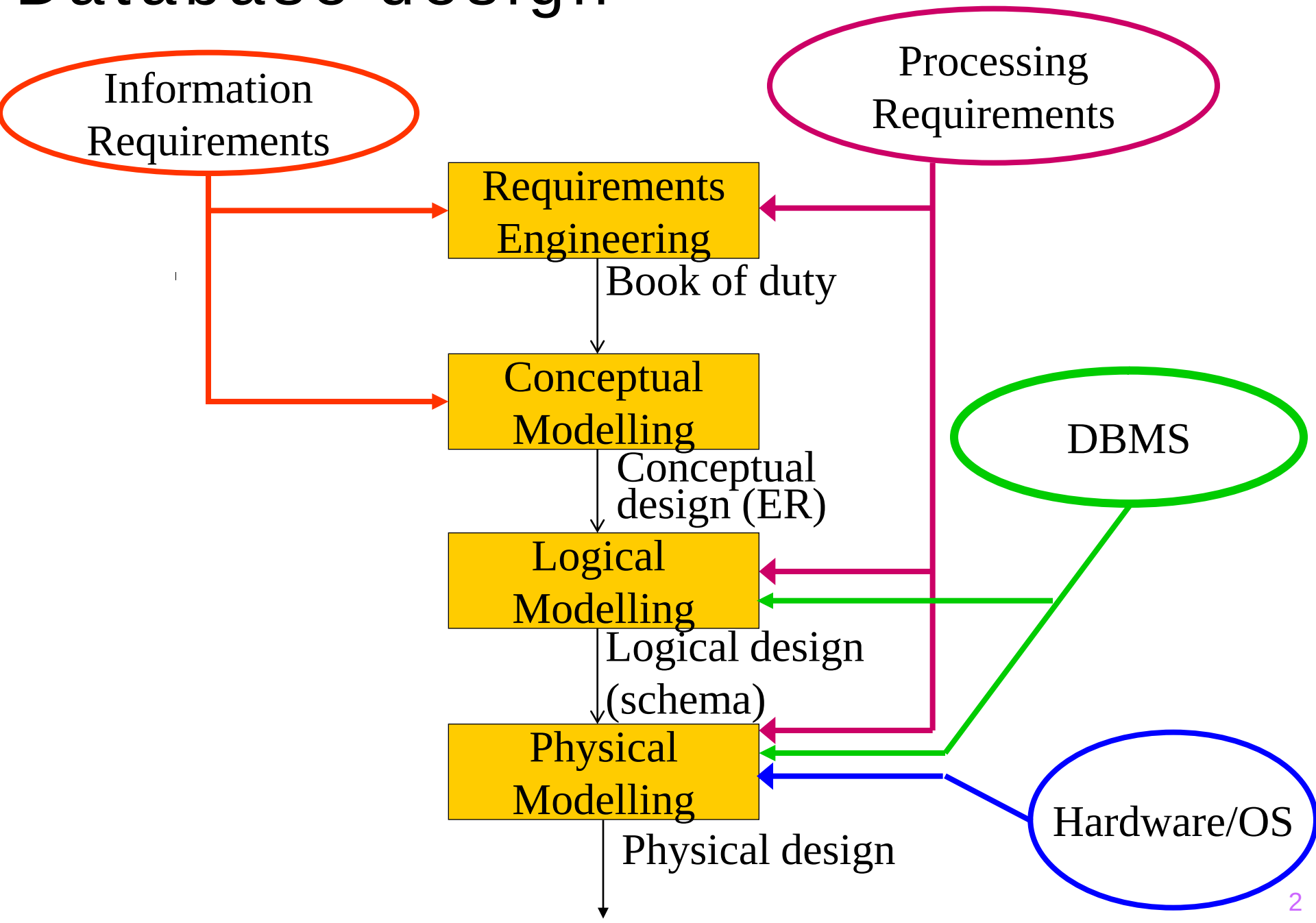
Database Design

Database Abstraction Layers

1. Conceptual Model
2. Logical Model
3. Physical Database Design

Database design



Book of Duty

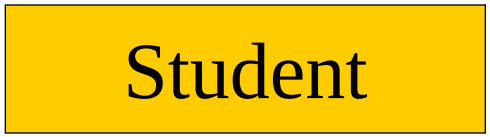
- **Describe information requirements**
  - **Objects used (e.g., student, professor, lecture)**
  - **Domains of attributes of objects**
  - **Identifiers, references / relationships**
- Describe processes
  - E.g., examination, degree, register course
- Describe processing requirements
  - Cardinalities: how many students?
  - Distributions: skew of lecture attendance
  - Workload: how often a process is carried out
  - Priorities and service level agreements

Entity/Relationship (ER) Models

- Entity (Gegenstandstyp)
- Relationship (Beziehungstyp)
- Attribute (Eigenschaft)
- Key (Identifikation)
- Role

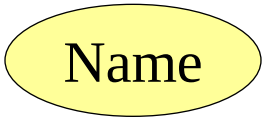
Entity/Relationship (ER) Models

- Entity (Gegenstandstyp)



- Relationship (Beziehungstyp)

- Attribute (Eigenschaft)

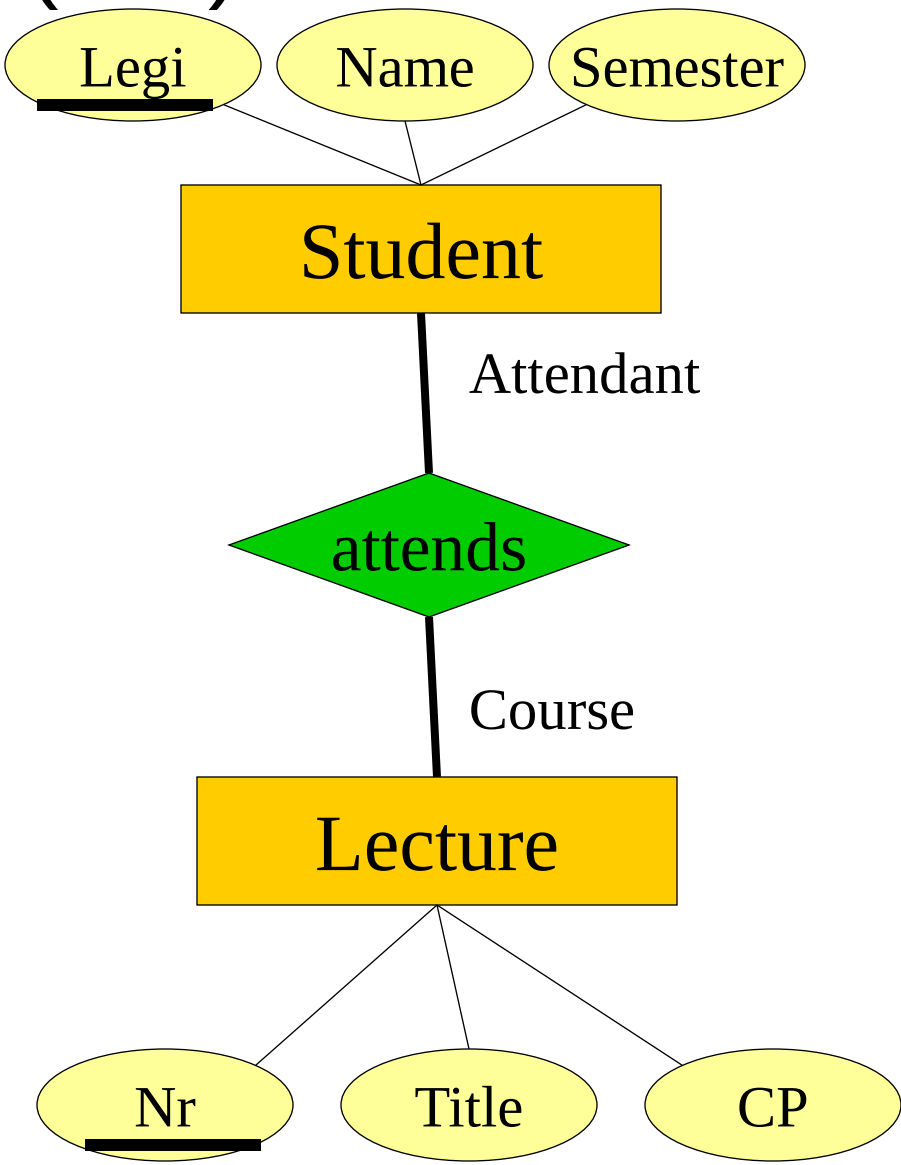


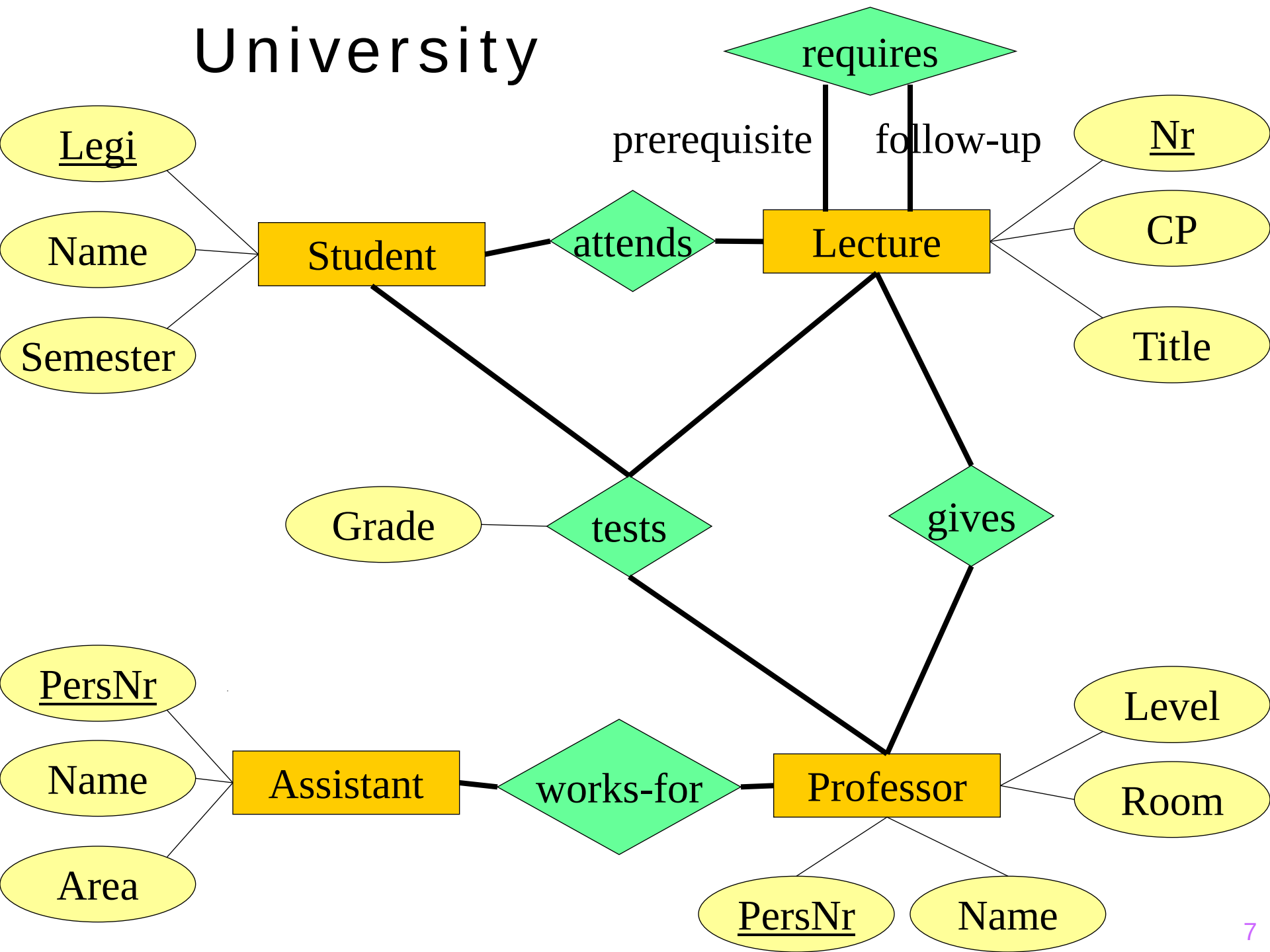
- Key (Identifikation)

- Role

Entity/Relationship (ER) Models

- Entity (Gegenstandstyp)
- Relationship (Beziehungstyp)
- Attribute (Eigenschaft)
- Key (Identifikation)
- Role




University
requires
prerequisite
follow-up
Legi
Name
Semester
Student
attends
Lecture
Nr
CP
Title
Grade
tests
gives
PersNr
Name
Area
Assistant
works-for
Professor
Level
Room
PersNr
Name

Natural Language Version

- Students have a LegiNr, Name and Semester. The Legi identifies a student uniquely.
- Lectures have a Nr, CP and Title. The Nr identifies a lecture uniquely.
- Professors have a PersNr, Name, Level and Room. The PersNr identifies a professor uniquely.
- Assistents have a PersNr, Name and (research) Area. The PersNr identifies an assistent uniquely.
- Students attend lectures.
- Lectures can be prerequisites for other lectures.
- Professors give lectures.
- Assistents work for professors.
- Students are tested by professors about lectures. Students receive grades as part of these tests.
- Is this the only possible interpretation?

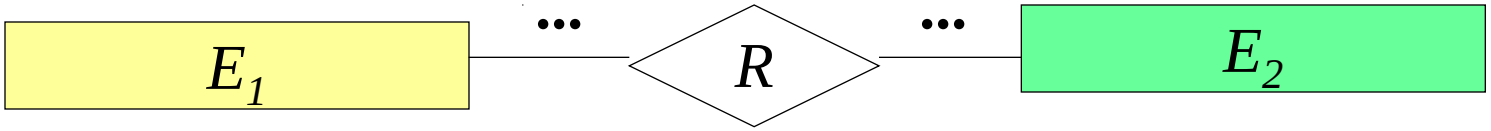
Why ER?

- Advantages
  - ER diagrams are easy to create
  - ER diagrams are easy to edit
  - ER diagrams are easy to read (from the layman)
  - ER diagrams express all information requirements

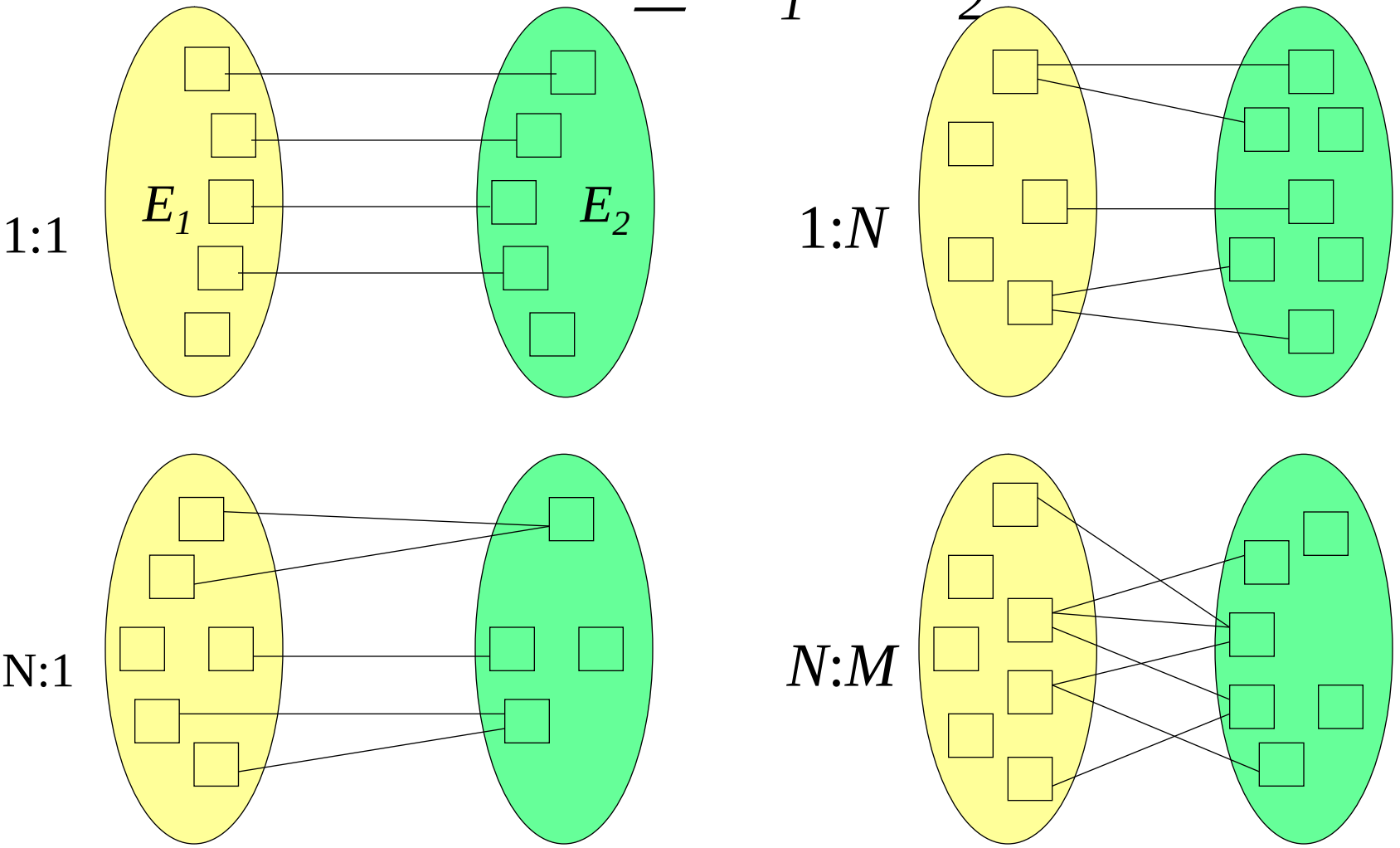
- Other aspects
  - Minimality
  - Tools (e.g., Visio)
  - Graphical representation

- General
  - Try to be consice, complete, comprehensible, and correct
  - Controversy whether ER/UML is useful in practice
  - No controversy that everybody needs to learn ER/UML

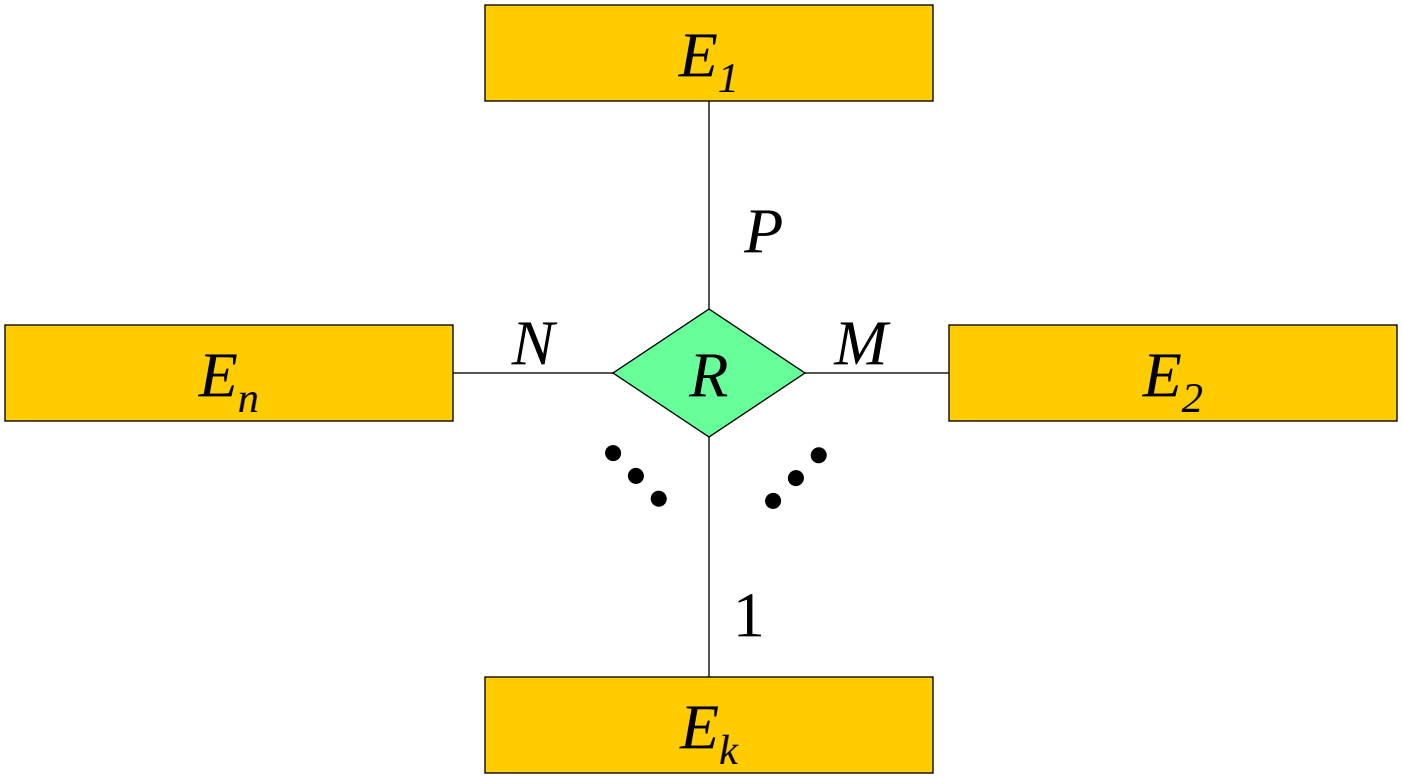
Functionalities



$$R \subseteq E_1 \times E_2$$

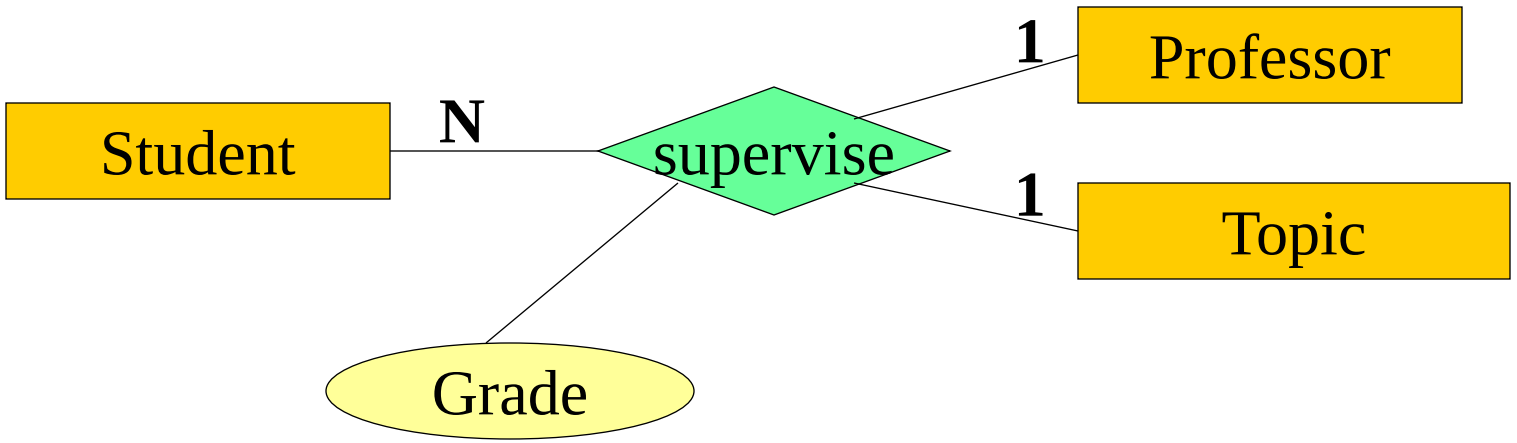


Functionalities of n-ary relationships



$$R : E_1 \text{ x } \ldots \text{ x } E_{k-1} \text{ x } E_{k+1} \text{ x } \ldots \text{ x } E_n \rightarrow E_k$$

Example: *seminar*



supervise : Professor x Student → Topic

supervise : Topic x Student → Professor

Constraints

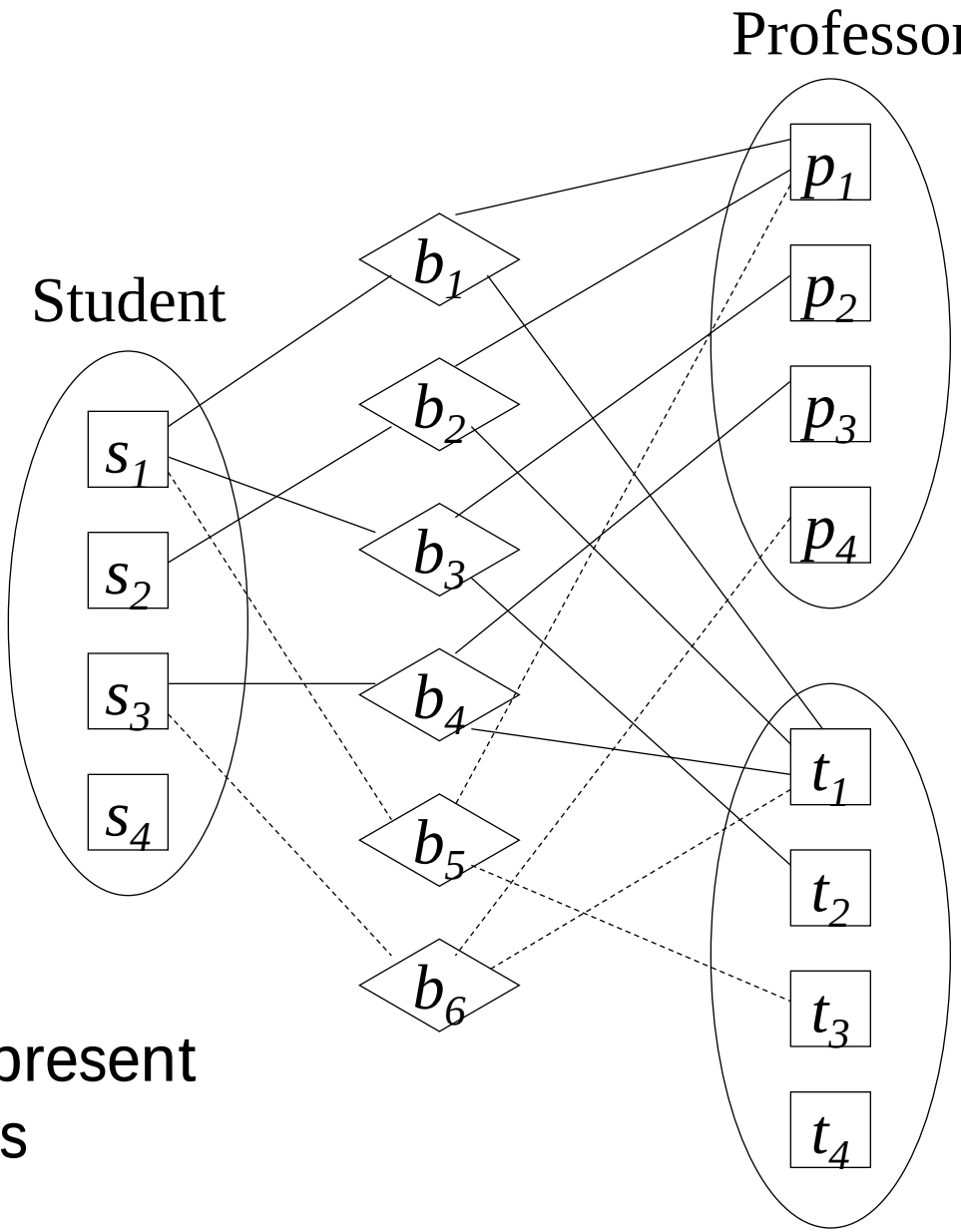
The following is not possible:

1. Students may only do at most one seminar with a prof.
1. Students may only work on a topic at most once.

The following is possible:

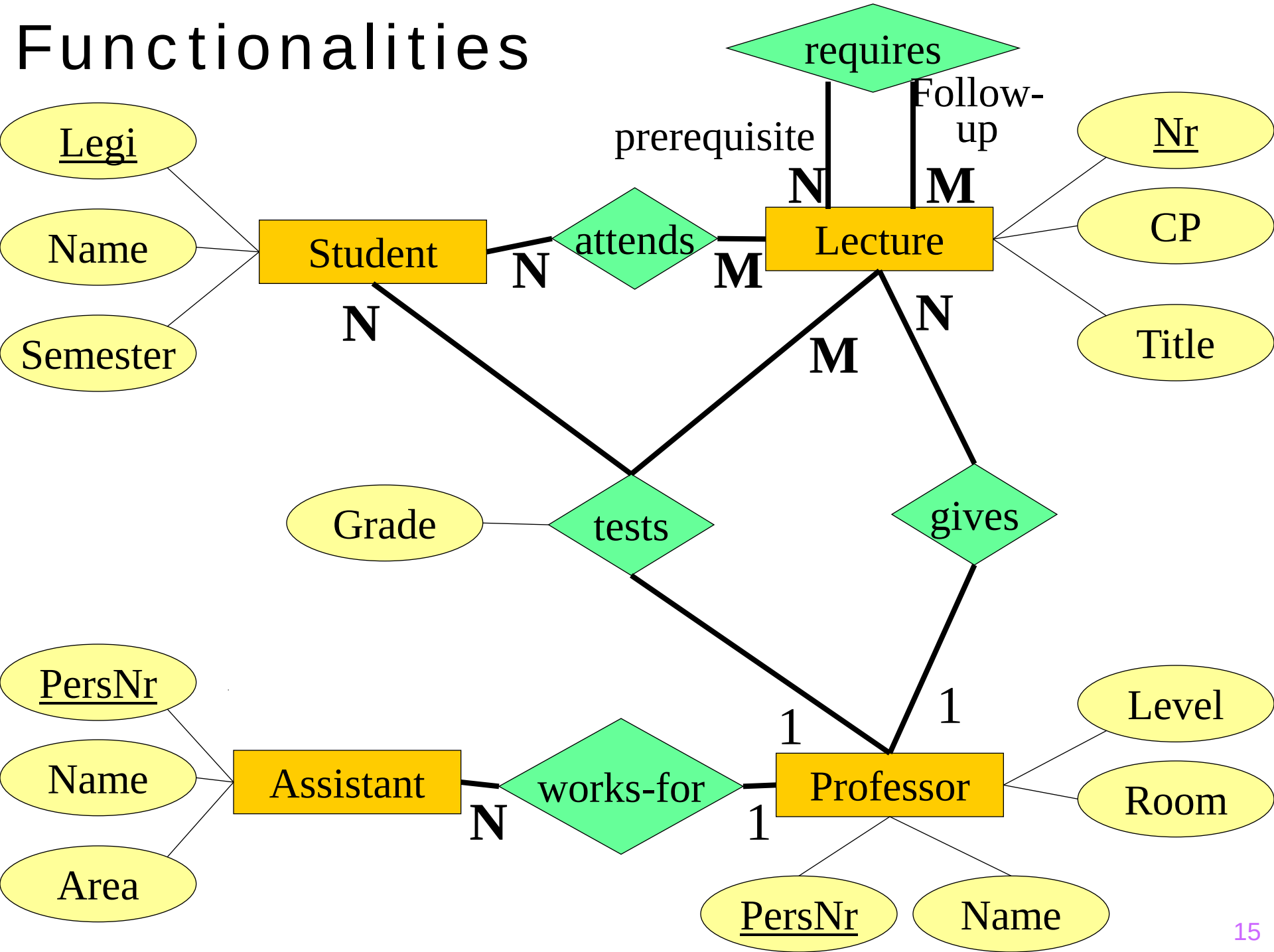
- Profs may recycle topics and assign the same topic to several students.
- The same topic may be supervised by several profs.

Example



Dashed lines represent illegal references

Functionalities



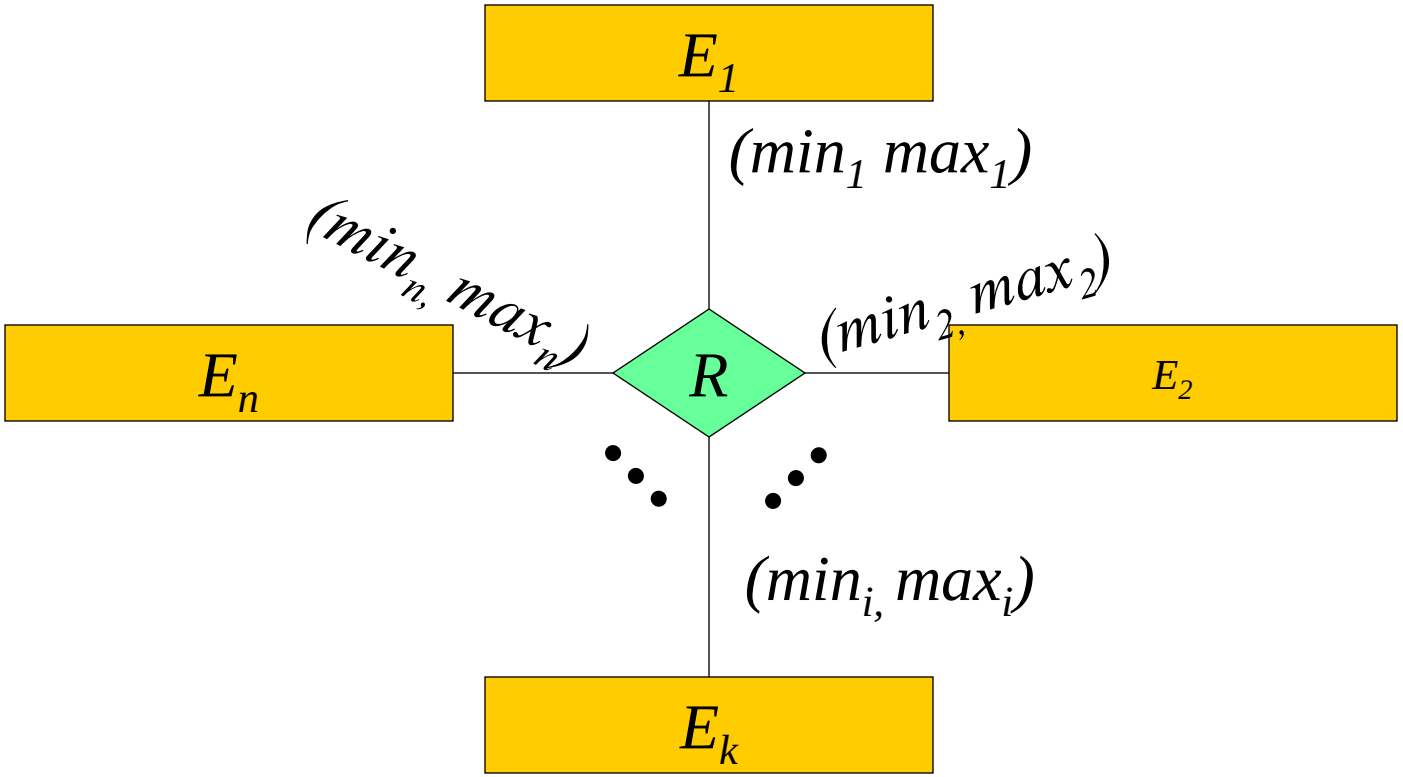
Two Binary vs. One Ternary Relat.

- A thief steals a painting as part of a theft.
  - Model as two binary relationships
  - Model as one ternary relationship
  - What is better?

Rules of thumb

- Attribute vs. Entity
  - Entity if the concept has more than one relationship
  - Attribute if the concept has only one 1:1 relationship
- Partitioning of ER Models
  - Most realistic models are larger than a page
  - Partition by domains (library, research, finances, ...)
  - I do not know of any good automatic graph partitioning tool
- Good vs. Bad models
  - Do not model redundancy or tricks to improve performance
  - Less entities is better (the fewer, the better!)
  - Remember the C4 rule. (concise, correct, complete, compr.)

(min, max)-Notation

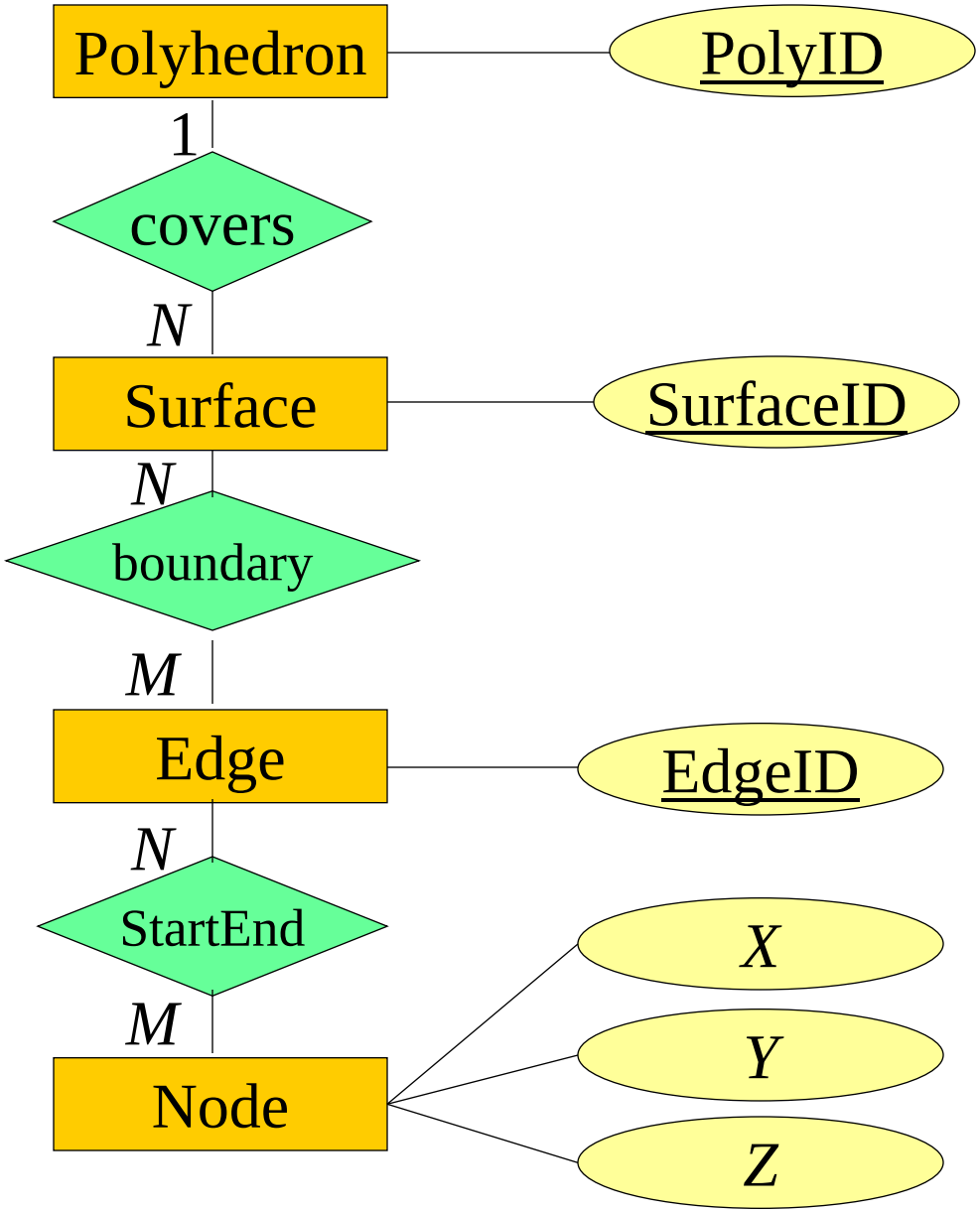


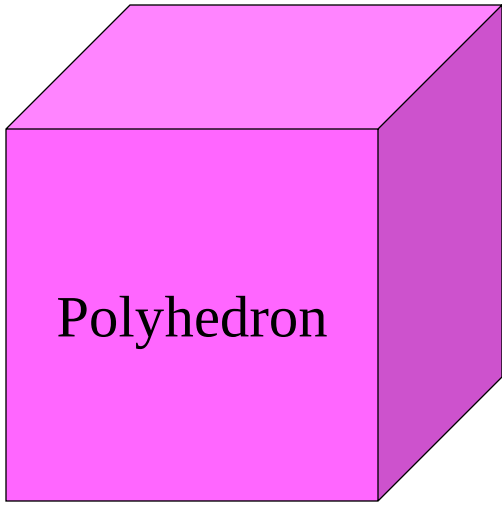
$$R \subseteq E_1 \text{ x } \ldots \text{ x } E_i \text{ x } \ldots \text{ x } E_n$$

For all $e_i \in E_i$:

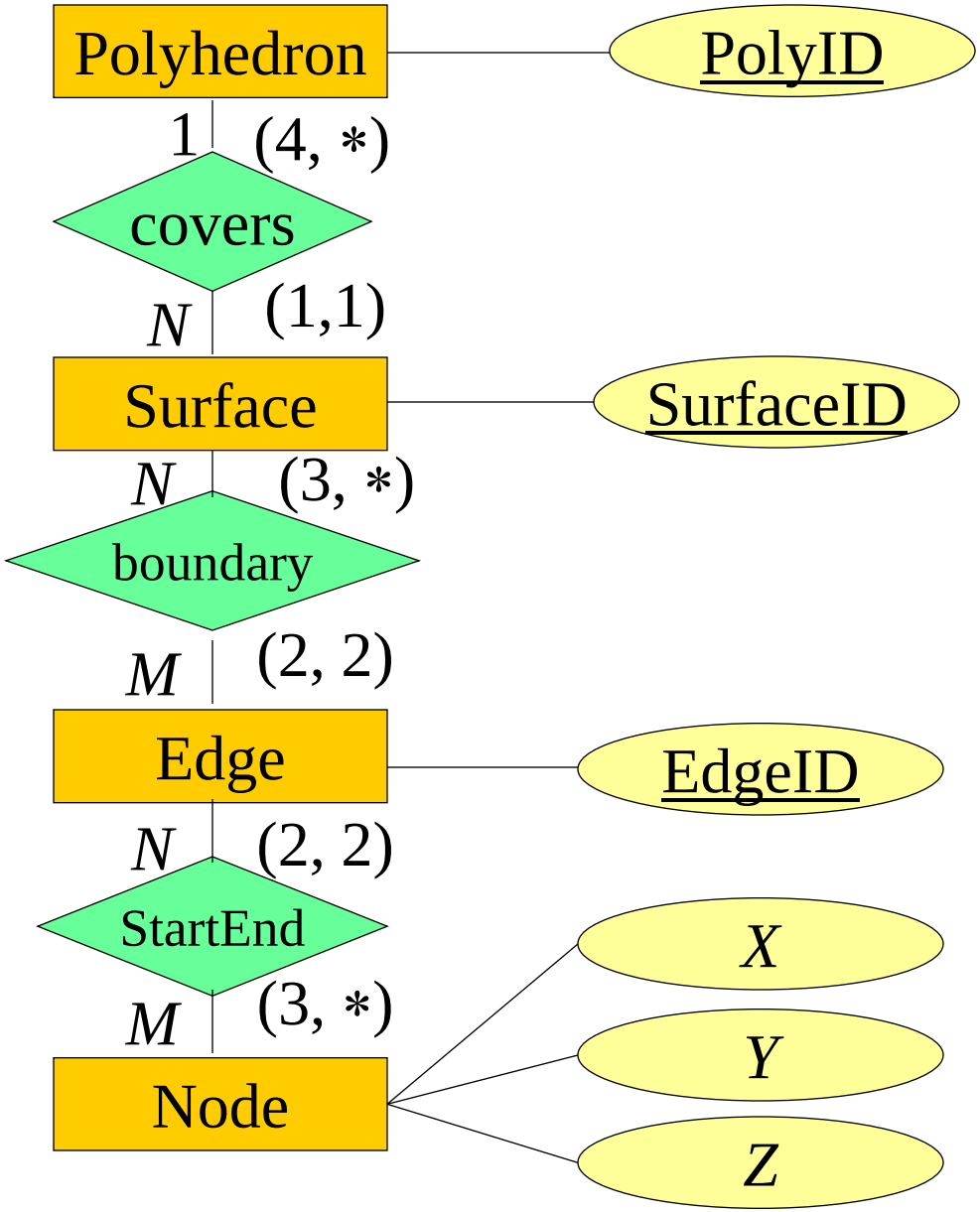
- At least min_i records $(\ldots, e_i, \ldots)$ exist in R AND
- At most max_i records $(\ldots, e_i, \ldots)$ exist in R

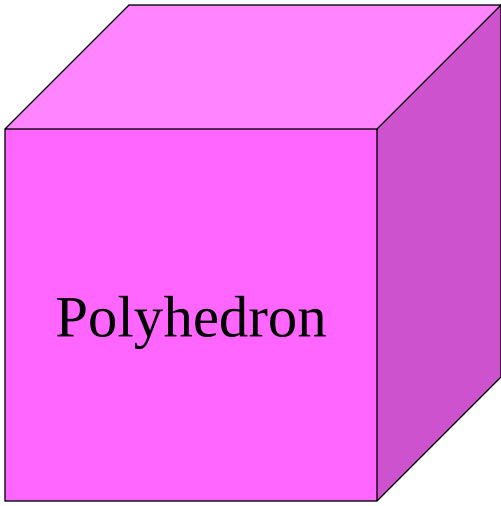
Geometric Modelling

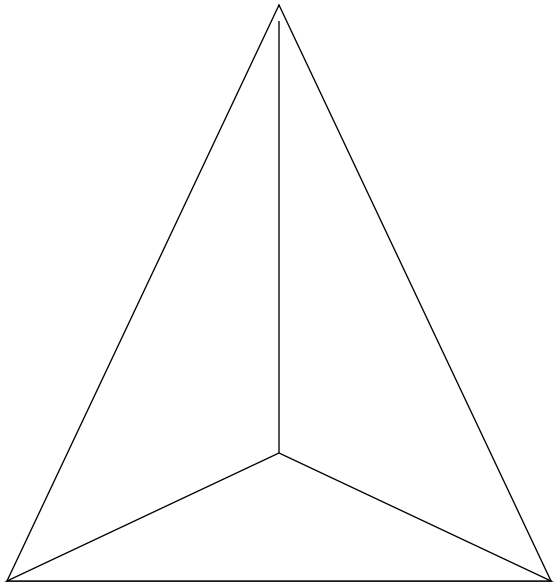




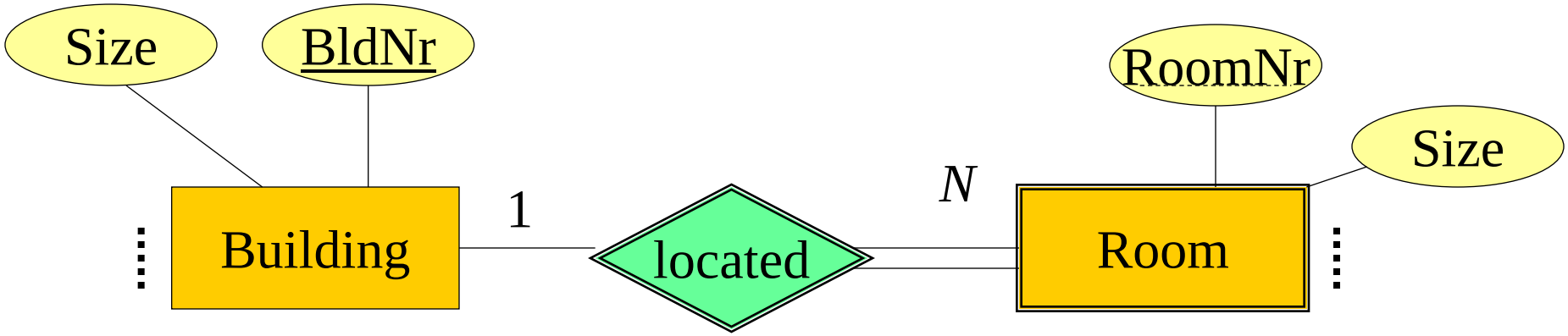
Geometric Modelling





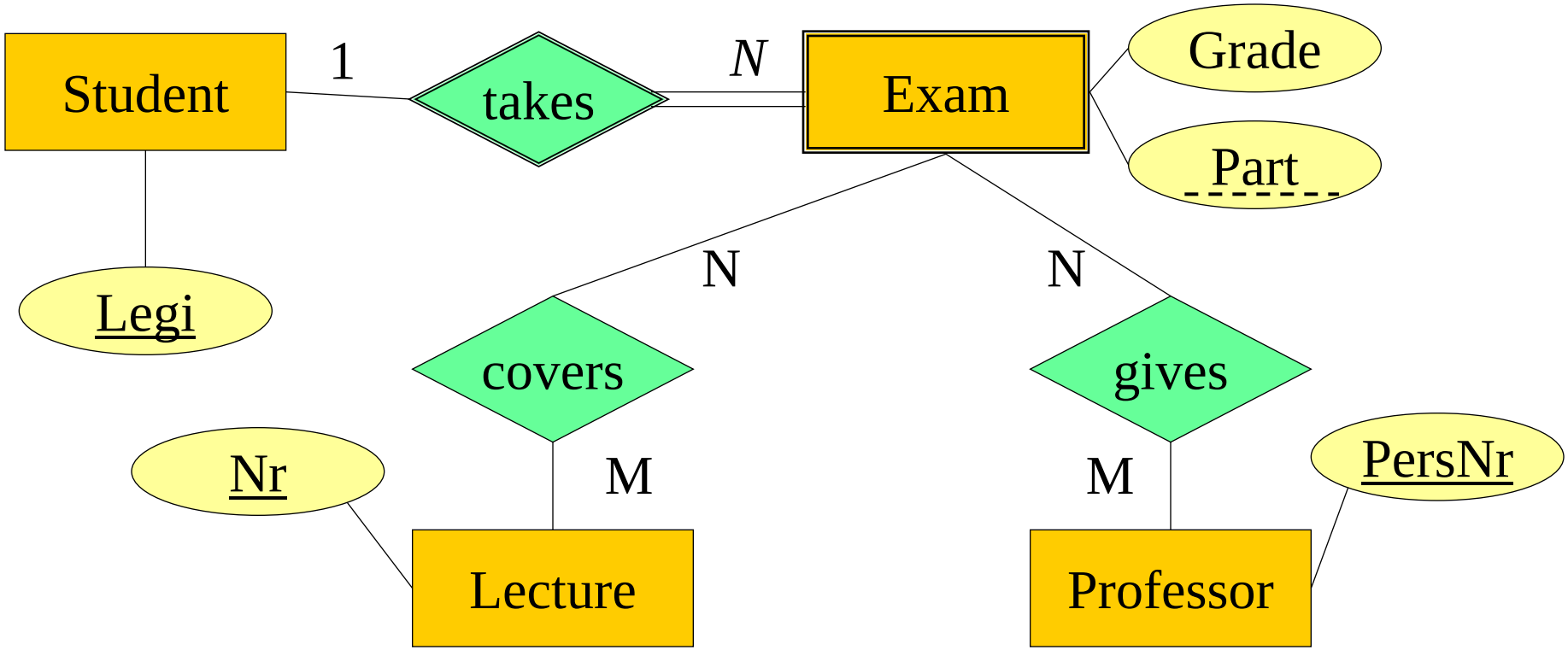


Weak Entities



- The existince of room depends on the existence of the associated building.
- Why must such relationships be N:1 (or 1:1)?
- RoomNr is only unique within a building.
- Key of a room: BldNr **and** RoomNr

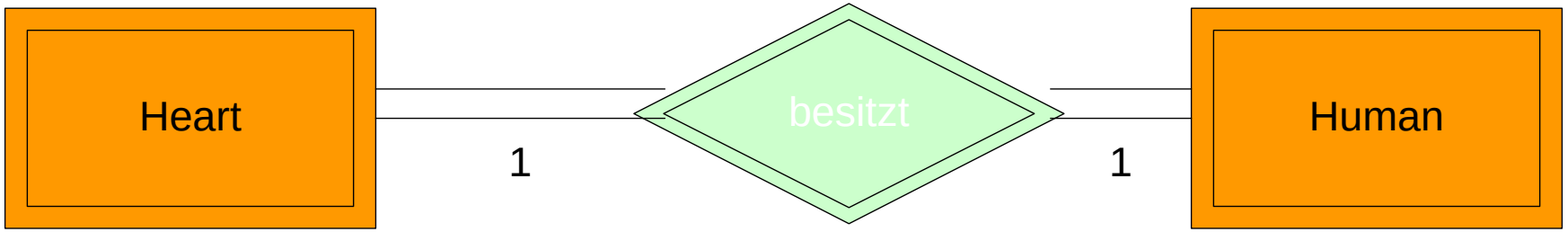
Exams depend on the student



- Can the existence of an entity depend on several other entities? (E.g., exam on student and prof?)

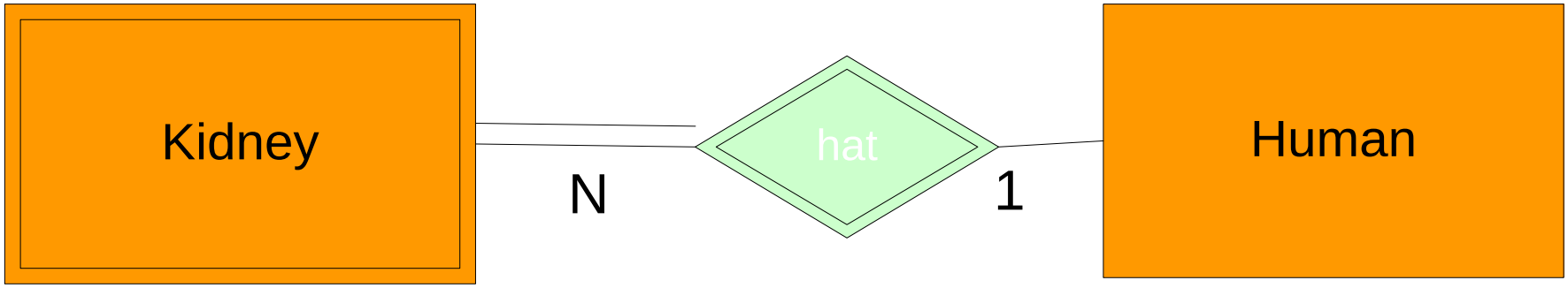
Corner Case 1

- A human cannot exist without a heart.
- A heart cannot exist without a human.
- Anne lives on Bob ‘s heart. Bob lives on Anne ‘s heart. Possible?

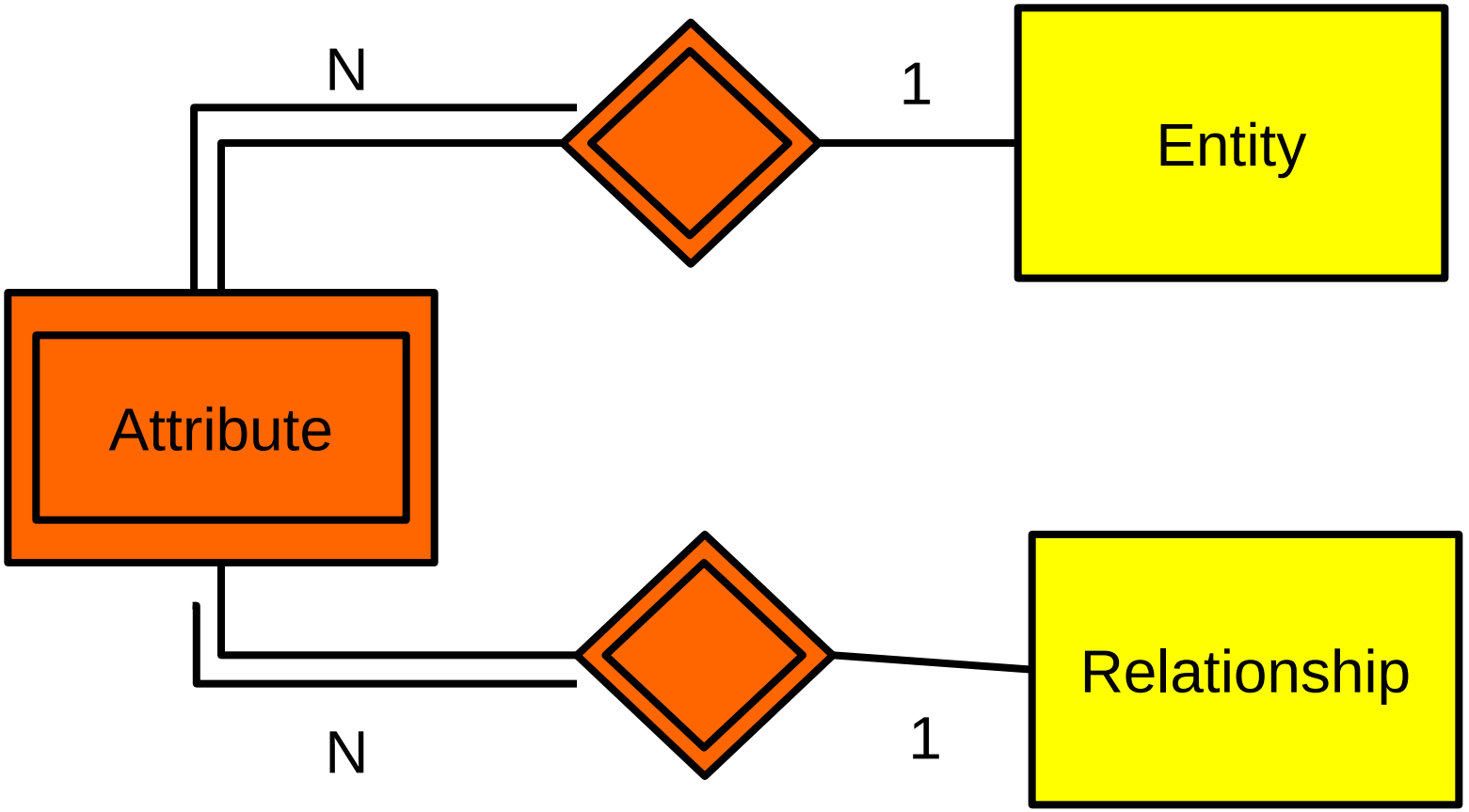


Corner Case 2

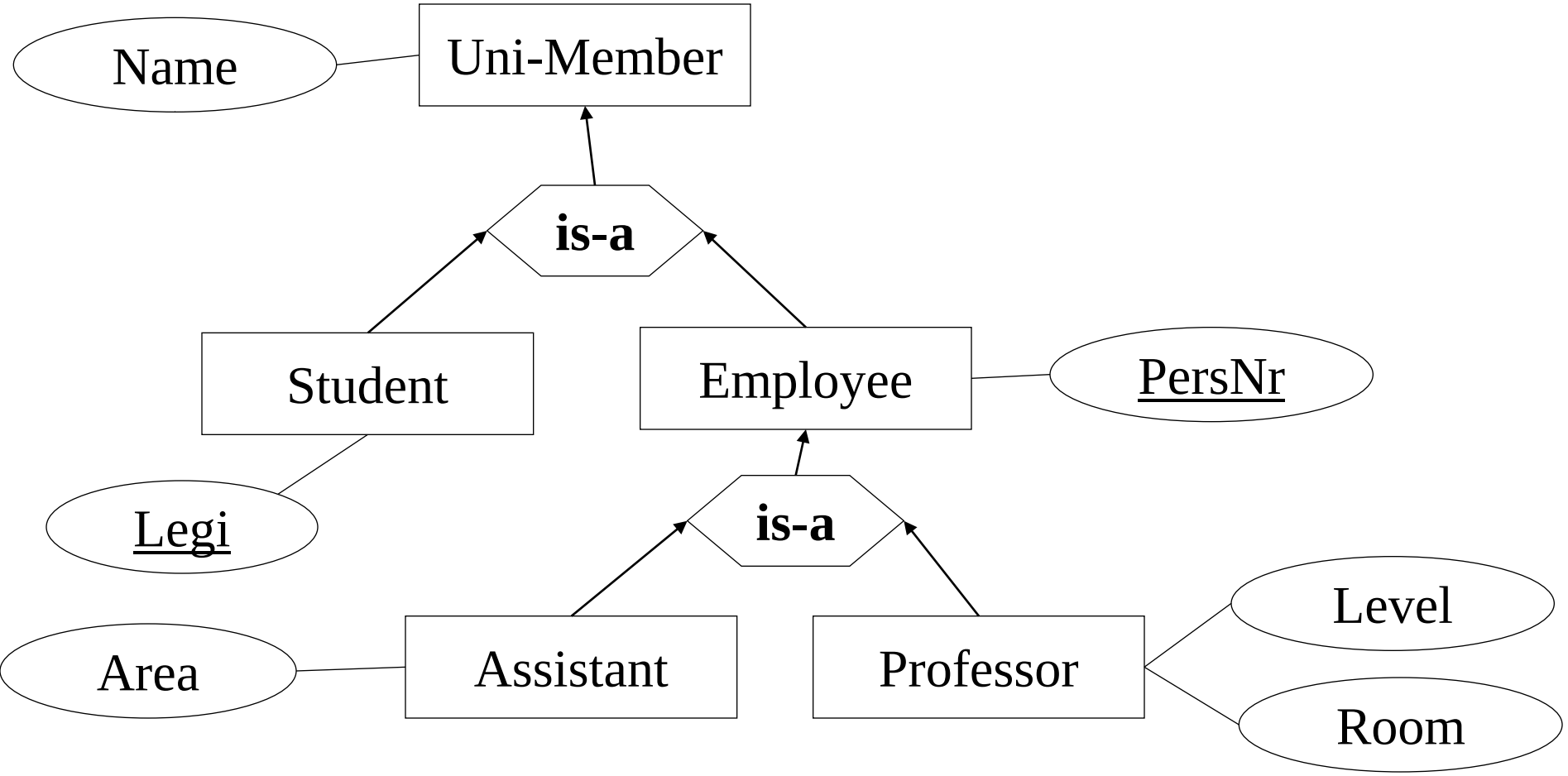
- A human can only survive with ***at least one*** kidney.
- A relationship can only survive with ***all*** its entities.
- **Not expressible with ER!** (Why not?)



Why is this a bad model?

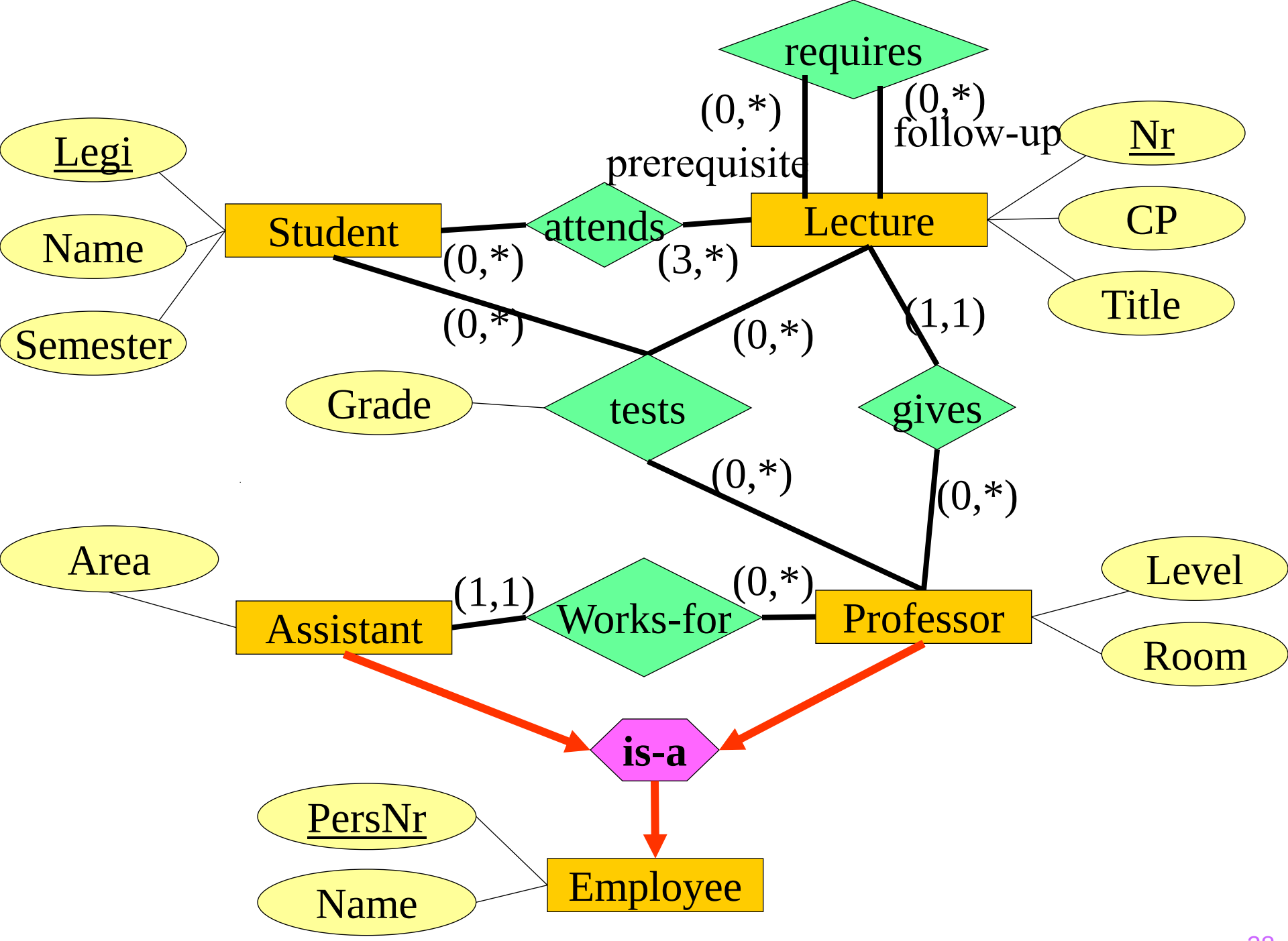


Generalization

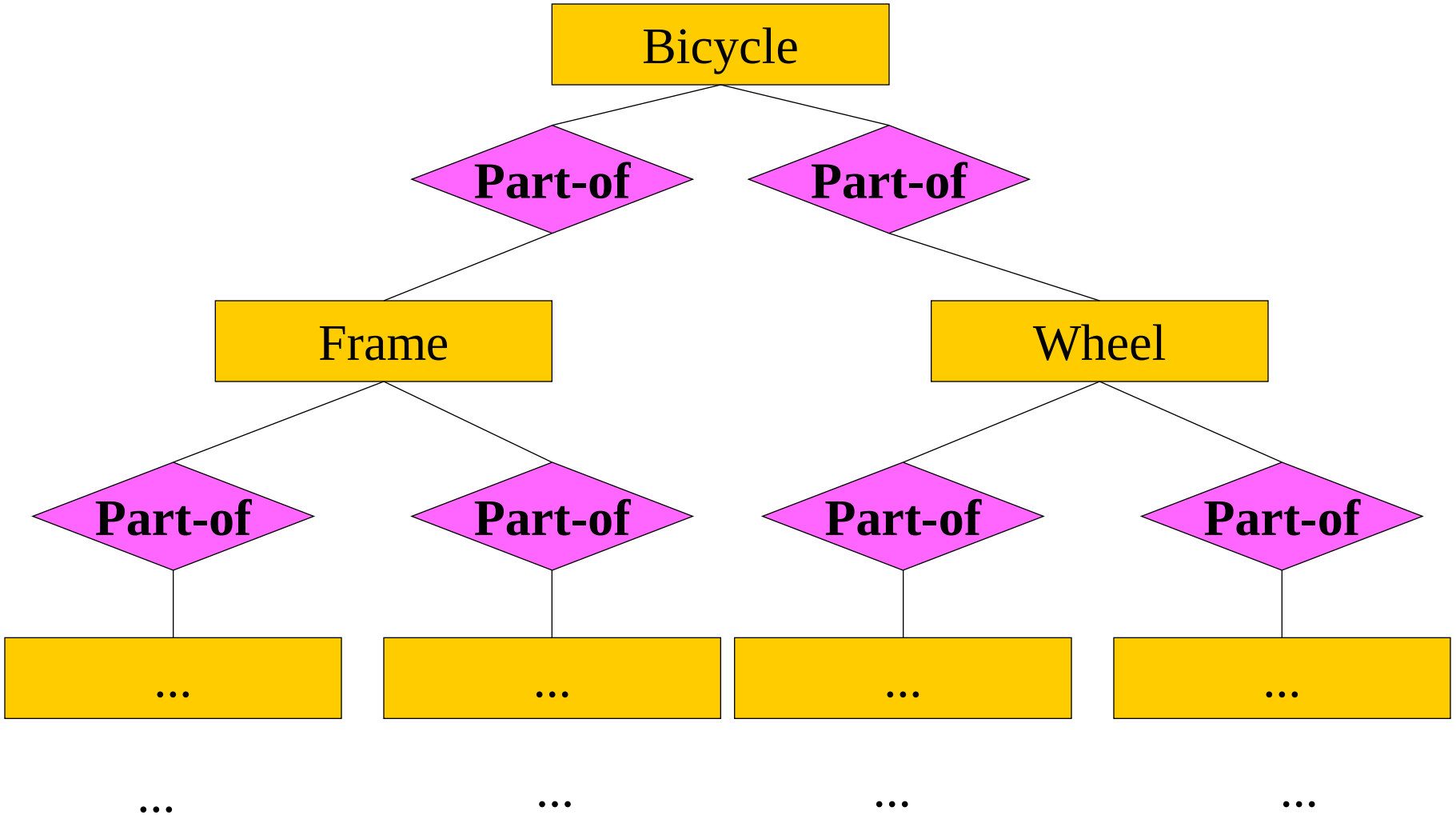


Comments on Generalization

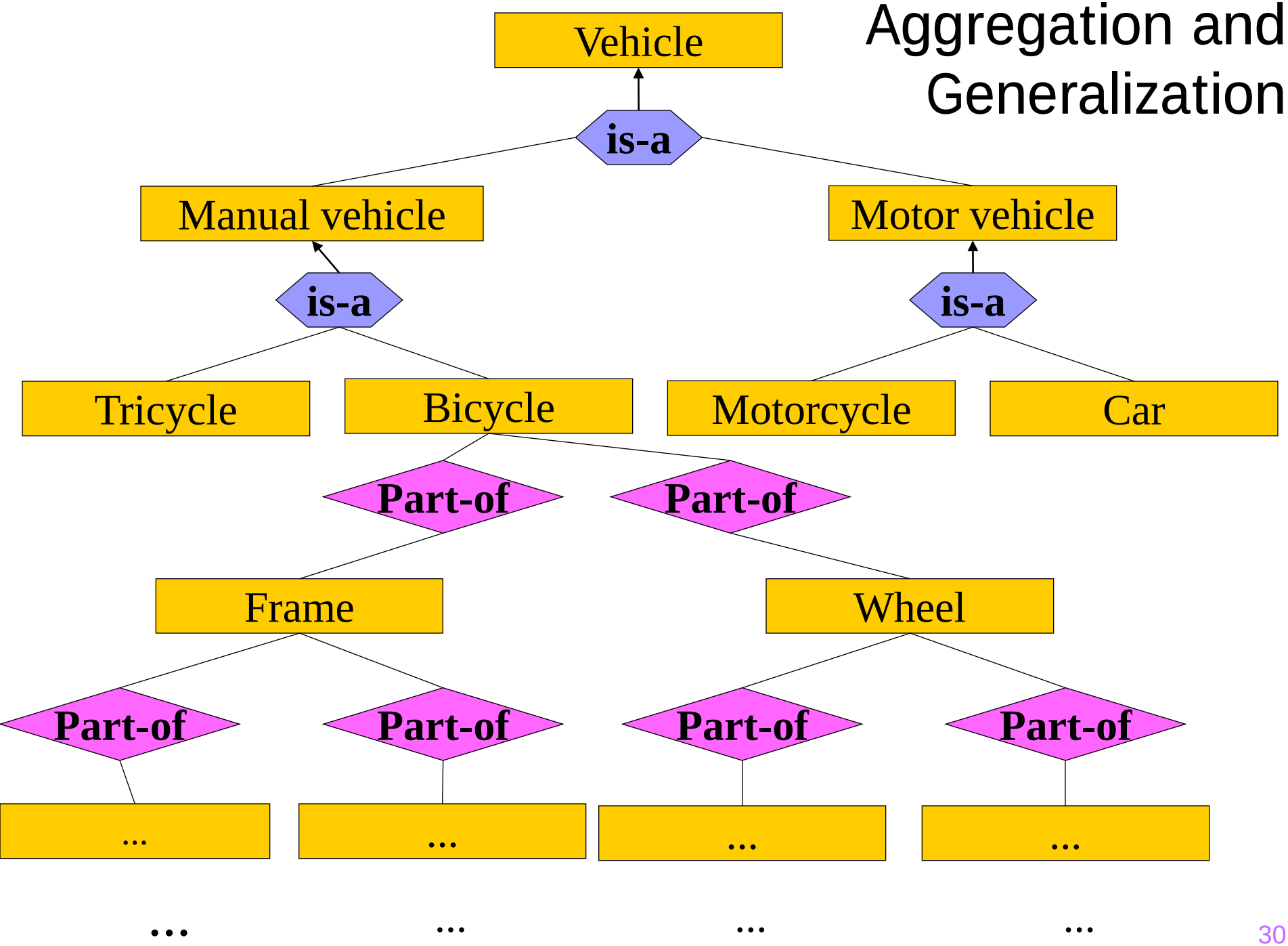
- There may be Employees who are neither Assistant nor Prof.
- There may be an Employee who is also an Assistant.
- There may be an Employee who is also a Prof.
- There may be an Employee who is also an Assistant and a Prof.
- **ER has no way to restrict to any of those four cases.**
  - If at all, done in the „quantative part“ of the book of duty
  - Does it matter?


requires
(0,*)
(0,*)
follow-up
prerequisite
Legi
Name
Semester
Student
attends
Lecture
Nr
CP
Title
(0,*)
(3,*)
(0,*)
(0,*)
(1,1)
Grade
tests
gives
(0,*)
(0,*)
Area
Assistant
(1,1)
Works-for
(0,*)
Professor
Level
Room
is-a
PersNr
Name
Employee

Aggregation



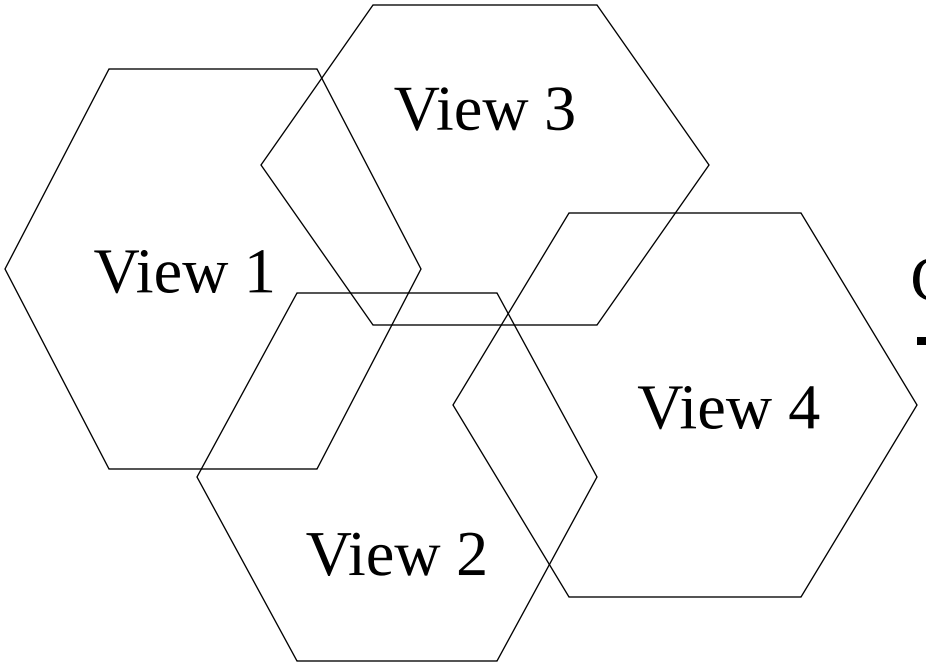
Aggregation and Generalization



Limitations of ER

- ER has no formal semantics
  - unclear whether this is a bug or a feature
  - (natural language has no formal semantics either)
- No way to express relationships between sets of entities
  - e.g., existence of person depends on a set of organs
  - sets of sets are notoriously hard to model
  - (more on that when we talk about 4 NF)
- No way to express negative rules
  - e.g., same entity cannot be an Assistant and Professor
  - again, negation notoriously hard (e.g., 2nd-order logic)
- ER has been around for 30+ years
  - maybe, ER hit sweet spot of expressivity vs. simplicity
  - (UML class diagrams inherit same weaknesses)

Why is ER modelling so difficult?

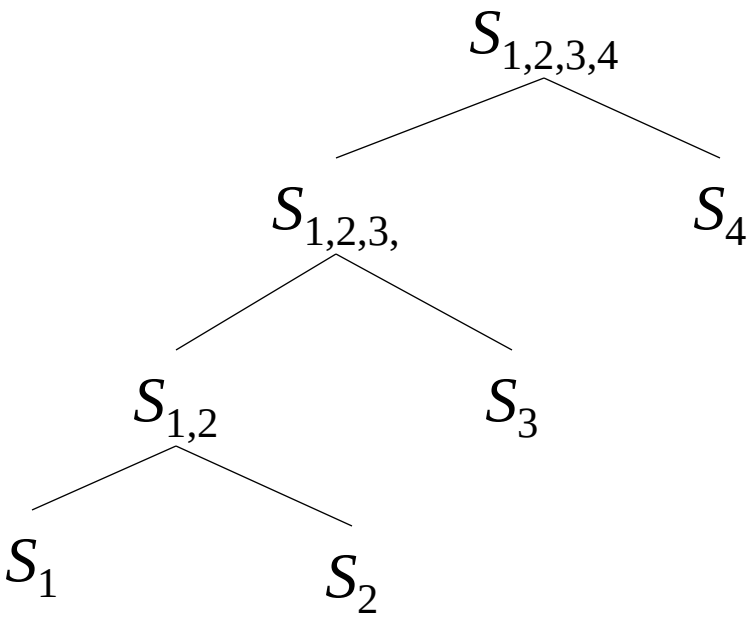


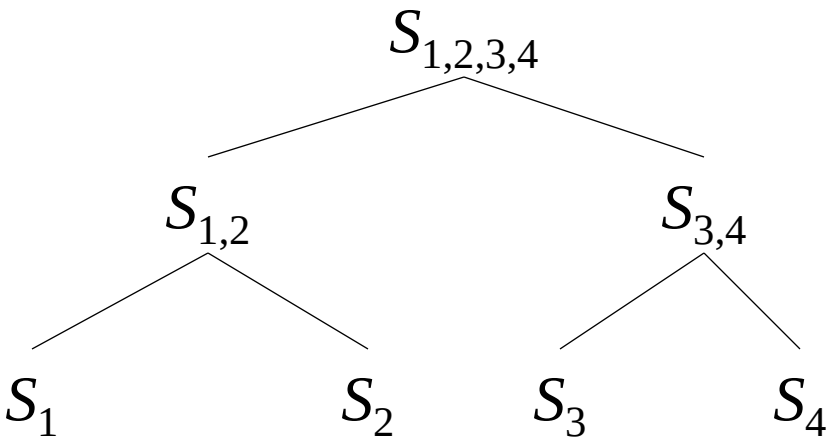
Consoli-
date

Global Schema

- **No redundancy**
- **No conflicts**
- **Avoid synonyms**
- **Avoid homonyms**

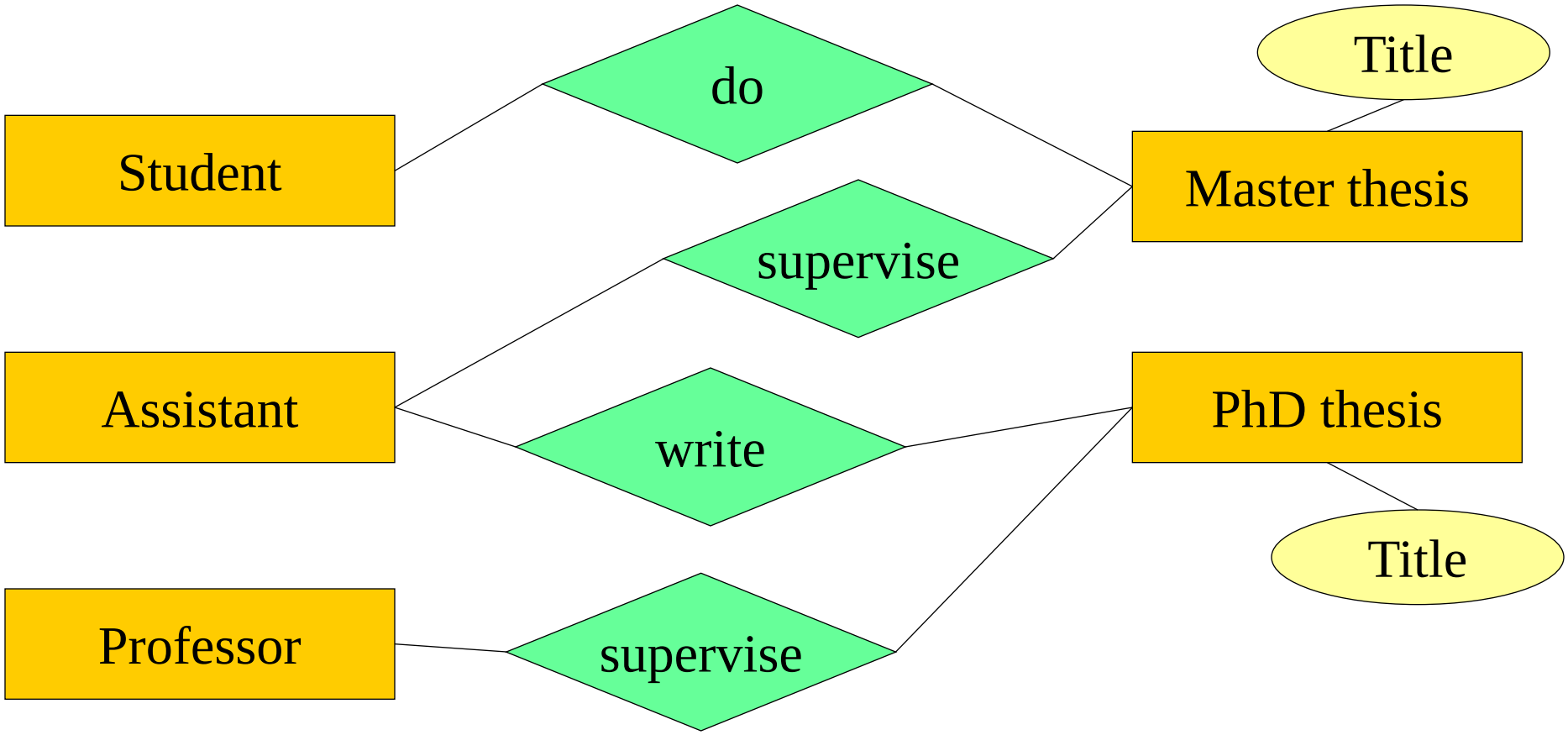
Consolidation Hierarchies



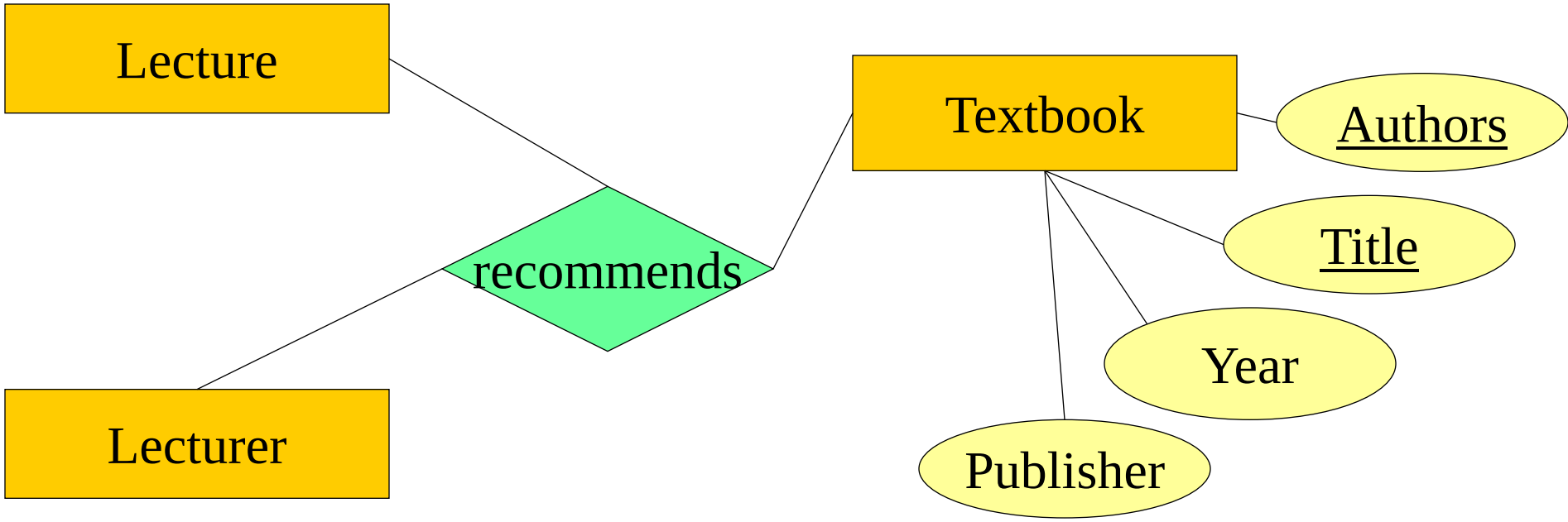


Problem: How to achieve multi-lateral consensus?

Example: Professor View

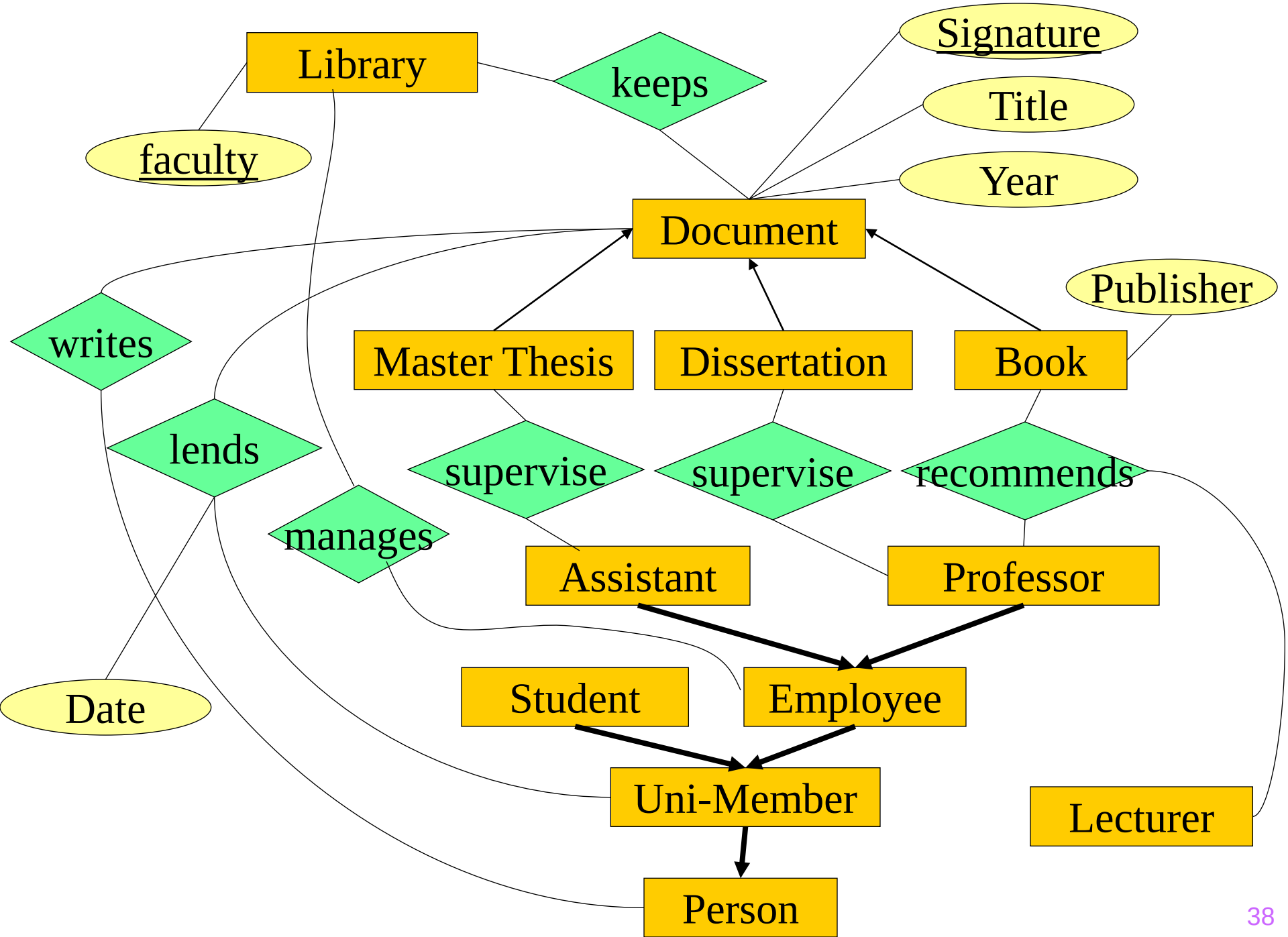


Example: Lecture View



Observations

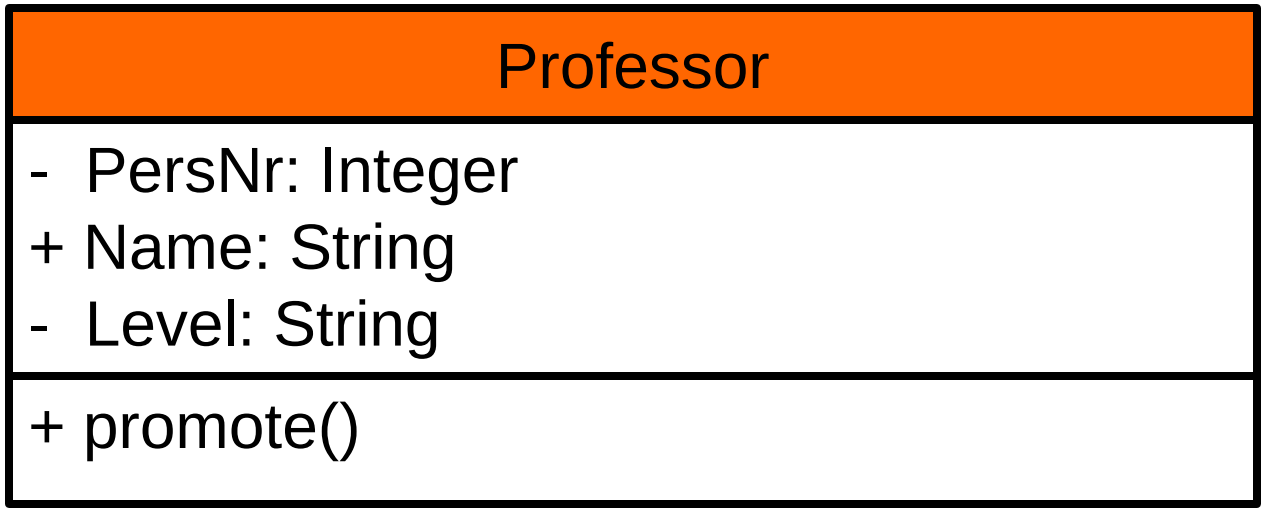
- *Lecturer* and *Professor* are synonyms.
- *Uni-Member* is a generalization of *Student, Professor* and *Assistant*.
- However, libraries are managed by *Employees.* (View 2 is imprecise in this respect.)
- *Dissertations, Master theses* and *Books* are different species of *Document*. All are held in libraries.
- *Do* and *Write* are synonyms in View 1.
- Things get complicated very quickly – requires „engineers“
  - Not unique
  - Need to invent new concepts
  - Need to compromise (e.g., authorship of documents)


Library
keeps
Signature
Title
Year
faculty
Document
Publisher
writes
Master Thesis
Dissertation
Book
lends
supervise
supervise
recommends
manages
Assistant
Professor
Date
Student
Employee
Uni-Member
Lecturer
Person

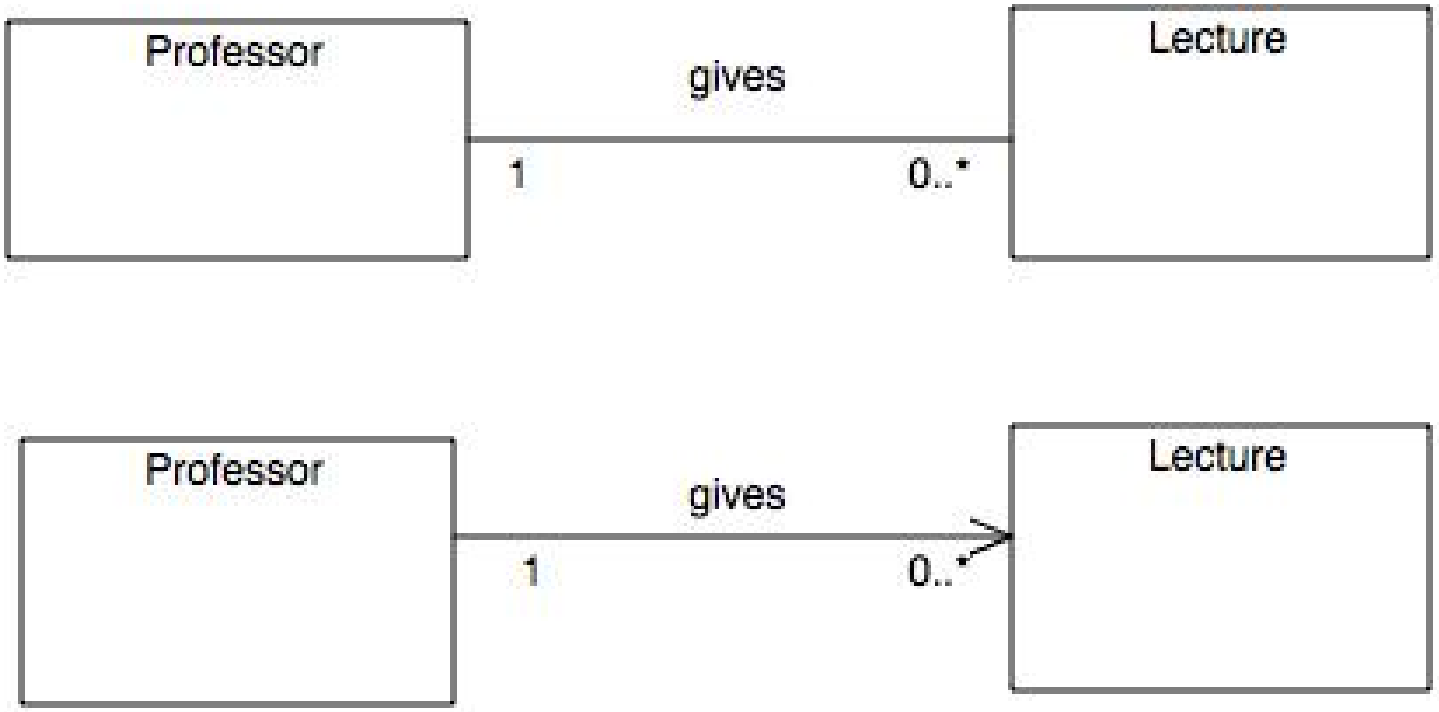
Data Modelling with UML

- Unified Modelling Language UML
- De-facto standard for object-orientierted design
- Data modelling is done with „class diagramms“
  - Class in UML ~ Entity in ER
  - Attribute in UML ~ Attribute in ER
  - Association in UML ~ Relationship in ER
  - Compositor in UML ~ Weak Entity in ER
  - Generalization in UML ~ Generalization in ER
- Key differences between UML class diagrams and ER
  - Methods are associated to classes in UML
  - Keys are not modelled in UML
  - UML explicitly models aggregation (part-of)
  - UML supports the modelling of instances (object diagrams)
- UML has much more to offer (use cases, sequence diagr., …)

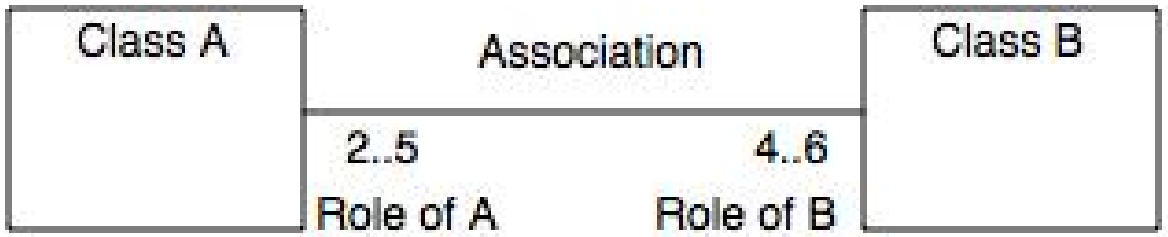
Class: Professor



Associations (directed, undirected)



Functionalities & Multiplicities



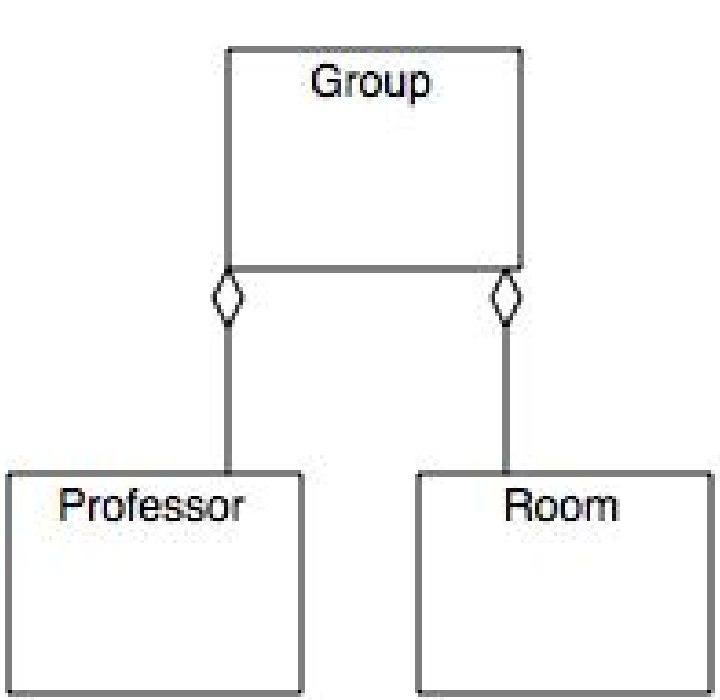
Multiplicities

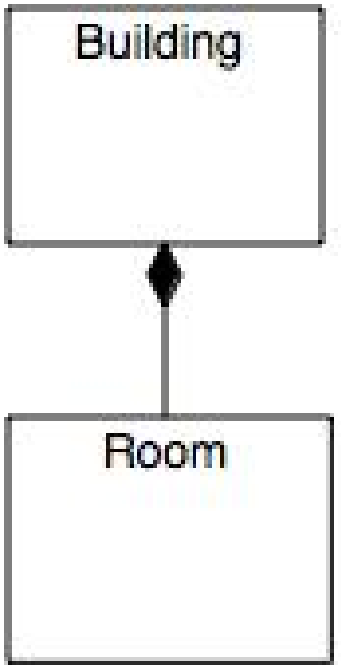
- Every instance of A is associated to 4 to 6 instances of B.
- Every instance of B is associated to 2 to 5 instances of A.
- Be careful: Flipped around as compared to ER.
- Be careful: Cannot be used for n-ary relationships.

Functionalities

- Represented as UML multiplicities: 1, *, 1..*, 0..*, or 0..1
- Otherwise, the same as in ER.

Aggregation





Generalization