

# Conceptual Modeling

## Entity Relationship Diagrams

**Orlando Karam**  
**[okaram@spsu.edu](mailto:okaram@spsu.edu)**

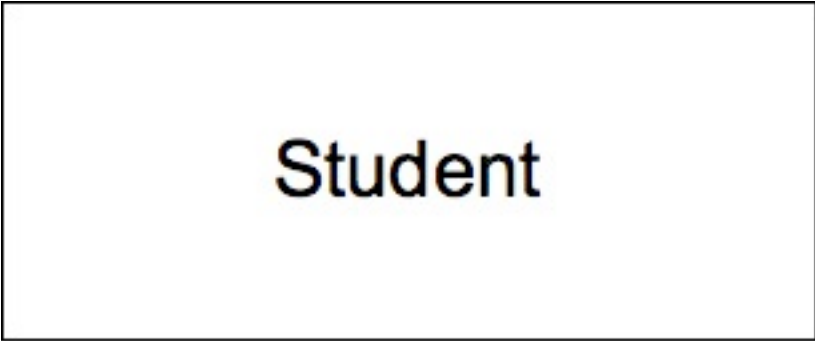
- Business Rule: Statement that constrains certain aspects of the business
  - Assert business structure or
  - Control business processes
  - Basically, Requirements from Business
- Our DB App will (hopefully) automate business rules

# ER modeling

- An ER Model is based on Entities, Relationships among entities, and attributes of entities and relationships
- An ER Diagram is a graphical representation of an ER Model

# Entities

- Things in the real world, physical or not
- Entity type is set of entities that share properties or characteristics
- Entity Instance is each of the instances of an entity type
- Represented by rectangle

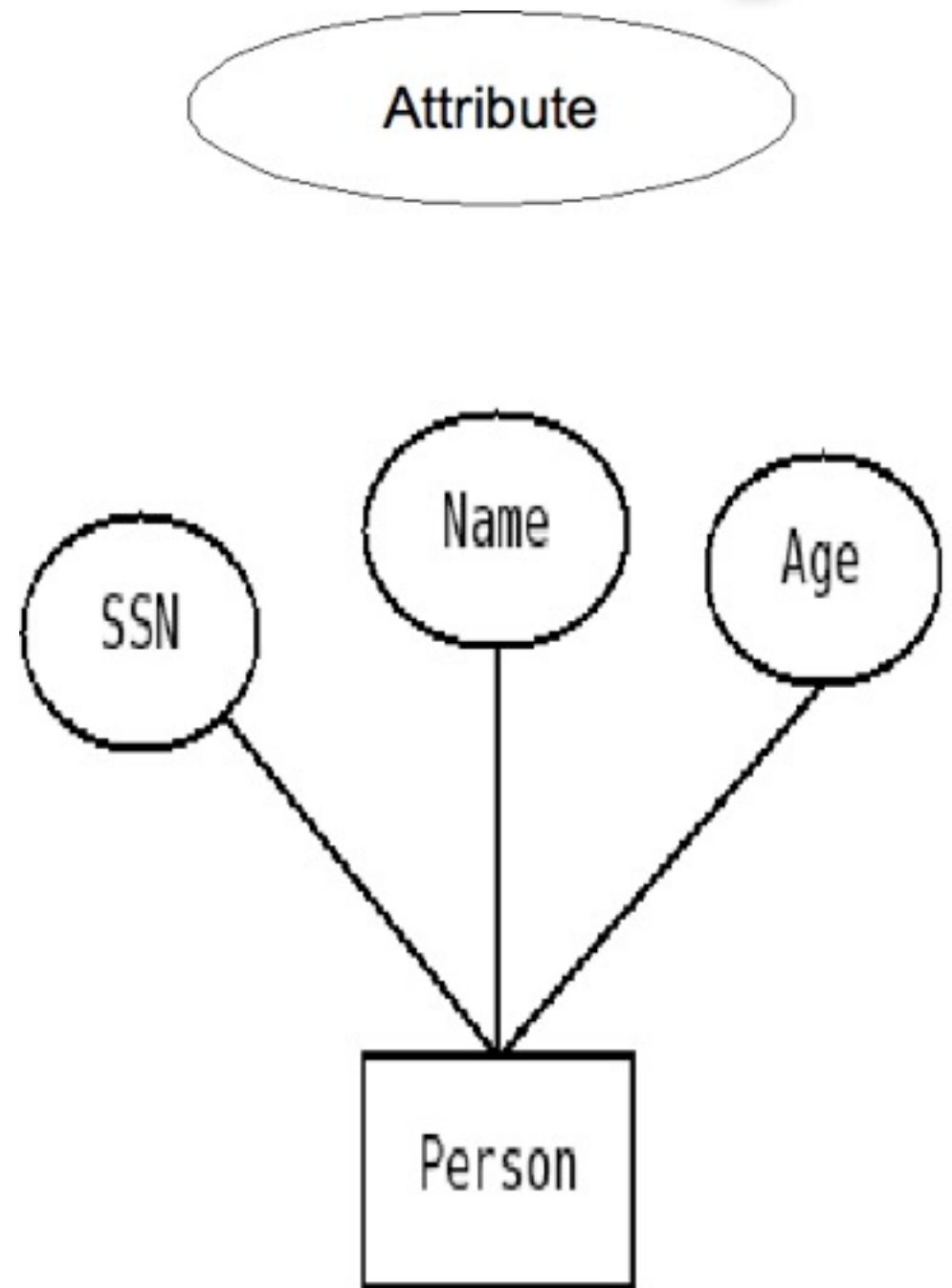
A diagram showing a rectangle representing an entity type. The word "Student" is centered inside the rectangle.

Student



# Attributes

- Properties of entities or relationships
- Represented by oval
- Can be (at the same time)
  - Optional or mandatory
  - Simple or composite
  - Single- or multivalued
  - Stored or Derived
- We also need to define an *identifier* which is a set of attributes for which no two entities of a given type will have the same value

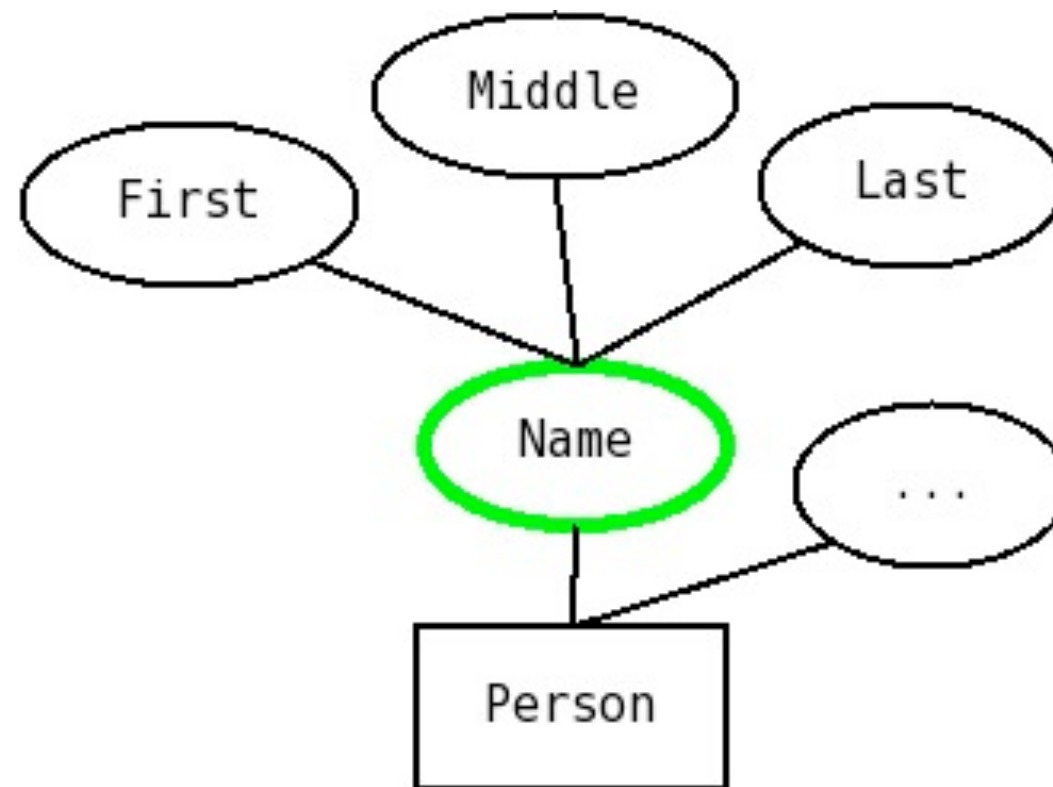


# Optional vs Mandatory

- For optional attributes, some entities may have no value at all.
- For mandatory attributes, a value has to be entered in DB.
- Not represented in ER diagram.
- By default, consider all attributes optional
- Examples of Mandatory: SSN, Name (for many applications)

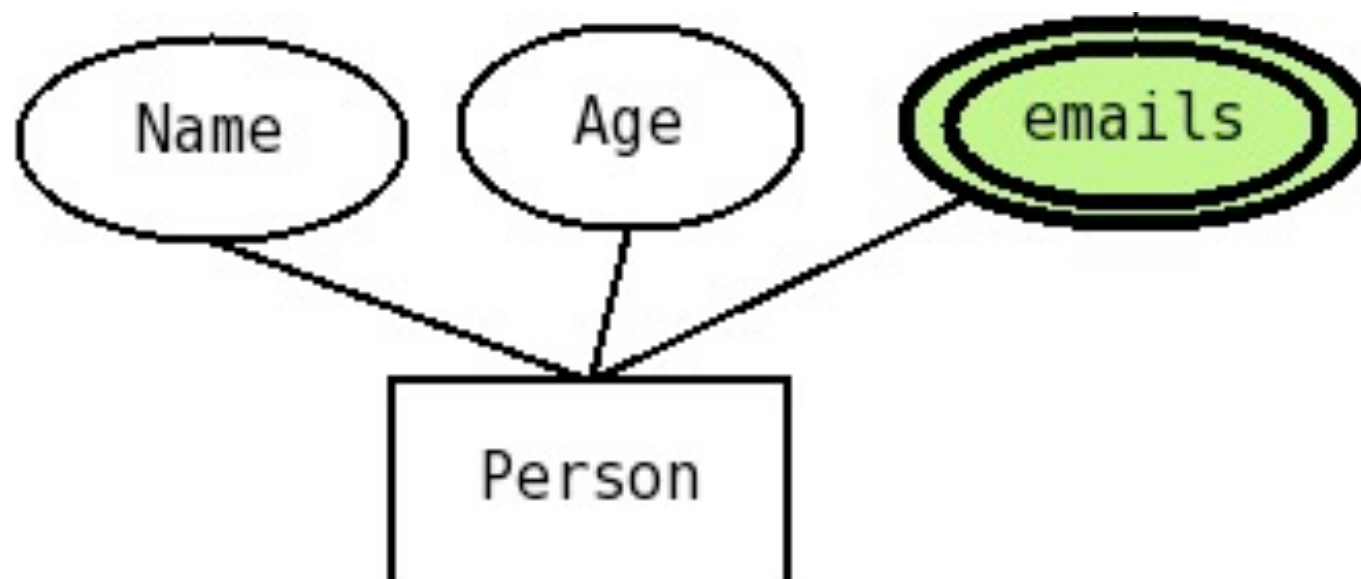
# Simple vs Composite

- Composite attributes can be broken-up into meaningful parts.
- Each part has a name, and they could be of different types.



# Single-valued vs multi-valued

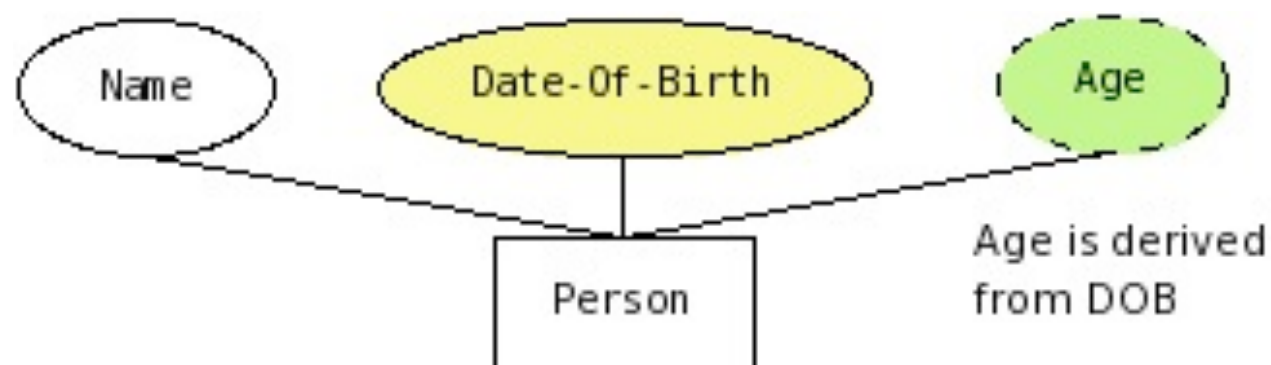
- For some attributes, we may have more than one value for the same entity.
- Here, all values are of the same type, and they are not identified by name; similar to an array or a set.
- Represented by double lines





# Derived vs Stored

- Many times we can calculate an attribute from some other attributes stored in the database
- We call those *derived* attributes (and use dash-lines around them).
- Sometimes you can choose which ones to store or derive



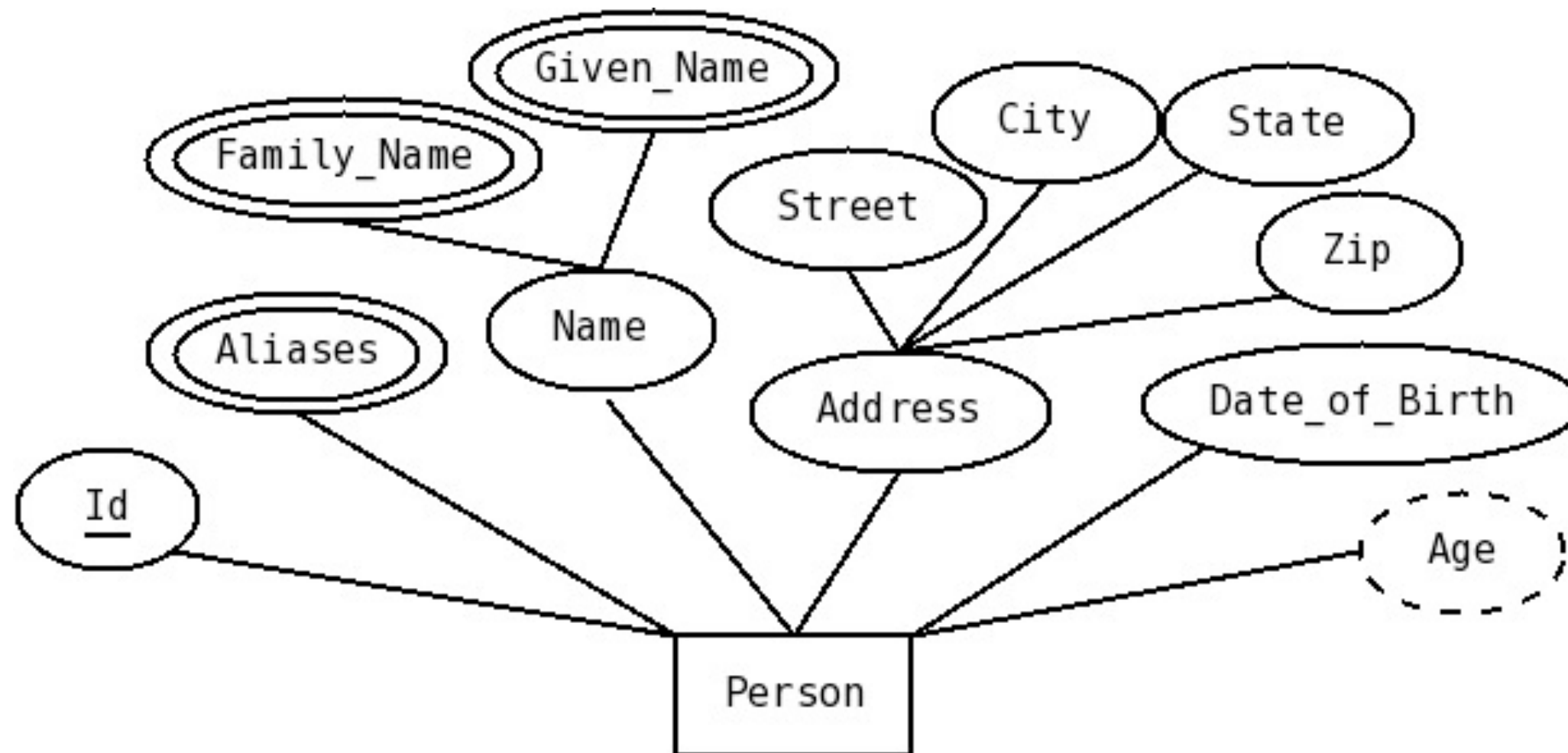
# Practice: Person

SPSU

- Produce an ER diagram for the following situation:
- We have only one entity, called **Person** (everything else is represented as attributes), with the following attributes: **Id** (the identifier); **Name**, which is composed of one or more **given names** and one or more **family names**; One or more **aliases**, an **address** (composed of street, city, state, zip); **date-of-birth**; and **age**, which can be calculated from the date of birth.

# Solution: Person

SPSU





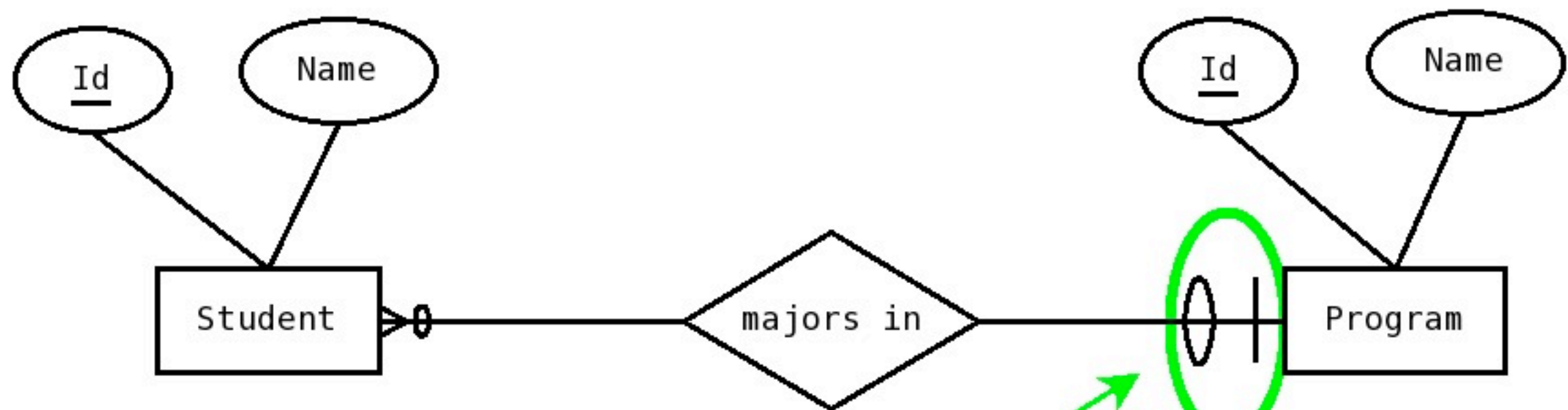
# Relationships

- Association between two or more entities
- Represented by diamond
- Degree: number of entity types that participate (mostly binary)
- Cardinality constraints



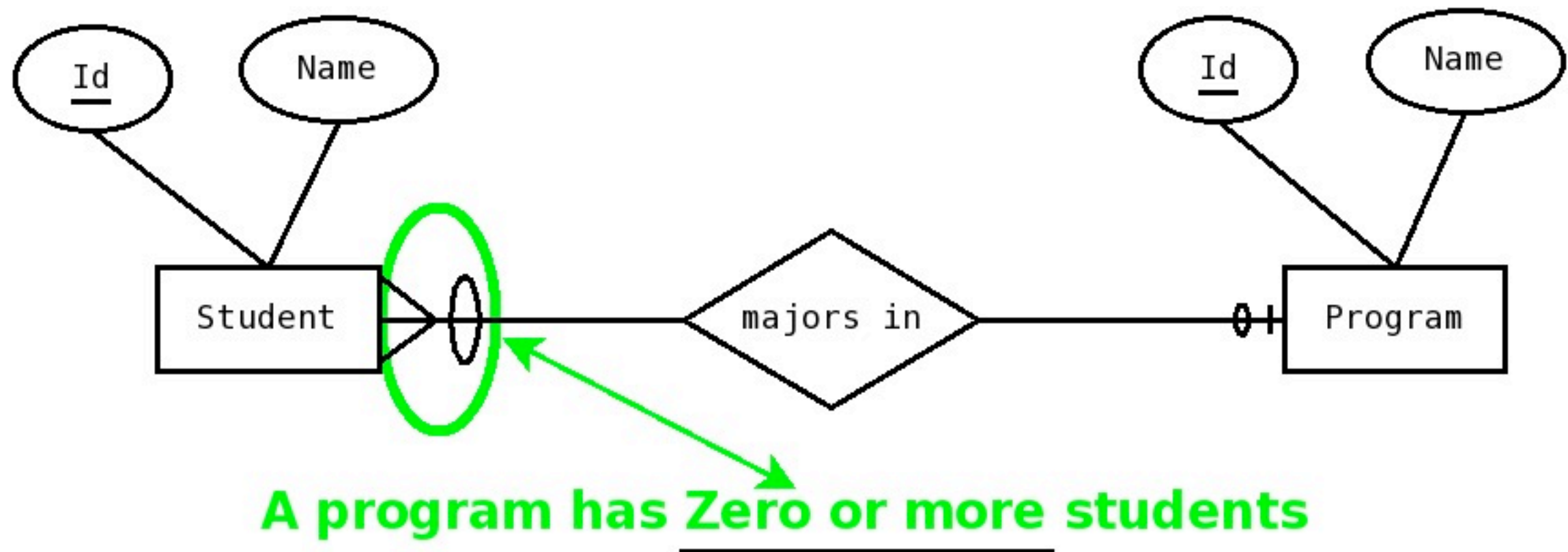


# Interpreting Cardinalities



**A Student majors in Zero or 1 programs**

# Interpreting Cardinalities

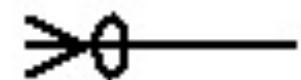


# Cardinality Constraints

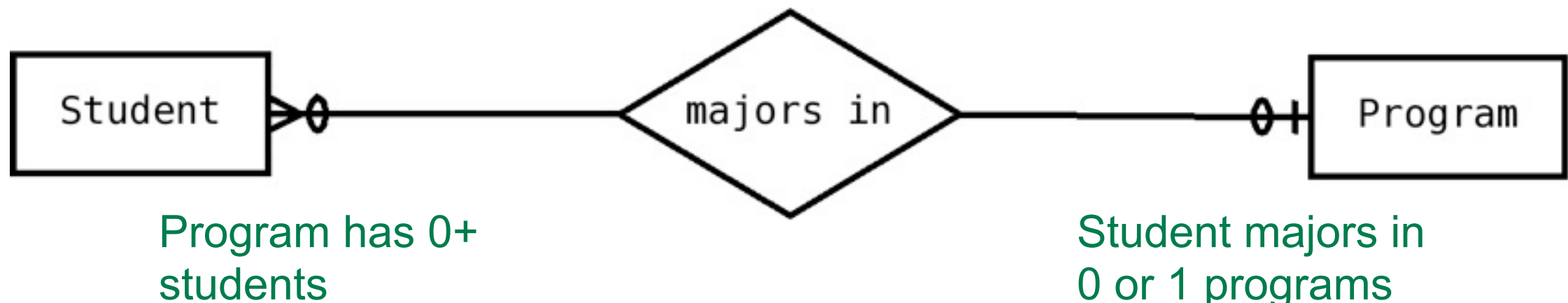
Constrain the number of other entities a given entities can be related with, for a given relationship.

 zero or 1

 exactly 1

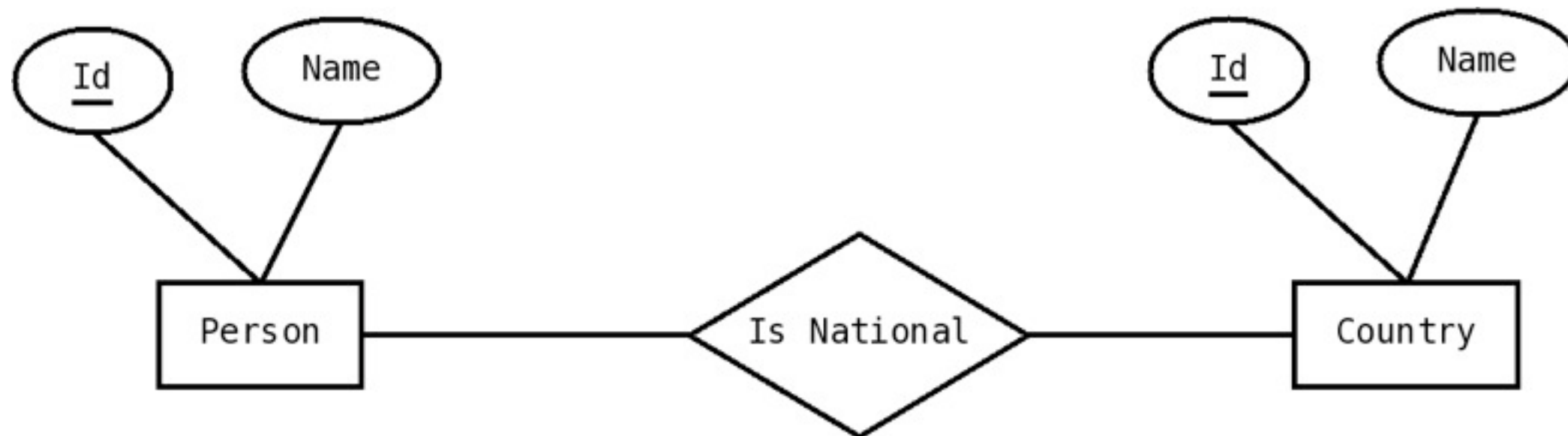
 zero or more

 1 or more



# Quick Check

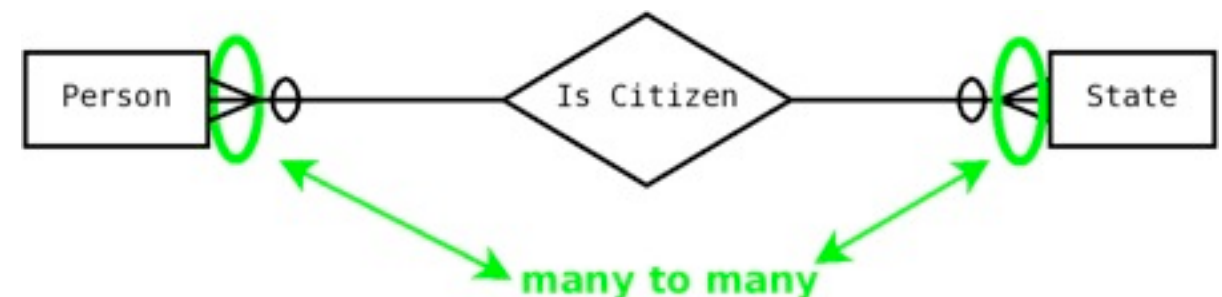
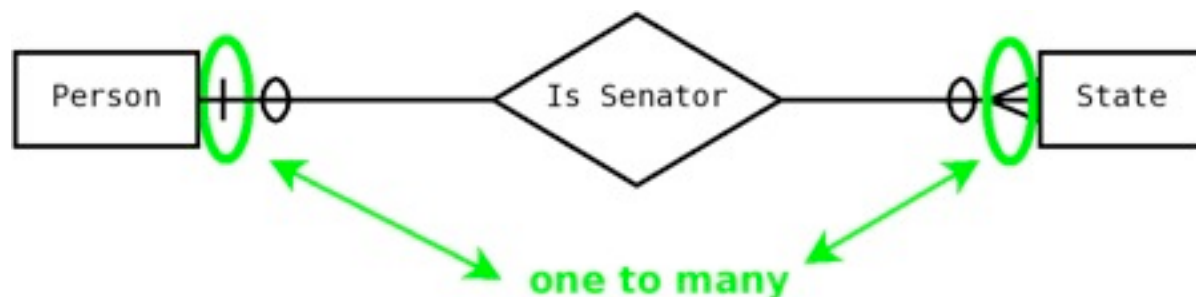
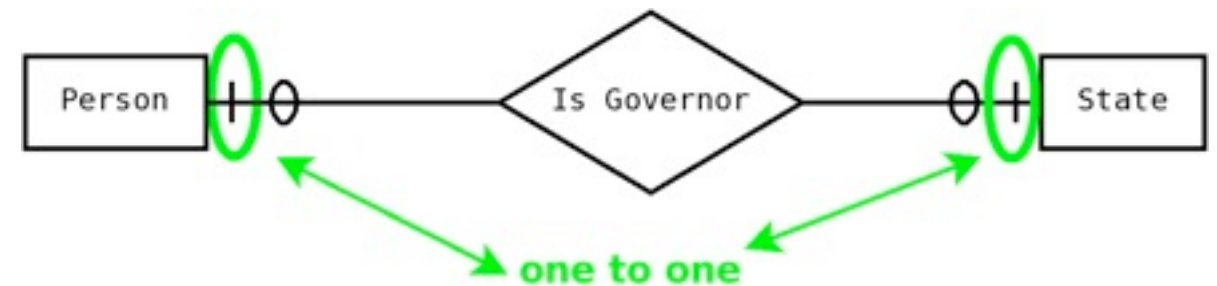
- Add cardinality constraints:
  - A person is a national of zero or more countries;
  - A country has one or more people.





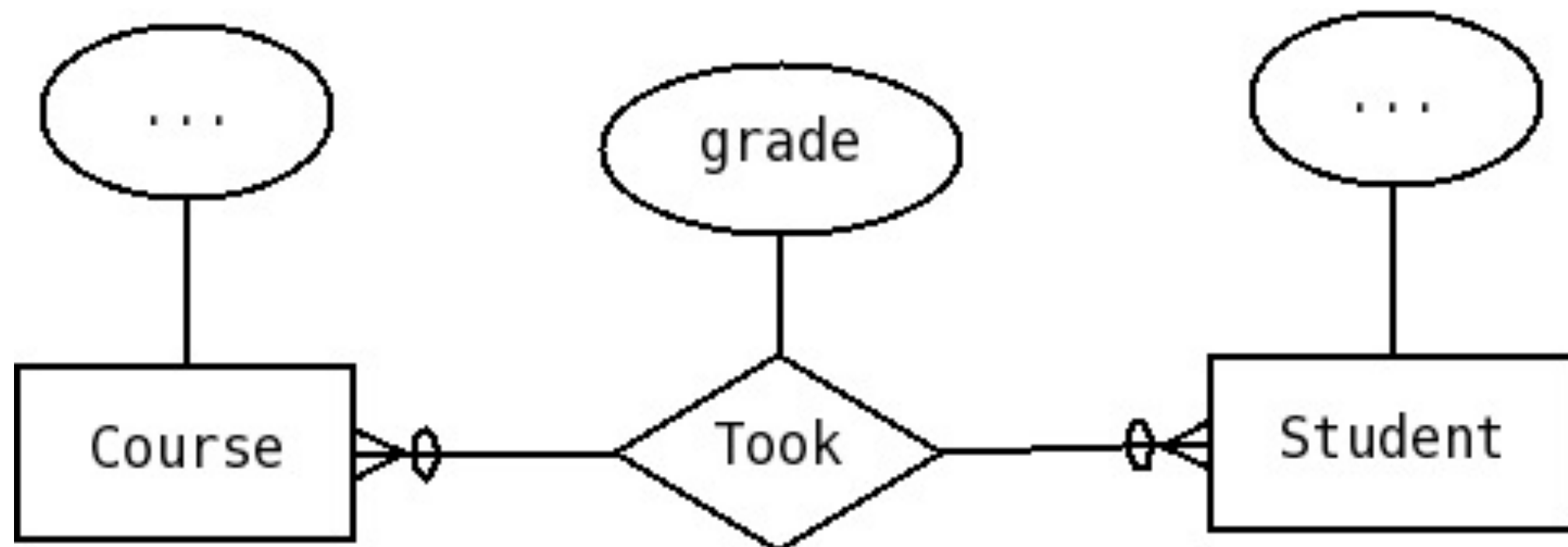
# Maximum cardinalities

- Looking at the MAXIMUM cardinality on BOTH sides, we classify relationships as:
  - one-to-one
  - one-to-many
  - many-to-many



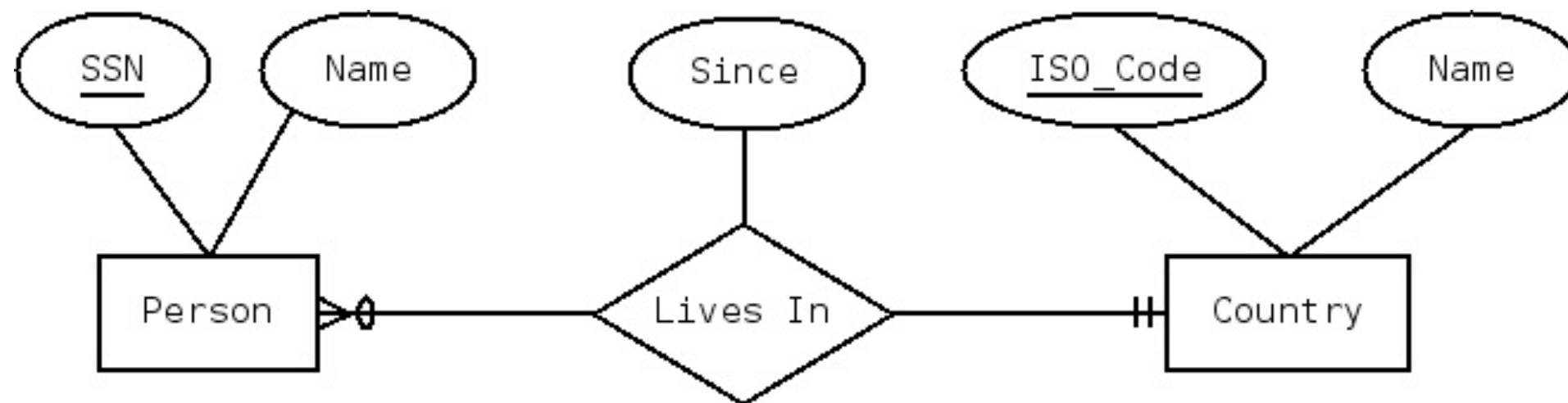
# Relationships and Attributes

- Relationships can have attributes
- These attributes are not associated with the related entities but with their relationship



# Relationships recap

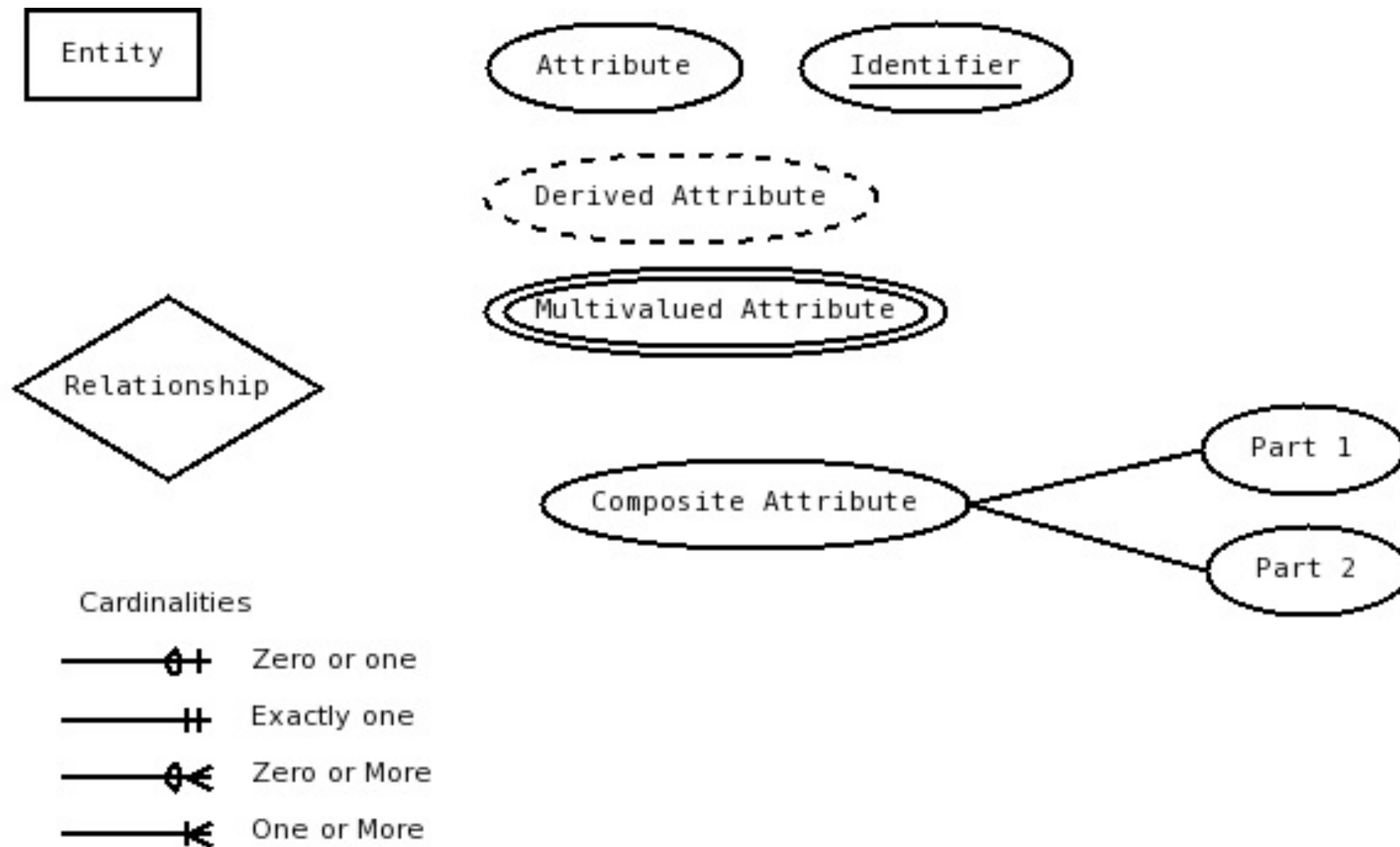
- Association between two or more entities
- Represented by diamond
- Degree
- May have attributes
- Cardinality constraints
- 1 to Many etc (use MAX on both sides)





# What we've covered

SPSU





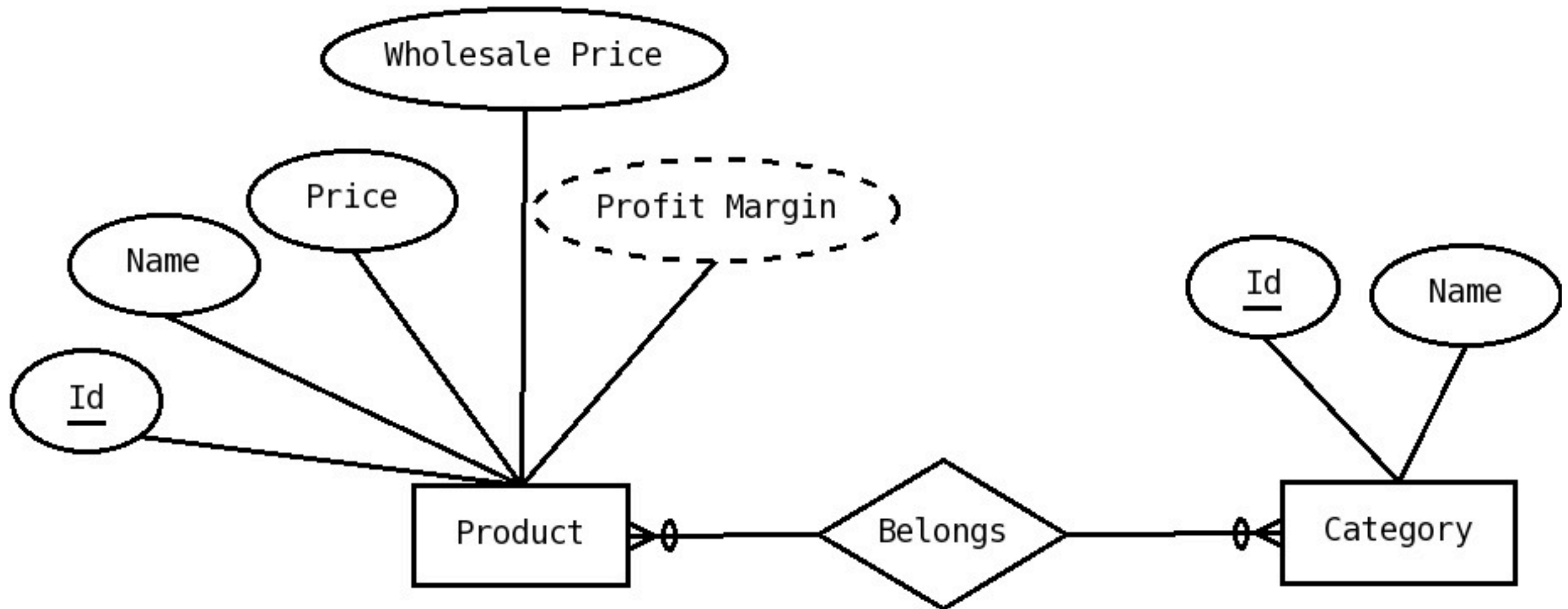
# Practice: Products

SPSU

- We have two kinds of entities: **Products** and **Categories**. For each product we keep its id (the identifier), name, price, wholesale price, and profit margin, which is calculated from the price and wholesale price. For each category we keep its id (the identifier) and its name. Each product belongs to zero or more categories and each category can have zero or more products.

# Solution: Products

SPSU



# Practice: CD 1

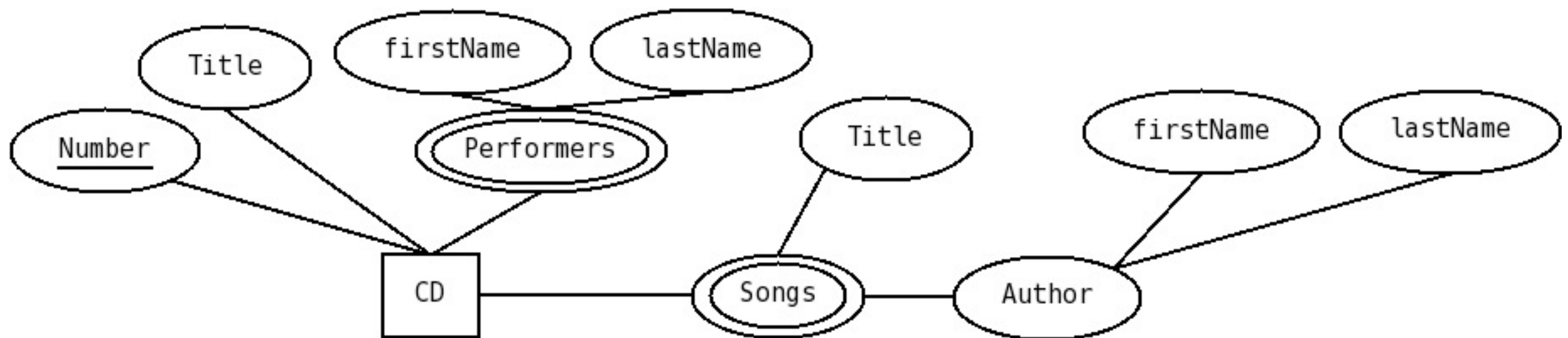
SPSU

- We want to model a CD as an entity. The only entity is the CD and everything else is modeled as attributes.
  - a) A CD has a number, which is its identifier.
  - b) A CD has a title.
  - c) A CD has zero or more performers; for each performer we keep their first and last name.
  - d) A CD has zero or more songs. For each song we keep its title and the name of its author (with name divided into first and last)



# Practice: CD 1

SPSU





# Practice: CD 2

SPSU

- Now let's model a CD, but with Person and Song as entities. So:
  - A CD has a number, which is its identifier, and a title.
  - A person has an id, and a name, divided in first and last.
  - A song has an id, a title and a length.
  - We keep track of which person is a song's author. A person can author many songs and a song has exactly one author.
  - We keep track of which people perform on a CD. Zero or more people perform on a CD and people can perform on zero or more CDs.
  - We keep track of which songs are included on a CD. One or more songs are included on a CD, and a song is included in one or more CDs.

# Practice: CD 2

SPSU

