E-R Diagram: relationships, weak entity, subclass, E-R to relational design

CS 4750 Database Systems

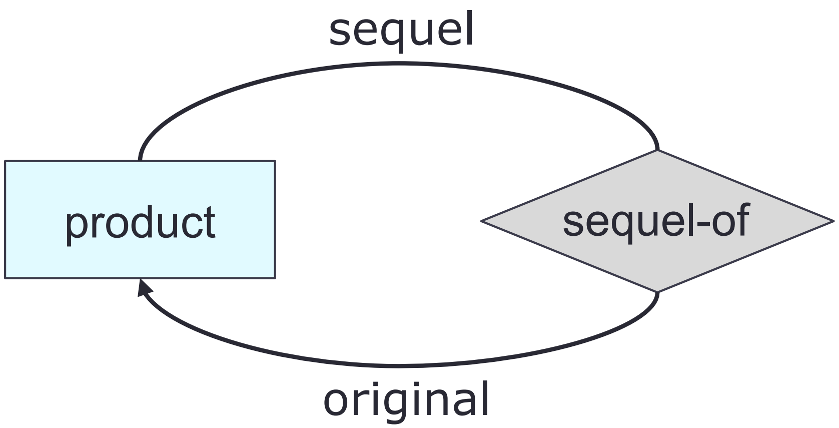
[A. Silberschatz, H. F. Korth, S. Sudarshan, Database System Concepts, Ch.6]
[C.M. Ricardo and S.D. Urban, Database Illuminated, Ch.3]

Roles in Relationships

- An entity set can appear two or more times in a single relationship

- Each edge to the entity set represents a different role that the entity set plays in the relationship

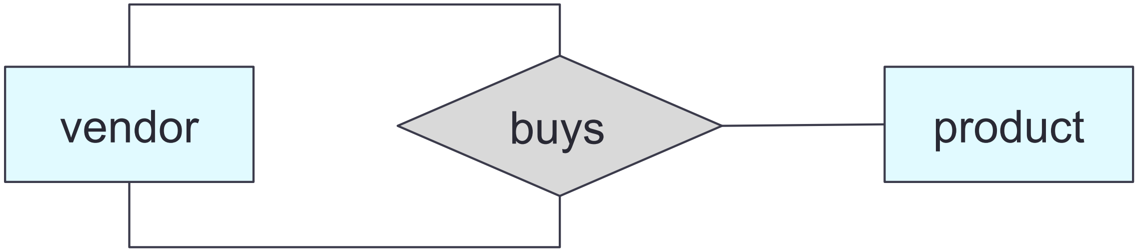
"self-referential relationship"



A product can have many sequels.
For each sequel, there is only one original product

Let’s try: Self-Referential, Multi-Way

Given the following E-R diagram. Come up with an example that can be represented by the diagram.



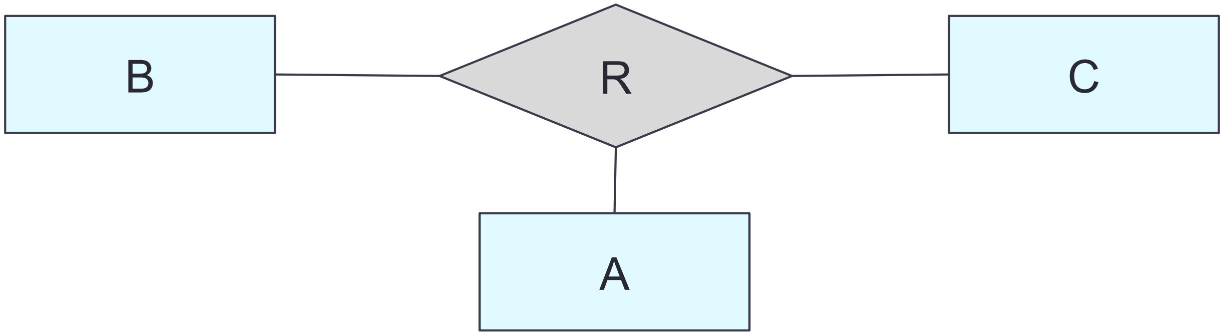
“Microsoft buys a printer from HP”

Binary vs. Unary Relationships

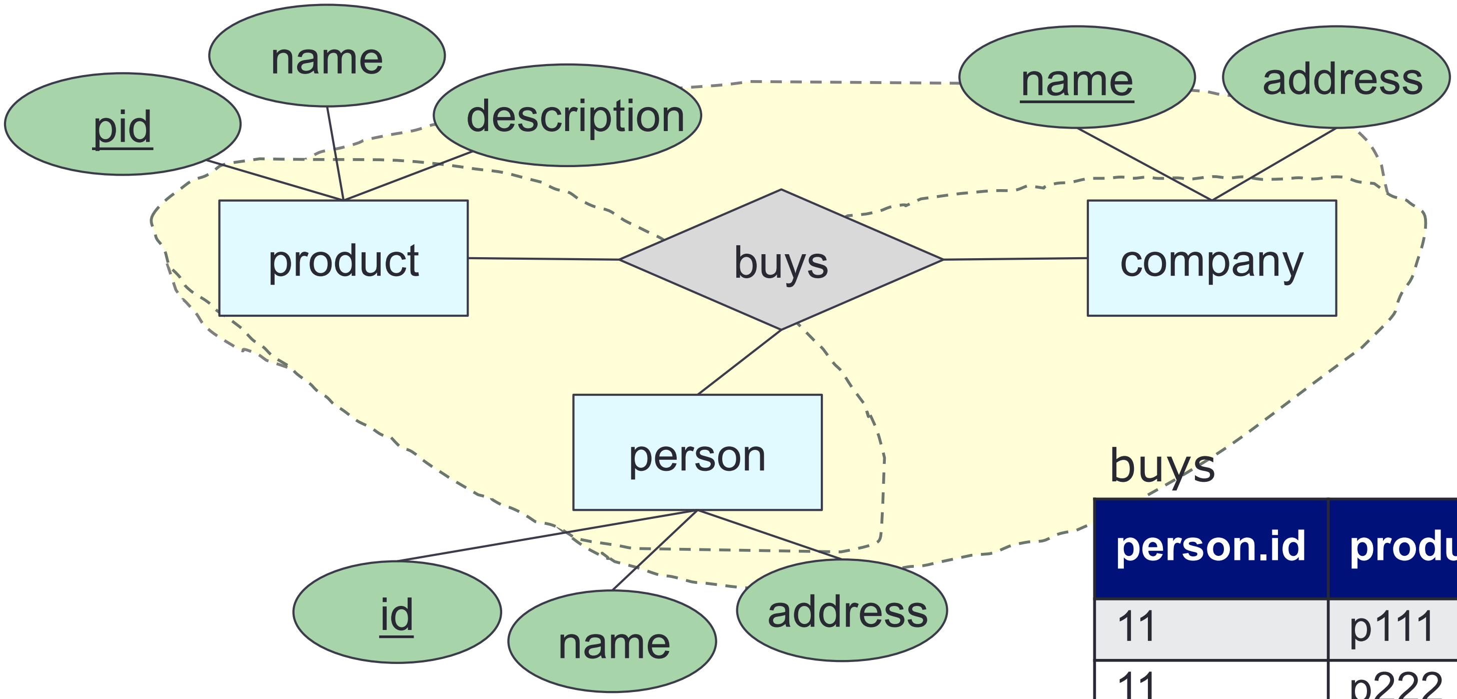
E-R model makes it convenient to define relationships involving more than two entity sets.

In practice, ternary (3-way) or higher-degree relationships are rare and usually add complexity to the design.

If *A*, *B*, and *C* are sets, a relationship *R* is a subset of *A X B X C*



Multi-Way Relationships



buys

person.id	product.pid	company.name
11	p111	Humpty Inc.
11	p222	Humpty Inc.
22	p111	Humpty Inc.
22	p111	Dumpty Shop

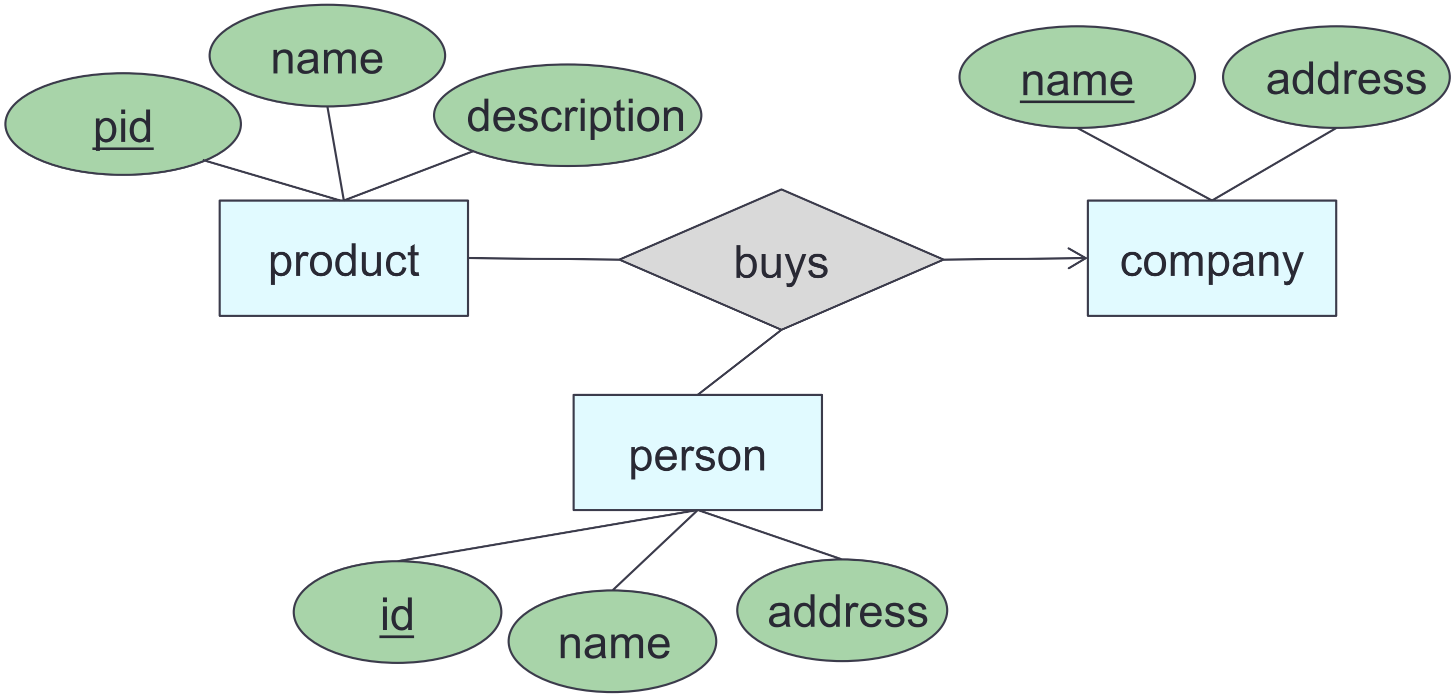
Each (person, product) pair can connect to many companies.
Each (person, company) pair can connect to many products.
Each (company, product) pair can connect to many persons.

Note: instances do not exist in E-R. These tables are only to help visualize the database being designed.

Let’s try: Multi-Way Relationships

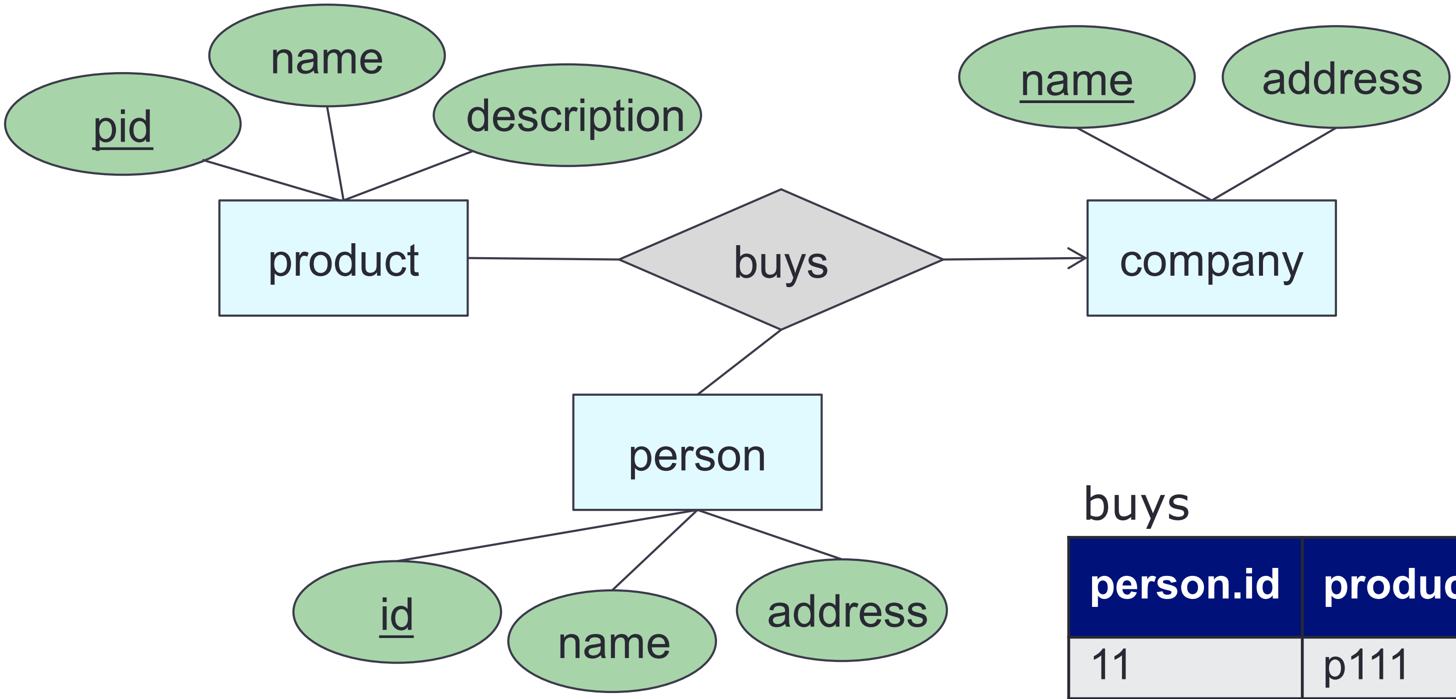
What if we want to ensure that each (person, product) pair comes from (or connects to) a single company?

Complete the diagram. (hint: don’t forget the cardinality)



Let’s try: Multi-Way Relationships

Based on the E-R diagram, **identify if any row in the given table is not allowed**



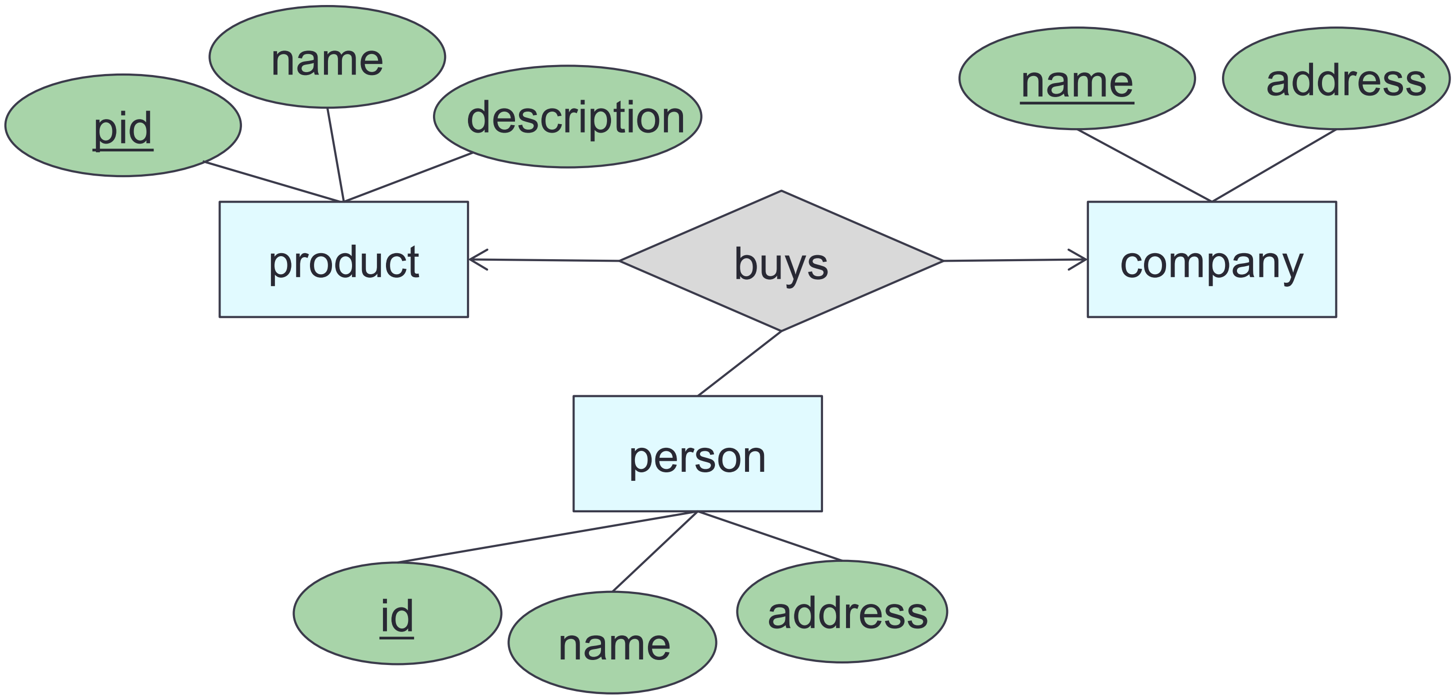
buys

person.id	product.pid	company.name
11	p111	Humpty Inc.
11	p222	Humpty Inc.
22	p111	Humpty Inc.
~~22~~	~~p111~~	~~Dumpty Shop~~

Note: instances do not exist in E-R. These tables are only to help visualize the database being designed.

Let’s try: Multi-Way Relationships

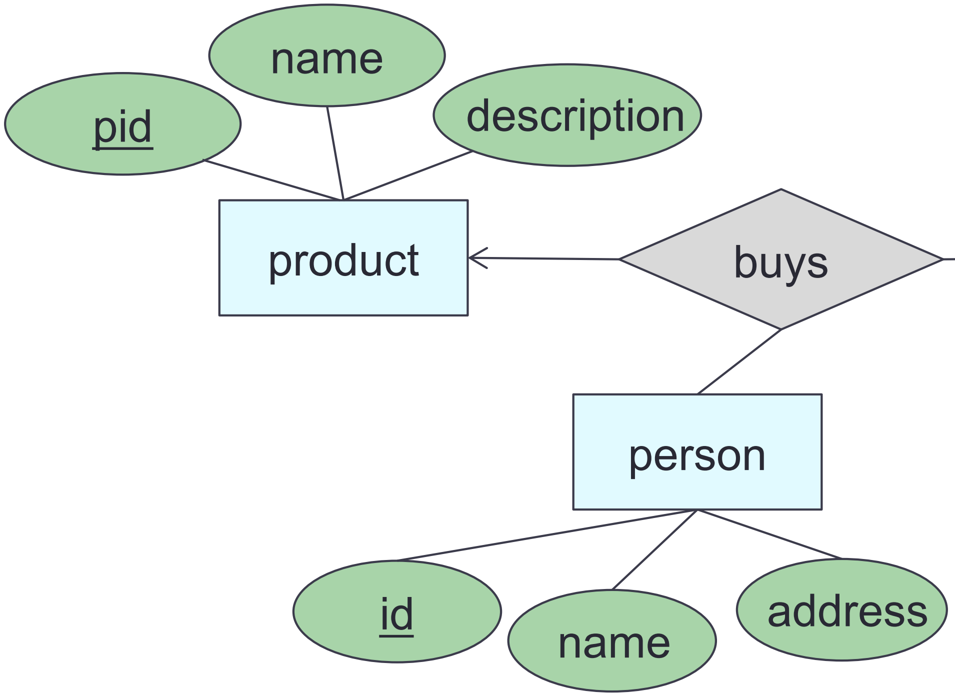
What can we interpret from the E-R diagram?



Each (person, product) pair connects to at most one company.
Each (person, company) pair connects to at most one product.
Each (company, product) pair connects to many persons.

Let’s try: Multi-Way Relationships

Based on the E-R diagram, **identify if any row in the given table is not allowed**



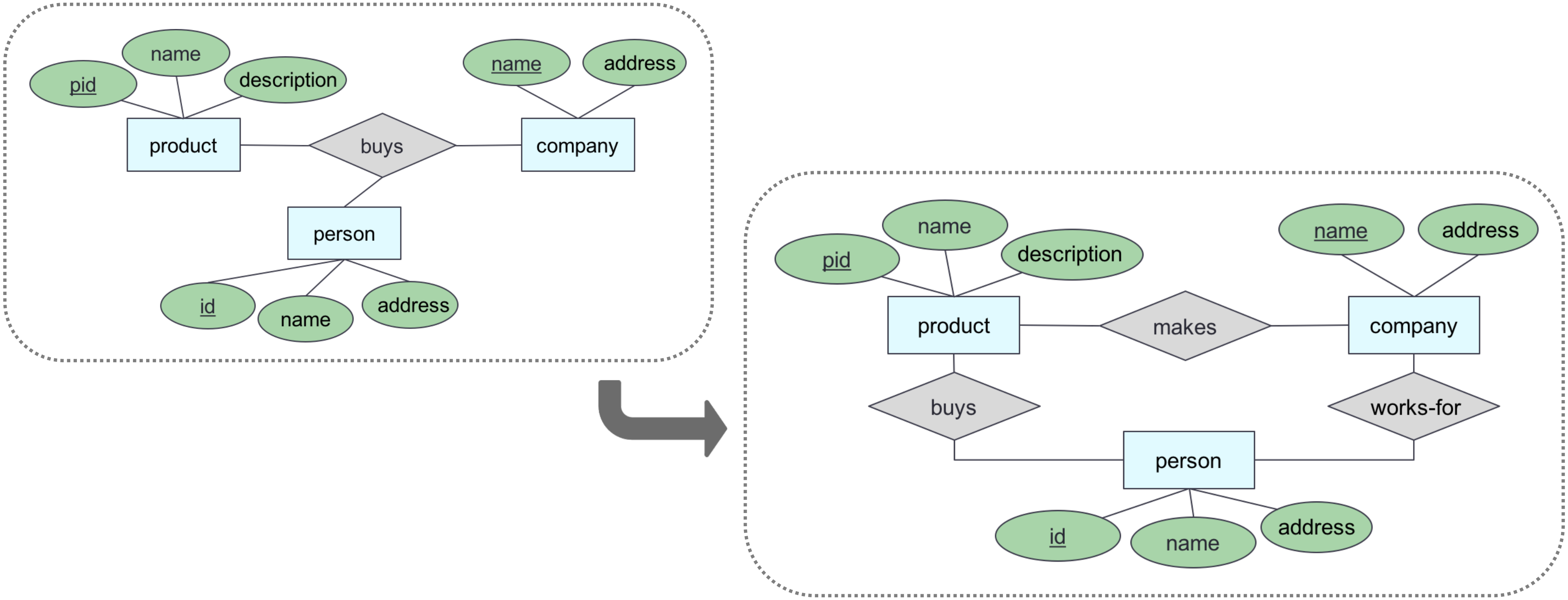
buys

person.id	product.pid	company.name
11	p111	Humpty Inc.
~~11~~	~~p222~~	~~Humpty Inc.~~
22	p111	Humpty Inc.
~~22~~	~~p111~~	~~Dumpty Shop~~
11	p333	Awesome Shop
33	p333	Awesome Shop

Note: instances do not exist in E-R. These tables are only to help visualize the database being designed.

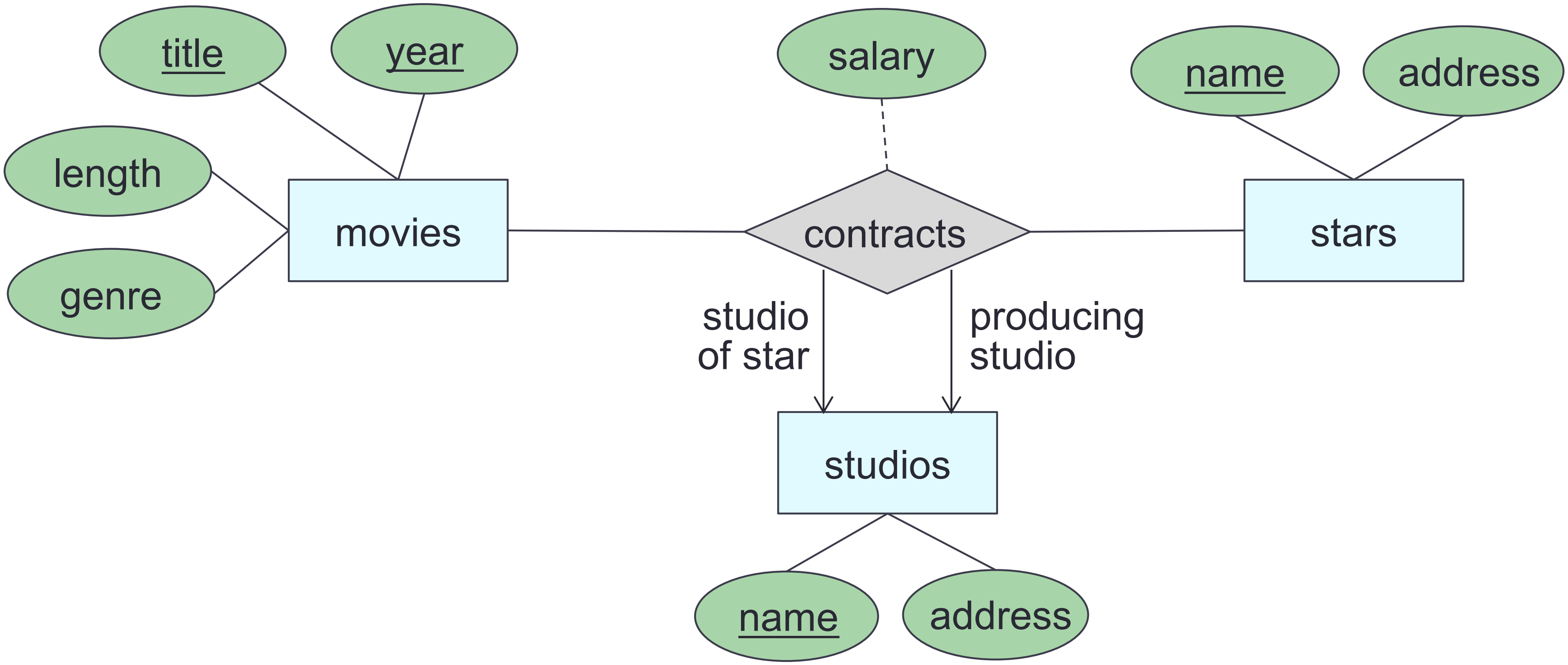
Converting Multi-Way to Binary

- E-R model does not require binary relationships
- It is useful to convert u-ary relationship to a collection of binary relationships
- To convert, replace a multi-way relationship with an entity set and binary relationships



Let’s try: Multi-Way Relationships

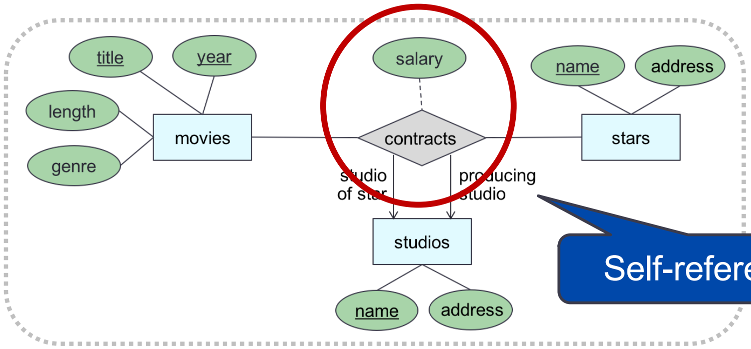
Suggest how we may convert the 4-way relationship into an E-R diagram with binary relationships



Ideas: Split multi-way relationship into multiple binary relationships.
Consider changing self-referential relationship into binary relationship(s).
Additional entity set(s) may be added.

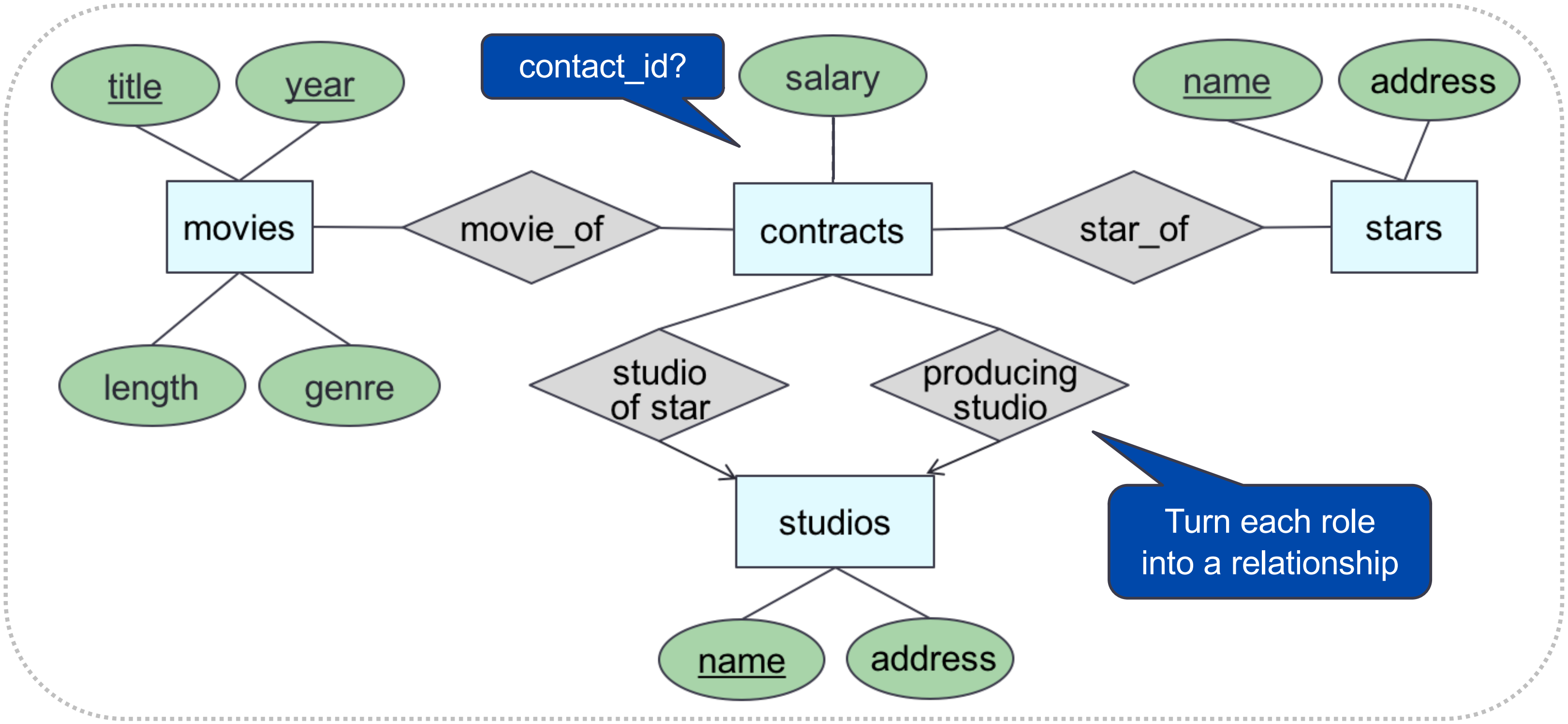
Let’s try: Multi-Way Relationships

Suggest how we may convert the 4-way relationship into an E-R diagram with binary relationships (from previous page)



Self-referential

Note: this is one idea. There are many ways to convert this diagram, mostly depending on design decision.



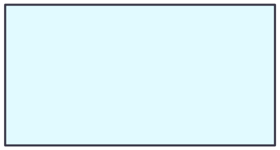
contact_id?

Turn each role into a relationship

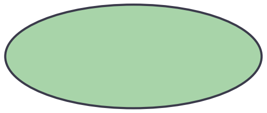
revisit

E-R Diagram: Building Blocks

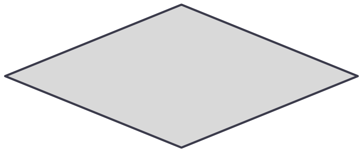
(strong) Entity set



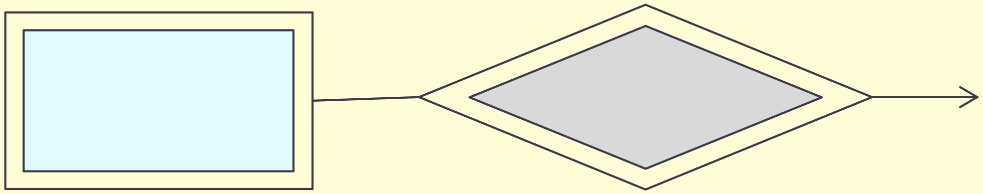
Attribute



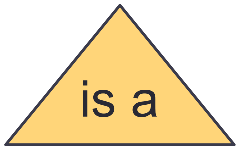
Relationship



Weak entity



Subclass



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

Strong and Weak Entity Sets

Strong entity set

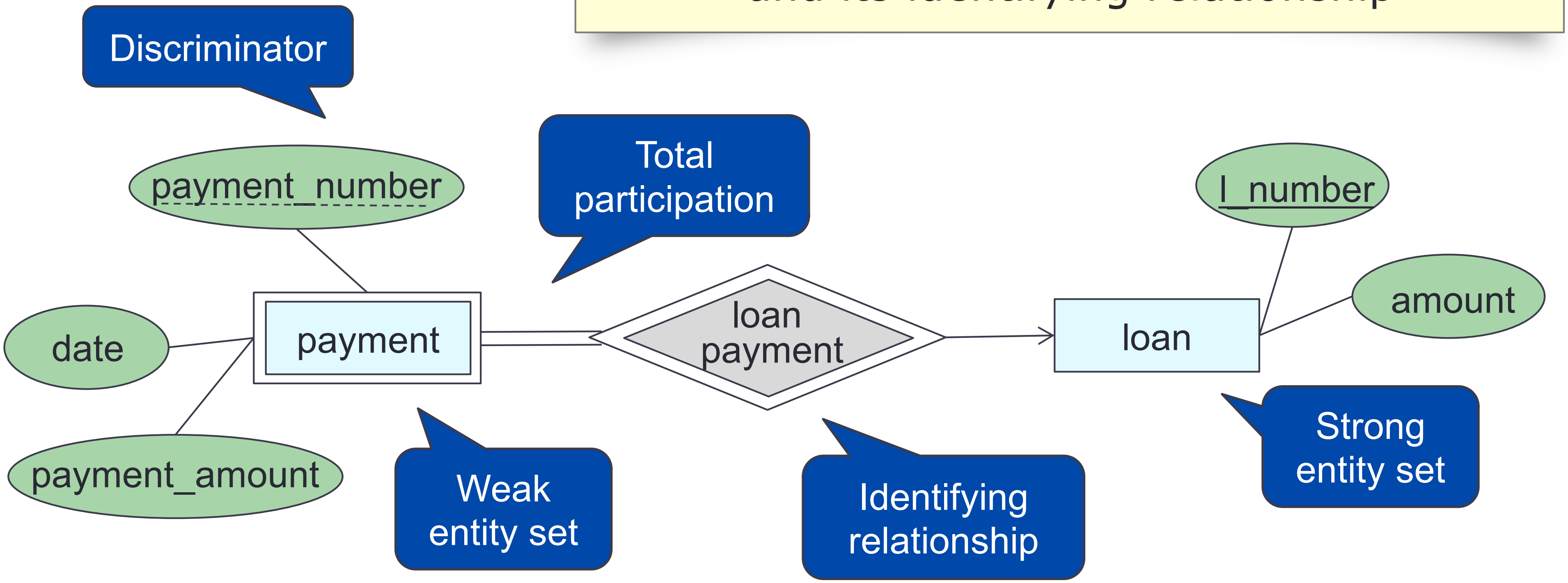
- Entities can be identified by the values of their attributes (a primary key)
- (what we have been discussing so far)

Weak entity set

- Entities cannot be identified by the values of their attributes
- There is no primary key made from its own attributes
- An entity can be identified by a combination of their attributes ("discriminator") and the relationship they have with another entity set ("identifying relationship")

Weak Entity Set

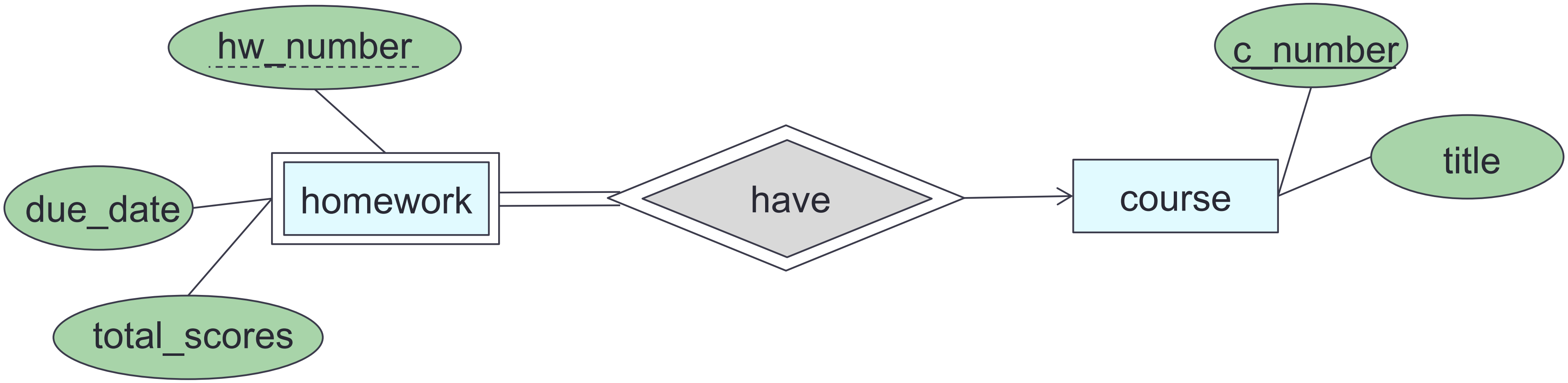
By definition, there must be total participation between the weak entity set and its identifying relationship



- Does not have sufficient attributes to form a primary key.
- Depends on the strong entity set it is associated with.
- Needs a discriminator and a primary key of the strong entity set.

Let’s try: Weak Entity Set

What can be concluded from the following E-R diagram?



Homework cannot exist without a course.
Every homework must belong to a single class.
A course can have many homework.
Different courses may have the same homework number.
To identify a homework, we need c_number and hw_number.

Let’s try: Weak Entity Set

Draw an E-R diagram for the following scenario

A movie studio might have several film crews.

The crews might be designated by a crew’s number as crew1, crew2, and so on.

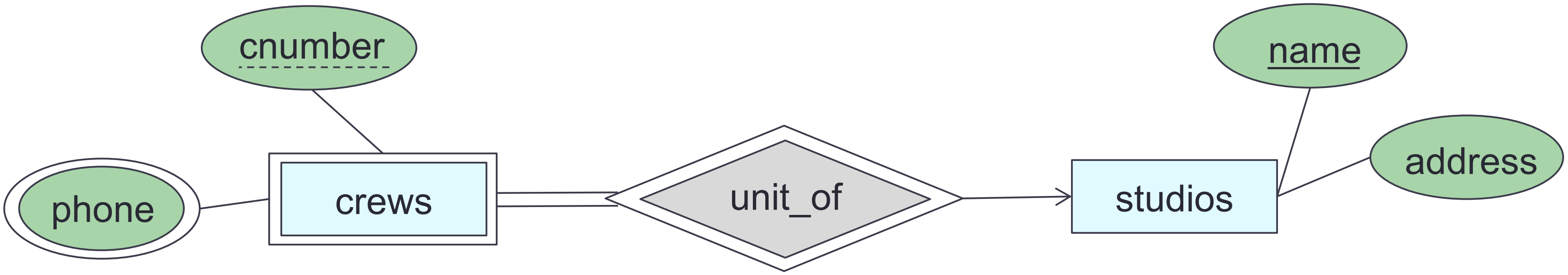
Each crew has multiple phone numbers.

Other studios might use the same designations for crews, so the attribute crew’s number is not a key for crews.

To name a crew uniquely, we need to give both the name of the studio to which it belongs and the crew’s number.

Let’s try: Weak Entity Set

Draw an E-R diagram for the following scenario



(from previous page)

A movie studio might have several film crews.

The crews might be designated by a crew’s number as crew1, crew2, and so on.

Each crew has multiple phone numbers.

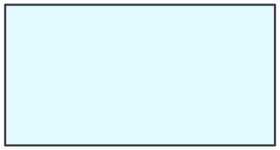
Other studios might use the same designations for crews, so the attribute crew’s number is not a key for crews.

To name a crew uniquely, we need to give both the name of the studio to which it belongs and the crew’s number.

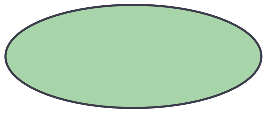
revisit

E-R Diagram: Building Blocks

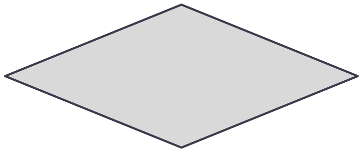
(strong) Entity set



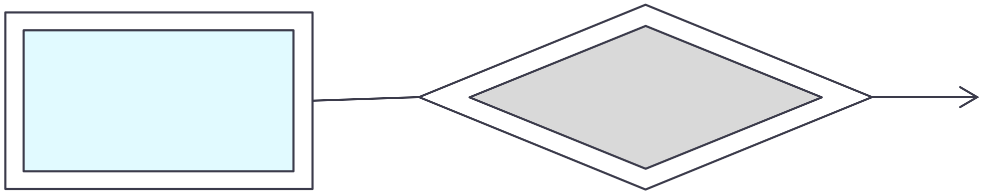
Attribute



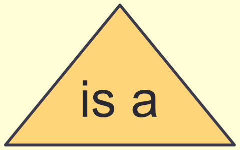
Relationship



Weak entity



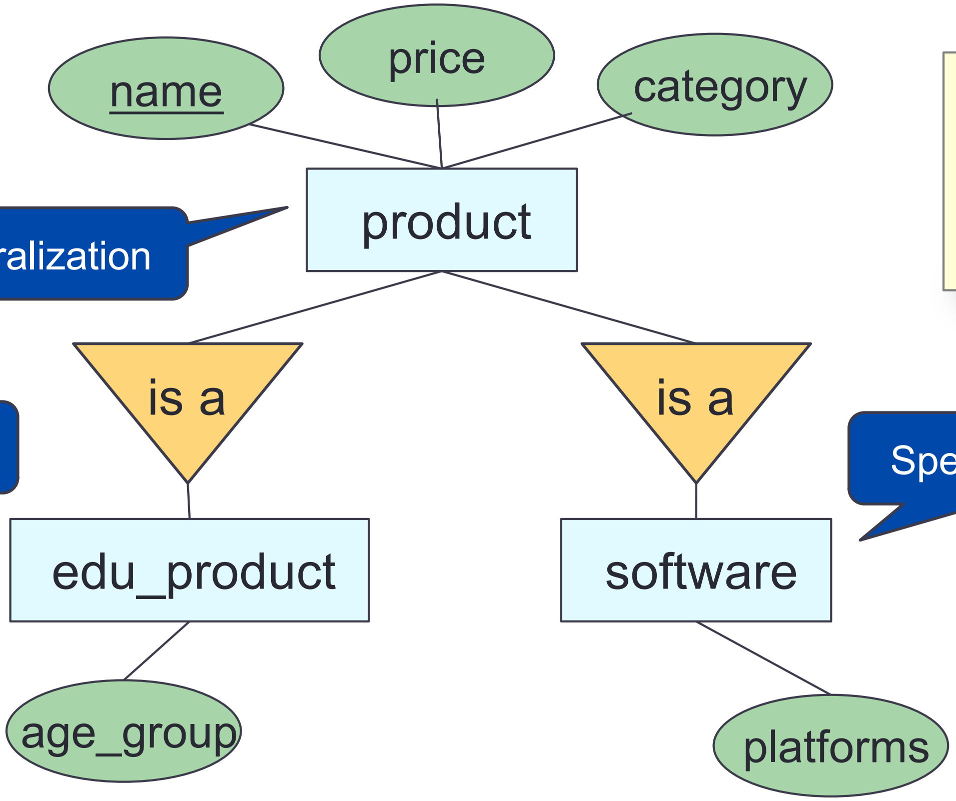
Subclass



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)



Generalization

Specialization

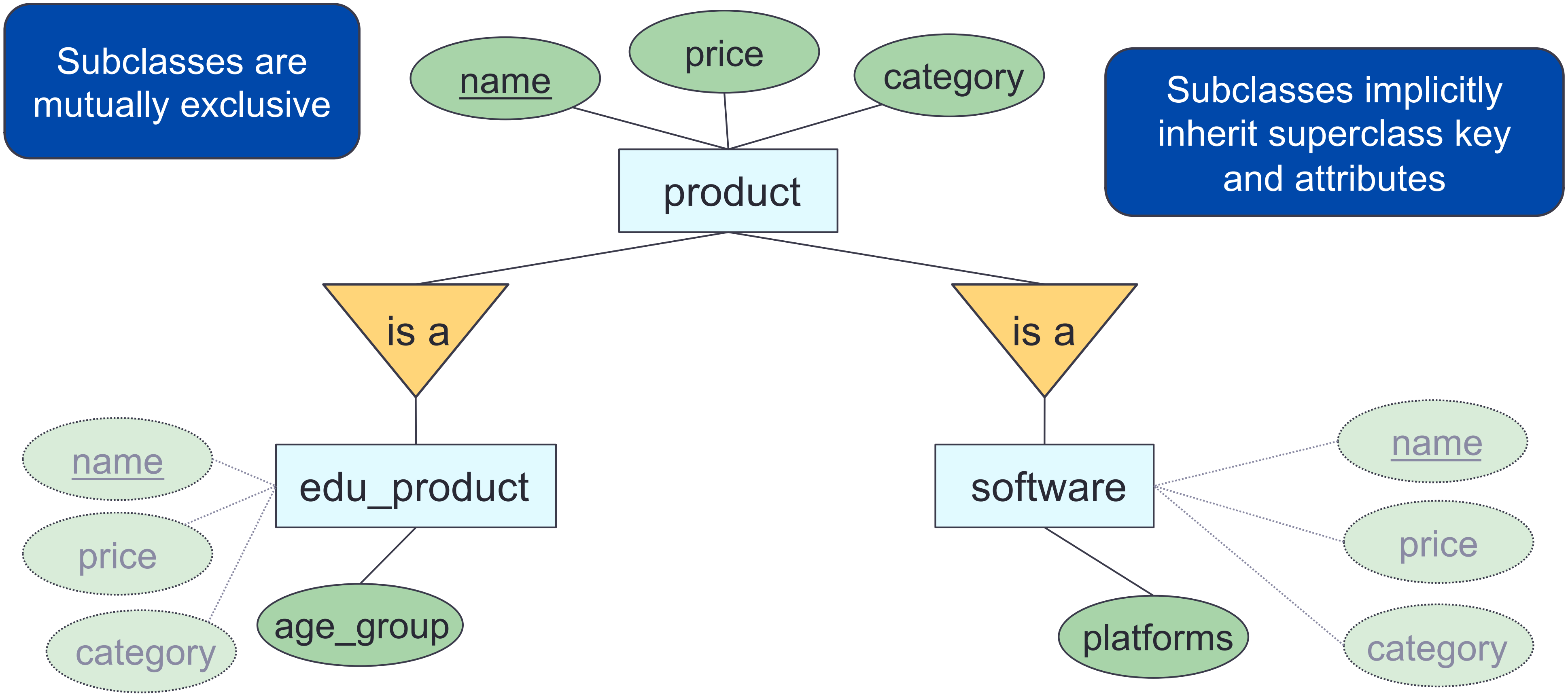
Specialization

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

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.

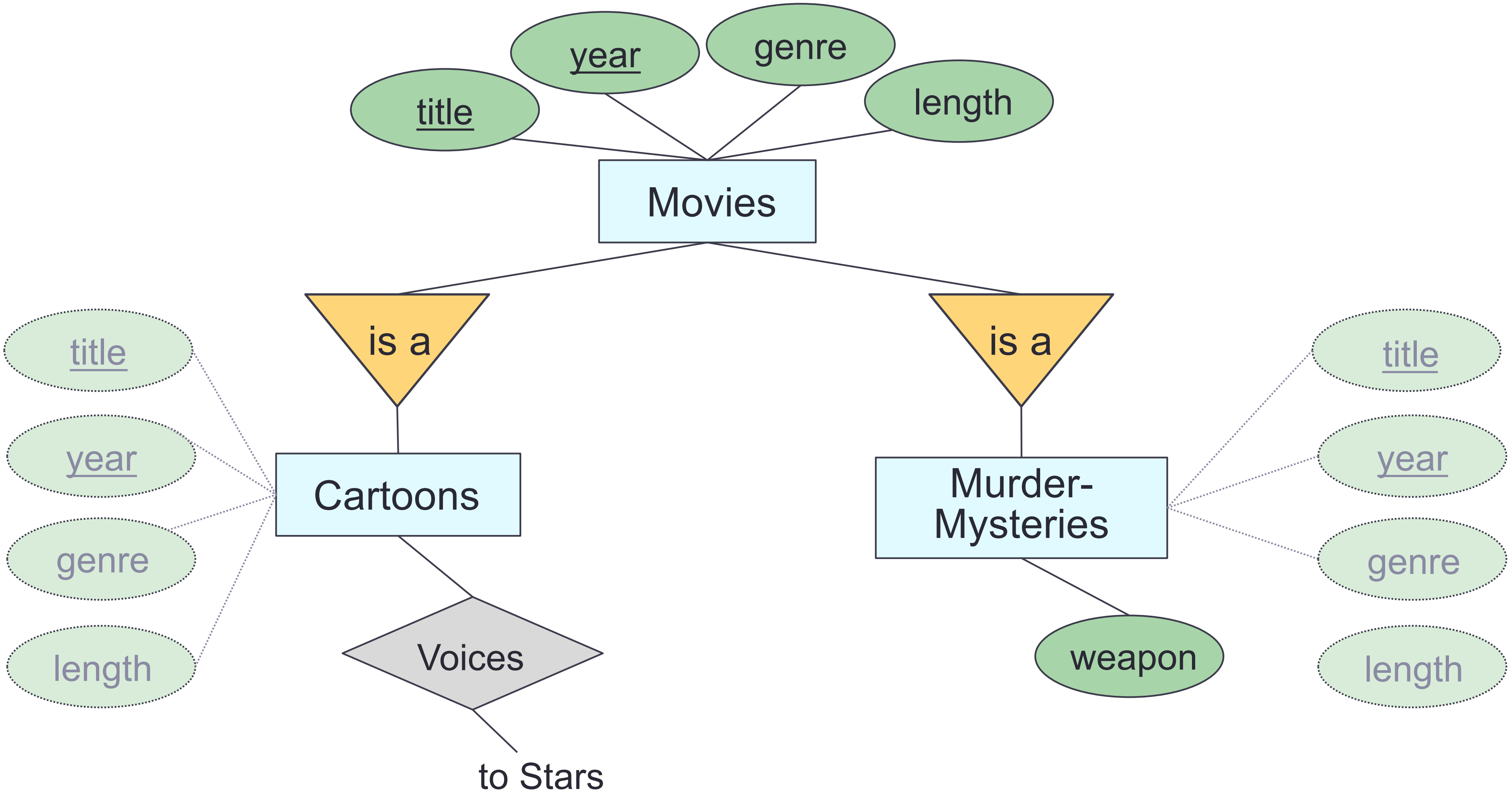
A cartoon has, in addition to the attributes and relationships of *Movies*, an additional relationship called *Voices* that gives us a set of stars who speak, but do not appear in the movie. Movies that are not cartoons do not have such stars.

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 three 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

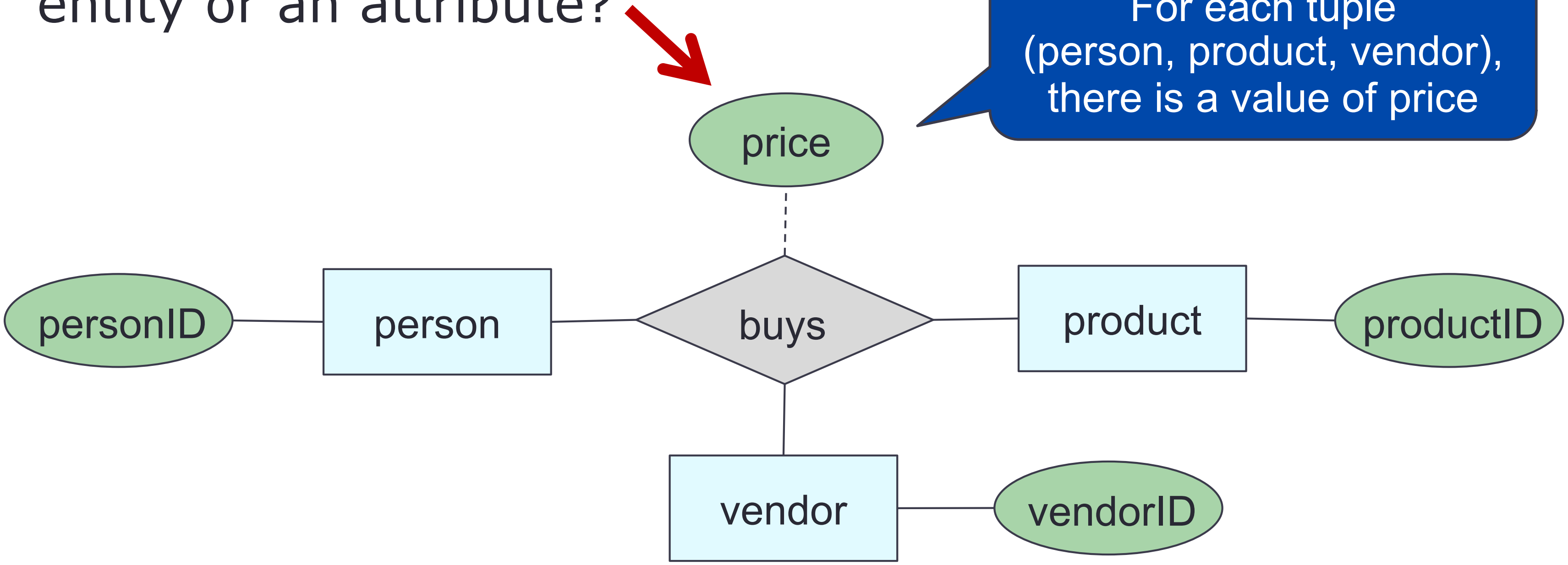
What are main differences between entities and attributes?

- Entities can model situations that attribute cannot model naturally
- Entities can participate in relationships
- Entities can have attributes
- Attributes cannot do any of these

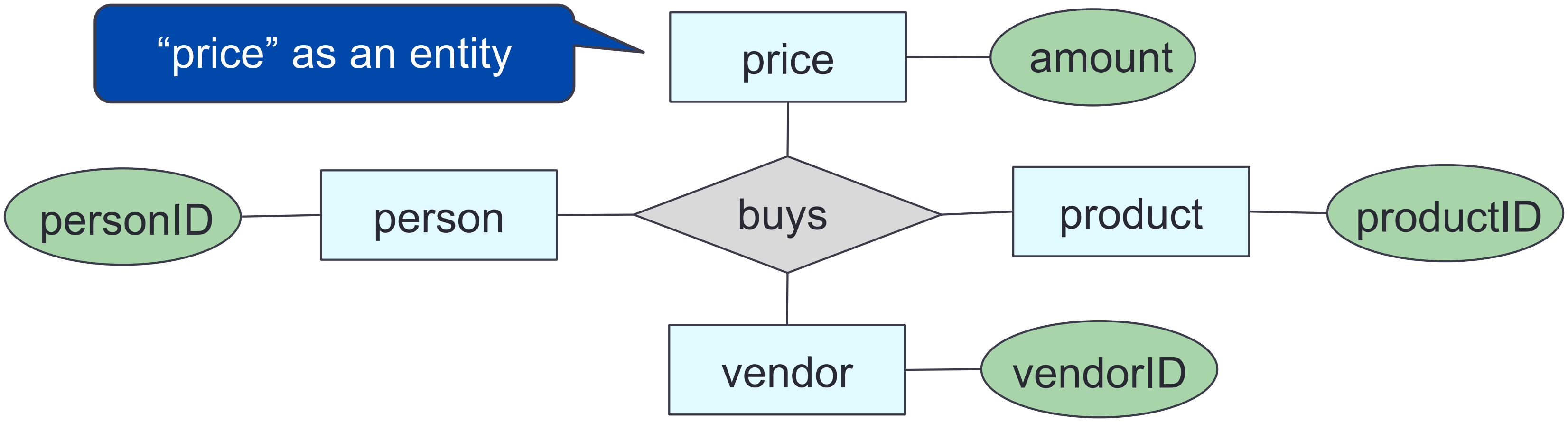
Design Decision

Should “price” be an entity or an attribute?

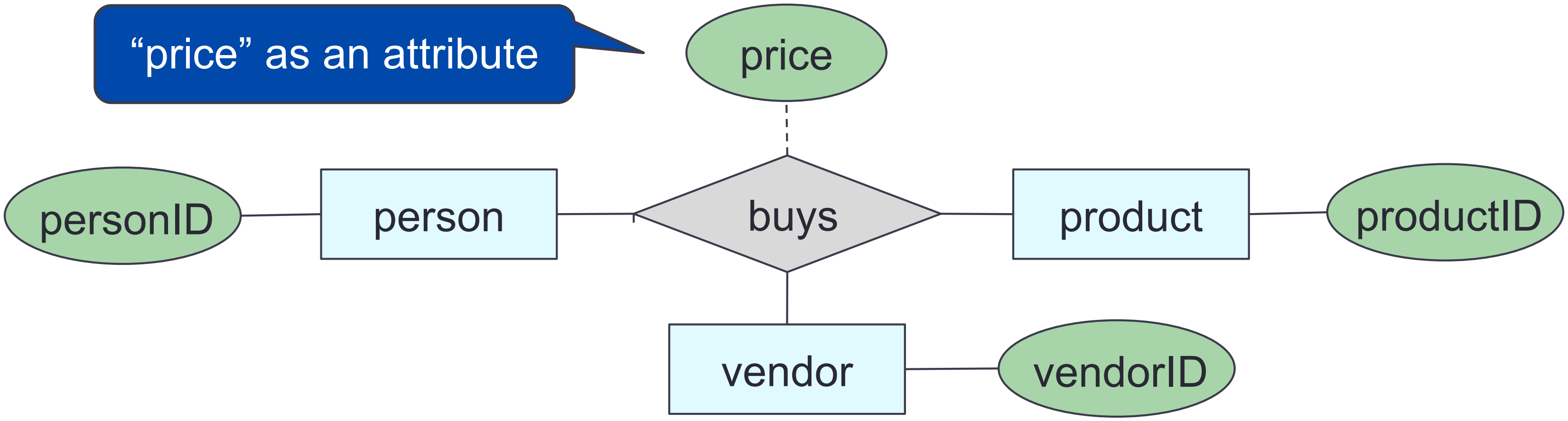
For each tuple (person, product, vendor), there is a value of price



Design Decision (2)

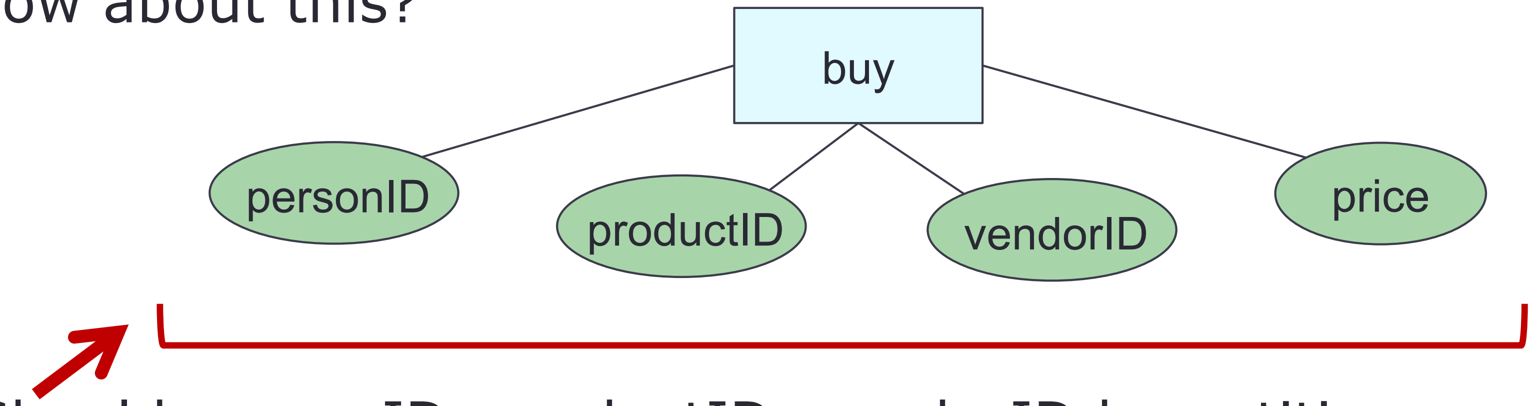

“price” as an entity

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

Design Decision (3)

How about this?



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, …)

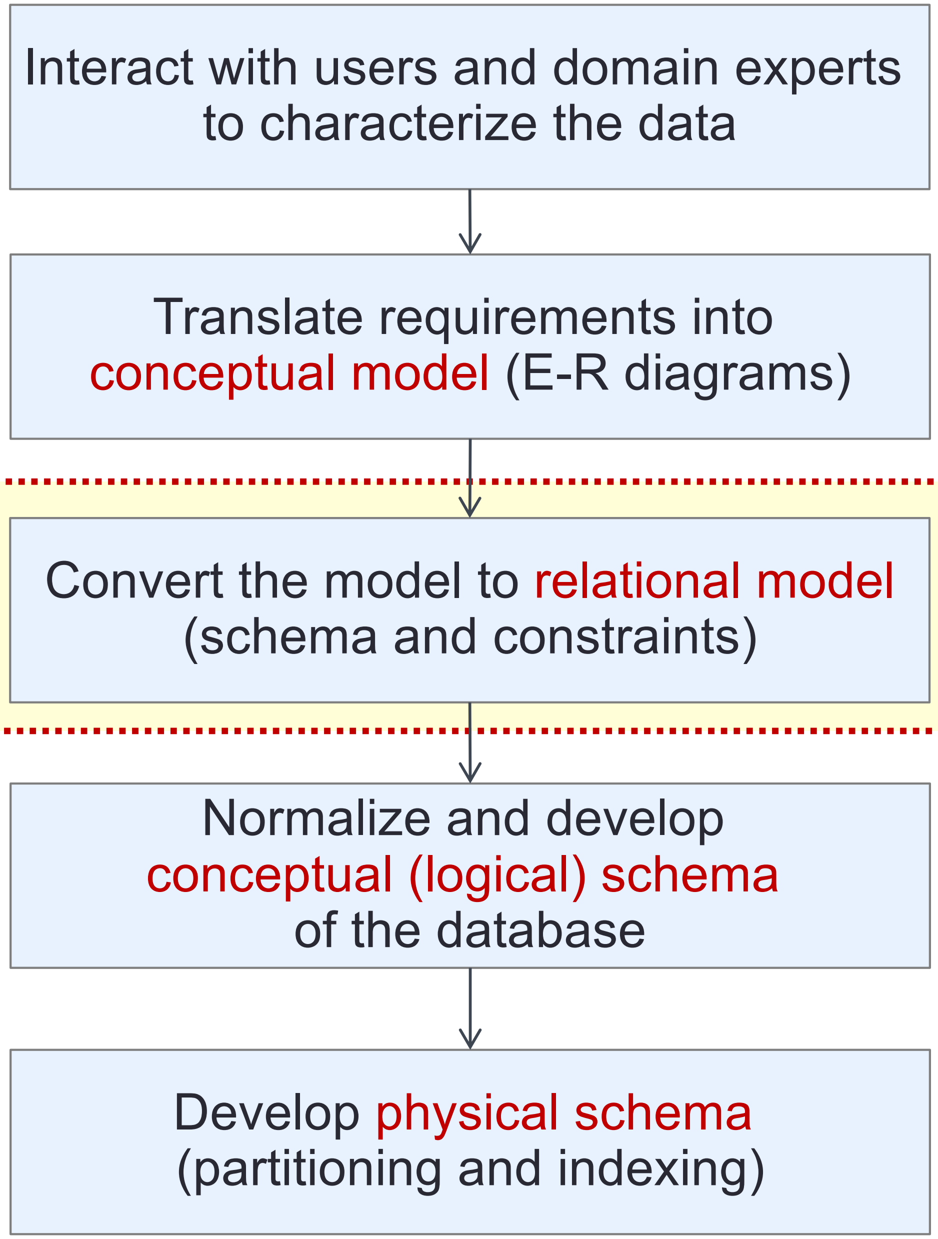
- Binary vs. n-ary relationship sets
- Specialization / generalization

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

revisit

Database Design Process



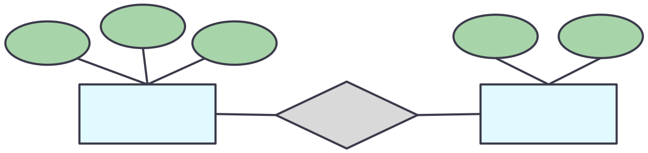
Interact with users and domain experts to characterize the data

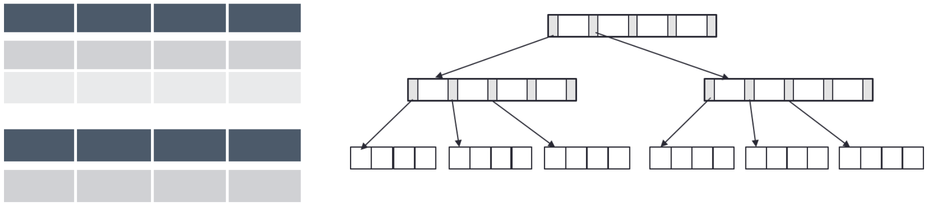
Translate requirements into conceptual model (E-R diagrams)

Convert the model to relational model (schema and constraints)

Normalize and develop conceptual (logical) schema of the database

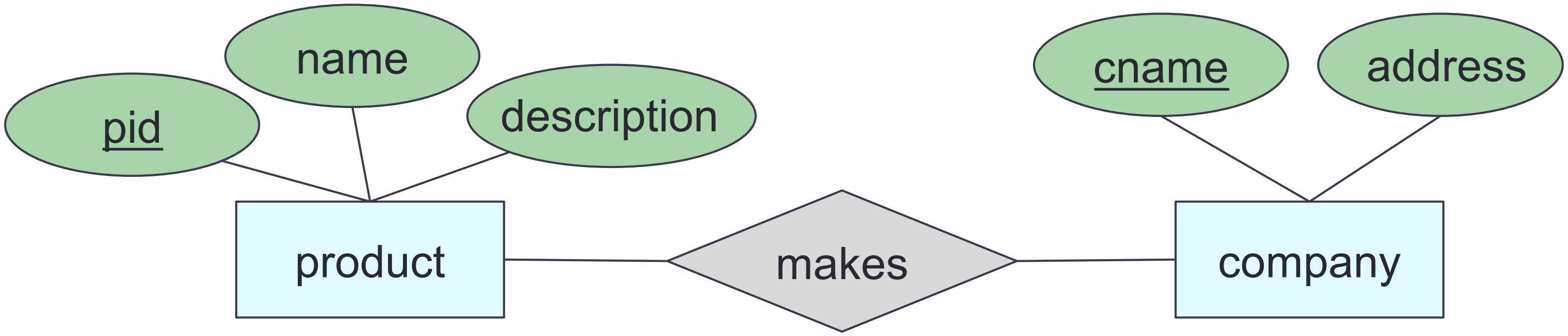
Develop physical schema (partitioning and indexing)





E-R Diagrams to Relations

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



product(pid, name, description)

company(cname, address)

makes(cname, pid)

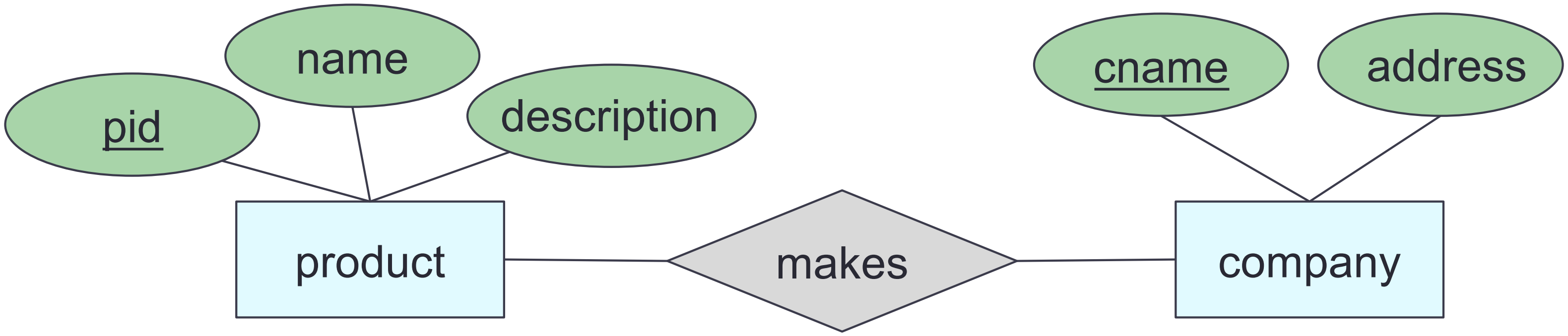
"Schema statement"

[ER to schema worksheet]

Strong Entity Set

Direct map:

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

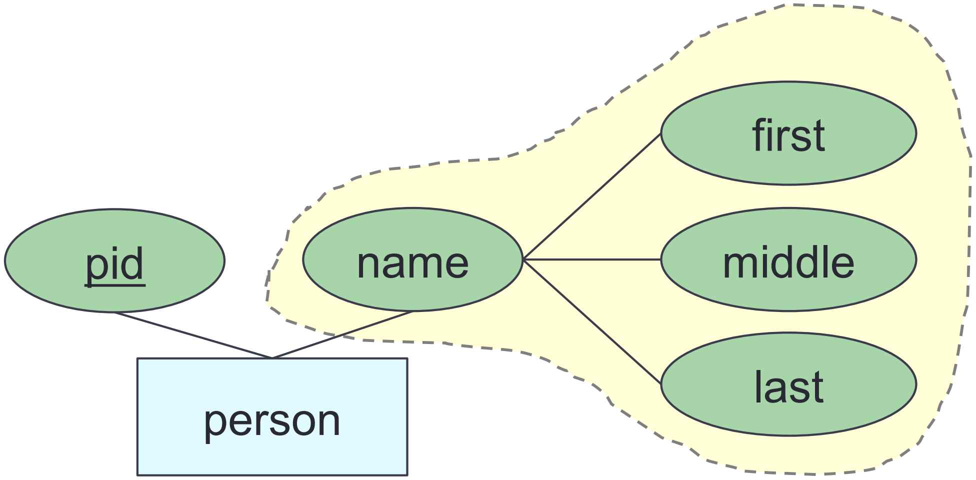


product(<u>pid</u>, name, description)
company(<u>cname</u>, address)
makes(<u>cname</u>, <u>pid</u>)

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)

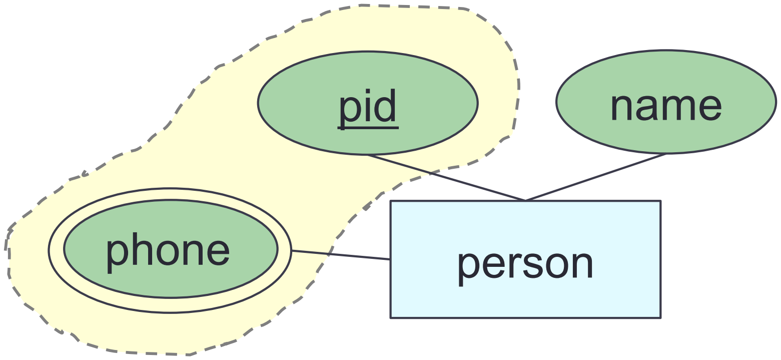
[ER to schema worksheet]

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



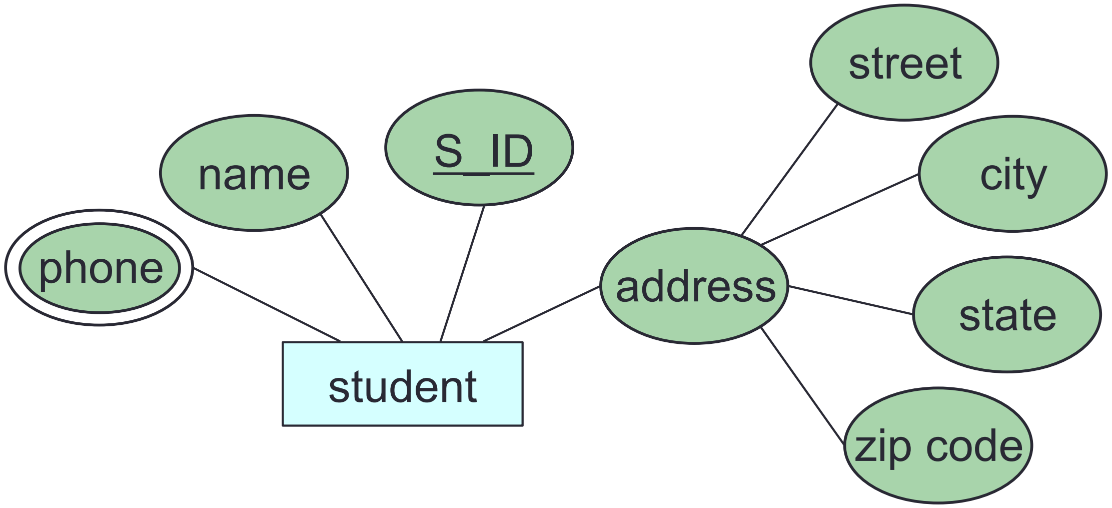
person(<u>pid</u>, name)

person_phone(<u>pid</u>, <u>phone_number</u>)

[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

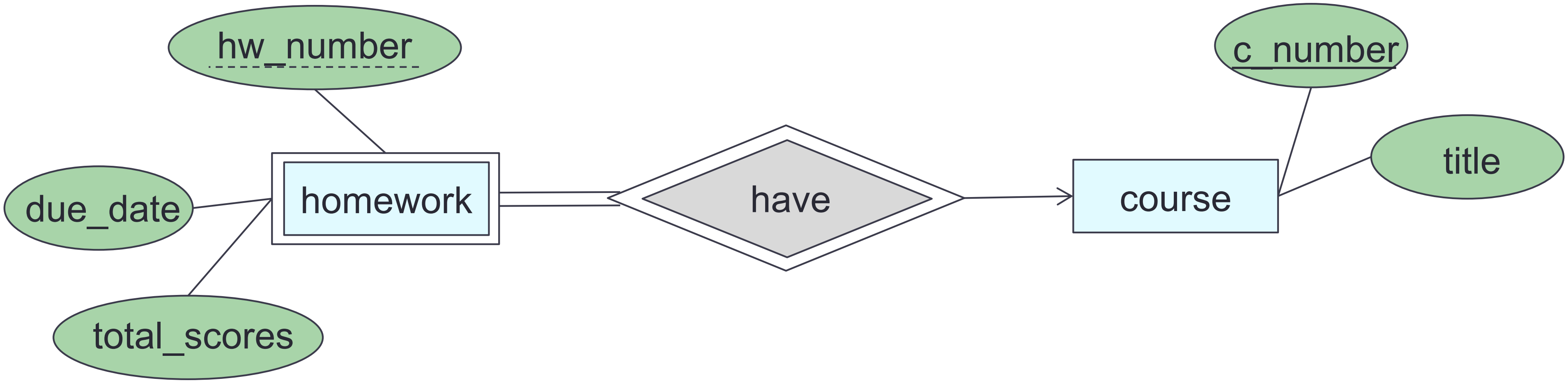
[ER to schema worksheet]

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*



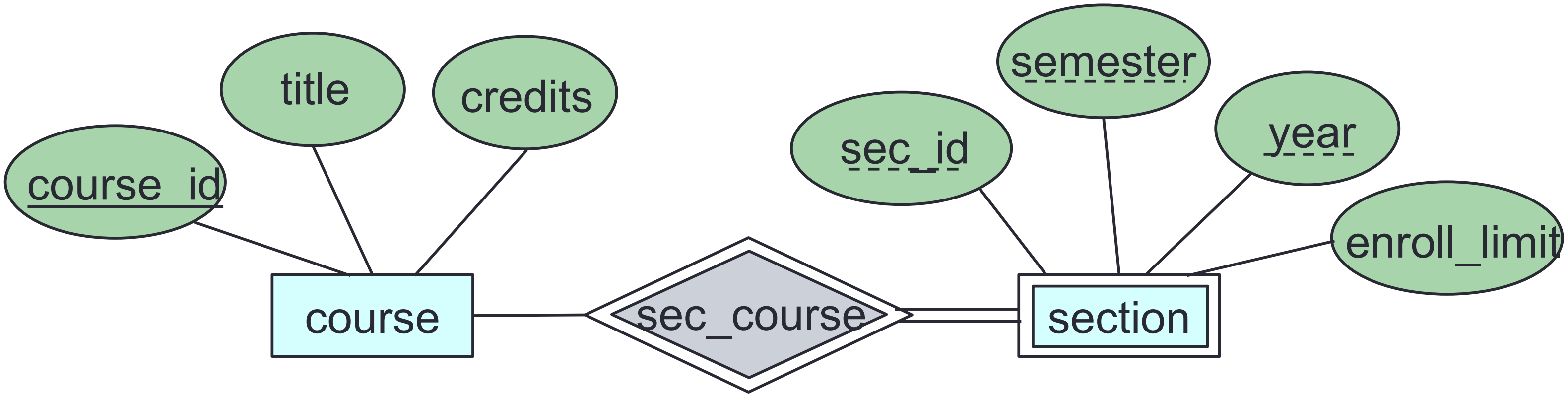
course(c_number, title)

homework(c_number, hw_number, due_date, total_scores)

[ER to schema worksheet]

Let’s try: E-R to Relations (2)

Convert the following E-R diagram into relations

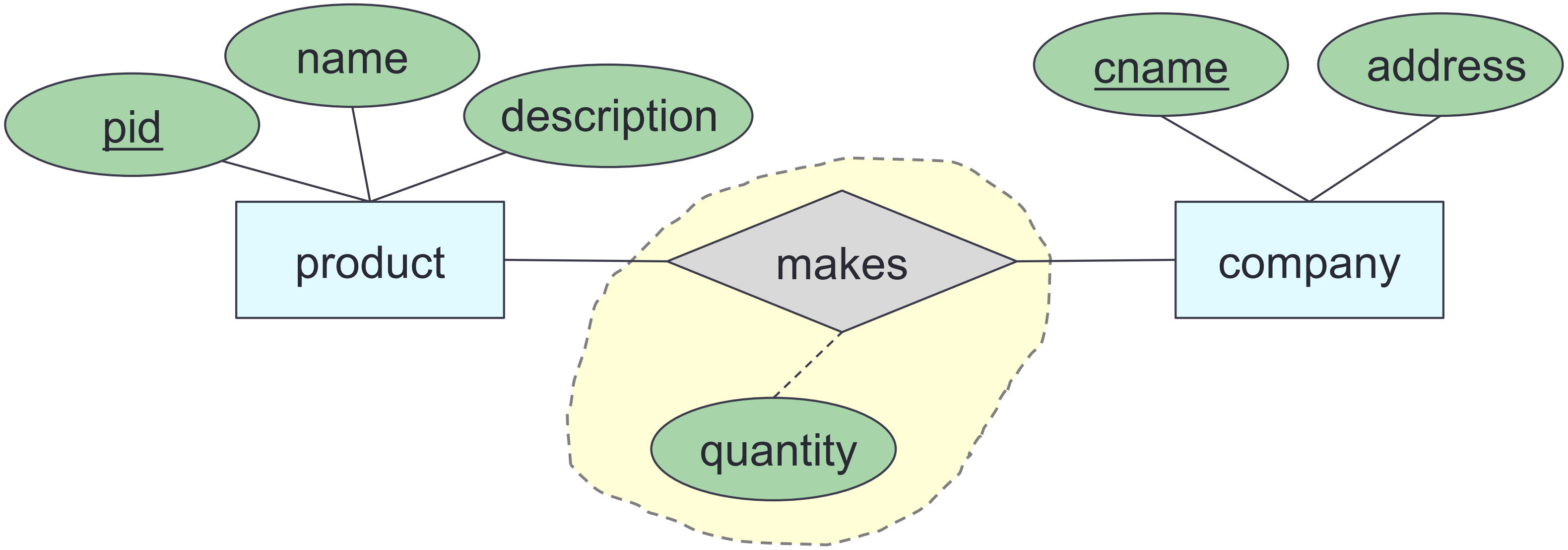


[ER to schema worksheet]

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



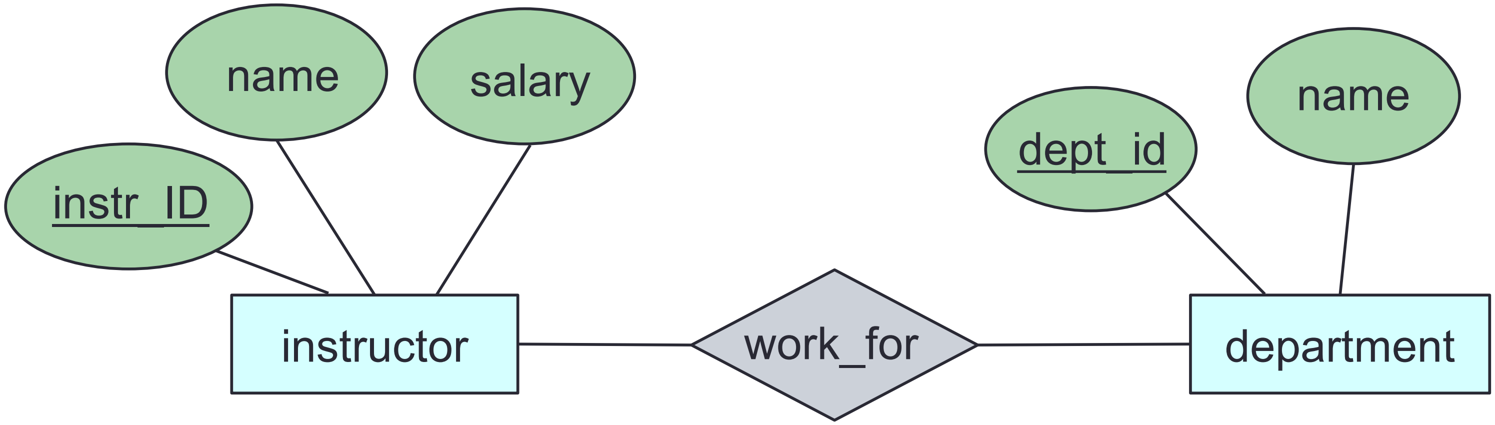
product(<u>pid</u>, name, description)
company(<u>cname</u>, address)
makes(<u>pid</u>, <u>cname</u>, quantity)

Primary keys of both entities

[ER to schema worksheet]

Let’s try: E-R to Relations (3)

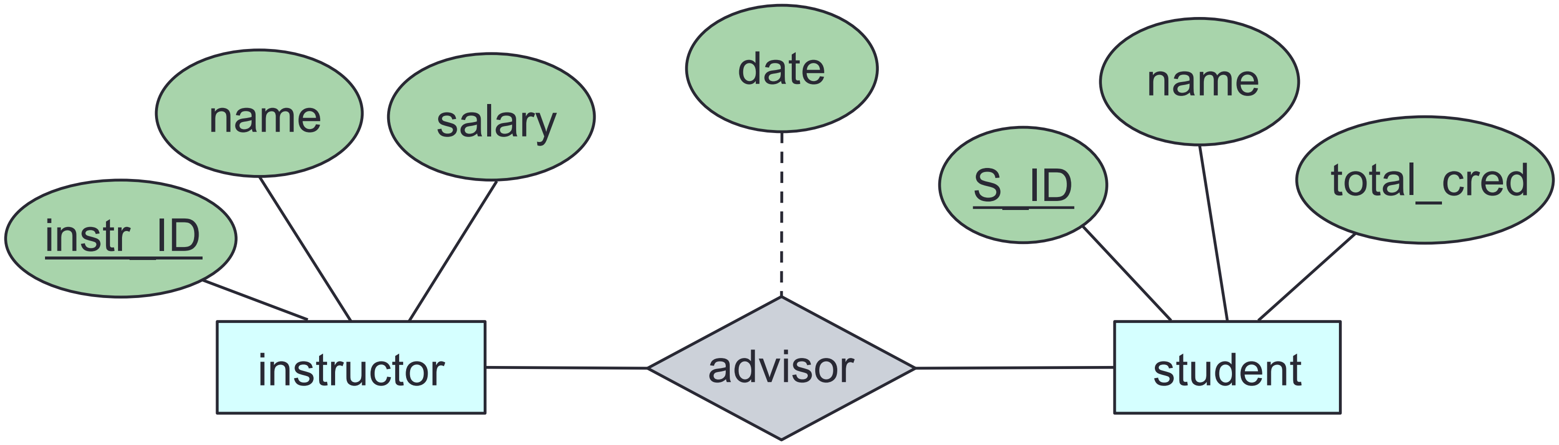
Convert the following E-R diagram into relations



[ER to schema worksheet]

Let’s try: E-R to Relations (4)

Convert the following E-R diagram into relations

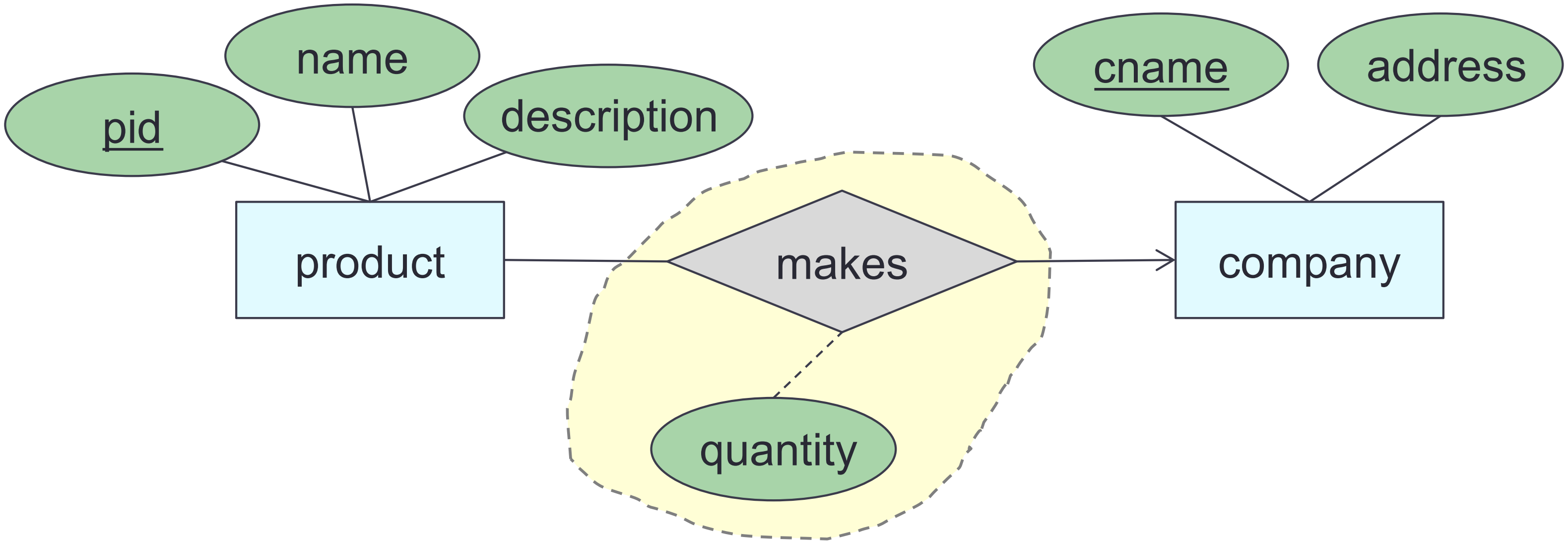


[ER to schema worksheet]

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(<u>pid</u>, name, description)

company(<u>cname</u>, address)

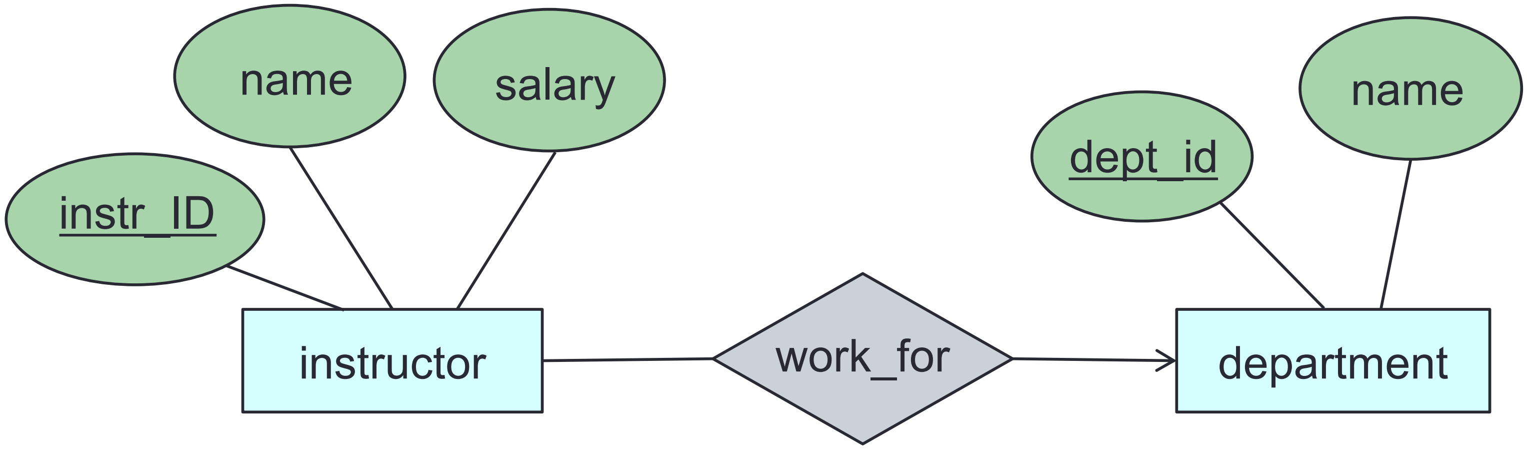
makes(<u>pid</u>, cname, quantity)

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

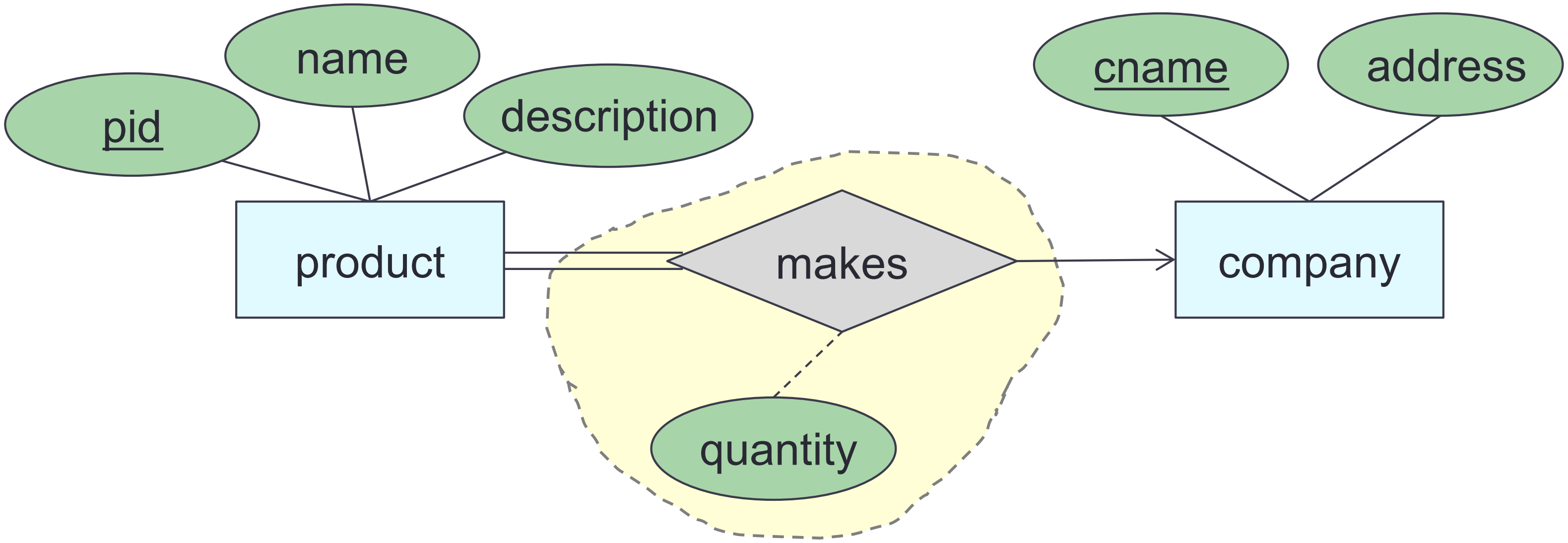


[ER to schema worksheet]

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

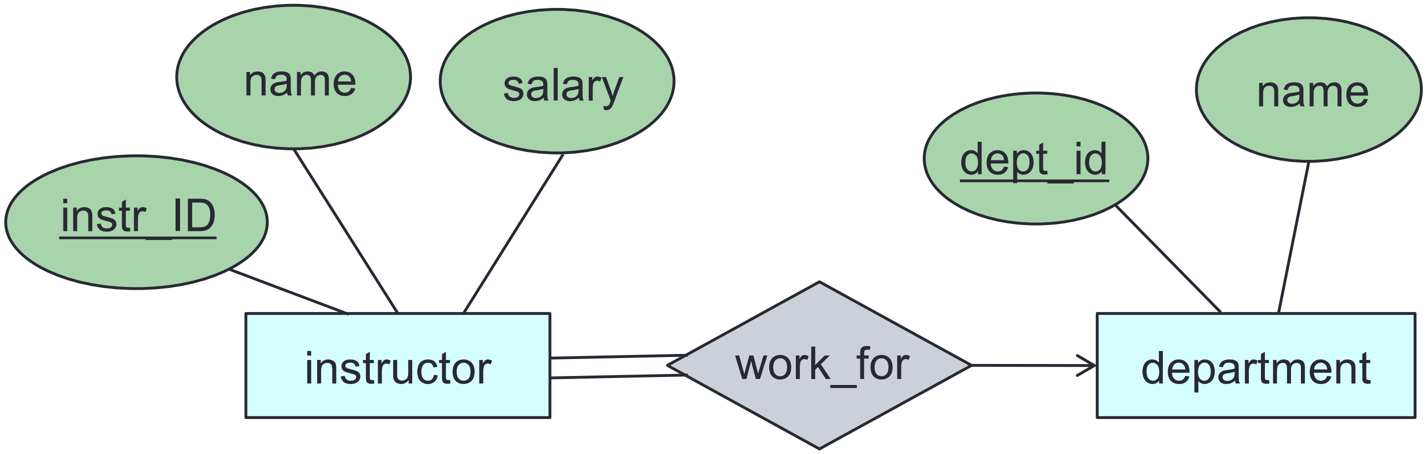


product(<u>pid</u>, name, description, cname, quantity)
company(<u>cname</u>, address)

[ER to schema worksheet]

Let’s try: E-R to Relations (6)

Convert the following E-R diagram into relations



[ER to schema worksheet]

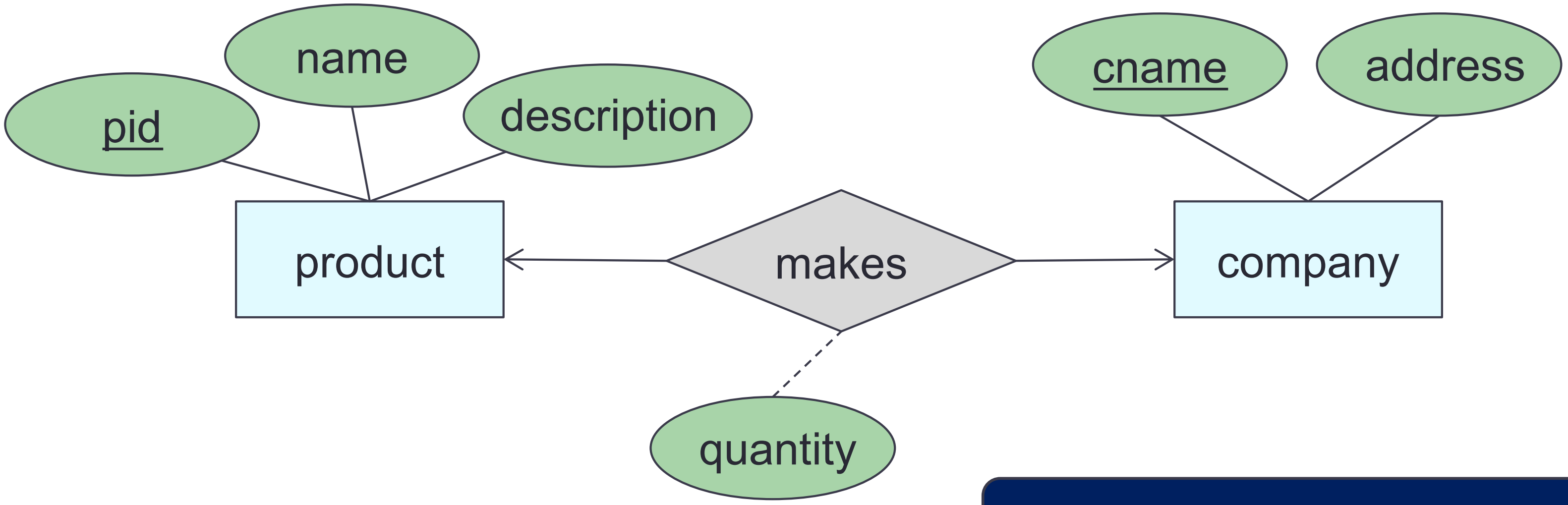
Relationship Set: One-to-One

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



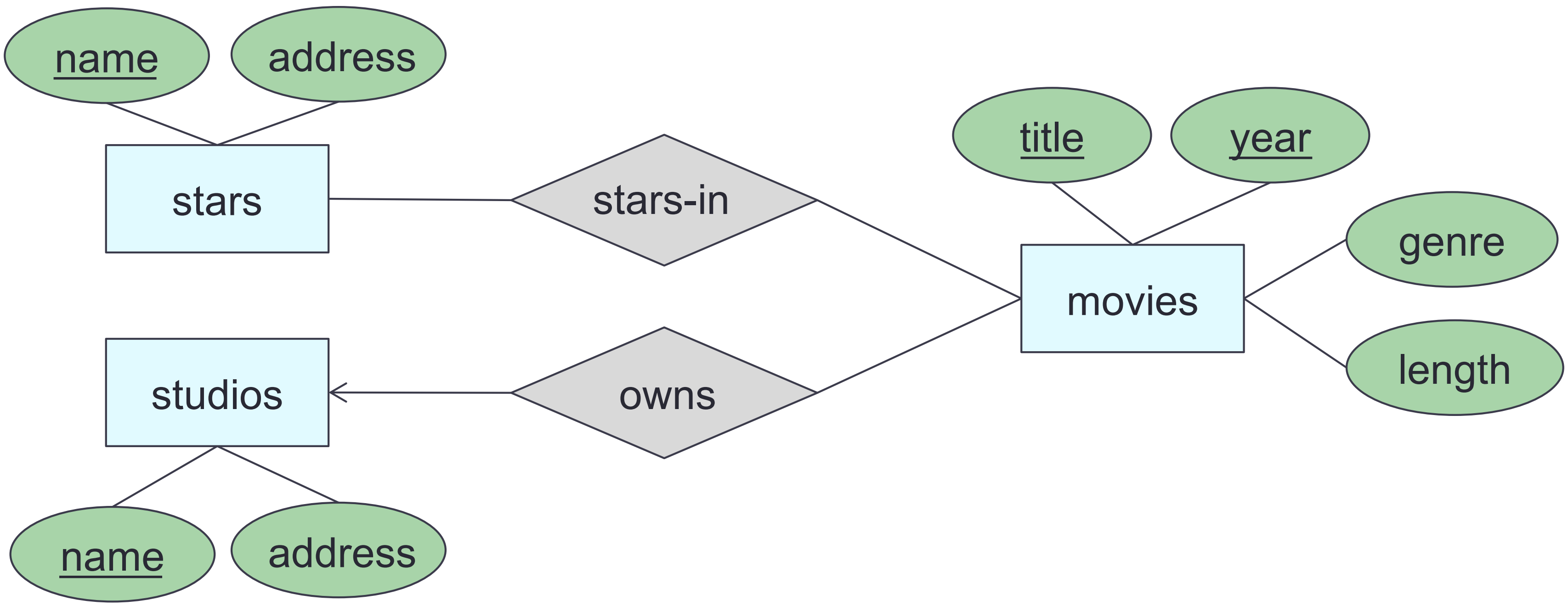
Primary key of the chosen entity

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

[ER to schema worksheet]

Let’s try: E-R to Relations (7)

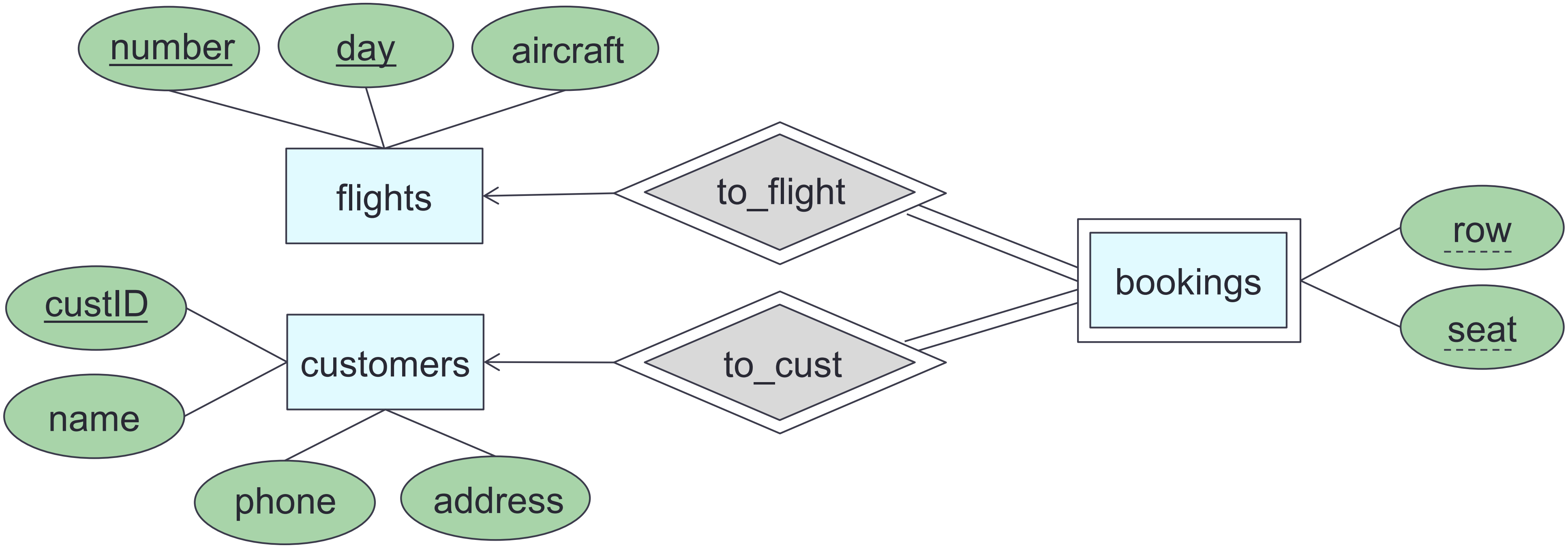
Convert the following E-R diagram into relations



[ER to schema worksheet]

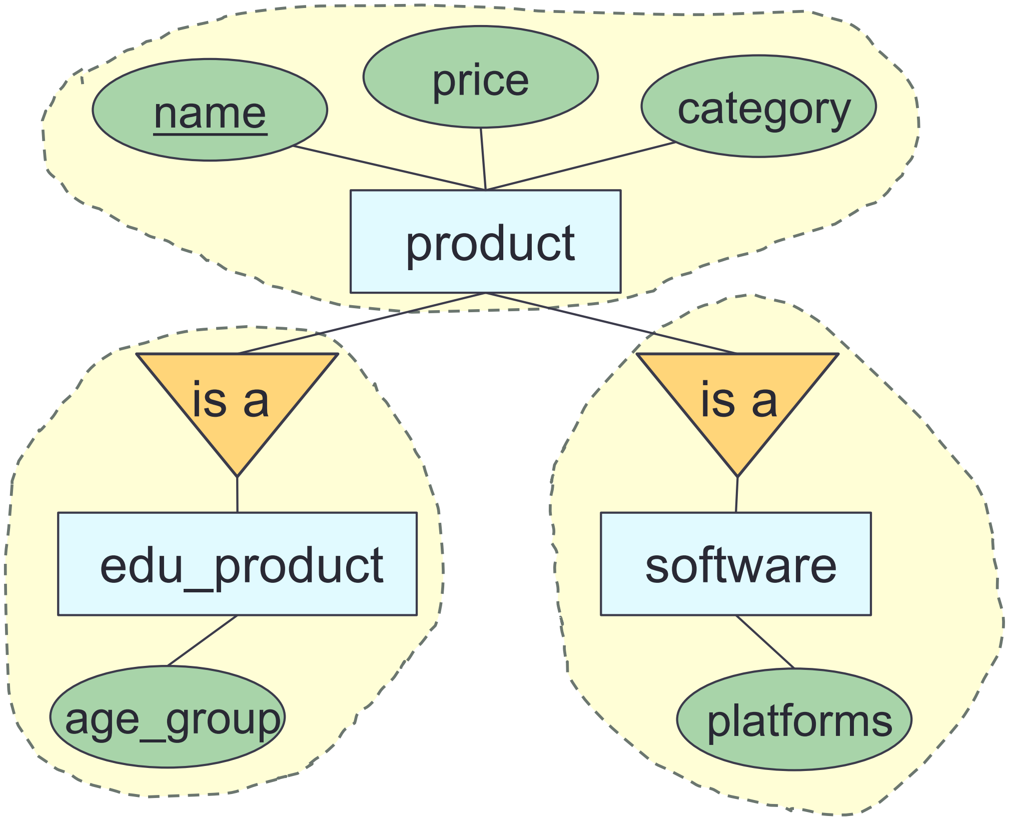
Let's try: E-R to Relations (8)

Convert the following E-R diagram into relations



[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)

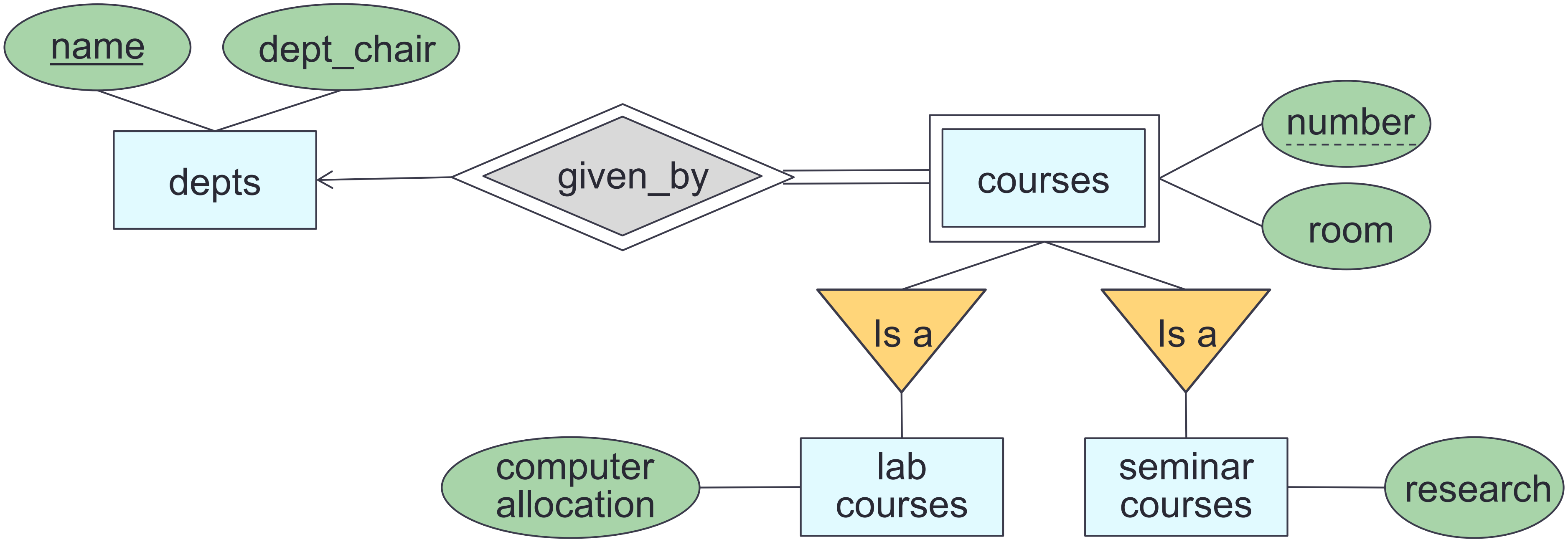
software(<u>name</u>, platforms)

Keep everything

[ER to schema worksheet]

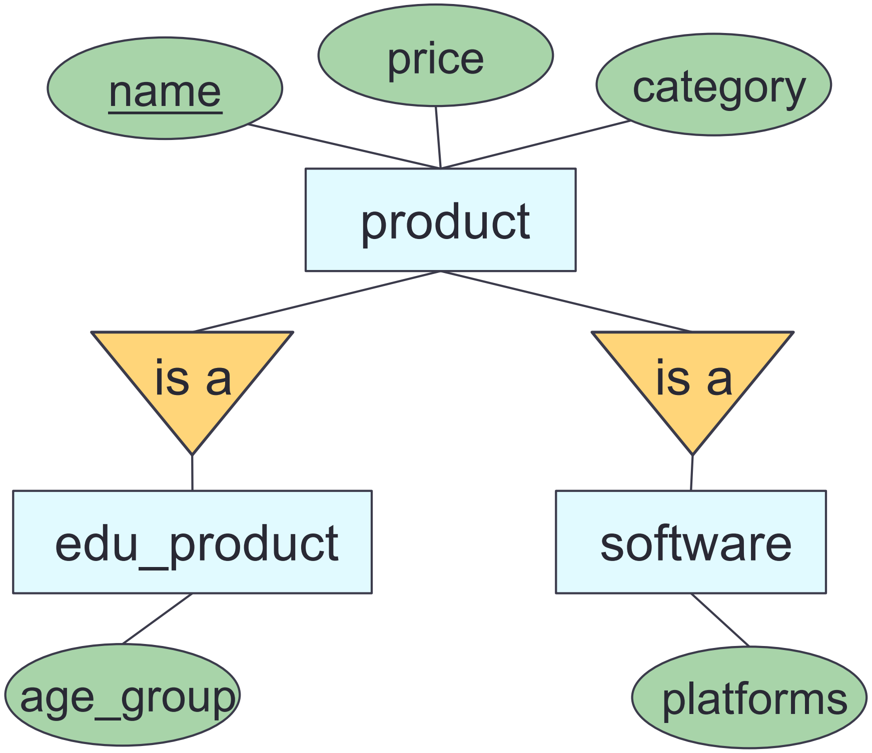
Let’s try: Subclasses (option 1)

Convert the following E-R diagram into relations



[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)

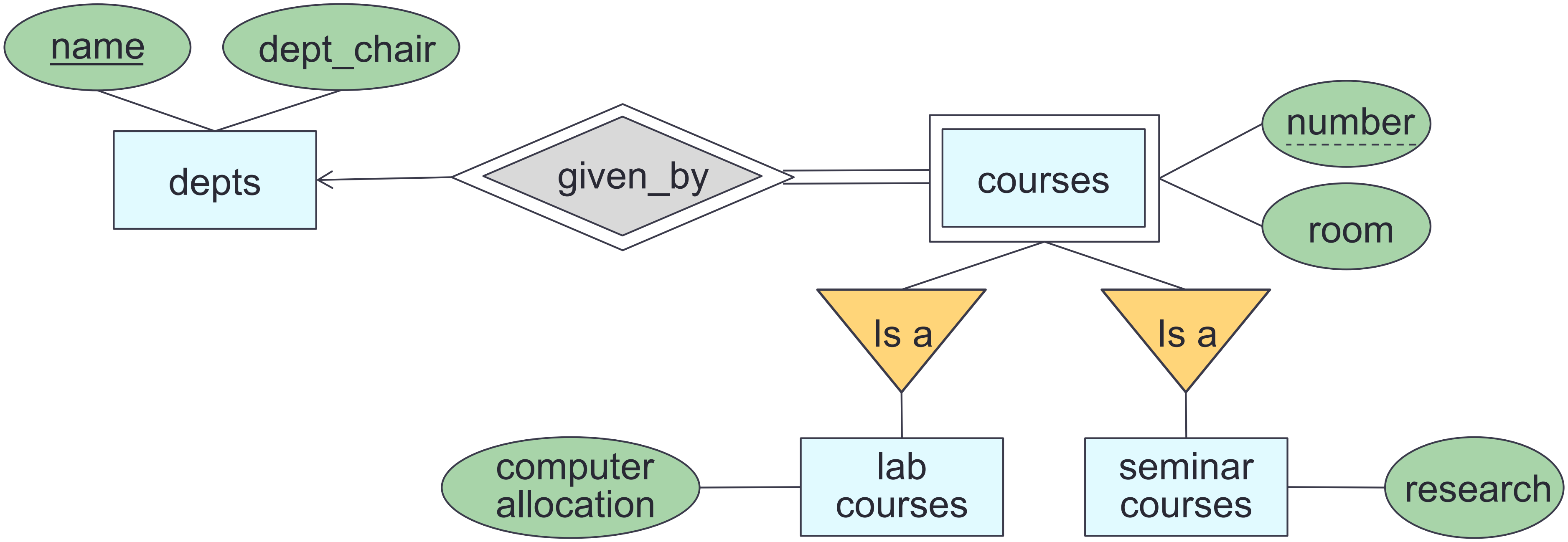
software(name, price, category, platforms)

Push down

[ER to schema worksheet]

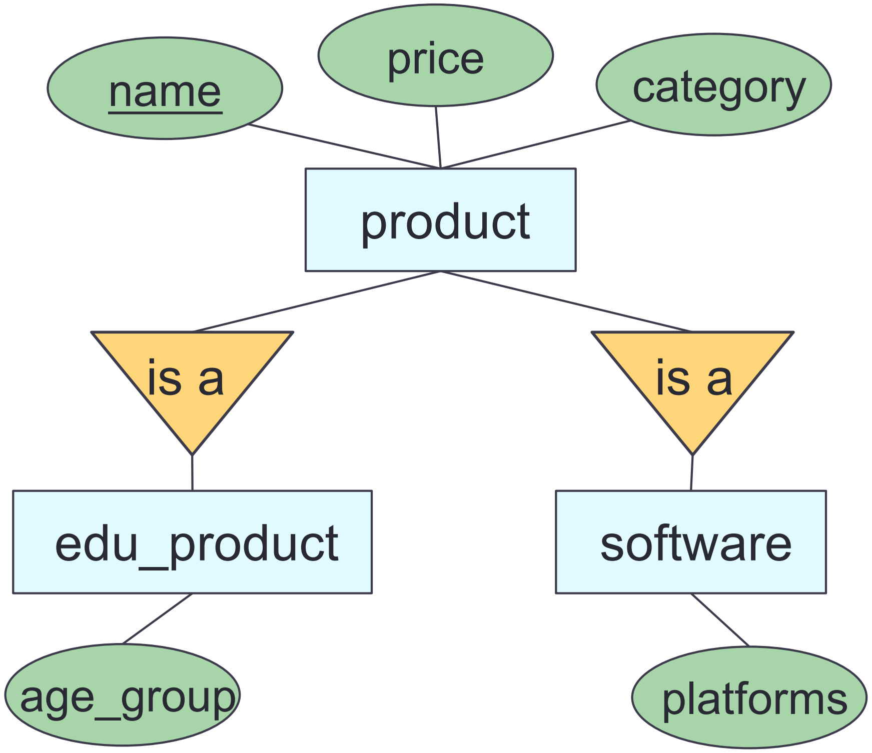
Let’s try: Subclasses (option 2)

Convert the following E-R diagram into relations



[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

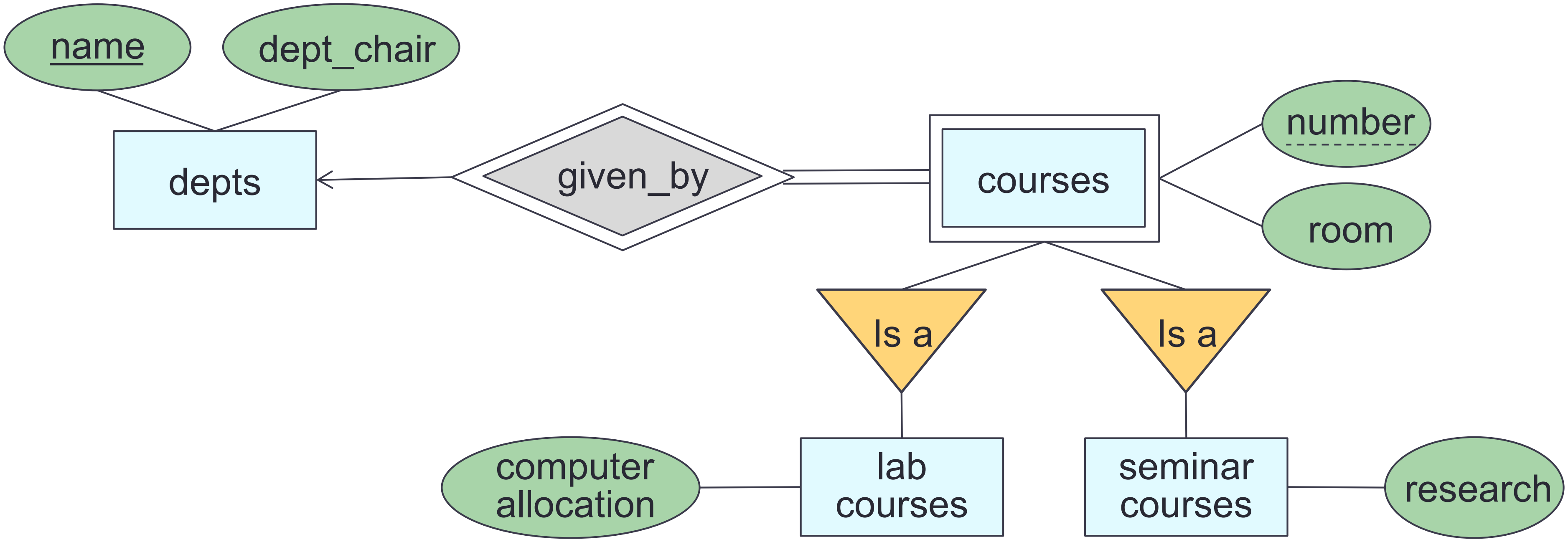
Push up

product(name, price, category, age_group, platforms)

[ER to schema worksheet]

Let’s try: Subclasses (option 3)

Convert the following E-R diagram into relations



[ER to schema worksheet]

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