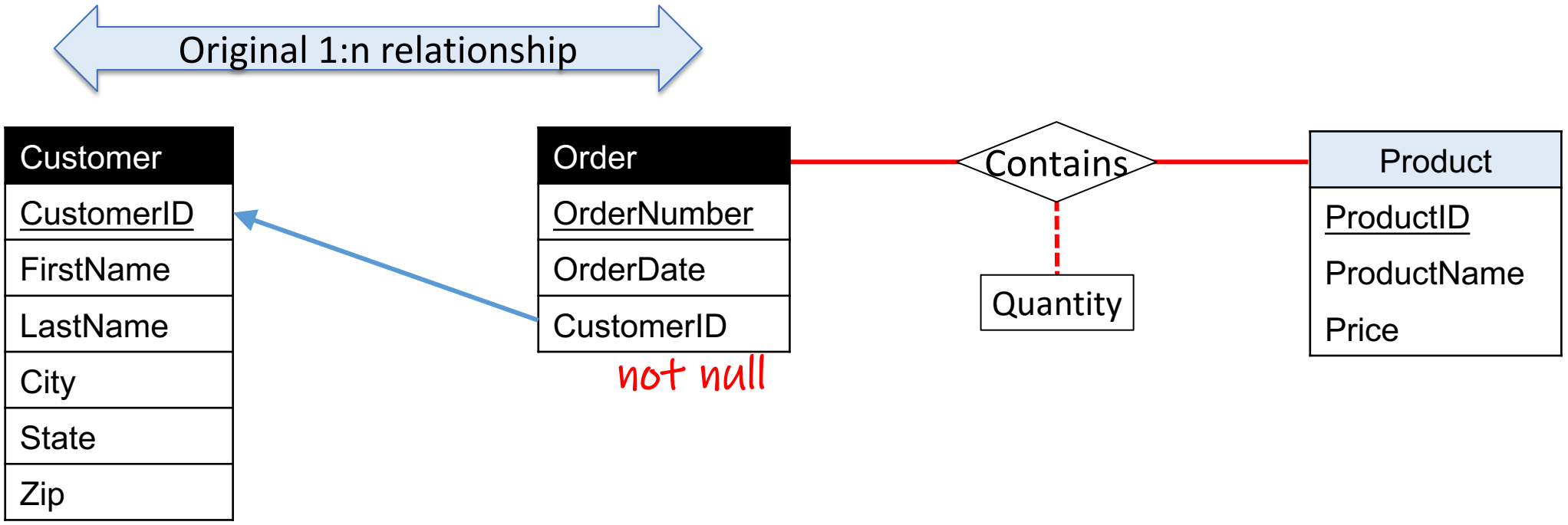
?

From ERD to tables (our textbok SDK notation)

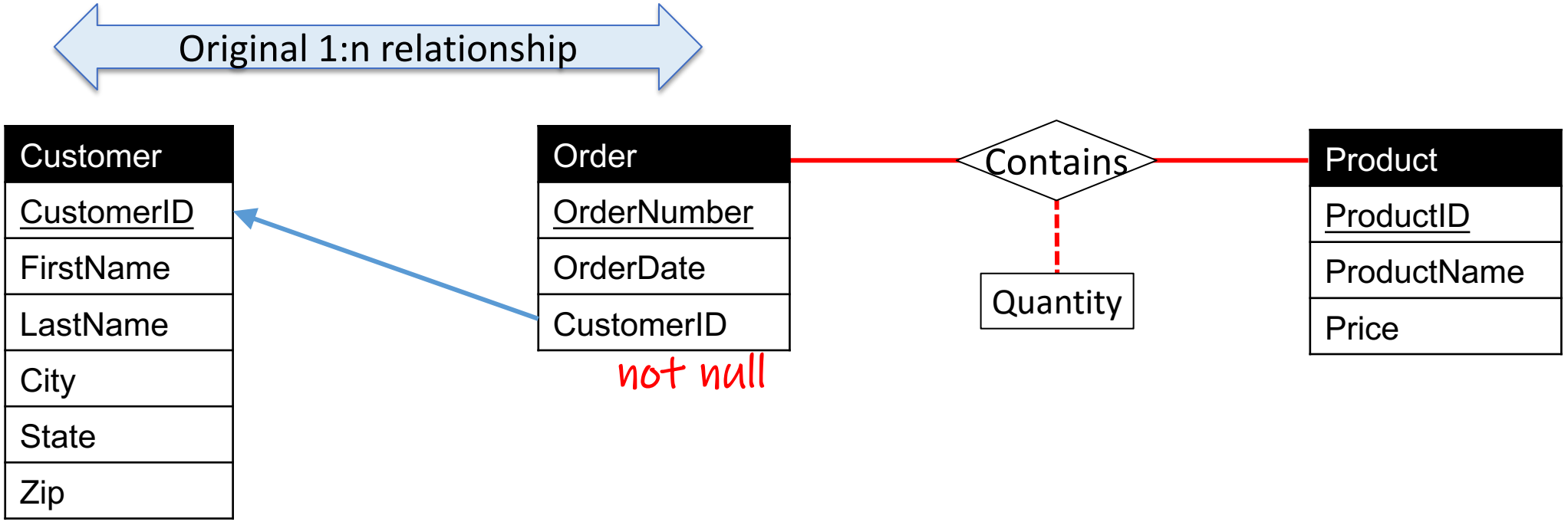


How to translate into tables ?

From ERD to tables (incomplete !)



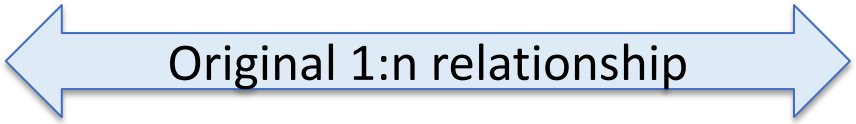
From ERD to tables (incomplete !)

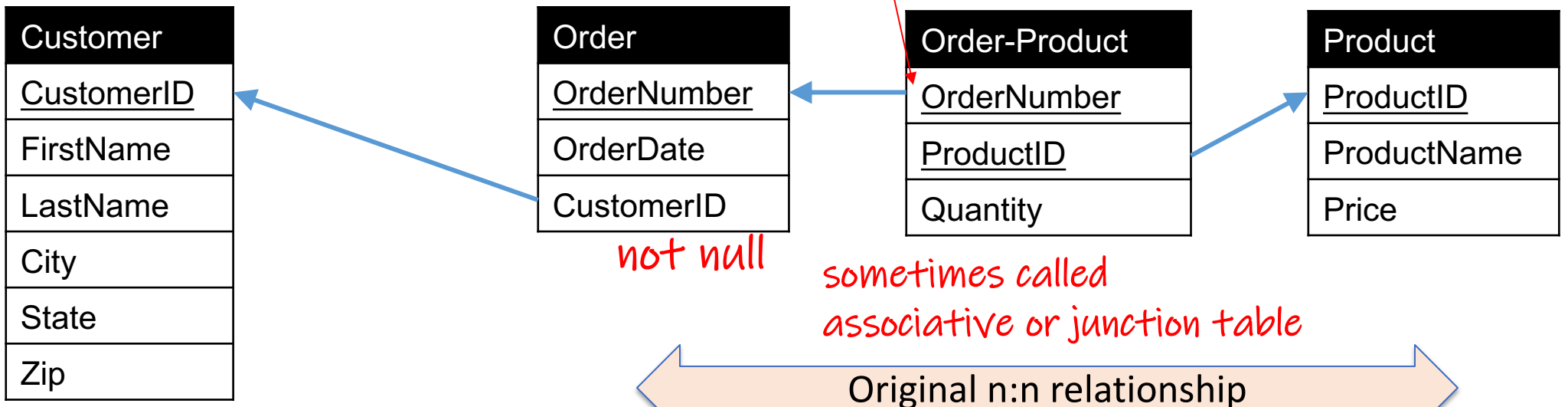


Relational schema

Which attributes are here FKs? ?

"not null" already take care off by being part of PK





- 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 FK are sometimes highlighted by an at sign (@) or (FK)

"not null" already take care off by being part of PK

Original 1:n relationship

Customer
CustomerID
FirstName
LastName
City
State
Zip

Order
OrderNumber
OrderDate
@CustomerID

not null

Order-Product
@OrderNumber
@ProductID
Quantity

Product
ProductID
ProductName
Price

sometimes called associative or junction table

Original n:n relationship

- 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 FK are sometimes highlighted by an at sign (@) or (FK)

"not null" already take care off by being part of PK

Original 1:n relationship

Customer
CustomerID
FirstName
LastName
City
State
Zip

Order
OrderNumber
OrderDate
CustomerID@

not null

Order-Product
OrderNumber@
ProductID@
Quantity

Product
ProductID
ProductName
Price

sometimes called associative or junction table

Original n:n relationship

- 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 FK are sometimes highlighted by an at sign (@) or (FK)

"not null" already take care off by being part of PK

Original 1:n relationship

Customer
CustomerID
FirstName
LastName
City
State
Zip

Order
OrderNumber
OrderDate
CustomerID (FK)

not null

Order-Product
OrderNumber (FK)
ProductID (FK)
Quantity

Product
ProductID
ProductName
Price

sometimes called associative or junction table

Original n:n relationship

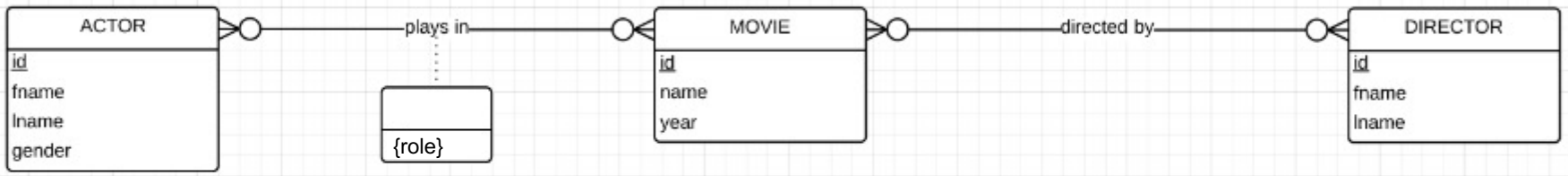
- Order-Product is a decomposed many-to-many relationship
  - Order-Product has a 1:n relationship with Order and Product
  - Now an order can have multiple products, and a product can be associated with multiple orders

The Rules

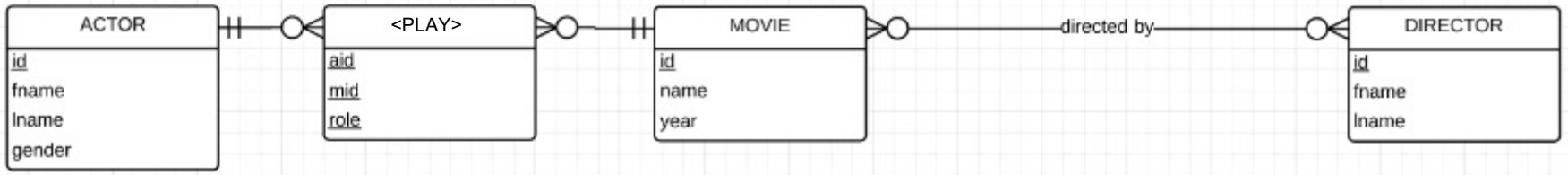
- Create a table for every **entity**
- Create table fields for every entity's **attributes**
- Implement **relationships** between the tables
  - 1:many relationships (1:1): primary key field of "1" table put into "many" (other) table as foreign key field
  - many:many relationships:
    - Create new table!
    - 1:many relationships with original table

Play table in our IMDB movie database

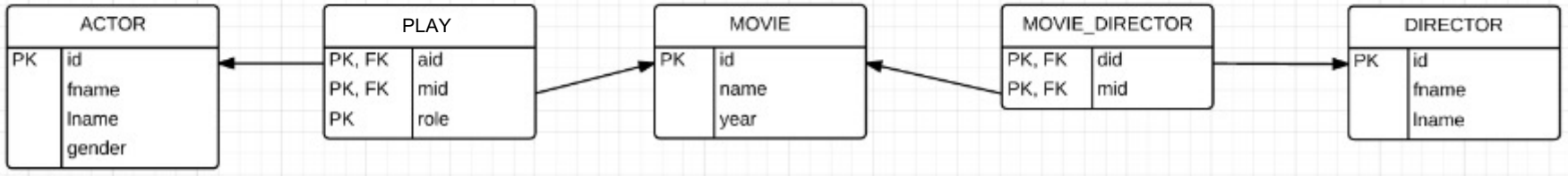
ER diagram: don't forget identifiers, but no FKs



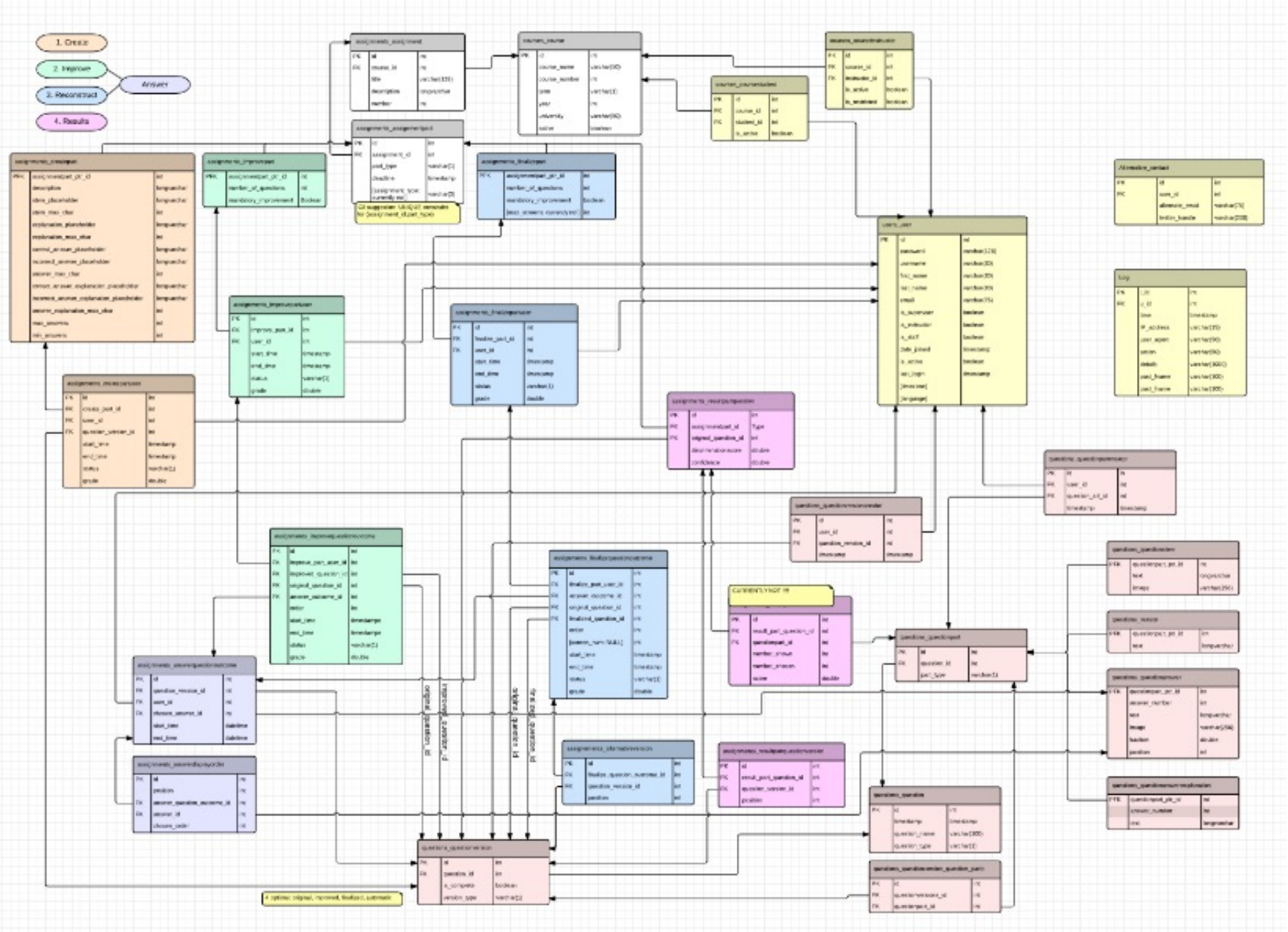
ER diagram: PLAY as associative entity can be justified



Relational schema: don't forget PKs and FKs



Why abstraction? Real relational schemas are huge



Details:
"Relational modeling":
From ERDs to Relations

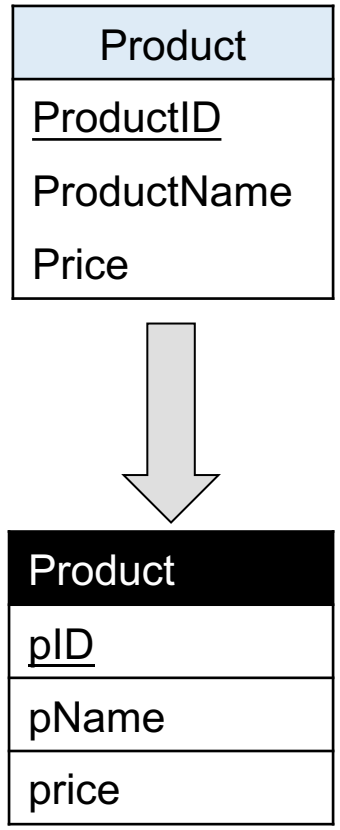
From ER Diagrams to Relational Schema

Product
ProductID
ProductName
Price

?

From ER Diagrams to Relational Schema

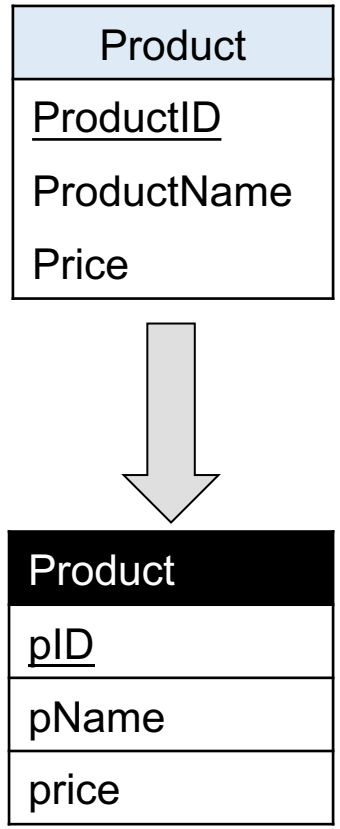
- An entity set becomes a table (= relation = multiset of tuples)
  - Each tuple is one entity
  - Each tuple is composed of the entity's attributes, and has the same PK (primary key)



From ER Diagrams to Relational Schema

```
CREATE TABLE Product(
pid INT PRIMARY KEY,
pname VARCHAR,
price decimal(8, 2))
```

```
CREATE TABLE Product(
pid INT,
pname VARCHAR,
price decimal(8, 2),
PRIMARY KEY (pid))
```

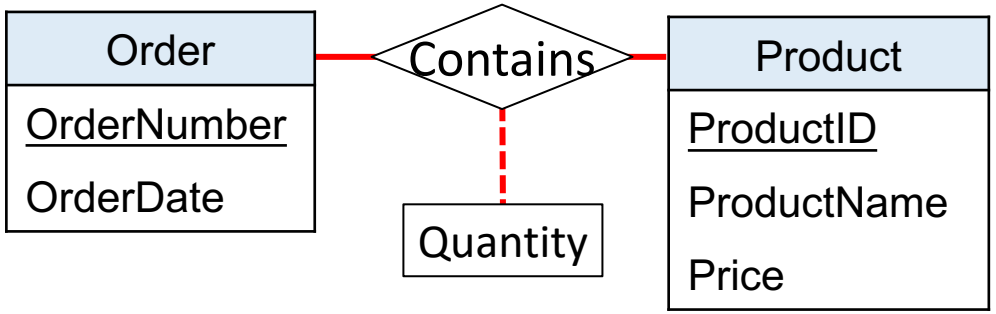



```
INSERT INTO Product VALUES
  (1, 'Gizmo', 19.99),
  (2, 'Supergizmo', 29.99)
```

Product

pid	pname	price
1	Gizmo	19.99
2	Supergizmo	29.99

From ER Diagrams to Relational Schema

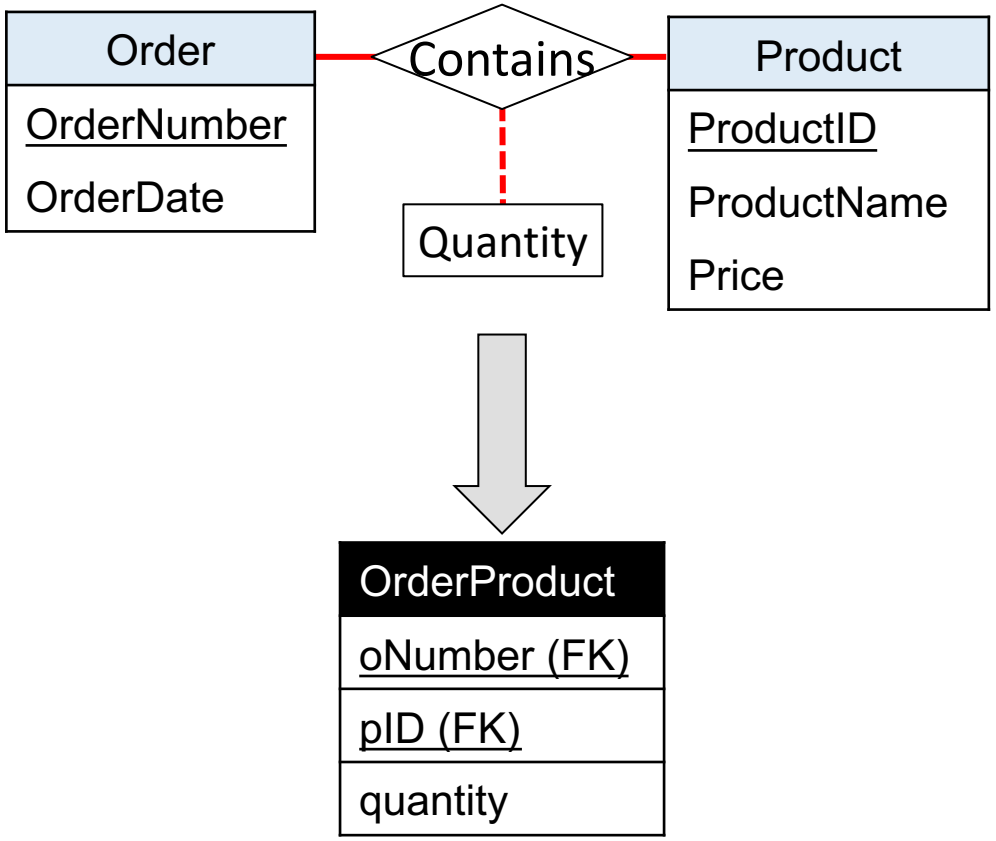


?

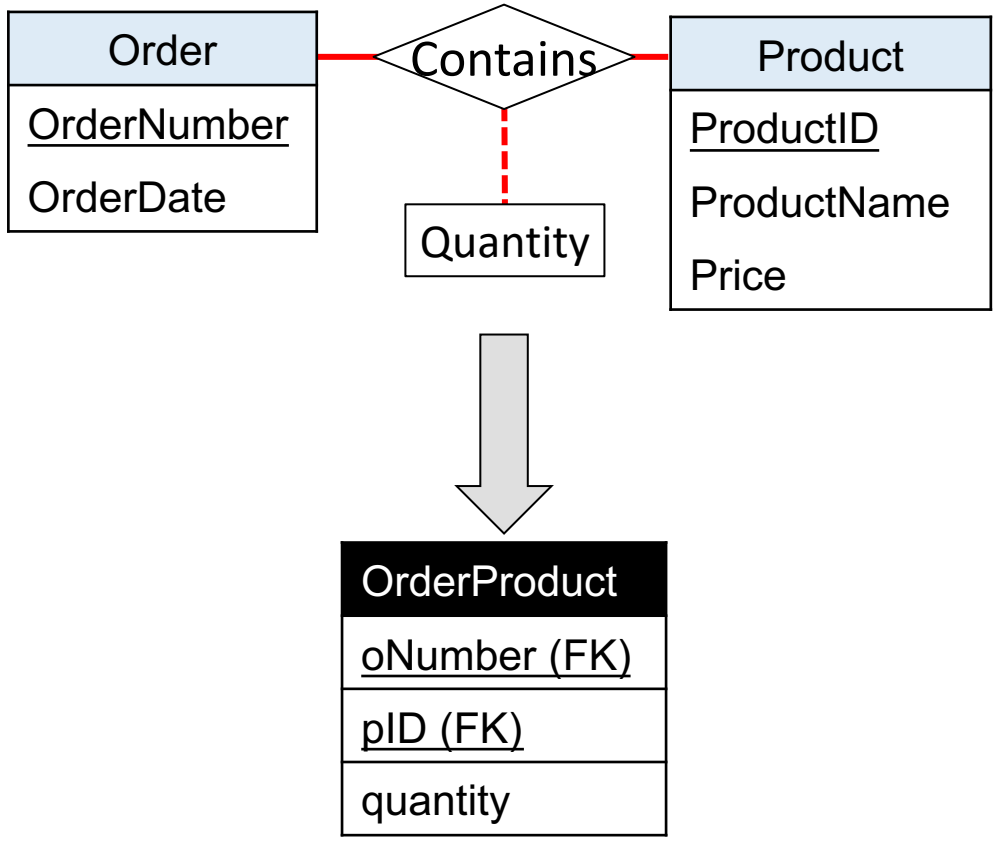
From ER Diagrams to Relational Schema

- A many-to-many relationship between entity sets A_1, …, A_N also becomes a table (i.e. a multiset of tuples)

- Each tuple in the table is one relation, i.e. one unique combination of entities $(a_1,…,a_N)$
  - attributes: union of the entity sets' PKs plus the attributes of the relationship
  - FKs (foreign keys): the entities' PKs (primary keys)
  - PK: union of the entity sets' PKs



From ER Diagrams to Relational Schema



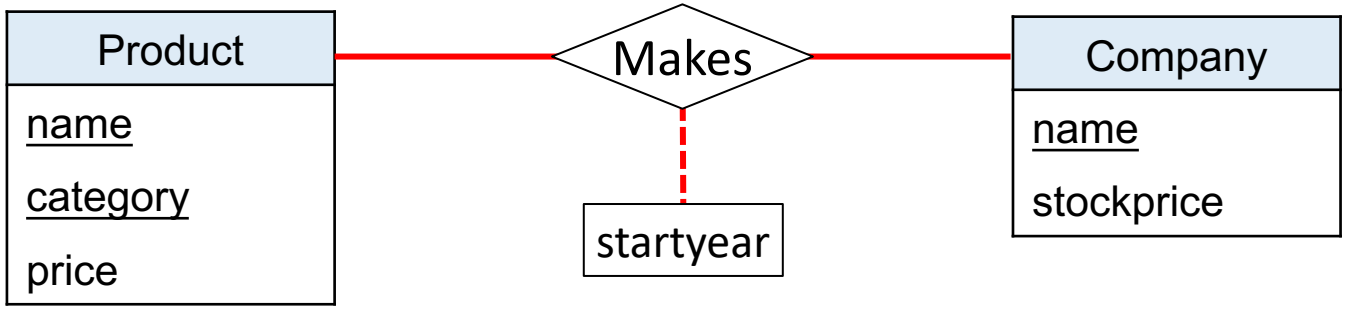

```
CREATE TABLE OrderProduct(
onumber INT,
pid INT,
quantity INT,
PRIMARY KEY (onumber, pid),
FOREIGN KEY (onumber) REFERENCES Order,
FOREIGN KEY (pid) REFERENCES Product)
```

```
INSERT INTO Ordernumber VALUES
  (456, 1, 2),
  (456, 2, 3)
```

OrderProduct

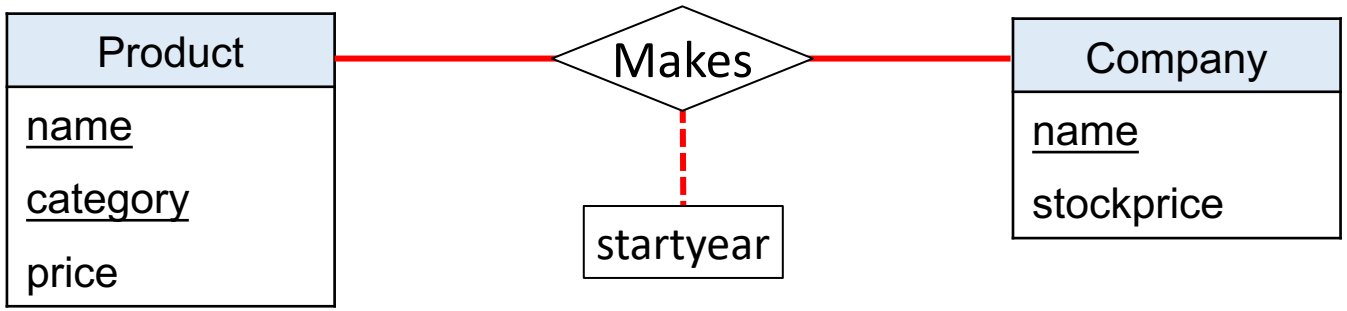
onumber	pid	quantity
456	1	2
456	2	3

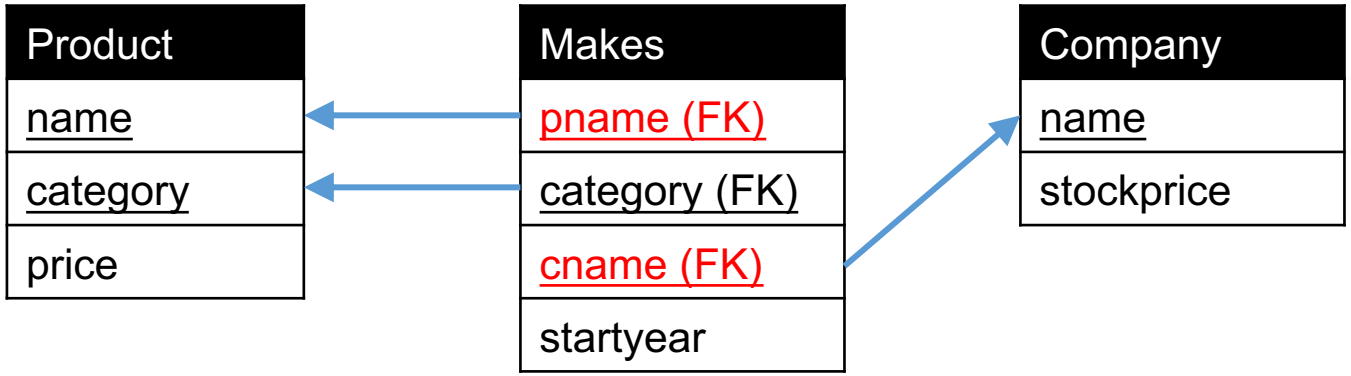
Relationships to Relations



?

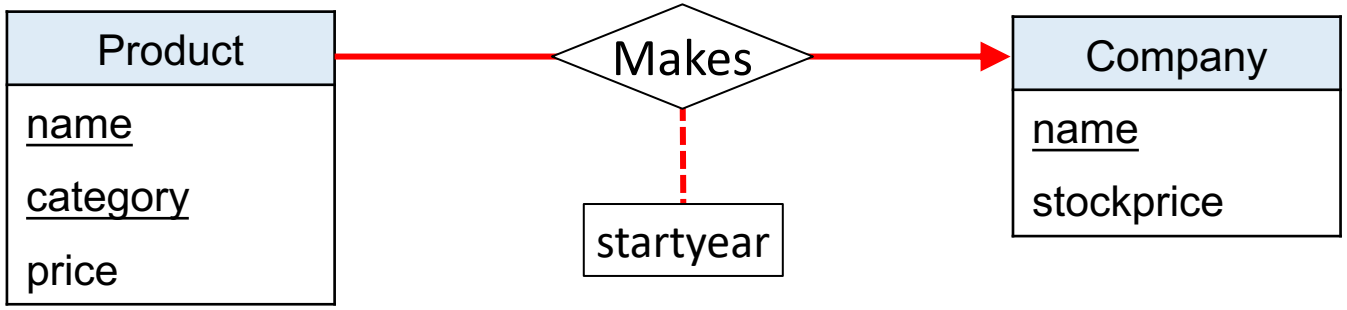
Relationships to Relations





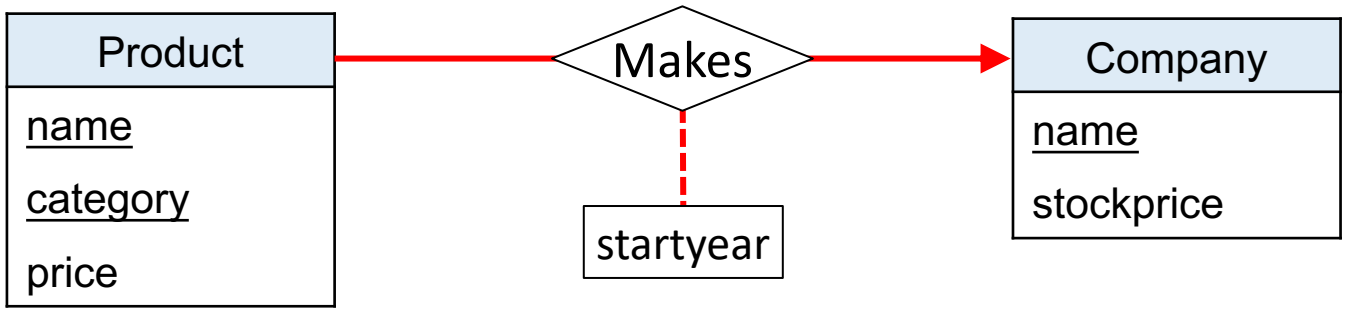
Watch out for attribute name conflicts! Need to rename the FKs.

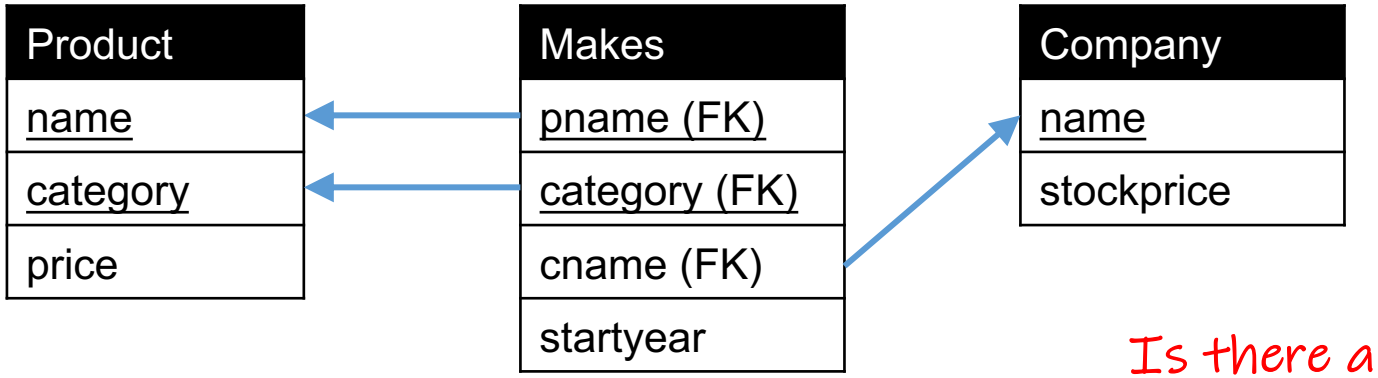
Relationships to Relations



?

Relationships to Relations

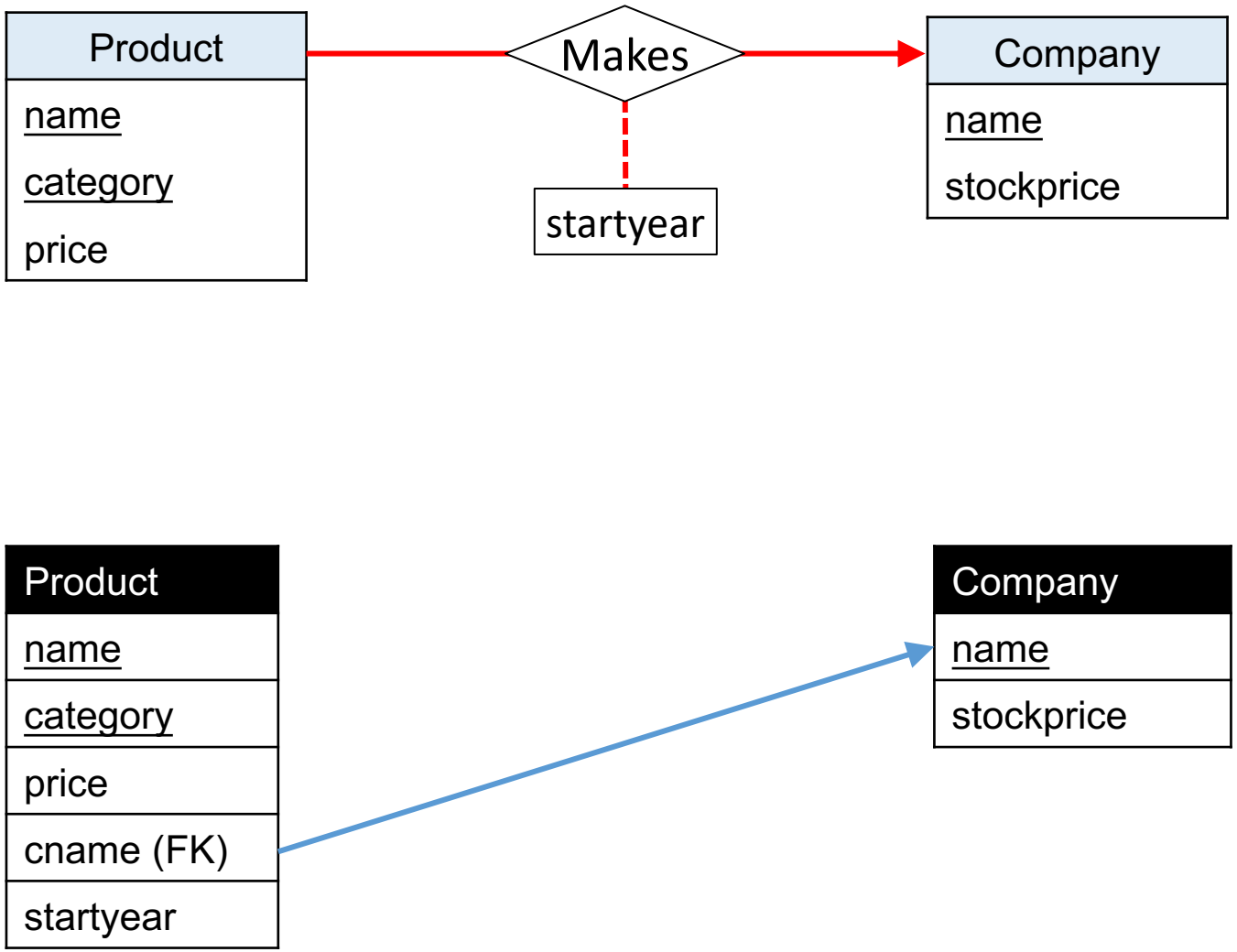




Is there a better way ?

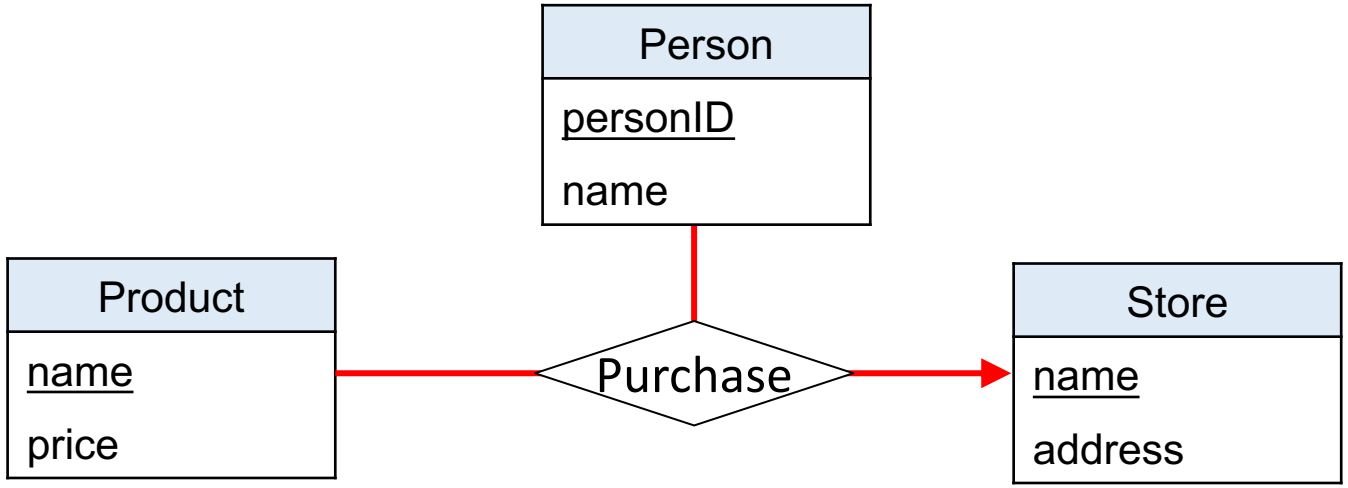
Only keep Product PK as PK

Relationships to Relations



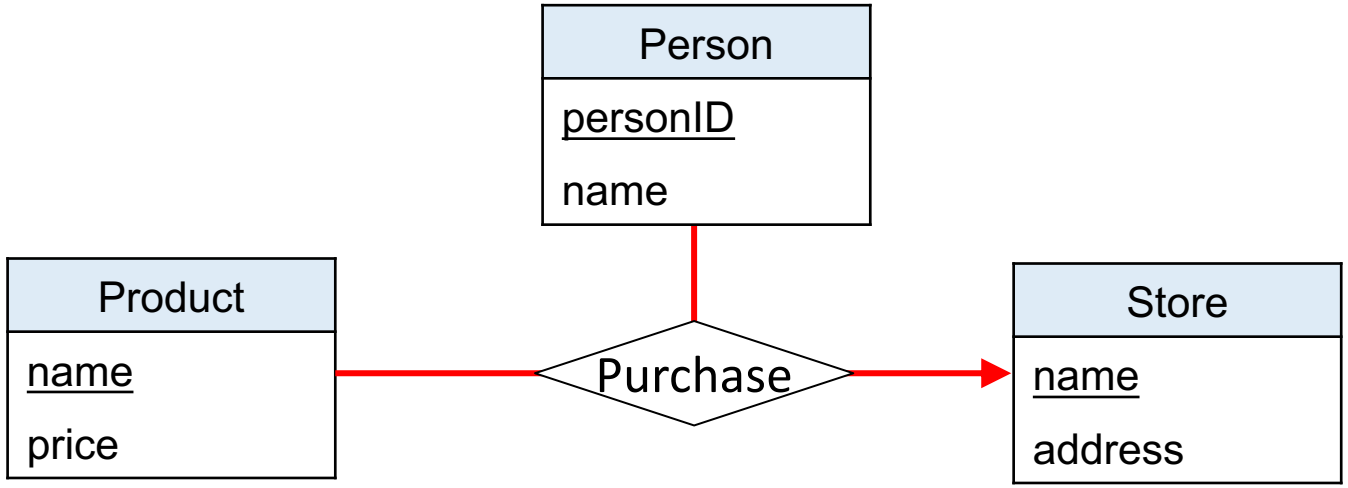
Correct solution: get rid of Makes and add FK and startyear to Product

Multi-way relationships to Relations



?

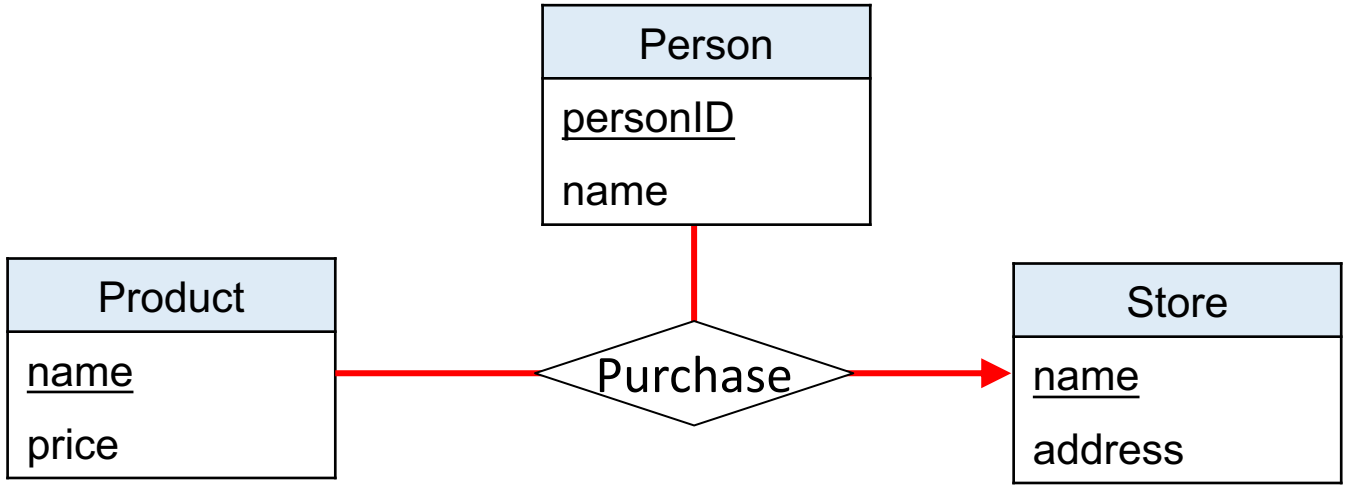
Multi-way relationships to Relations



Purchase
pname (FK)
pid (FK)
sname (FK)

What should be the PK? ?

Multi-way relationships to Relations



Purchase
pname (FK)
pid (FK)
sname (FK)

Notice the composite PK

Exercise (Part 2 & 3): create an ERD & relations

The following grade report below is mailed to students of Millennium College at the end of each semester. Prepare an ERD reflecting the data contained in the grade report (capturing Entities, Attributes, and Relationships). Assume that each course is taught by one instructor and a student can take a class once. Include PKs for each entity type.

MILLENNIUM COLLEGE
GRADE REPORT
FALL SEMESTER 200X

NAME: Emily Williams ID: 268300458
CAMPUS ADDRESS: 208 Brooks Hall
MAJOR: Information Systems

COURSE ID	TITLE	INSTRUCTOR NAME	INSTRUCTOR LOCATION	GRADE
IS 350	Database Mgt.	Codd	B104	A
IS 465	System Analysis	Parsons	B317	B

?

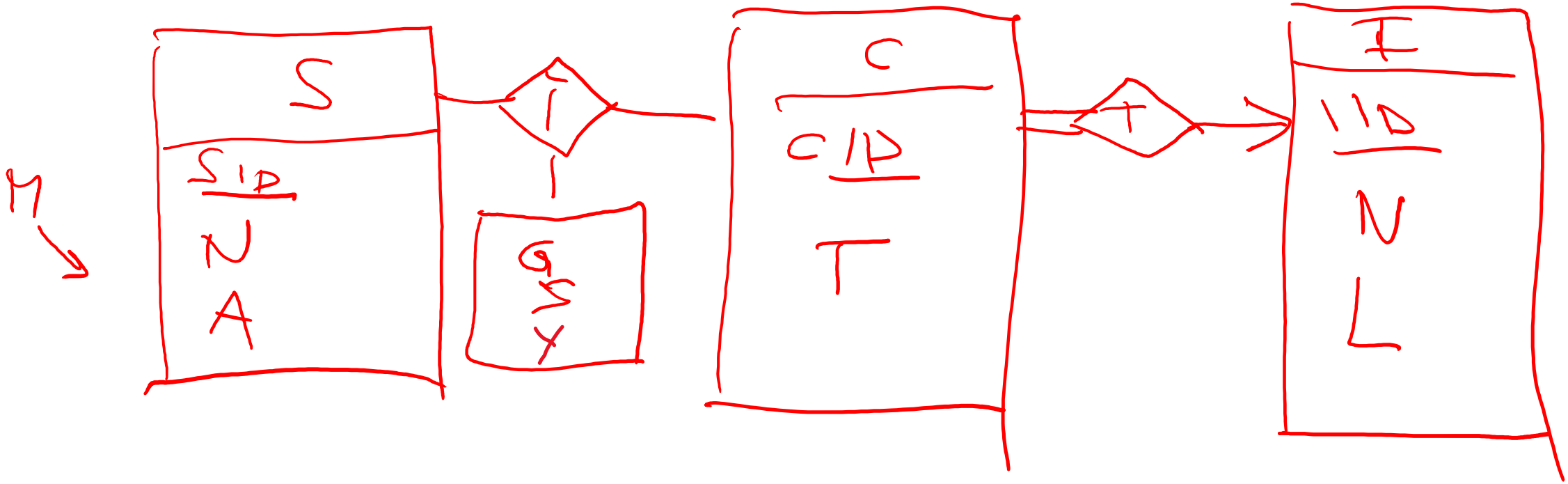
Source: Hoffer, Ramesh, Topi, "Modern database management," 10th ed, 2010.
Wolfgang Gatterbauer. Database design: https://northeastern-datalab.github.io/cs3200/

Exercise (Part 2): create an ERD

MILLENNIUM COLLEGE
GRADE REPORT
FALL SEMESTER 200X

NAME: Emily Williams ID: 268300458
CAMPUS ADDRESS: 208 Brooks Hall
MAJOR: Information Systems

COURSE ID	TITLE	INSTRUCTOR NAME	INSTRUCTOR LOCATION	GRADE
IS 350	Database Mgt.	Codd	B104	A
IS 465	System Analysis	Parsons	B317	B



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

Exercise (Part 2): create an ERD

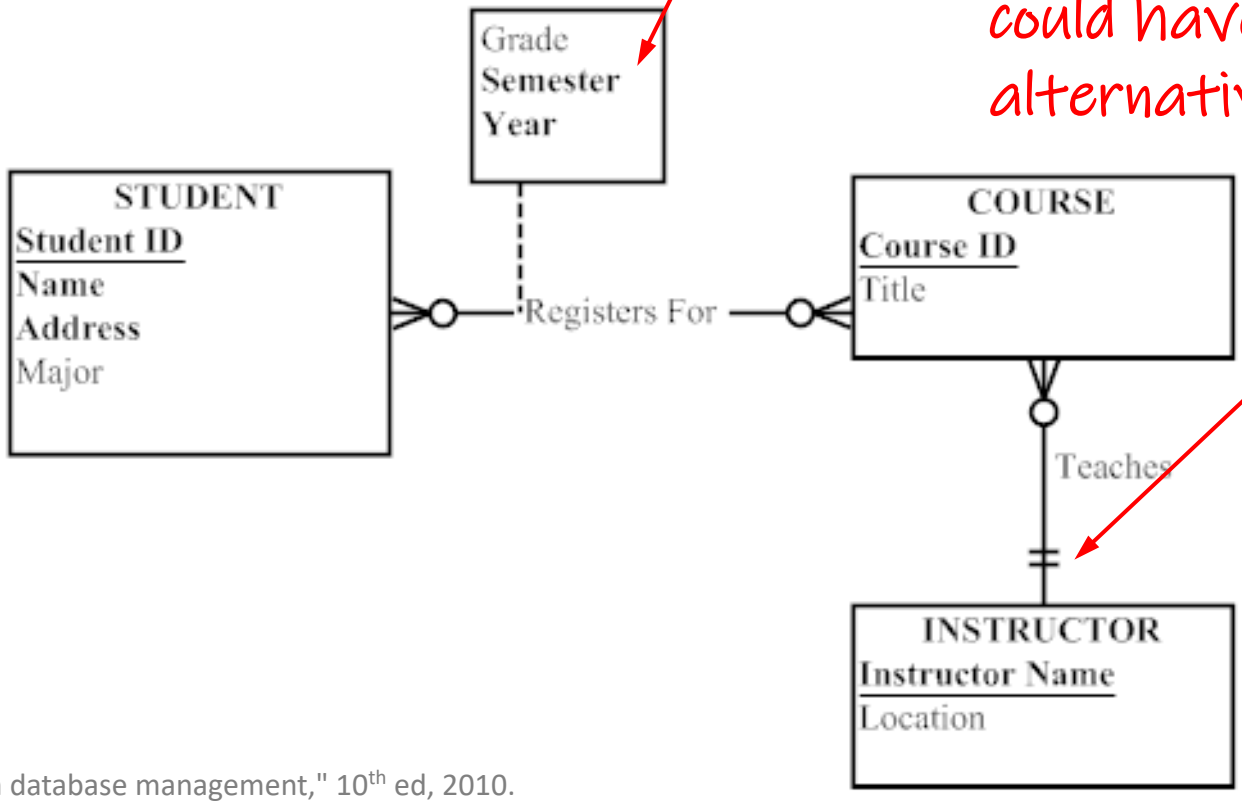
MILLENNIUM COLLEGE
GRADE REPORT
FALL SEMESTER 200X

NAME: Emily Williams ID: 268300458
CAMPUS ADDRESS: 208 Brooks Hall
MAJOR: Information Systems

COURSE ID	TITLE	INSTRUCTOR NAME	INSTRUCTOR LOCATION	GRADE
IS 350	Database Mgt.	Codd	B104	A
IS 465	System Analysis	Parsons	B317	B

Hoffer notation: bold for necessary attributes (not NULL), non-bold for optional ones

If we had not specified that a course is taught by 1 instructor, it could have multiple sections, then alternatively many-to-many possible



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

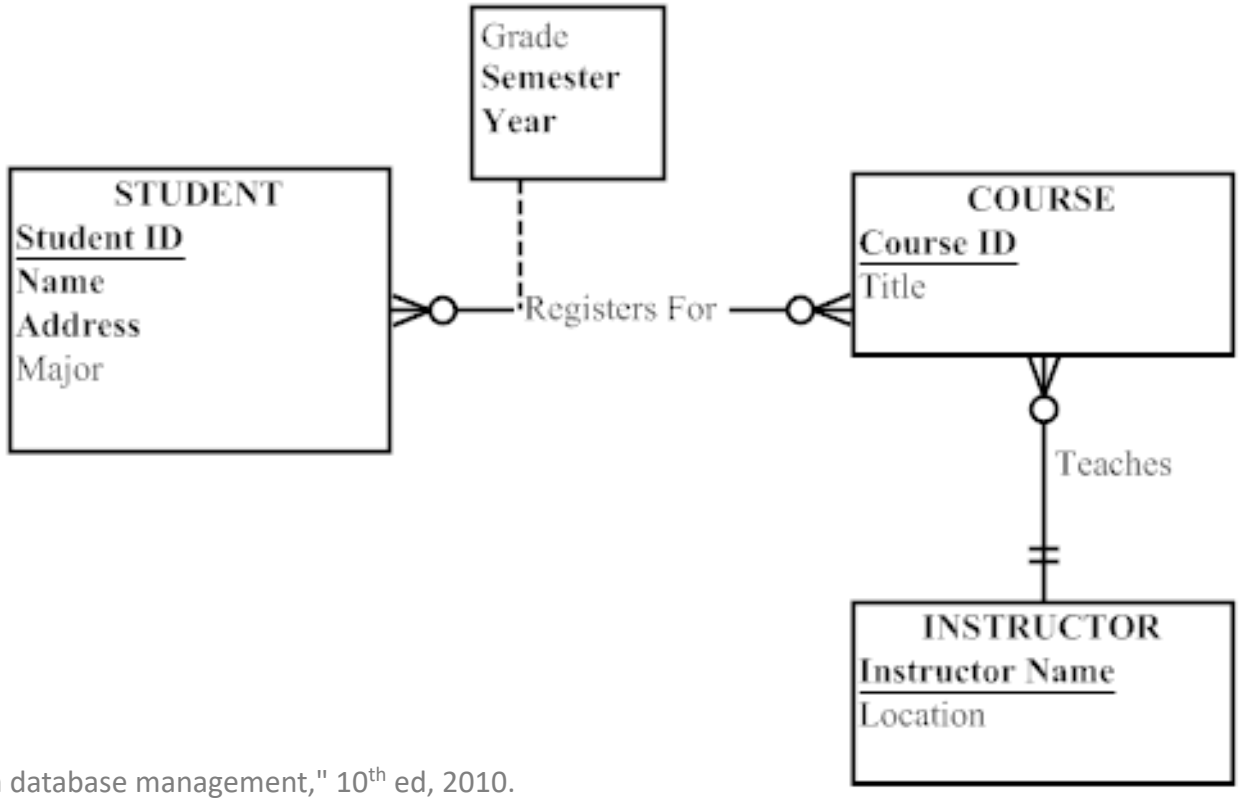
Exercise (Part 3): create relations

MILLENNIUM COLLEGE
GRADE REPORT
FALL SEMESTER 200X

NAME: Emily Williams ID: 268300458
CAMPUS ADDRESS: 208 Brooks Hall
MAJOR: Information Systems

COURSE ID	TITLE	INSTRUCTOR NAME	INSTRUCTOR LOCATION	GRADE
IS 350	Database Mgt.	Codd	B104	A
IS 465	System Analysis	Parsons	B317	B

Translate into relatoin. Also pay attention to which FKs can be NULL ?



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

In-Class Exercise (Part III): relational schema

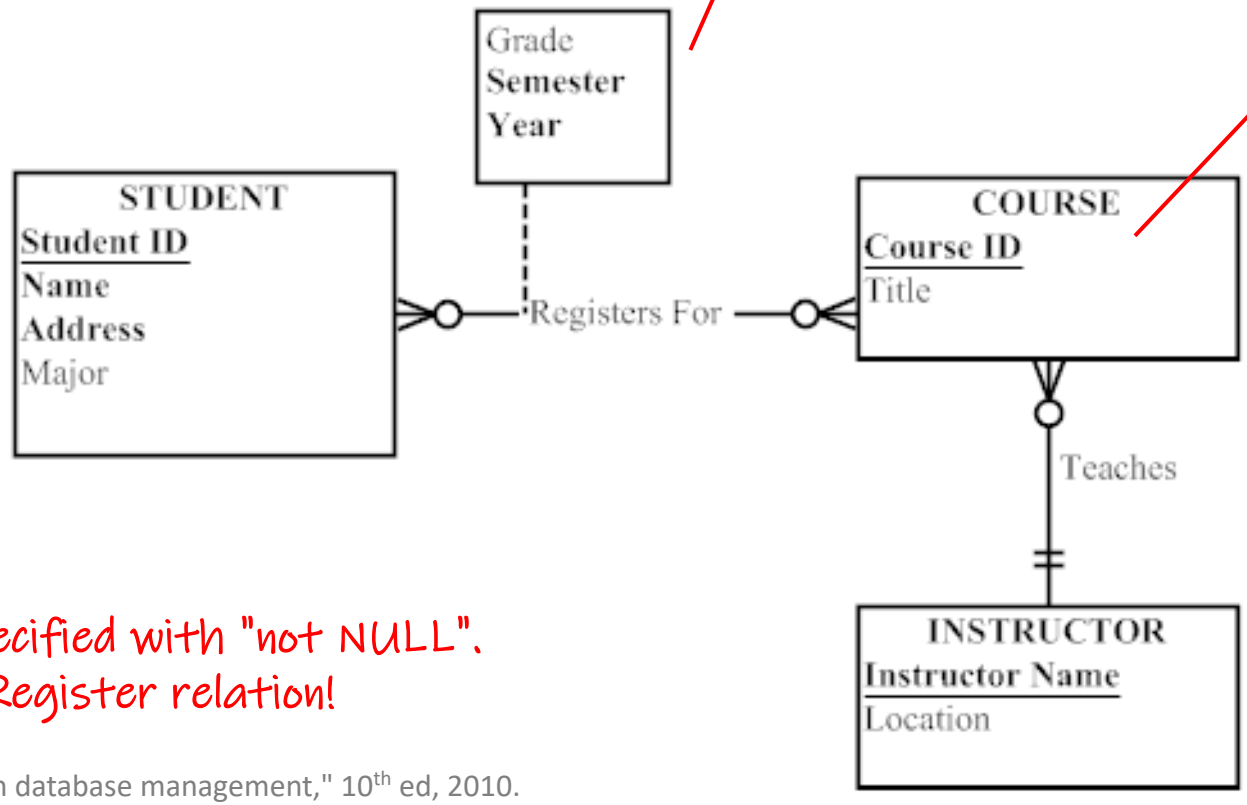
MILLENNIUM COLLEGE
GRADE REPORT
FALL SEMESTER 200X

NAME: Emily Williams ID: 268300458
CAMPUS ADDRESS: 208 Brooks Hall
MAJOR: Information Systems

COURSE ID	TITLE	INSTRUCTOR NAME	INSTRUCTOR LOCATION	GRADE
IS 350	Database Mgt.	Codd	B104	A
IS 465	System Analysis	Parsons	B317	B

Registration(@sid, @cid, semester, year, grade)

Course(cid, title, @instructor_name)



All FKs are specified with "not NULL".
Even for the Register relation!

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

In-Class Exercise (Part III): relational schema

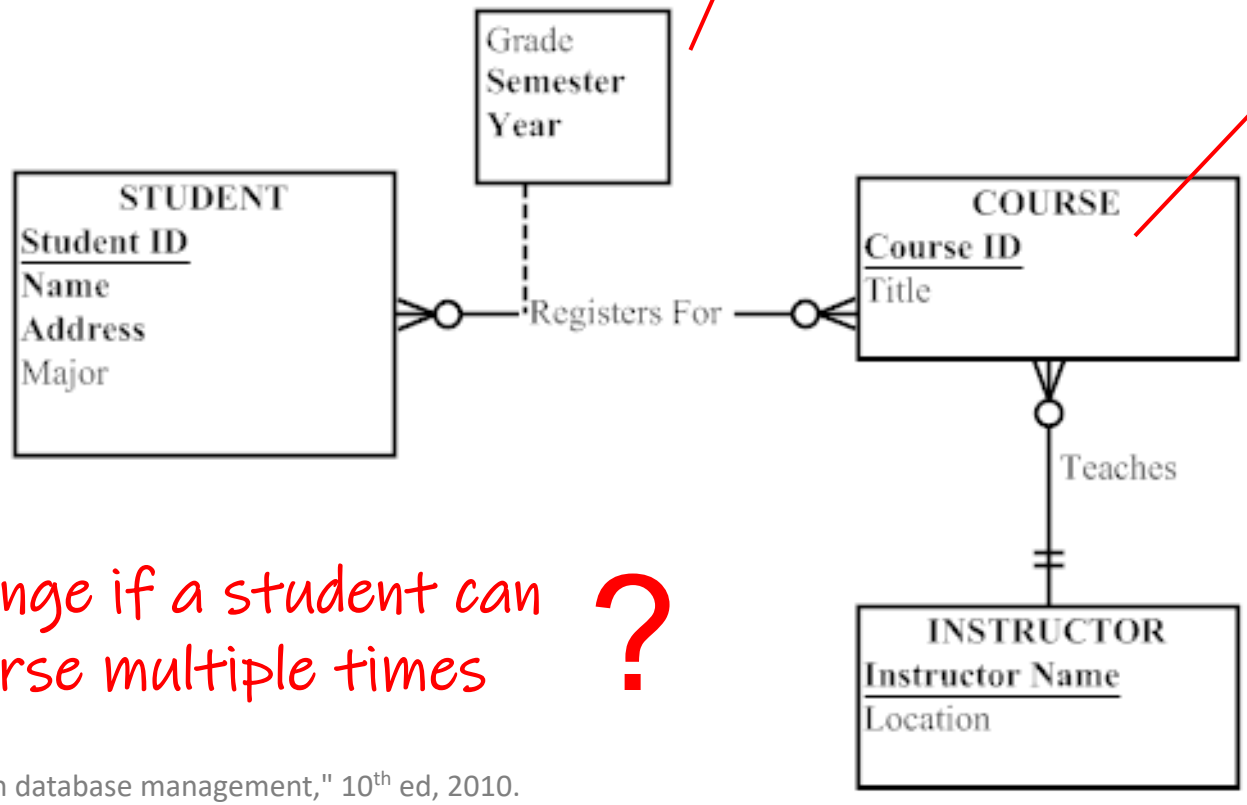
MILLENNIUM COLLEGE
GRADE REPORT
FALL SEMESTER 200X

NAME: Emily Williams ID: 268300458
CAMPUS ADDRESS: 208 Brooks Hall
MAJOR: Information Systems

COURSE ID	TITLE	INSTRUCTOR NAME	INSTRUCTOR LOCATION	GRADE
IS 350	Database Mgt.	Codd	B104	A
IS 465	System Analysis	Parsons	B317	B

Registration(@sid, @cid, semester, year, grade)

Course(cid, title, @instructor_name)



What needs to change if a student can take the same course multiple times ?

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

In-Class Exercise (Part III): relational schema

MILLENNIUM COLLEGE
GRADE REPORT
FALL SEMESTER 200X

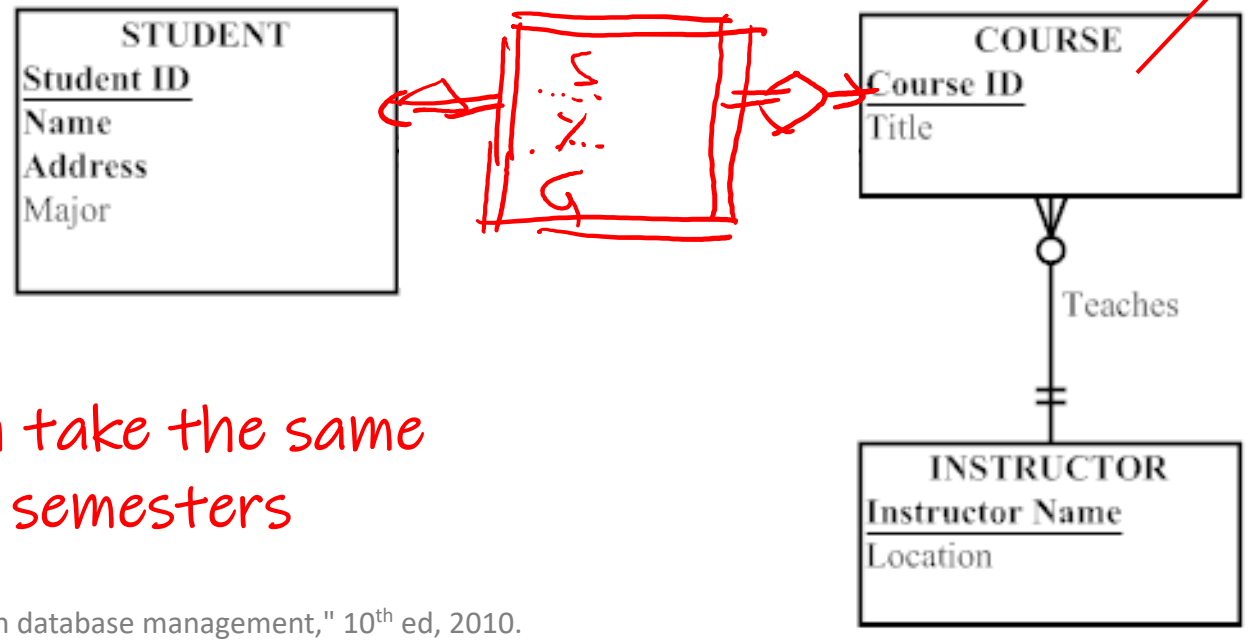
NAME: Emily Williams ID: 268300458
CAMPUS ADDRESS: 208 Brooks Hall
MAJOR: Information Systems

COURSE ID	TITLE	INSTRUCTOR NAME	INSTRUCTOR LOCATION	GRADE
IS 350	Database Mgt.	Codd	B104	A
IS 465	System Analysis	Parsons	B317	B

Registration(@sid, @cid, semester, year, grade)

Course(cid, title, @instructor_name)

or with multivalued attribute on relation!



Now a student can take the same course in different semesters

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

More Practice

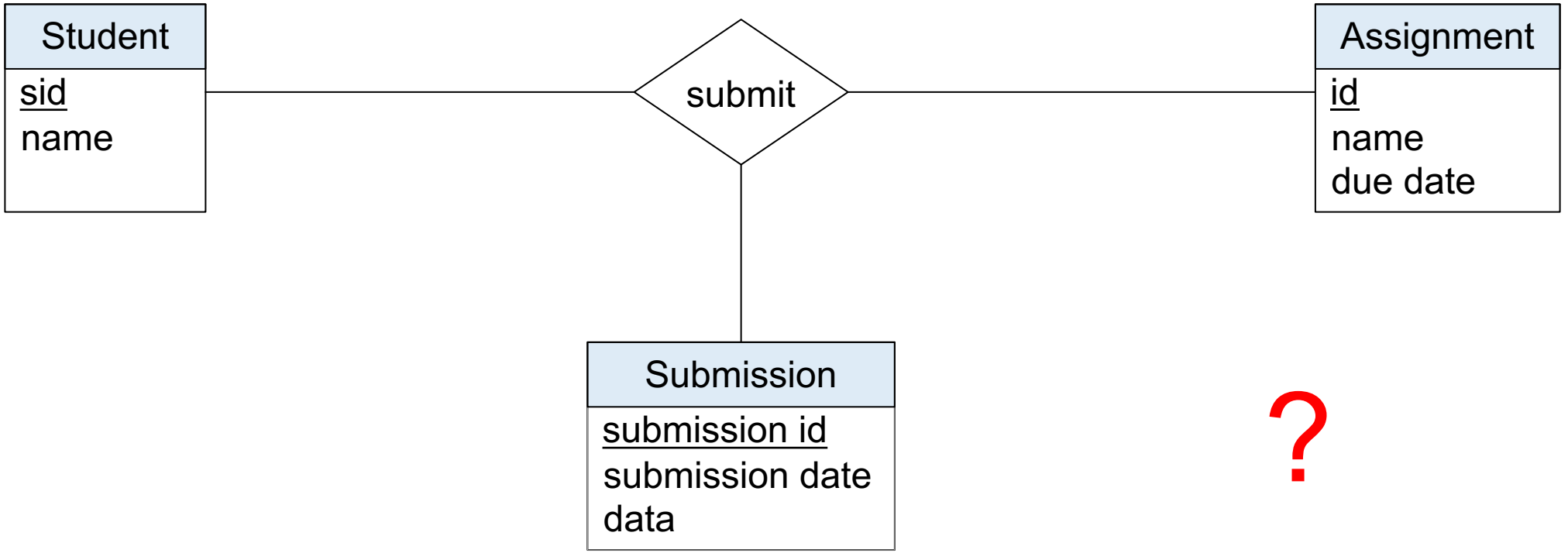
Example ERD

- Students (id, name) submit homework submissions (date, some "data") for assignments (id, name, due date)
- Students can submit multiple versions

?

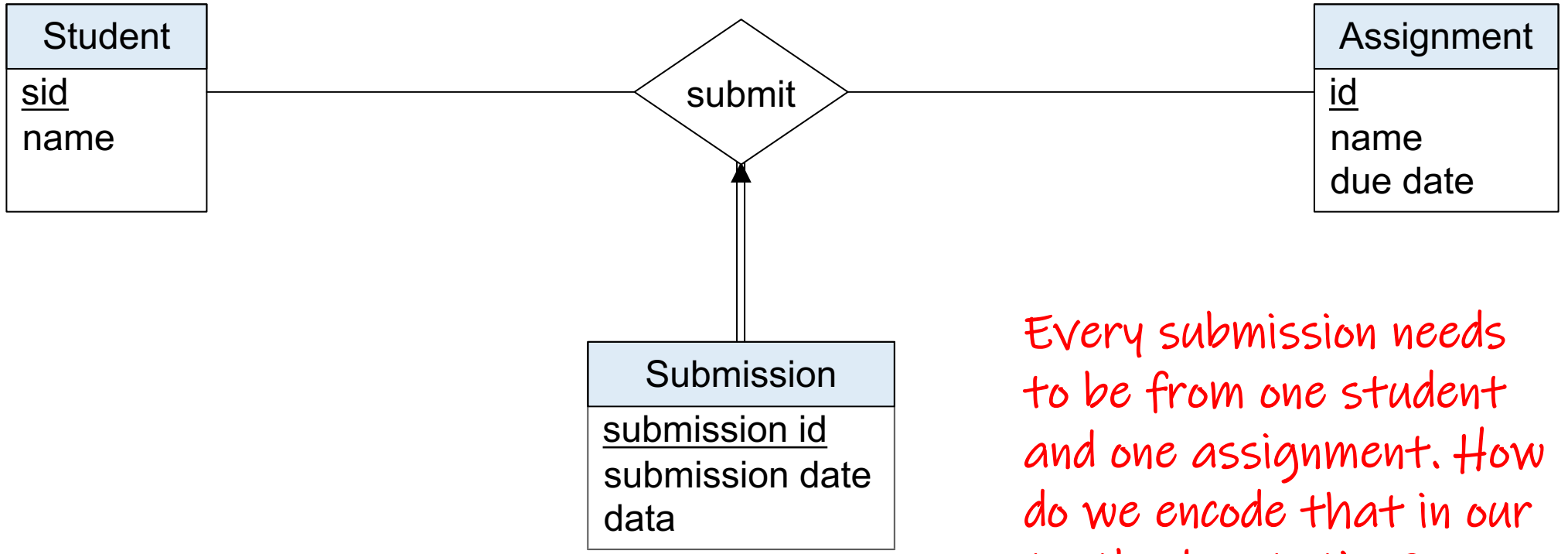
Example ERD

- Students (id, name) submit homework submissions (date, some "data") for assignments (id, name, due date)
- Students can submit multiple versions



Example ERD

- Students (id, name) submit homework submissions (date, some "data") for assignments (id, name, due date)
- Students can submit multiple versions

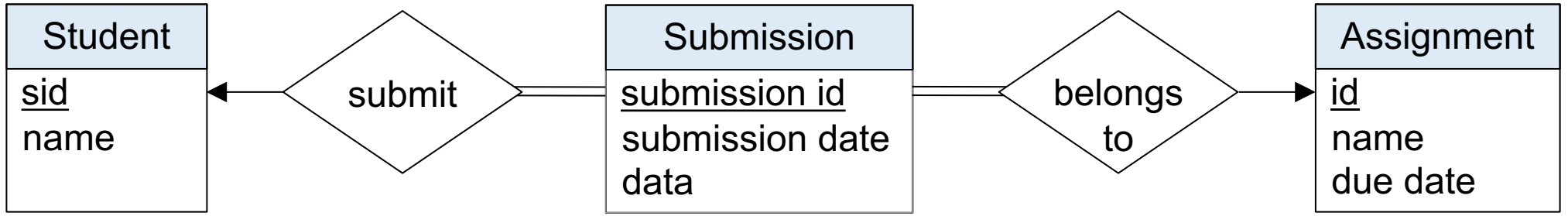


Every submission needs to be from one student and one assignment. How do we encode that in our textbook notation?

?

Example ERD

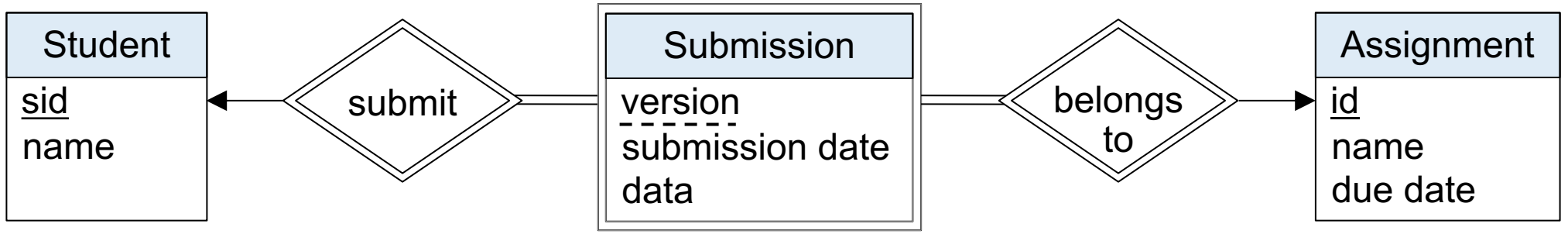
- Students (id, name) submit homework submissions (date, some "data") for assignments (id, name, due date)
- Students can submit multiple versions



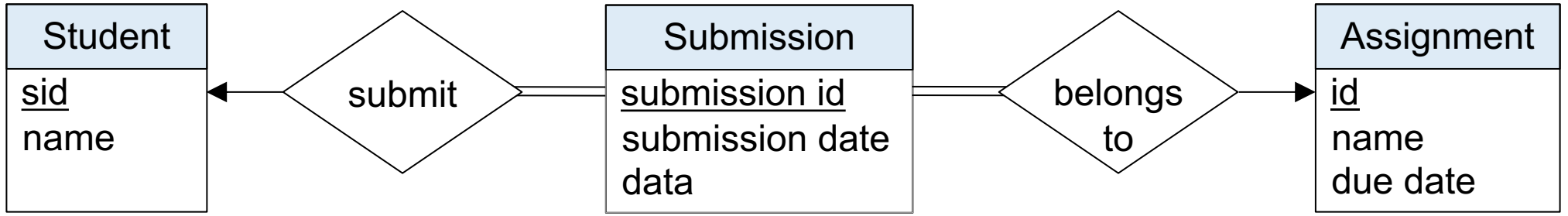
Assume we want to keep explicity submission versions (1, 2, 3, ...) instead of completely new ids ?

Example ERD

- Students (id, name) submit homework submissions (date, some "data") for assignments (id, name, due date)
- Students can submit multiple versions

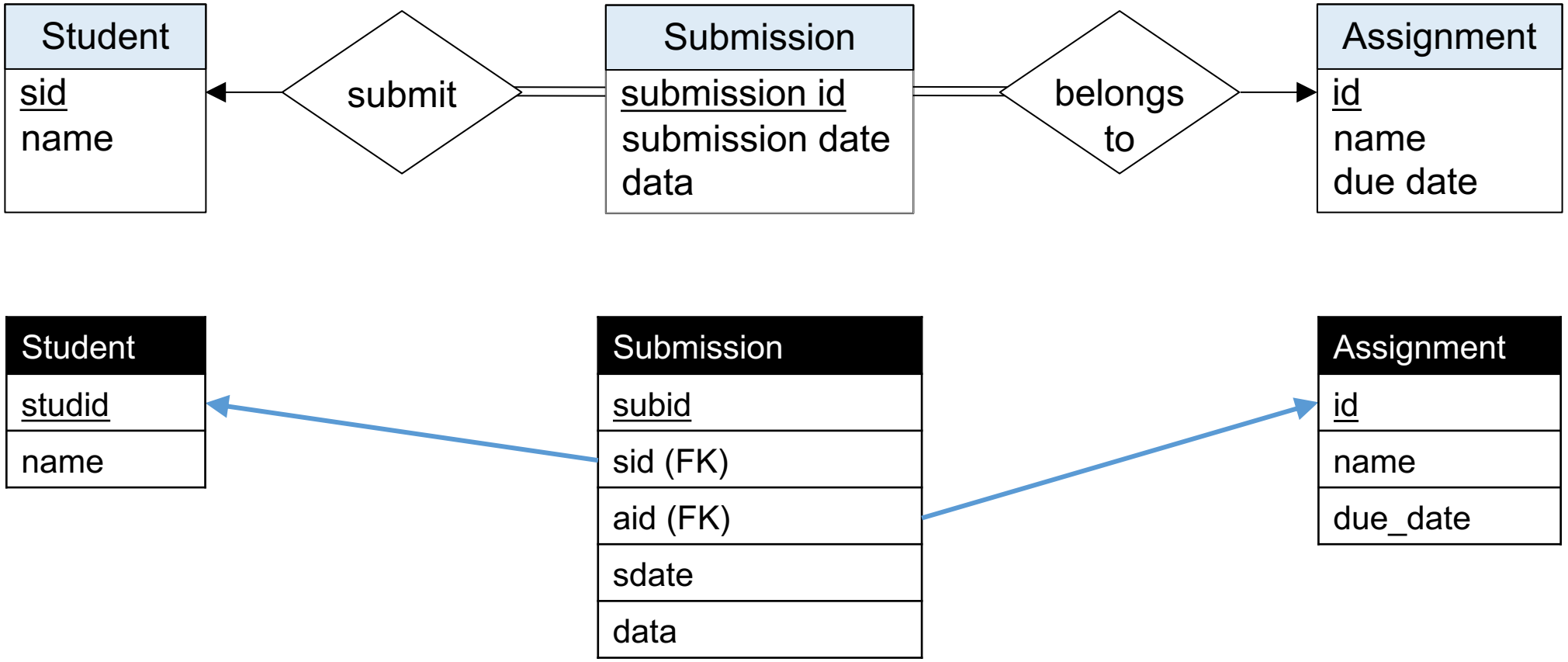


Example ERD to relations

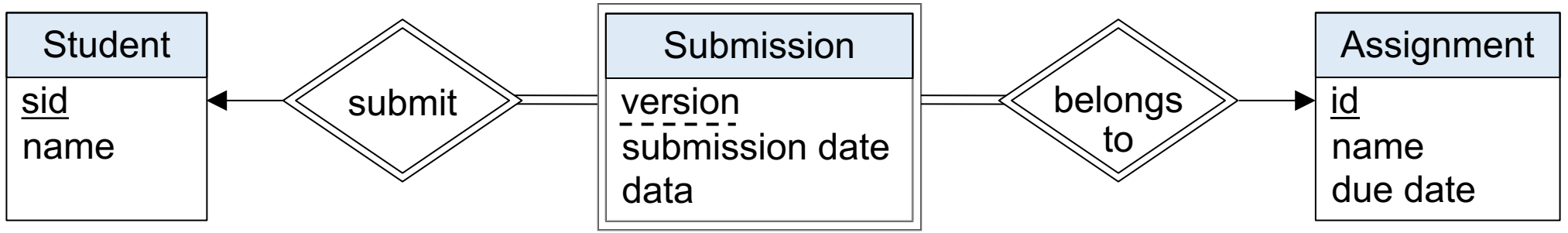


?

Example ERD to relations



Example ERD to relations



?

Example ERD to relations