E-R Diagram: Subclass, E-R to relational design
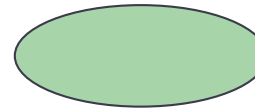
CS 4750
Database Systems

[A. Silberschatz, H. F. Korth, S. Sudarshan, Database System Concepts, Ch.6]

[C.M. Ricardo and S.D. Urban, Database Illuminated, Ch.3]
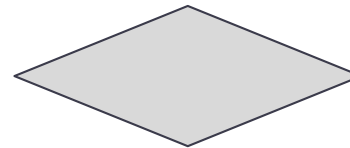
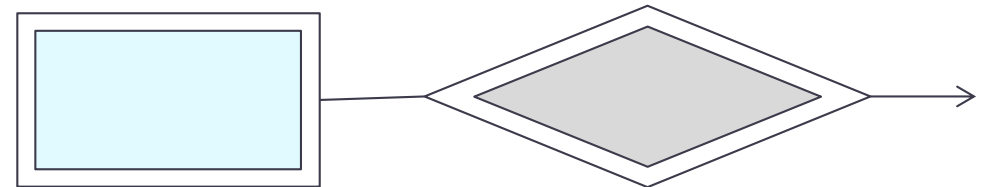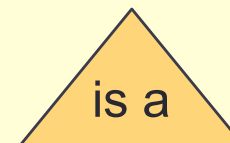# E-R Diagram: Building Blocks

(strong) Entity set
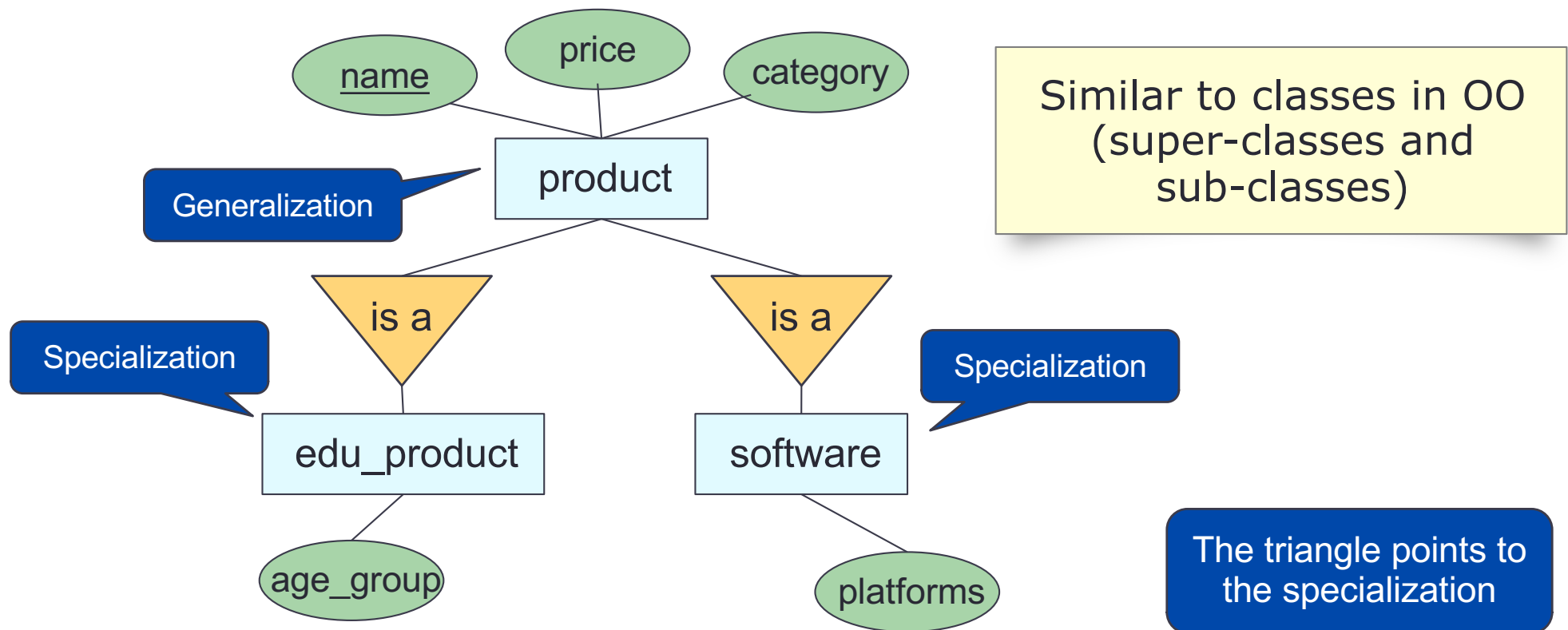
Attribute

Relationship

Weak entity

Subclass

is a

Note: colors are not part of E-R Diagram. They simply are used to increase readability.
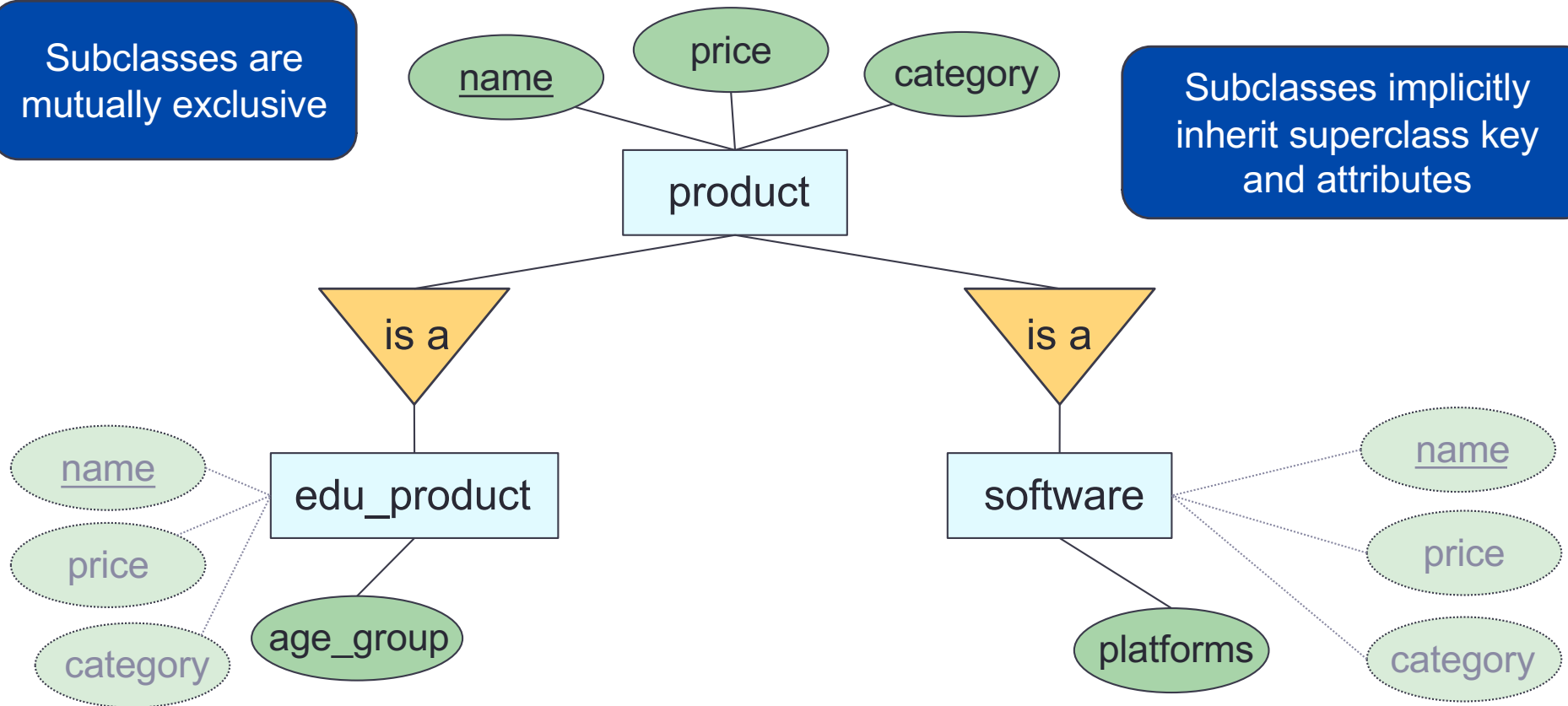
Subclassing

- An entity set may contain entities that have special properties not associated with all members of the set

- Subclasses ~ special-case entity sets

- Isa (or Is-a) ~ special kind of relationship (one-to-one)

Similar to classes in OO (super-classes and sub-classes)

The triangle points to the specialization

Subclassing

Subclasses are mutually exclusive

Subclasses implicitly inherit superclass key and attributes

name

price

category

product

is a

is a

name

price

category

edu_product

software

name

price

category

age_group

platforms

Generalization / Specialization
The triangle points to the specialization

Let's try: Subclassing (Movies)

Consider the following (partial) description of a movie scenario.

Each movie has a *title* and *year*; *title* and *year* together uniquely identify the movie. *Length* and *genre* are maintained for each movie.

Among the special kinds of movies, we might store in our database are cartoons and murder mysteries.

A cartoon has, in addition to the attributes and relationships of *Movies*, an additional relationship called *Voices* that gives us a set of stars who speak, but do not appear in the movie. Movies that are not cartoons do not have such stars.

Murder-mysteries have an additional attribute *weapon*.

Draw an E-R diagram to show the connections among the three entity sets: *Movies, Cartoons,* and *Murder-Mysteries.*

Let's try: Subclassing (Movies)

Draw an E-R diagram to show the connections among the 3 entity sets: *Movies, Cartoons,* and *Murder-Mysteries* (from previous page)
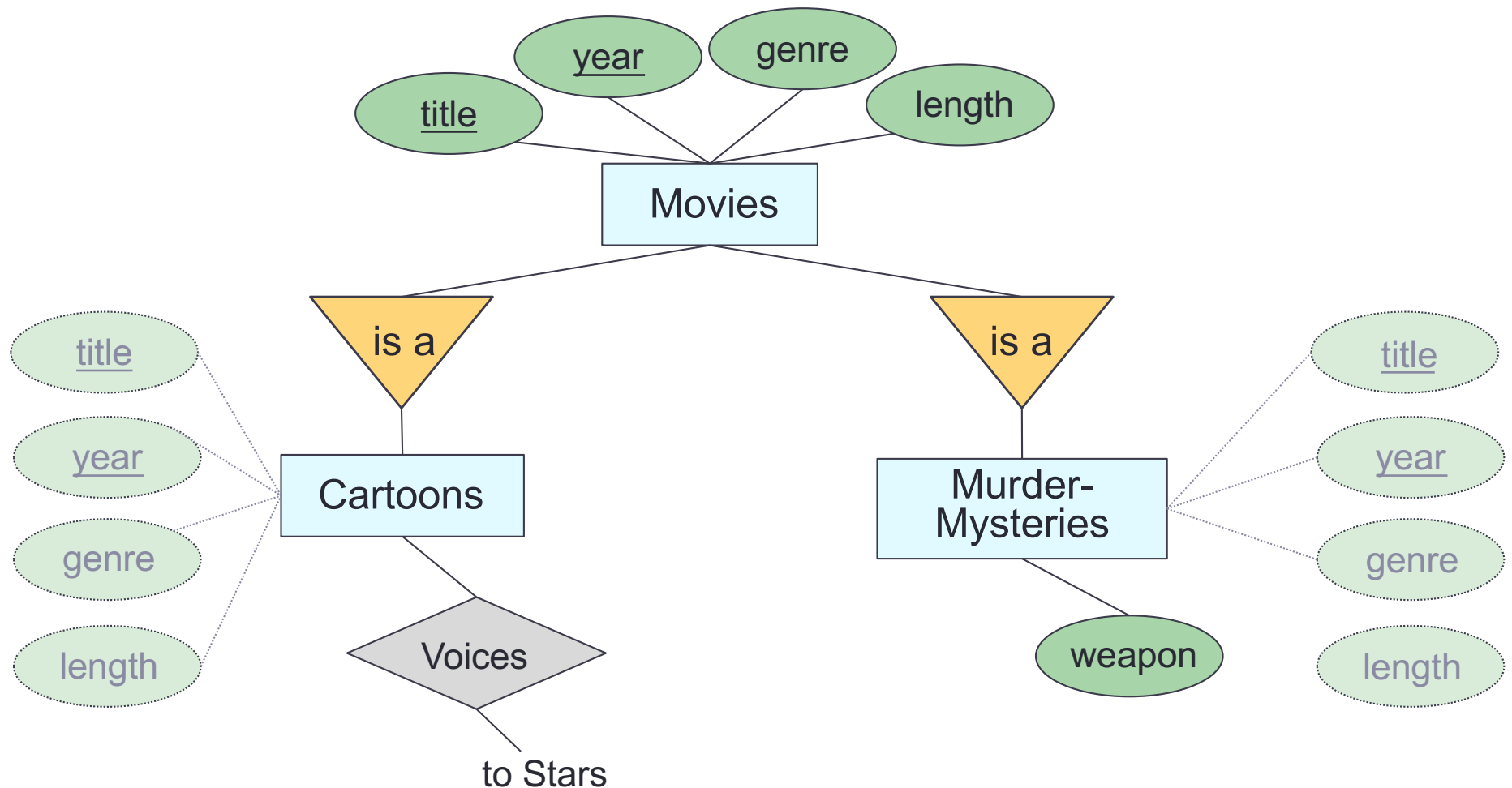
Done with the building blocks

Let's transition to design decision

and converting E-R diagram into Relational designs

Recap: Entity vs. Attribute

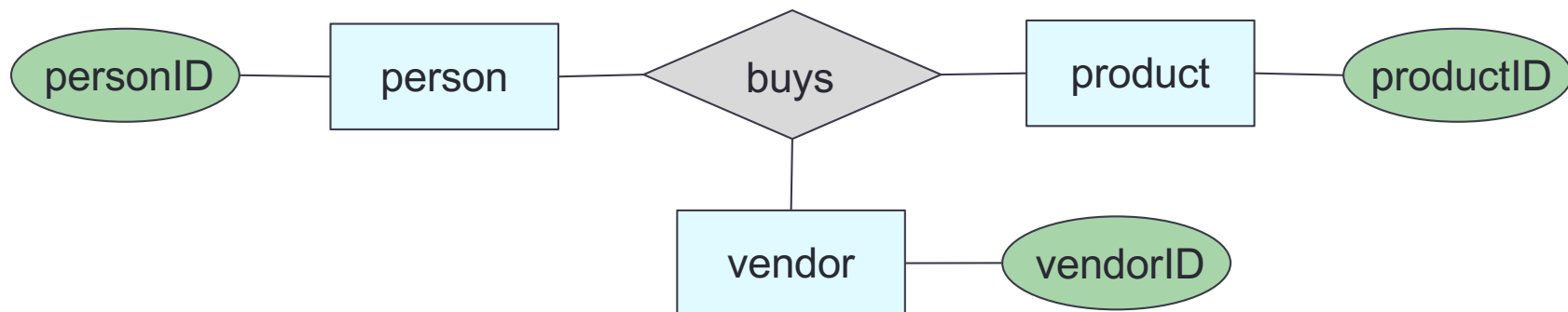**What are main differences between entities and attributes?**

- Entities can model situations that attribute cannot model naturally

- Entities can participate in relationships

- Entities can have attributes
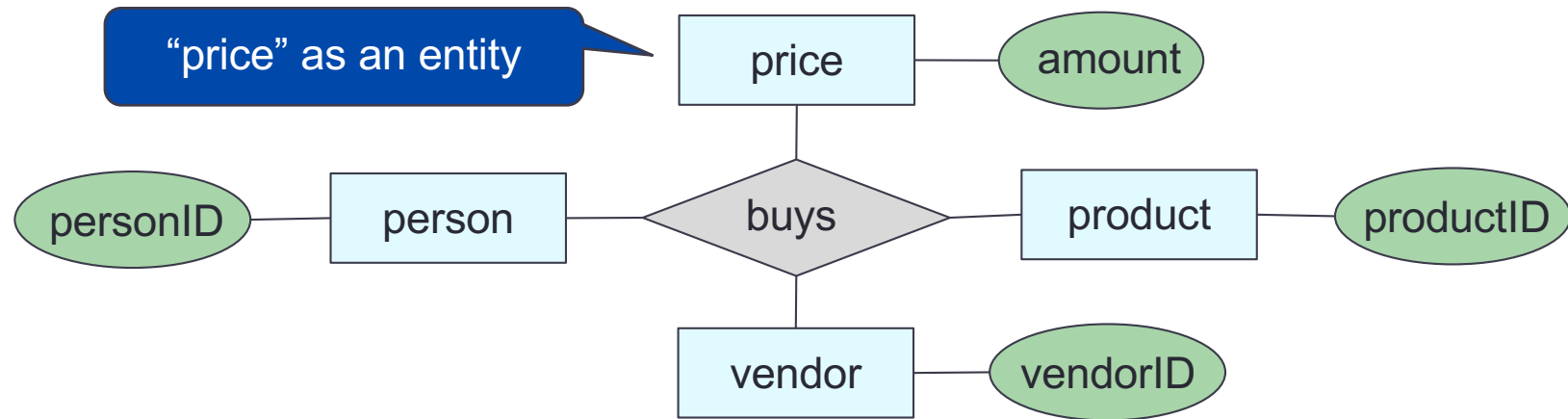

- Attributes cannot do any of these

Design Decision

Should "price" be an entity or an attribute?

price

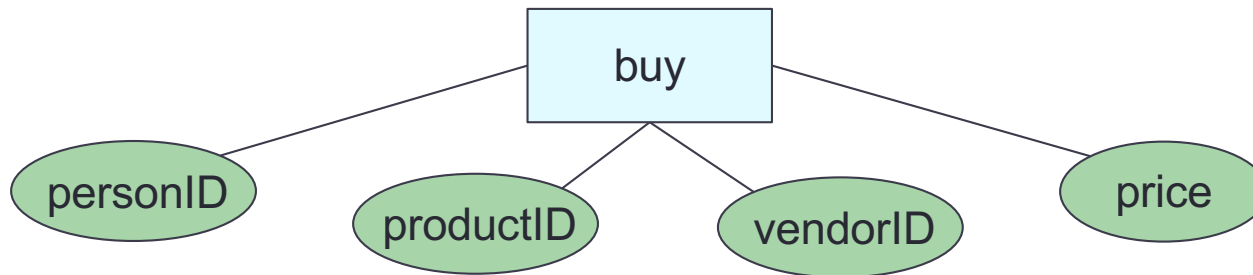For each tuple (person, product, vendor), there is a value of price

personID — person — buys — product — productID

vendor — vendorID

Design Decision (2)

"price" as an entity

price — amount

personID — person — buys — product — productID

vendor — vendorID

Since "price" is just the actual amount, treating it as an attribute is adequate. No need to make it an entity

"price" as an attribute

price

personID — person — buys — product — productID
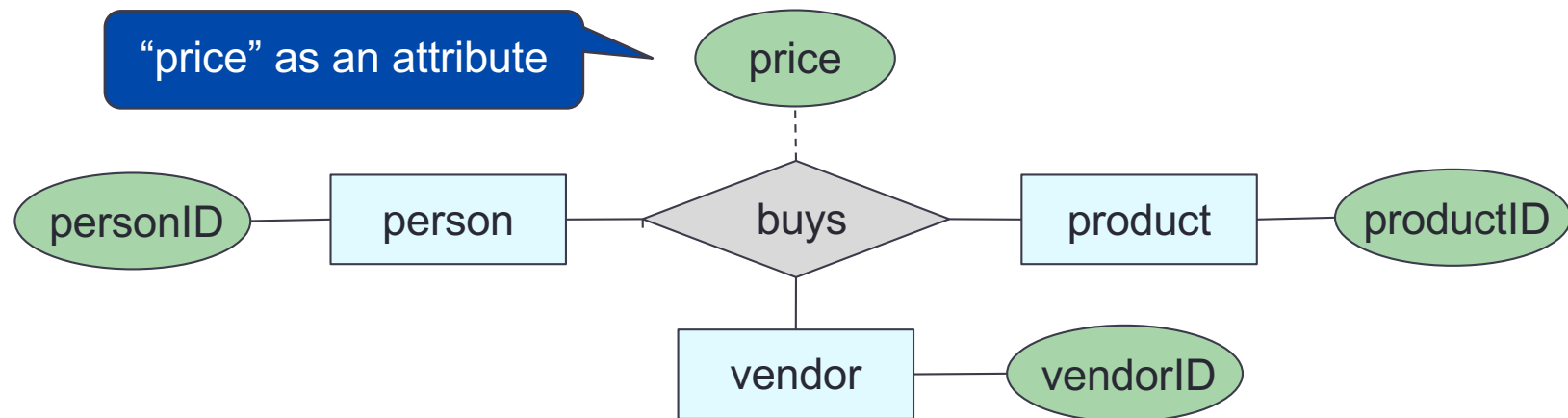
vendor — vendorID

Design Decision (3)

How about "buy" as an entity?

Should personID, productID, vendorID be entities or attributes?

- A "person" is an attribute of "buy"
- A "vendor" is an attribute of "buy"
- A "product" is an attribute of "buy"
- Cannot model something about a "person" (or "vendor" or "product") such as date-of-birth, address
- A "person" will involve in any relationship "buy" is associated with

Decisions to Make

- Entity set vs. attributes

    - Has more data   → entity set

    - Is the data   → attribute

- Entity set vs. relationship set

    - Entity set   → nouns (students, faculty, loads, …)

    - Relationship   → possession verbs (teaches, advises, owns, works for, …)

- Binary vs. n-ary relationship sets

- Specialization / generalization

Rules of Thumb

- Pick the right entities

- Keep it simple

- Don't over complicate things

- Choose the right elements (entities vs. attributes)

- Choose the right relationships

- Follow the specification of the application to be built

- Avoid NULL value
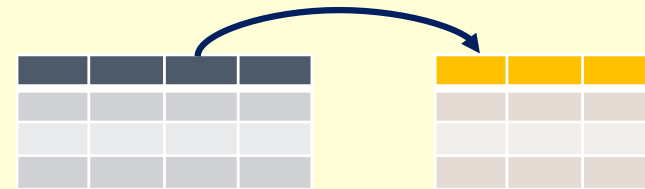
- Avoid redundancy

- Consider small number of tables

Database Design Process

Interact with users and domain experts to characterize the data

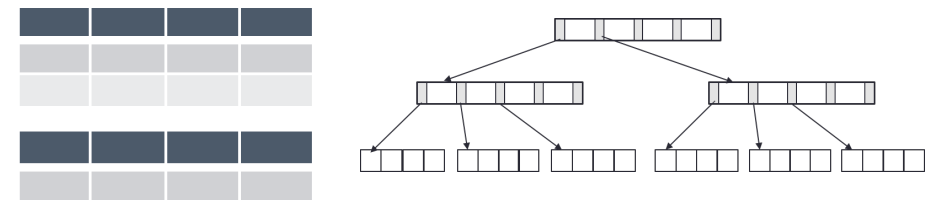Translate requirements into conceptual model (E-R diagrams)
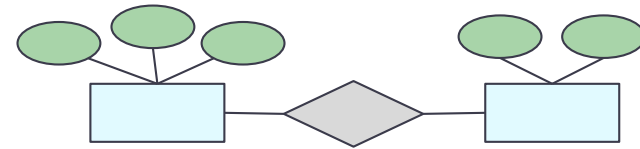
Convert the model to relational model (schema and constraints)

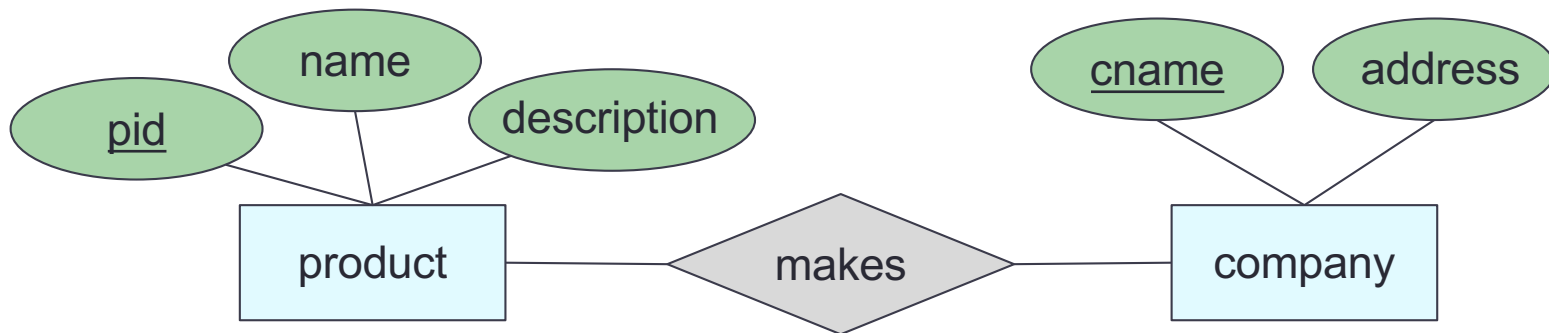Normalize and develop conceptual (logical) schema of the database

Develop physical schema (partitioning and indexing)

E-R Diagrams to Relations

There is a unique table which is assigned the name of the corresponding entity set or relationship set

product(<u>pid</u>, name, description)
company(<u>cname</u>, address)
makes(<u>cname</u>, <u>pid</u>)

"Schema statement"

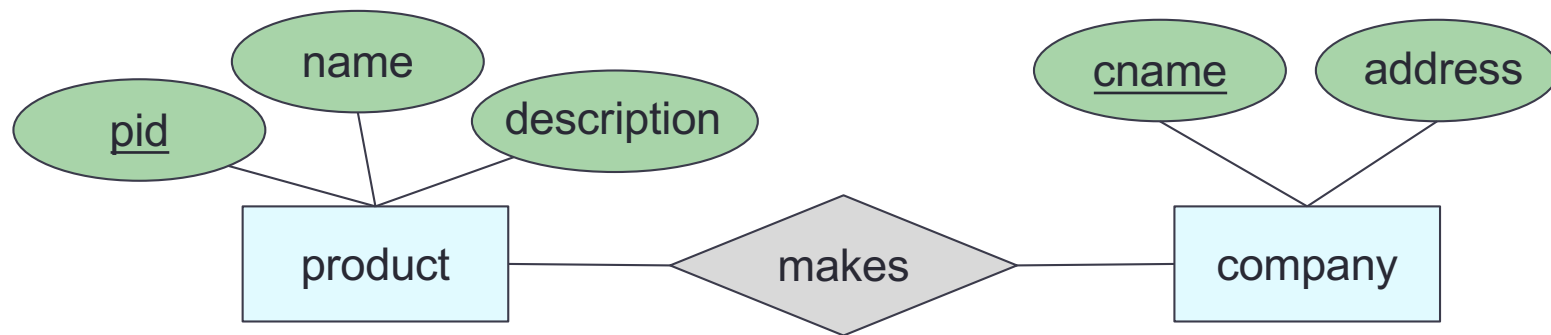[ER to schema worksheet]

Strong Entity Set

Direct map:

Entity name → relation name
Attributes → columns
Primary key: same as entity
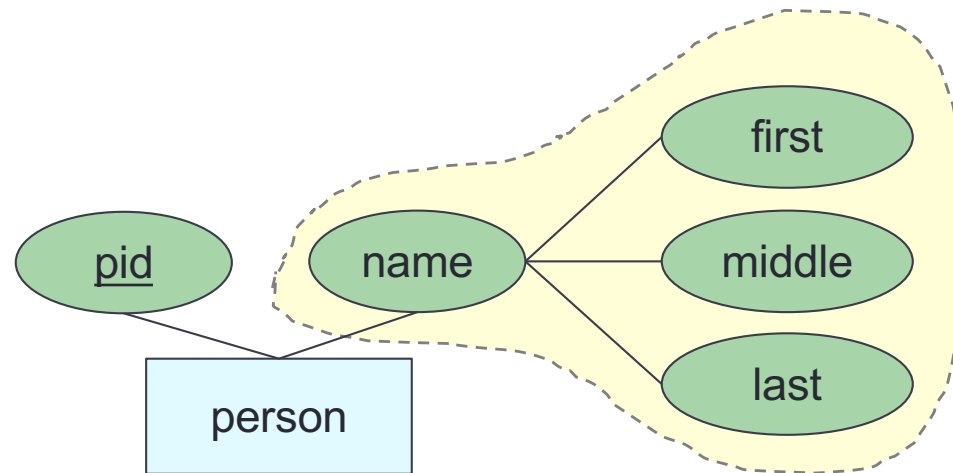

product(pid, name, description)
company(cname, address)
makes(cname, pid)

Strong Entity Set with Composite Attribute

Create separate attributes for each component

Don't include the higher level attribute

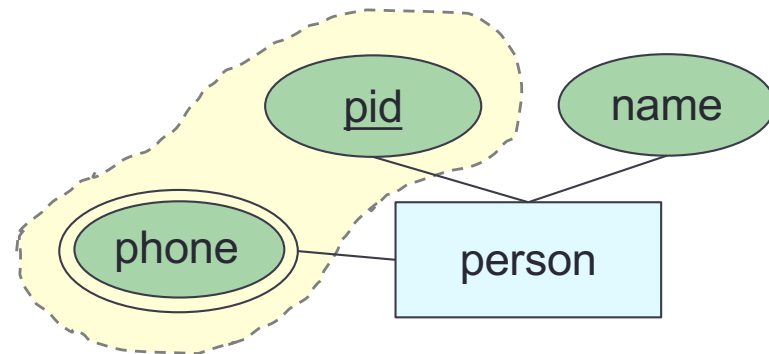person(<u>pid</u>, first_name, middle_name, last_name)

[ER to schema worksheet]

Strong Entity Set with Multivalued Attribute

Create a separate table for the multivalued attribute

Name the table with the concatenation, separated by "_"
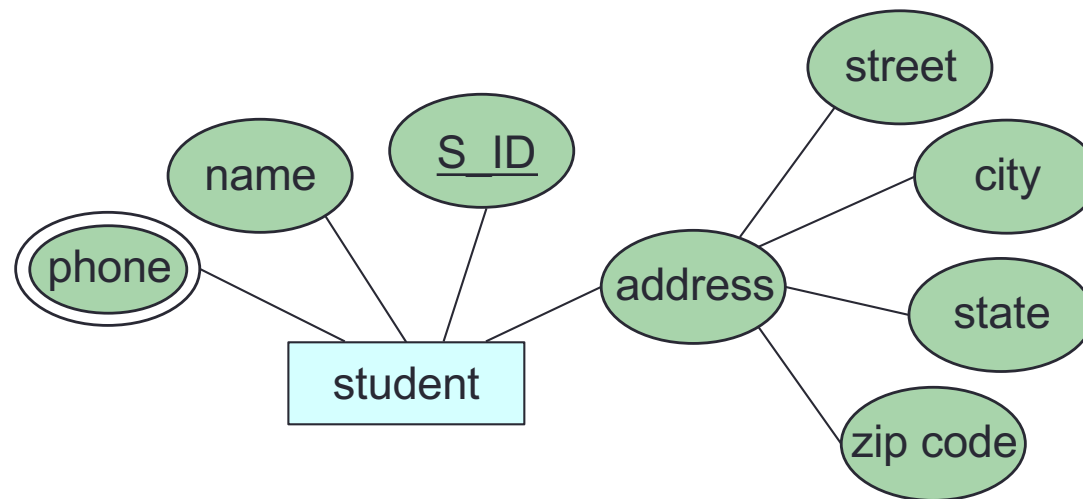   *entityname_attributename*

Primary key: all attributes

person(pid, name)
person_phone(pid, phone_number)

[ER to schema worksheet]

Let's try: E-R to Relations (1)

Convert the following E-R diagram into relations

student (<u>S_ID</u>, name, street, city, state, zip code)
student_phone (<u>S_ID</u>, <u>phone</u>)

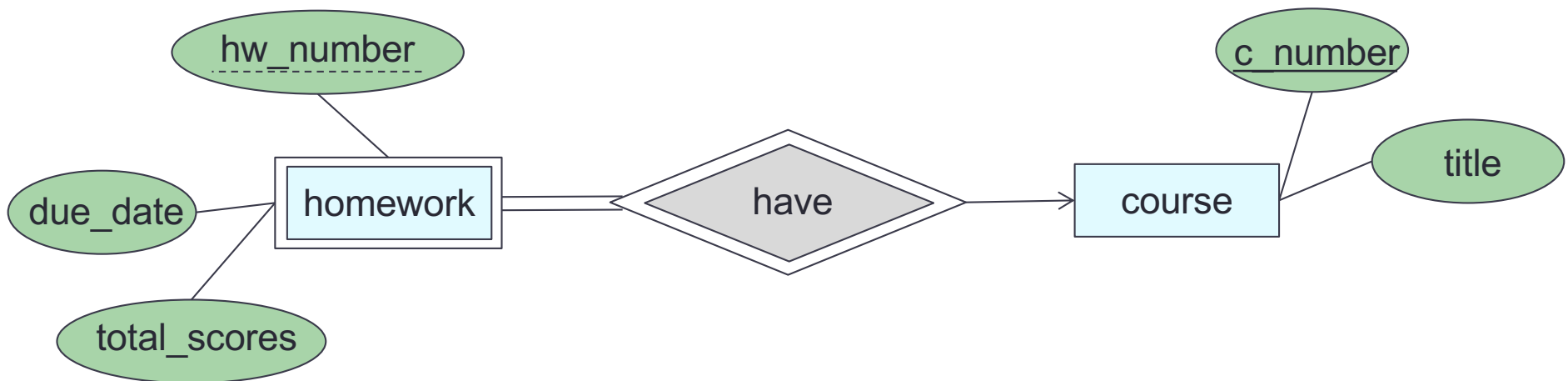PK of this table is a combination of all attributes

[ER to schema worksheet]

Weak Entity Set

Let *A* be a weak entity set and *B* be the identifying strong entity set on which *A* depends

Create a table with primary key of *B* and all *A*'s attributes

Primary key: primary key of *B* (strong entity) and discriminator of *A*
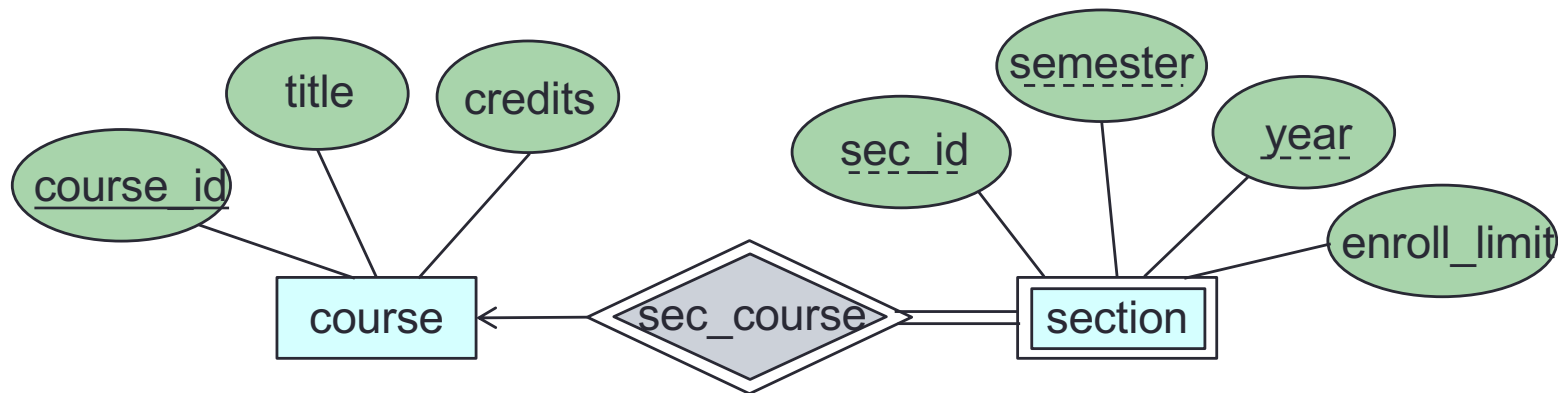

course(<u>c_number</u>, title)

homework(<u>c_number</u>, <u>hw_number</u>, due_date, total_scores)

[ER to schema worksheet]

Let's try: E-R to Relations (2)

Convert the following E-R diagram into relations

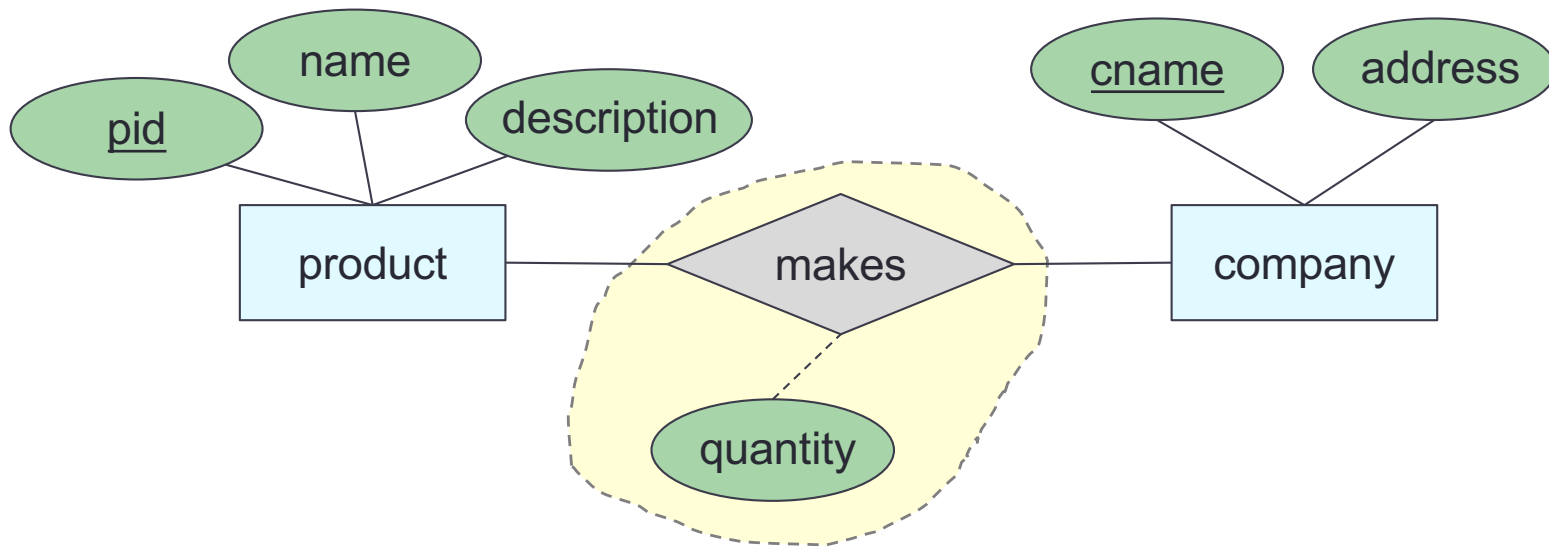course (<u>course_id</u>, title, credits)
section (<u>course_id</u>, <u>sec_id</u>, <u>semester</u>, <u>year</u>, enroll_limit)

[ER to schema worksheet]

Relationship Set: Many-to-Many

Table: primary keys of both participating entity sets and any attributes on the relationship itself

Primary key: primary keys of both participating entity sets

product(<u>pid</u>, name, description)
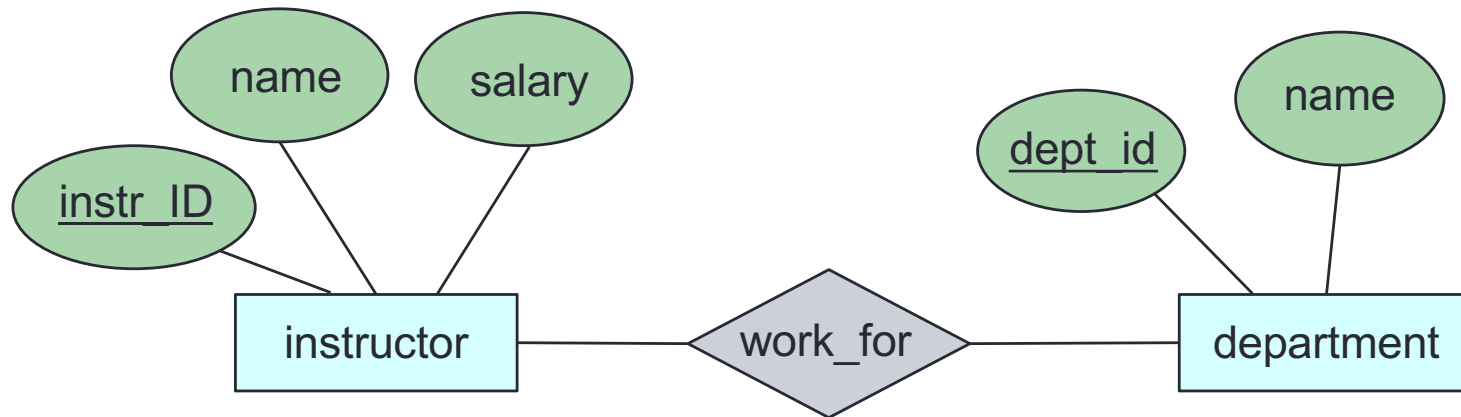company(<u>cname</u>, address)
makes(<u>pid</u>, <u>cname</u>, quantity)

Primary keys of both entities

[ER to schema worksheet]

Convert the following E-R diagram into relations
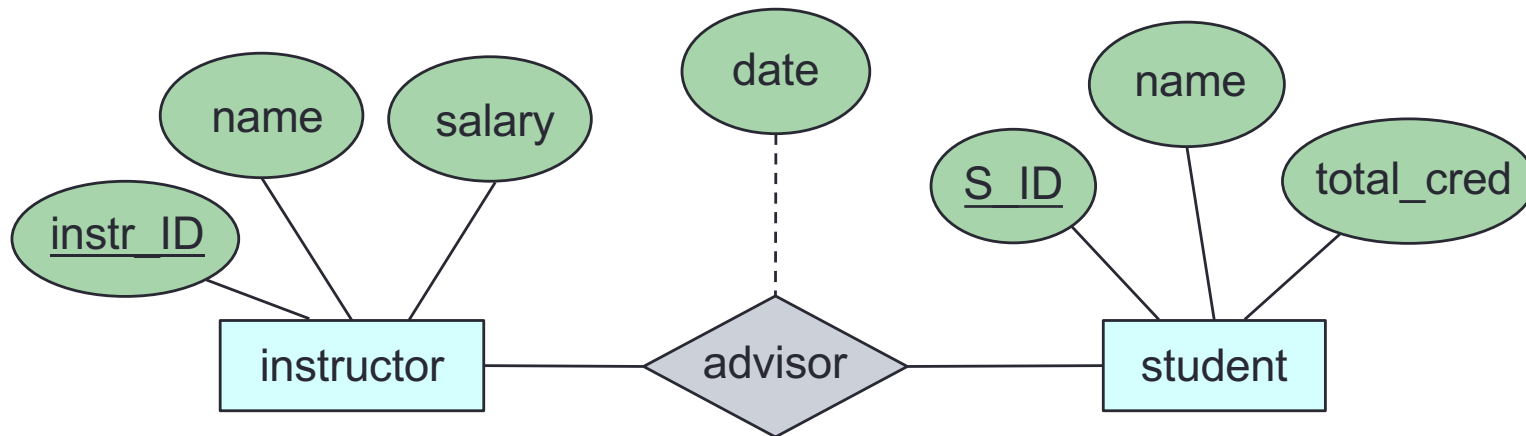

instructor (<u>instr_ID</u>, name, salary)
department (<u>dept_id</u>, name)
work_for (<u>instr_ID</u>, <u>dept_id</u>)

[ER to schema worksheet]

Let's try: E-R to Relations (4)
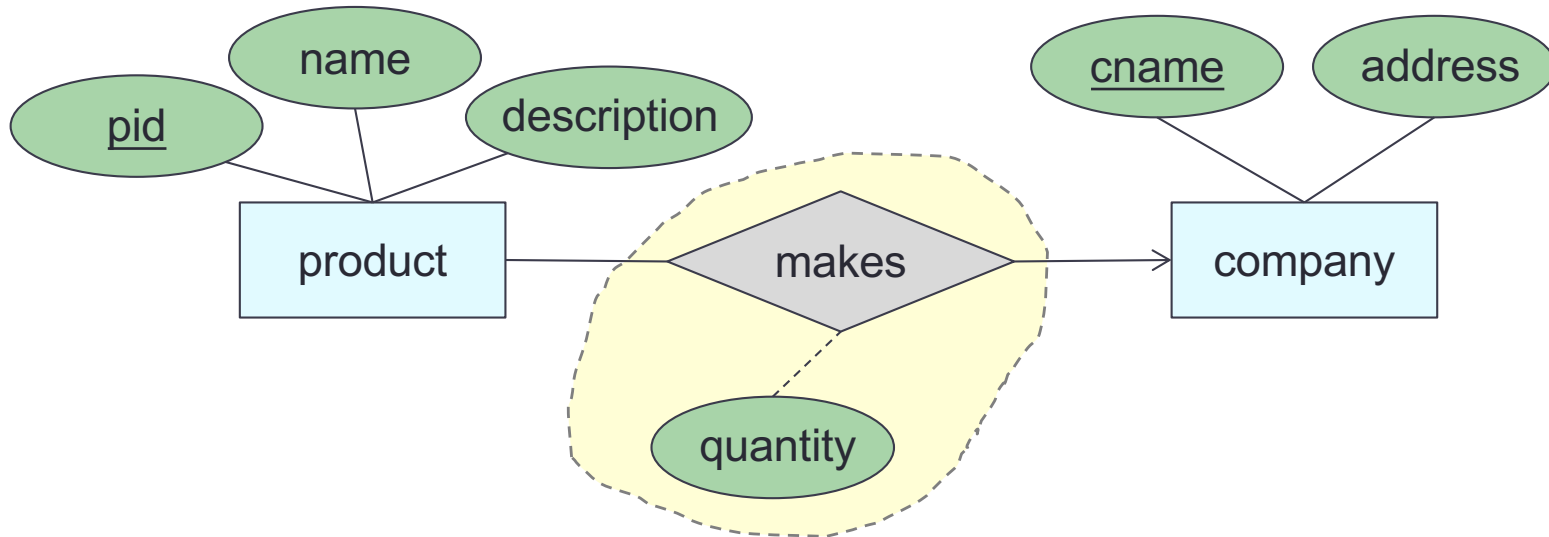
Convert the following E-R diagram into relations

instructor (<u>instr_ID</u>, name, salary)
student (<u>S_ID</u>, name, total_cred)
advisor (<u>instr_ID</u>, <u>S_ID</u>, date)

[ER to schema worksheet]

Relationship Set: Many-to-One / One-to-Many

Table: primary keys of both participating entity sets and any attributes on the relationship itself

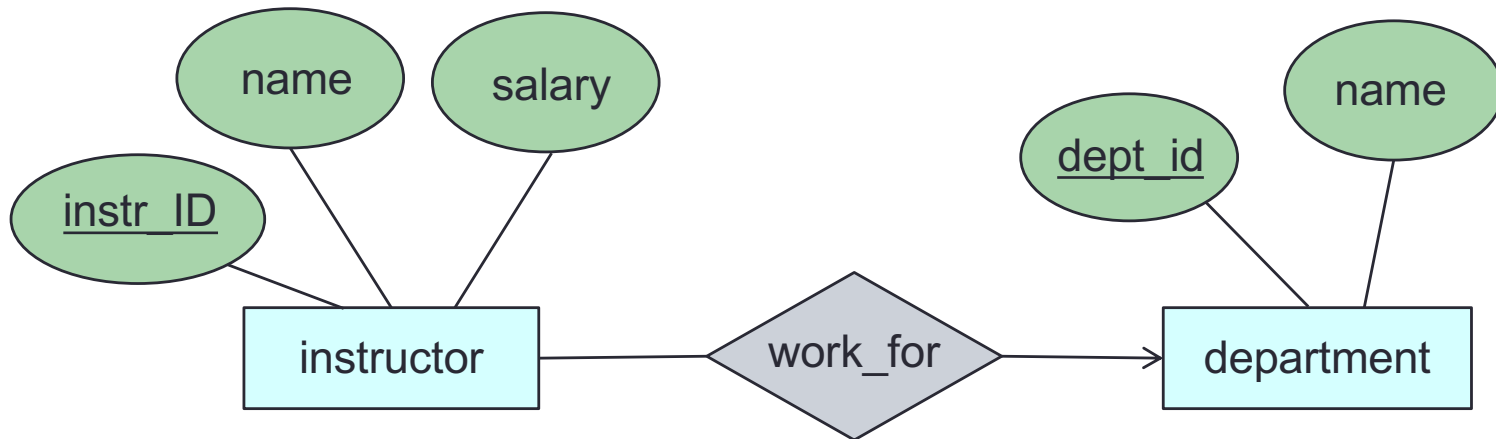Primary key: primary keys of the entity set on the "many" side

product(pid, name, description)
company(cname, address)
makes(pid, cname, quantity)

Primary key of the "many" side

[ER to schema worksheet]

Let's try: E-R to Relations (5)

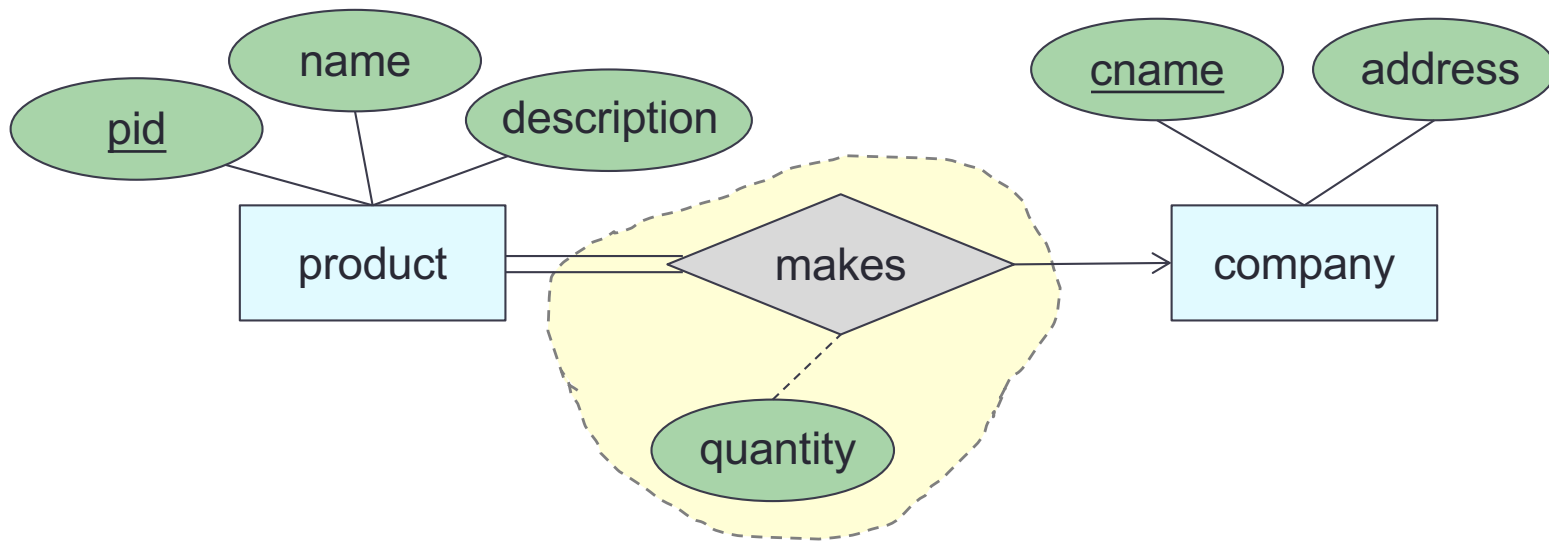**Convert the following E-R diagram into relations**

instructor (<u>instr_ID</u>, name, salary)
department (<u>dept_id</u>, name)
work_for (<u>instr_ID</u>, dept_id)

[ER to schema worksheet]

Relationship Set: Total Participation

Because the total participation requires all entity to be participated in the relationship

→ add the primary key of the "one" side to the "many" side entity set, no table for relationship needed
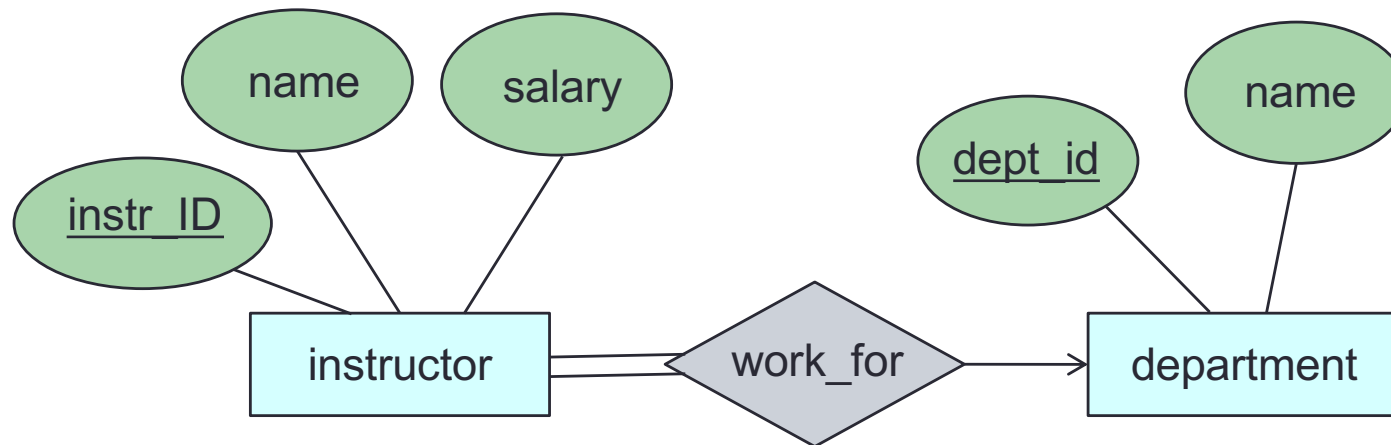

product(<u>pid</u>, name, description, cname, quantity)
company(<u>cname</u>, address)

[ER to schema worksheet]

Let's try: E-R to Relations (6)

Convert the following E-R diagram into relations

instructor (<u>instr_ID</u>, name, salary, dept_id)
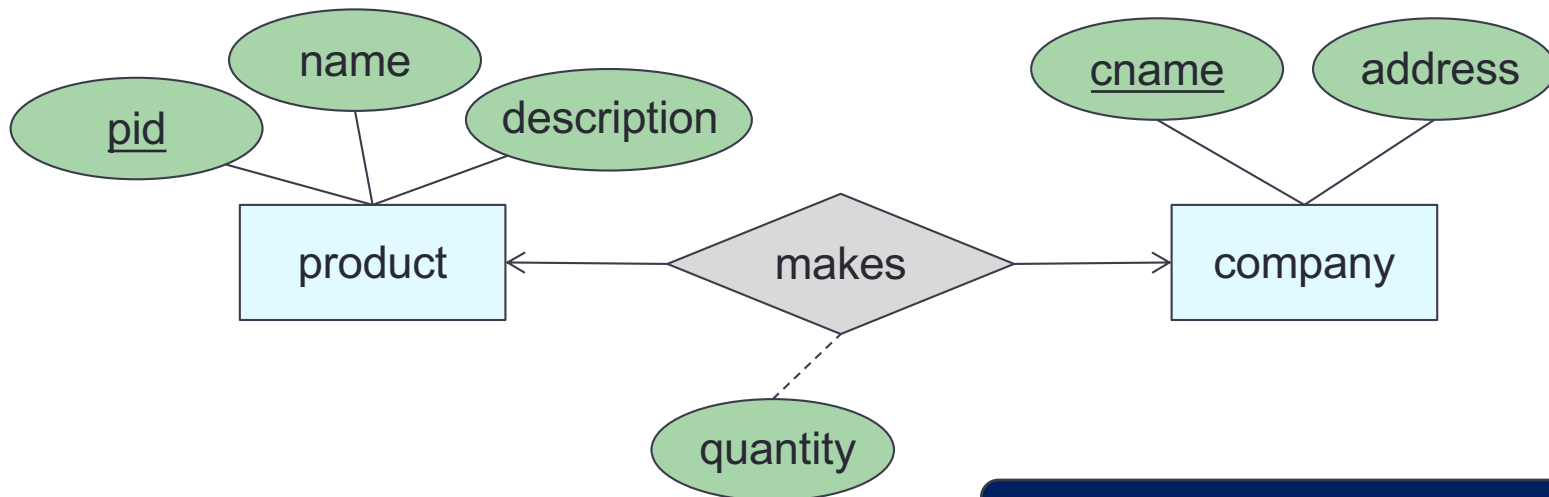department (<u>dept_id</u>, name)

Relationship Set: One-to-One

Table: Either side can be used as the main table

> (Which side? doesn't matter. Pick the one that makes the most sense)

> Add the other side's primary key to it

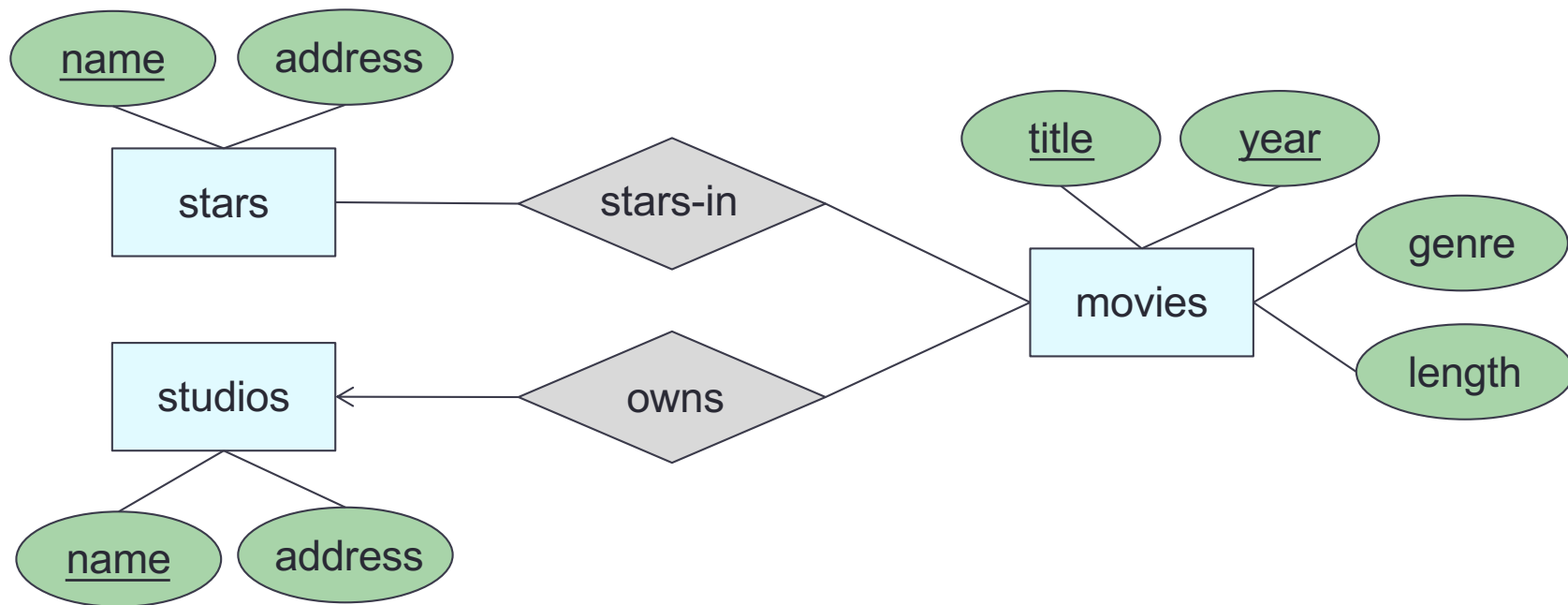Primary key: primary keys of the entity set you pick

Primary key of the chosen entity

product(<u>pid</u>, name, description)
company(<u>cname</u>, address, pid, quantity)

[ER to schema worksheet]

Convert the following E-R diagram into relations

stars(<u>name</u>, address)

studios(<u>name</u>, address)

movies(<u>title</u>, <u>year</u>, genre, length)

stars-in(<u>title</u>, <u>year</u>, <u>starsName</u>)

owns(<u>title</u>, <u>year</u>, studioName)

[ER to schema worksheet]

Let's try: E-R to Relations (8)

Convert the following E-R diagram into relations
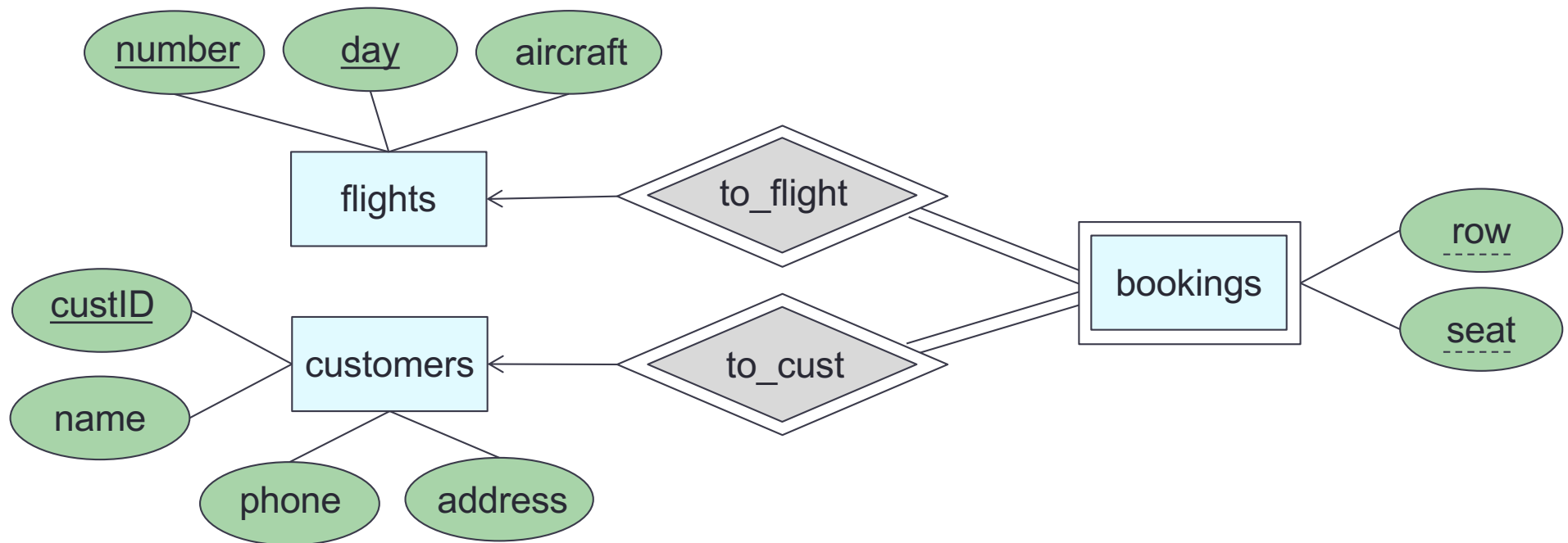
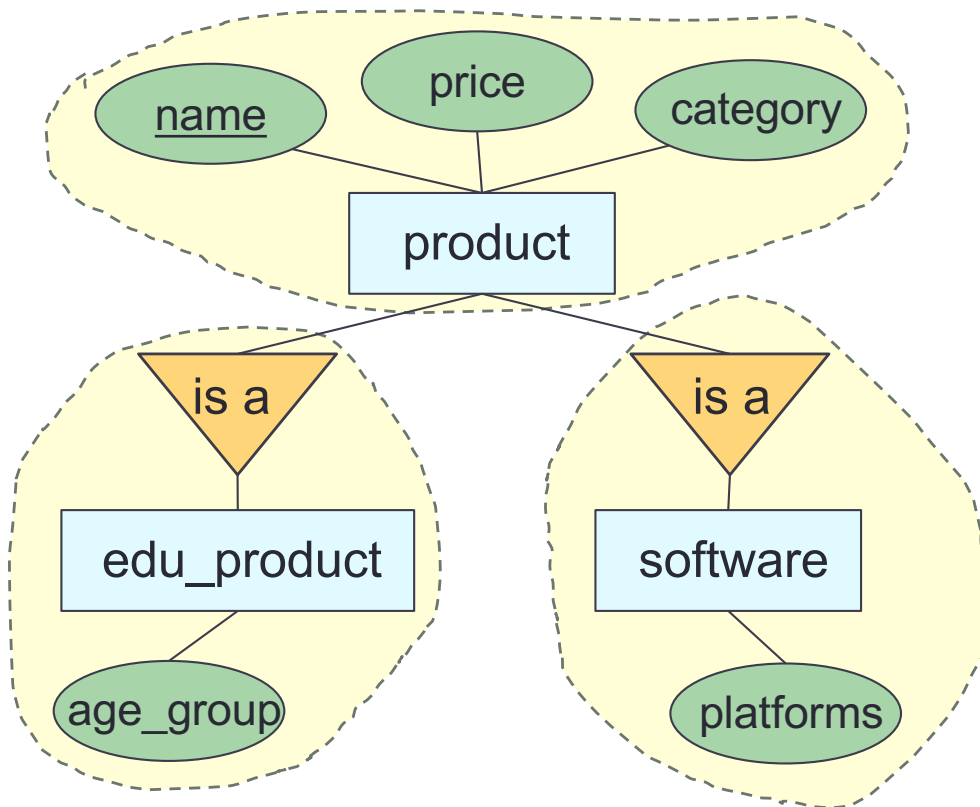flights(<u>number</u>, <u>day</u>, aircraft)

customer(<u>custID</u>, name, phone, address)

bookings(<u>number</u>, <u>day</u>, <u>custID</u>, <u>row</u>, <u>seat</u>)

[ER to schema worksheet]

Subclass (Option 1)

Keep everything

Primary key of the lower level entity set: from the higher level

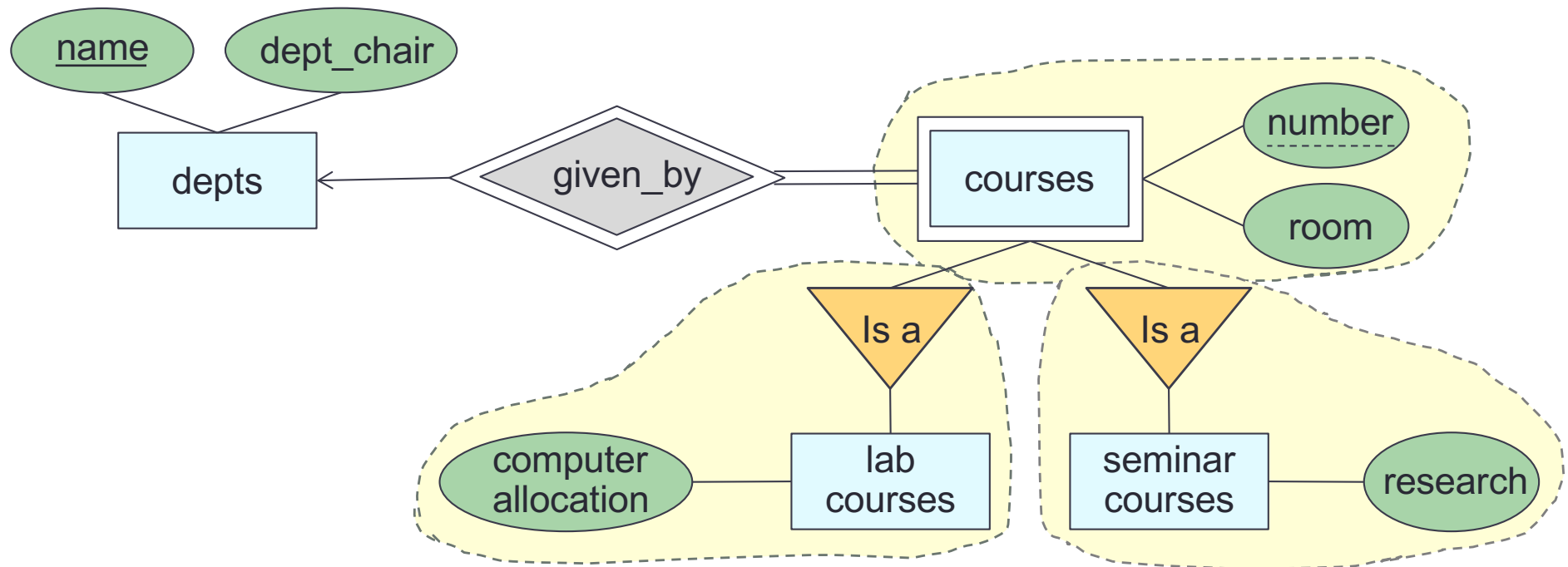Drawback: need to access more tables to get info about the lower levels

product(<u>name</u>, price, category)
edu_product(<u>name</u>, age_group)
software(<u>name</u>, platforms)

Keep everything

[ER to schema worksheet]

Let's try: Subclasses (option 1)

Convert the following E-R diagram into relations
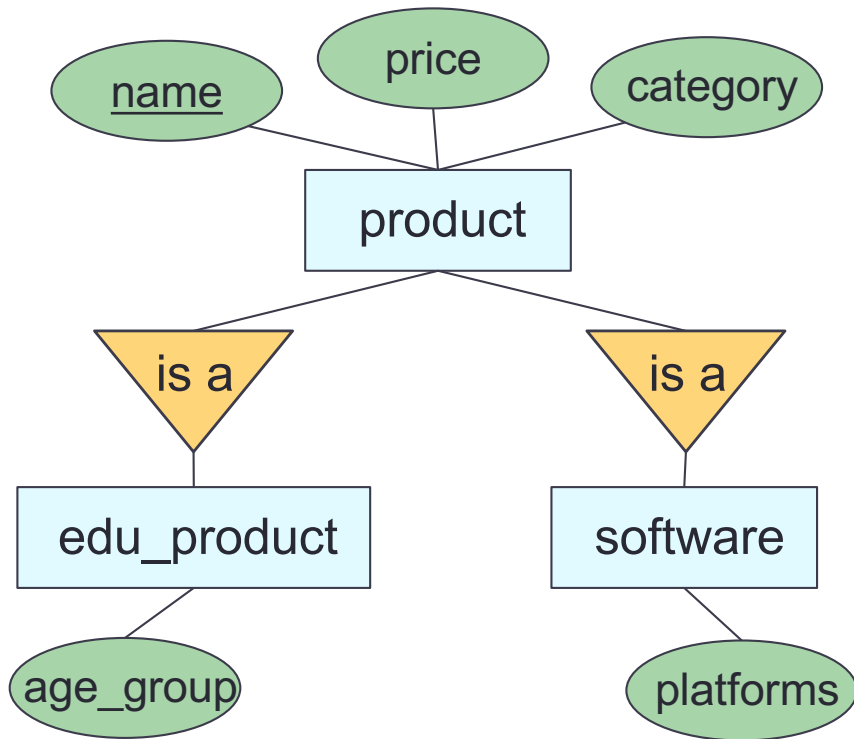

depts(<u>name</u>, dept_chair)

courses(<u>deptName</u>, <u>number</u>, room)

labCourses(<u>deptName</u>, <u>number</u>, computerAllocation)

seminarCourses(<u>deptName</u>, <u>number</u>, research)

[ER to schema worksheet]

Subclass (Option 2)

Keep specialization entity sets

No table for generalization entity set

Primary key of the lower level entity set: from the higher level

Drawback: redundancy if entities have more than one specialization

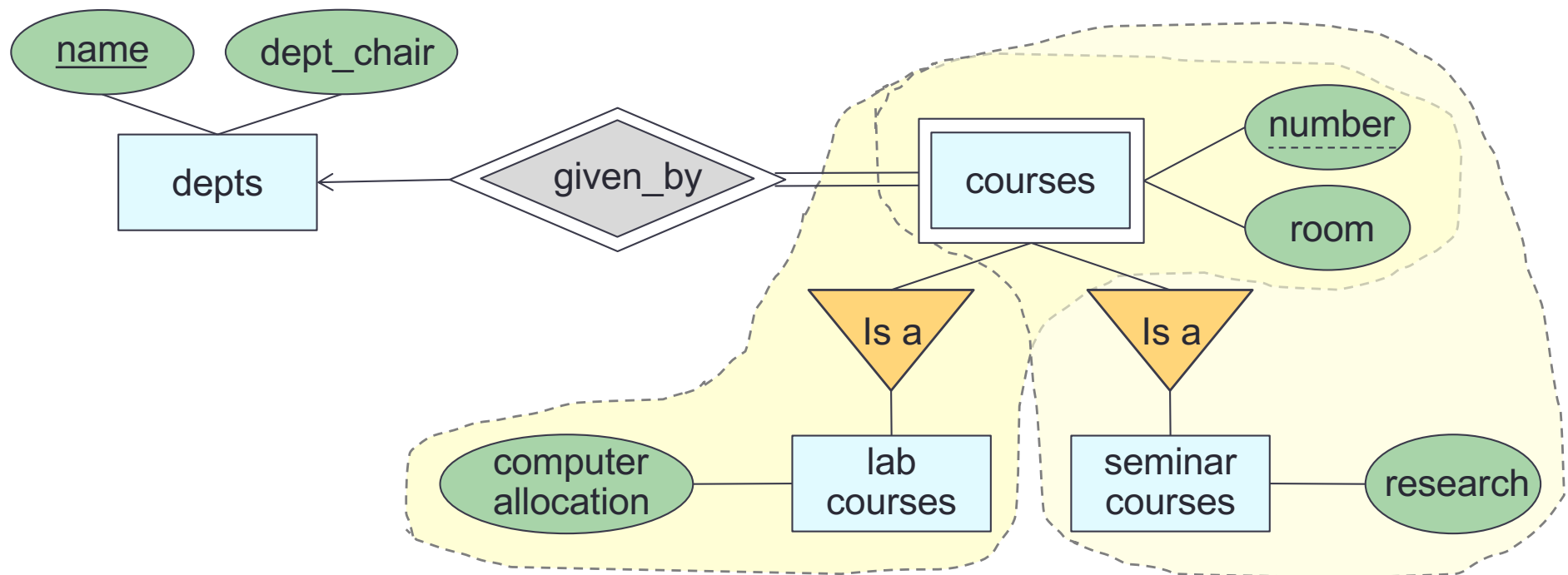edu_product(name, price, category, age_group)
software(name, price, category, platforms)

Push down

Let's try: Subclasses (option 2)

Convert the following E-R diagram into relations

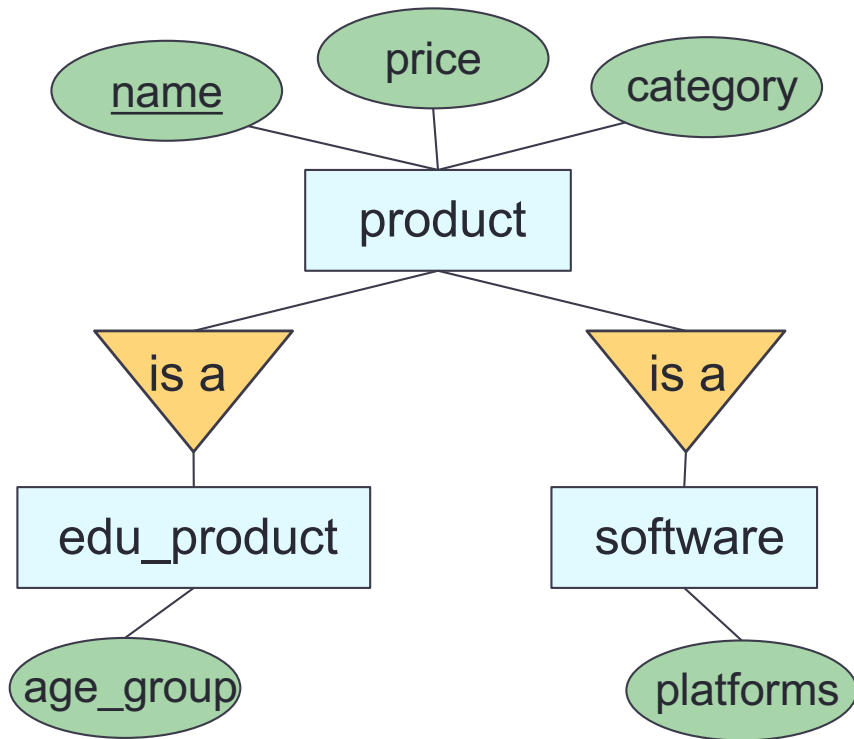depts(<u>name</u>, dept_chair)

labCourses(<u>deptName</u>, <u>number</u>, room, computerAllocation)

seminarCourses(<u>deptName</u>, <u>number</u>, room, research)

[ER to schema worksheet]

Subclass (Option 3)

Keep generalization entity set

No table for specialization entity sets

Drawback: NULL in attributes from specialization entity sets

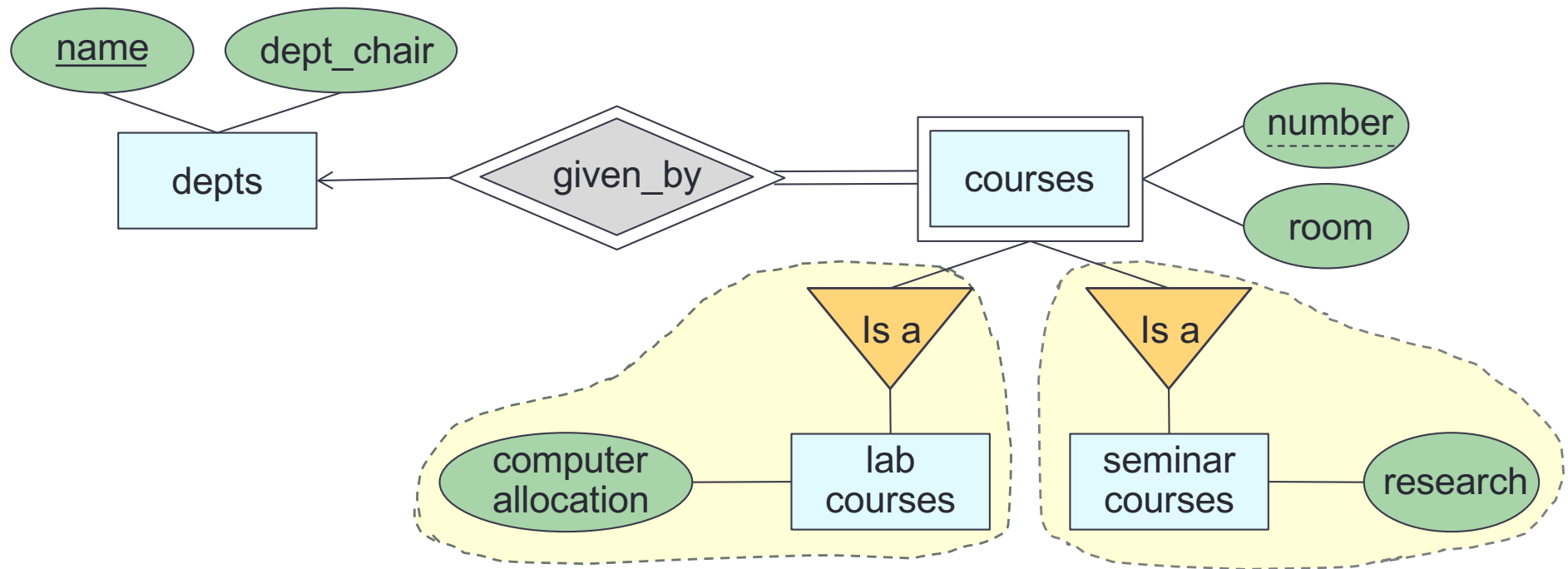Although less duplication of data, need to handle NULL value

product(<u>name</u>, price, category, age_group, platforms)

Push up

[ER to schema worksheet]

Let's try: Subclasses (option 3)

Convert the following E-R diagram into relations

depts(<u>name</u>, dept_chair)

courses(<u>deptName</u>, <u>number</u>, room, computerAllocation, research)

Subclass: Design Decision

Depending on the number of attributes of the generalization entity set and specialization entity set

- If balanced → do option 1 (create all)
- If more attributes in specialization → do option 2
- If more attributes in generalization → do option 3

In general, design decision depends on

- The number of attributes
- DB administrator's decision

Overall goal: minimize duplication
(there is no one correct way)

Wrap-Up

- Roles in Relationships

- Relationships: binary, u-ary

- Weak entity

- Subclasses

- Converting from E-R diagrams to relational designs
  - Turn each entity set into a relation with the the same set of attributes
  - Replace a relationship by a relation whose attributes are the keys for the connected entity sets
  - Weak entity sets cannot be translated straightforwardly to relations
  - "Is a" relationships and subclasses require careful treatment

What's next?

- Apply the concept to database scenarios and fine-tuning database structure