Chapter 2: ER-Diagrams

Content:

- Learn how to draw ER diagrams

Database Design

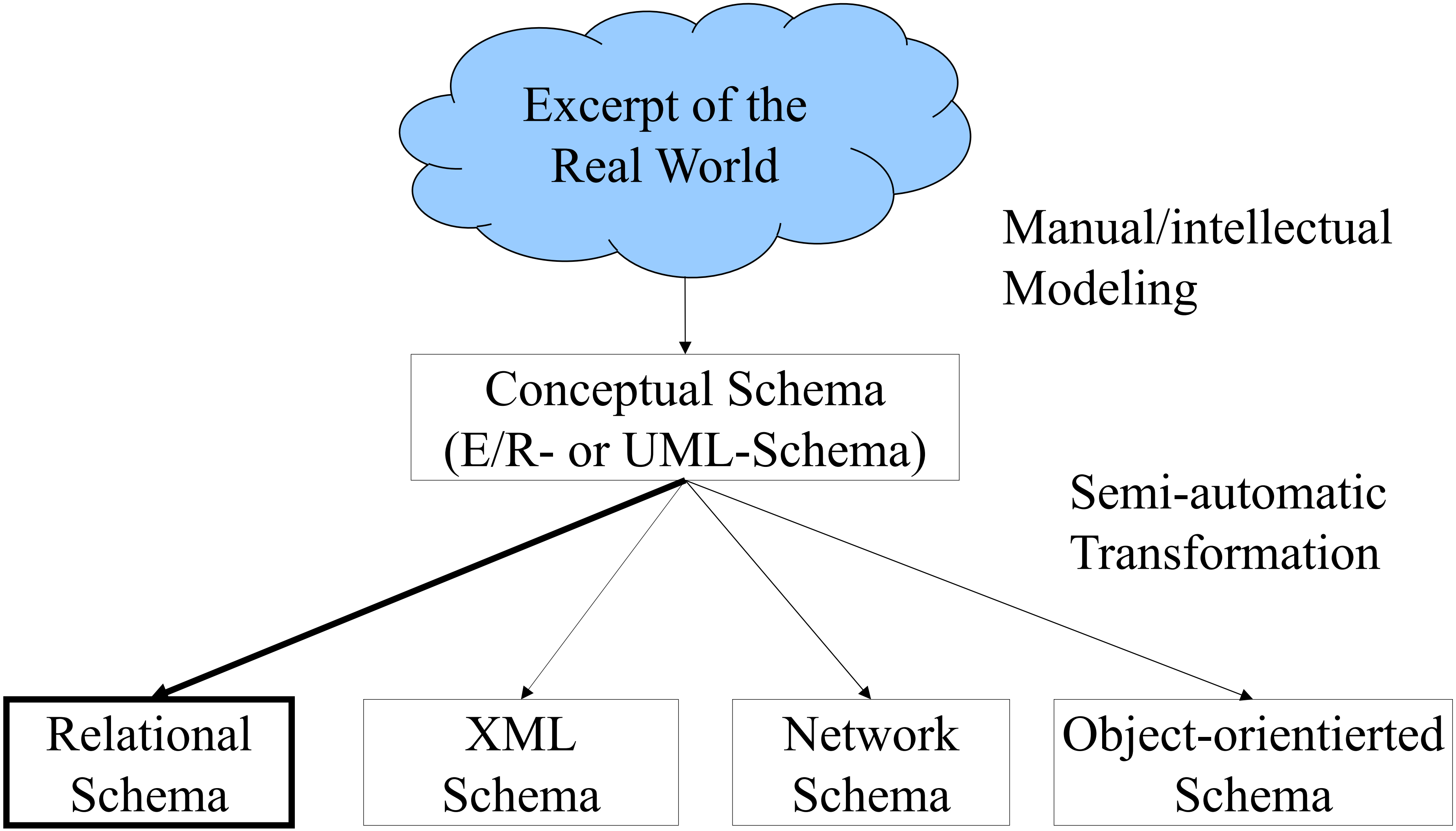
DBS can take care automatically of many things – but the user has to specify

- Requirements of the application
- Characteristics of the data

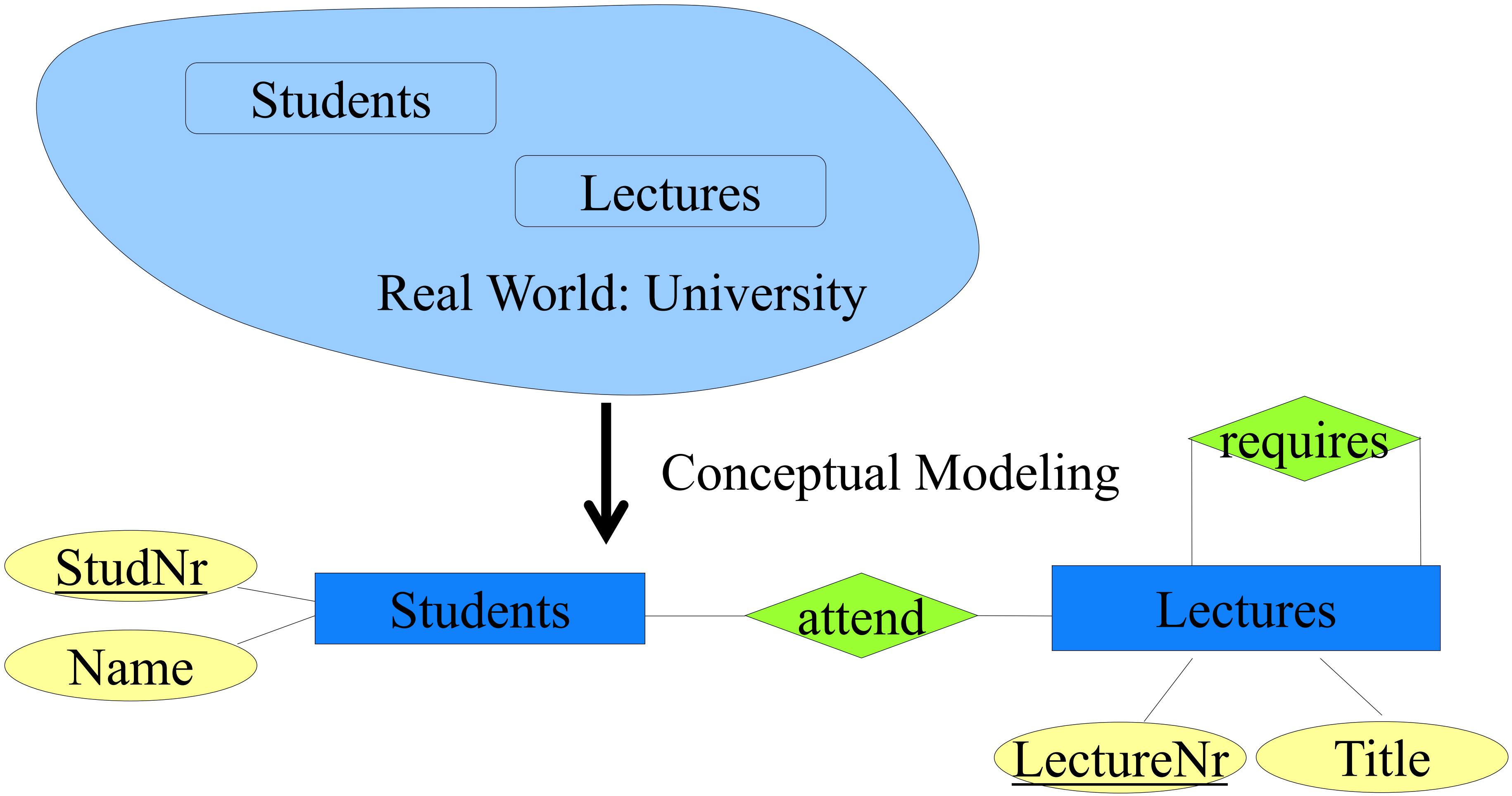
Two important concepts during DBS design:

- Data Model: How to describe the data?
- Data Schema: Concrete description of the data (using the chosen data model)

Data modeling



Modeling a small example application: E/R



Relational Data Model

Students	
StudNr	Name
26120	Fichte
25403	Jonas
...	...

attend	
StudNr	Lecture Nr
25403	5022
26120	5001
...	...

Lectures	
Lecture Nr	Title
5001	Grundzüge
5022	Glaube und Wissen
...	...

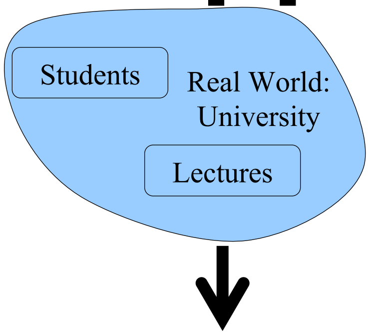
Select Name
From Students, attend, Lectures
Where Students.StudNr = attend.StudNr **and**
attend.LectureNr = Lectures.LectureNr **and**
Lectures.Title = ‘Grundzüge‘**;**

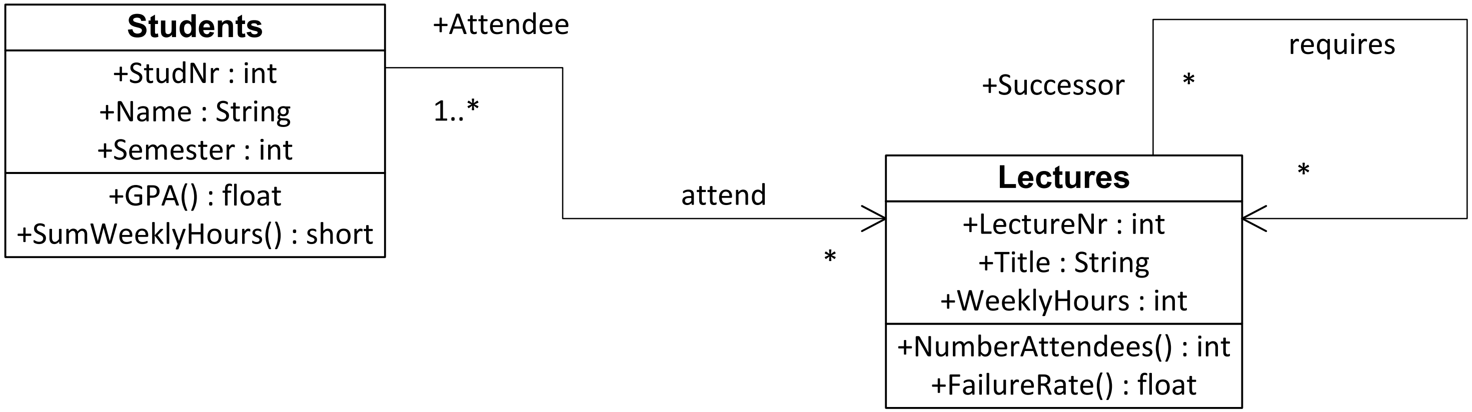
Logical Data Models

- Network Model
- Hierarchical Model
- **Relational Data Model**
- XML Model
- Object-orientierted Data Model
  - Object-relational Schema
- Deductive Data Model

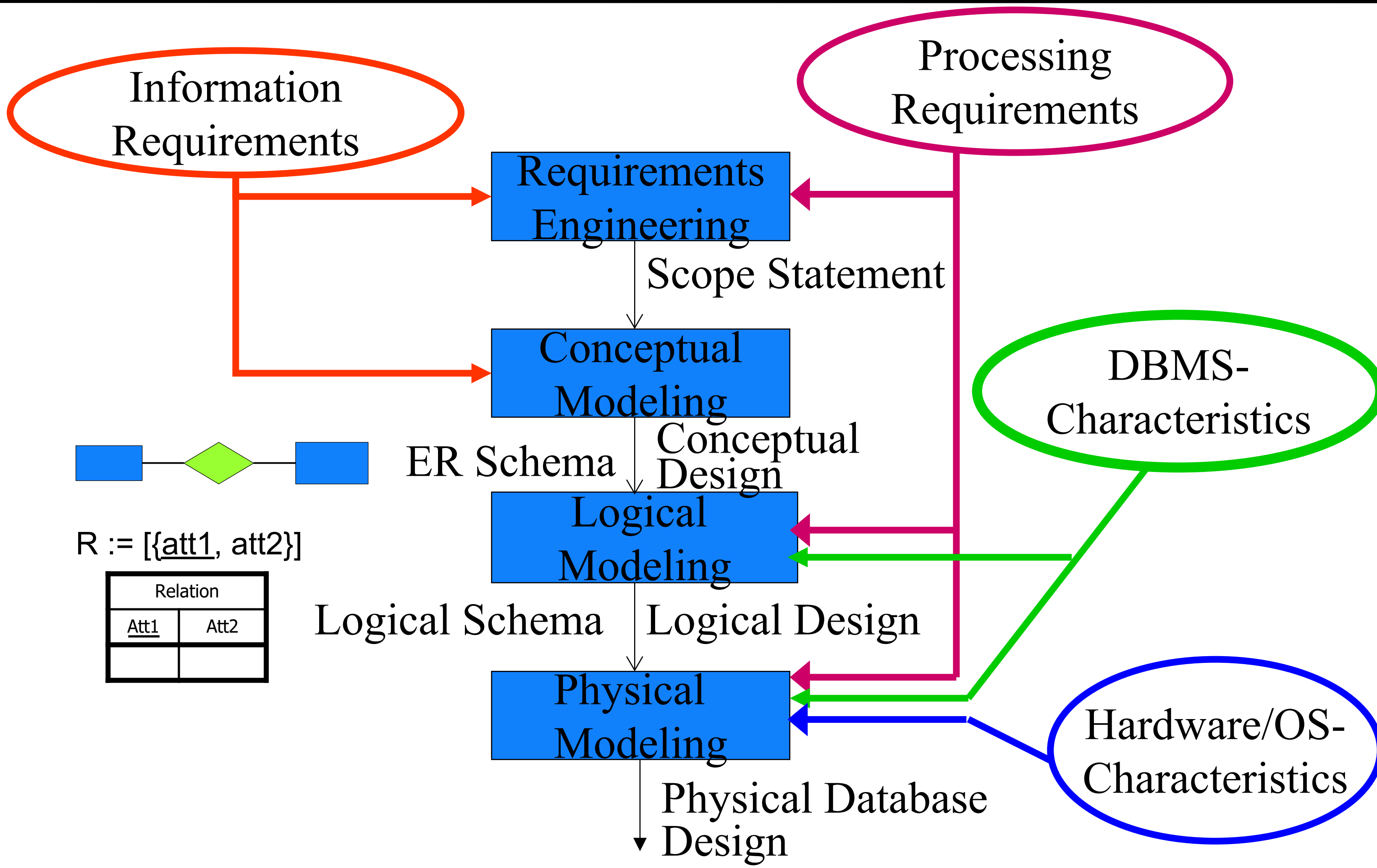
* [Michael Stonebraker: What Goes Around Comes Around]

Modeling a small example application: UML

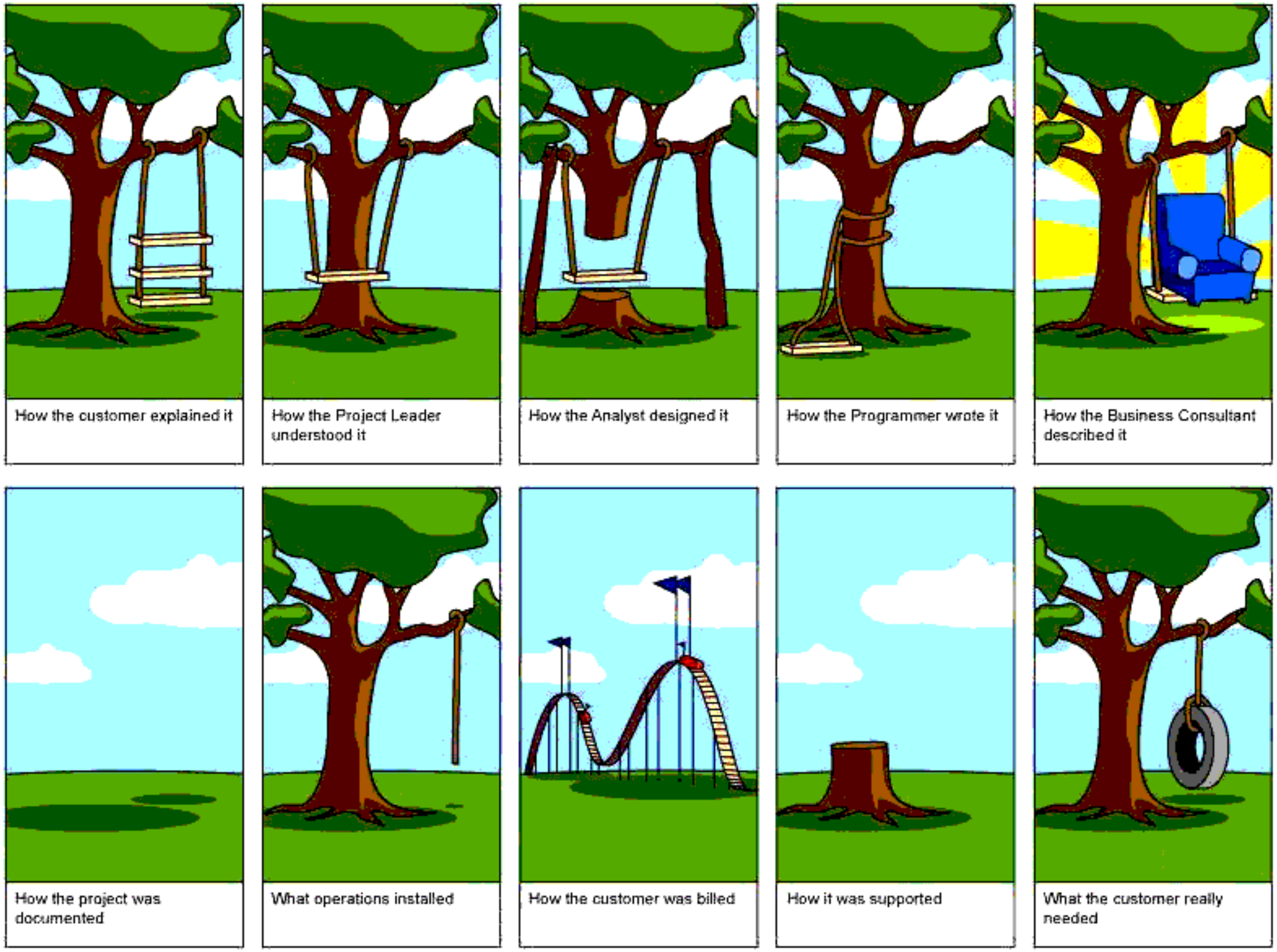




Phases of Database Design



Software Development and Ability to Communicate



Requirements Engineering

Create a ”Scope Statement” consisting of:

- Entity description
- Relation description
- Process description

Entity Description

University Employees

- -Quantity: 1000
- -Attributes
  - ❖ EmpNumber
    - •Type: Integer
    - •Domain: 0...999.999.99
    - •Defined: 100%
    - •Identifying: yes
    - •Example: 007
  - ❖ Salary
    - •Type: decimal
    - •Length: (7,2)
    - •Unit: Euro per month
    - •Defined: 10%
    - •Identifying: no
  - ❖ Level
    - •Type: String
    - •Length: 2
    - •Defined: 100%
    - •Identifying: no
    - •Example: W2

Relation Description: *exam*

Involved Objects:

- Professor as Tester
- Student as Testee
- Lecture as Test Subject

Attributes of the Relation:

- Date
- Time
- Grade

Quantity: 100 000 per year

Process Description : *Issue a Certificate*

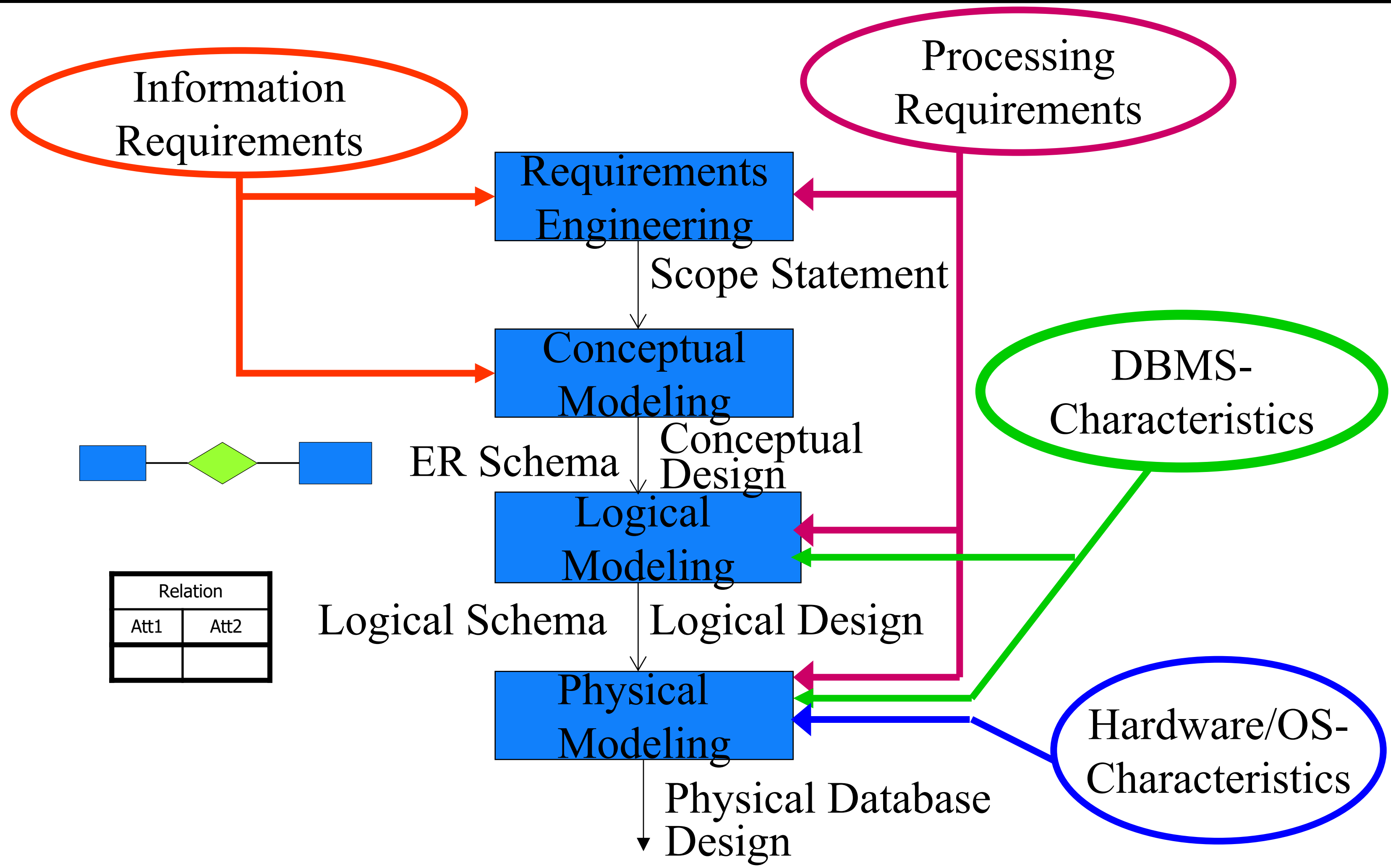
- Frequency: semiannually
- Required Data
  - Tests
  - Examination Rules
  - Student‘s Records
  - ...
- Priority: high
- Data Volume to be processed
  - 500 Students
  - 3000 Tests
  - 10 Versions of Examination Rules

Creating a Specification

The actual analysis is an iterative process:

- Customer tells developer his/her needs
- Developer notes everything down (s/he understood) in his/her „language" . . .
- . . . and translates it into the "language" of the customer
- This is shown to the customer who does not agree with everything
- Change requests are agreed on
- Back to step 2

Phases of Database Design



Conceptual Design

The ideal design (the ideal specification) is

- unique
- complete
- comprehensible (for all participants)
- nonredundant
- . . . and not reachable in reality

Entity/Relationship-Modeling

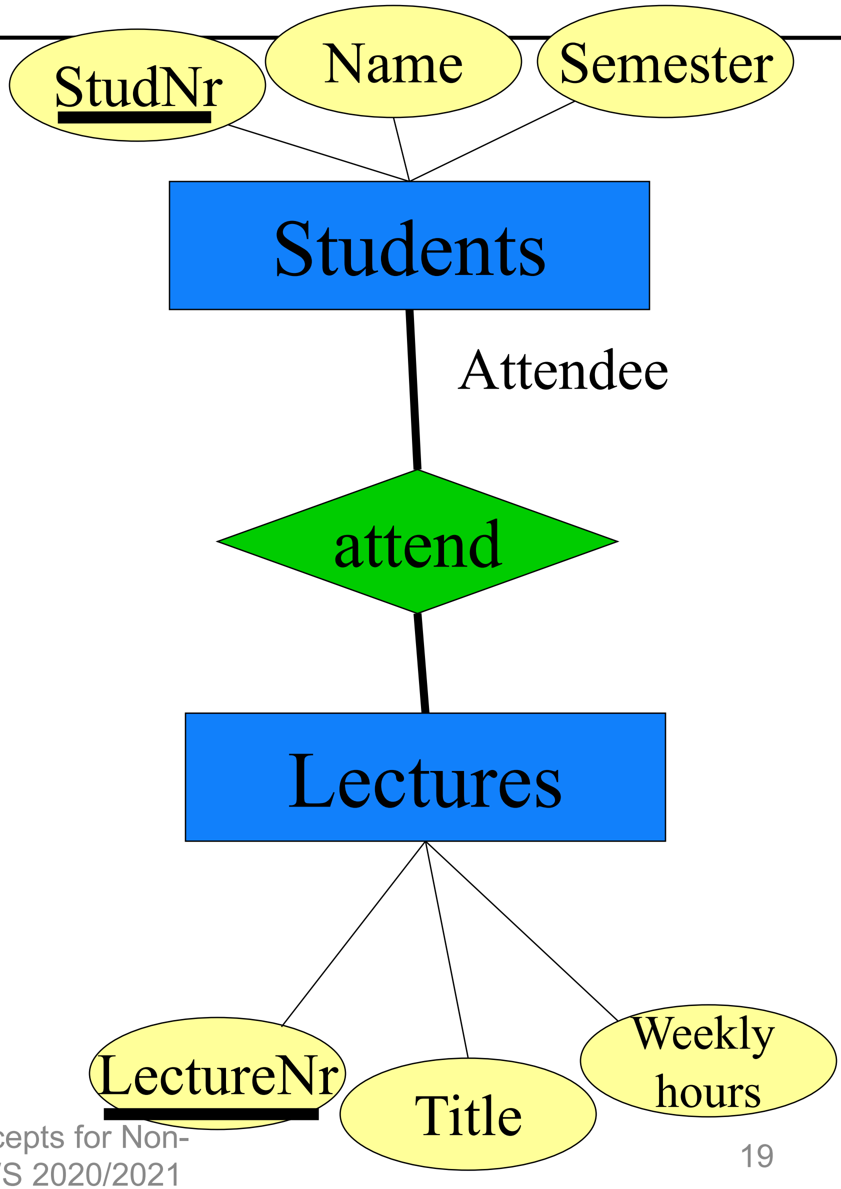
Entity

Relationship

Attribute (property)

Key (identification)

Role

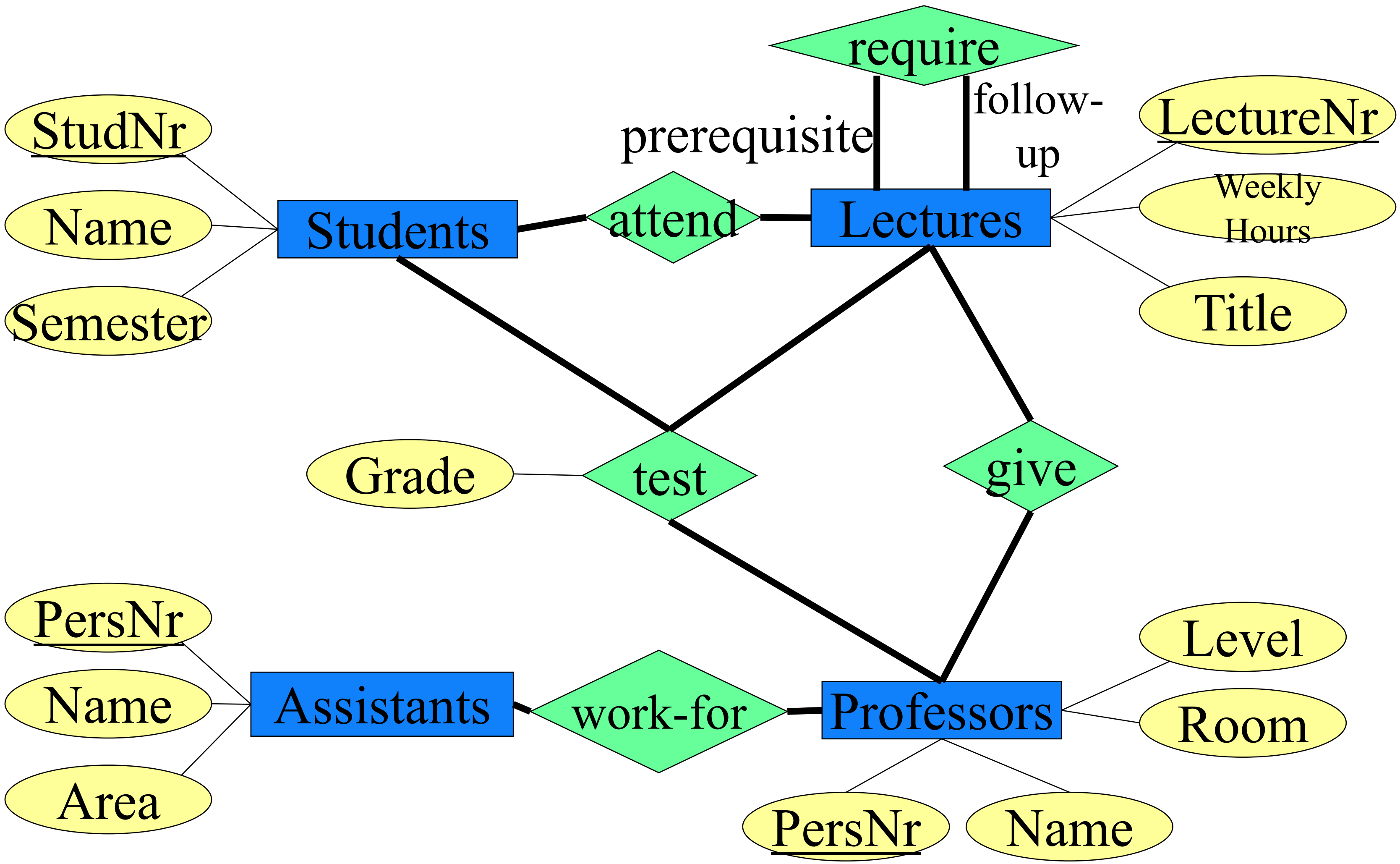


Entity/Relationship-Modeling

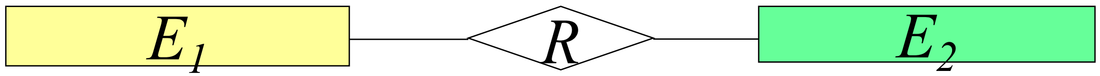
Mathematically: “Relational Schema”

- *Entities* are sets of n-ary tuples:
  - Students = `{[1, „Sam“, 3], [2, „Jack“, 5], …}`
- *Relationships* are n-ary relations:
  - `attend ⊆ Students × Lectures = {[1, 101], [1, 102], [2, 101]}`

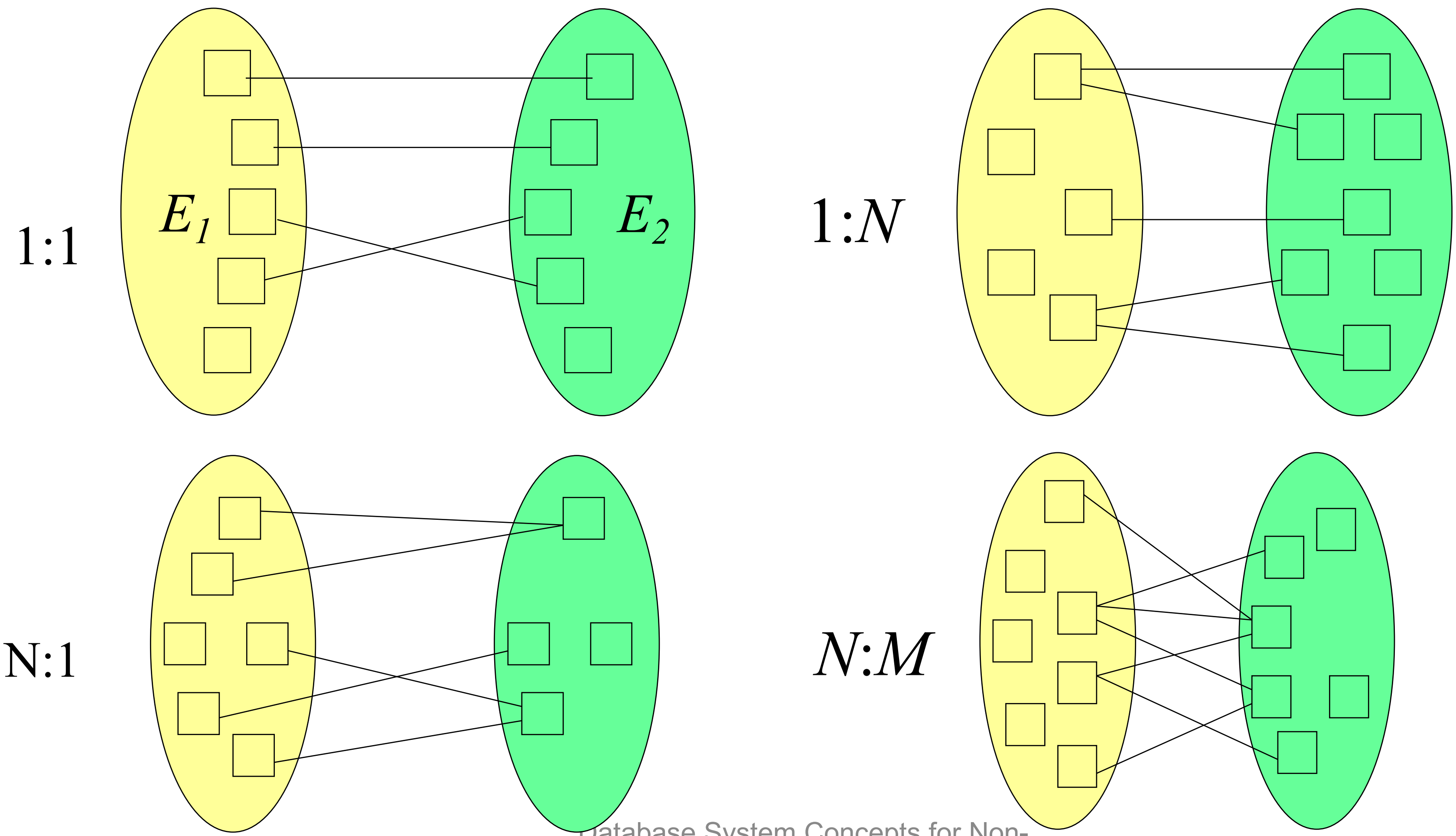
University Schema



Functionalities

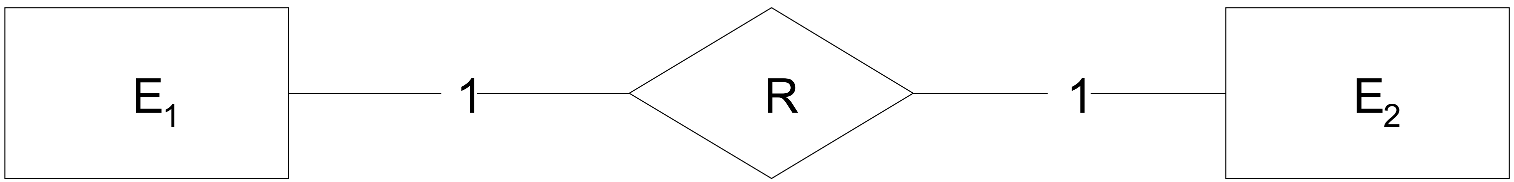


$R \subseteq E_1 \times E_2$



Relationship 1:1

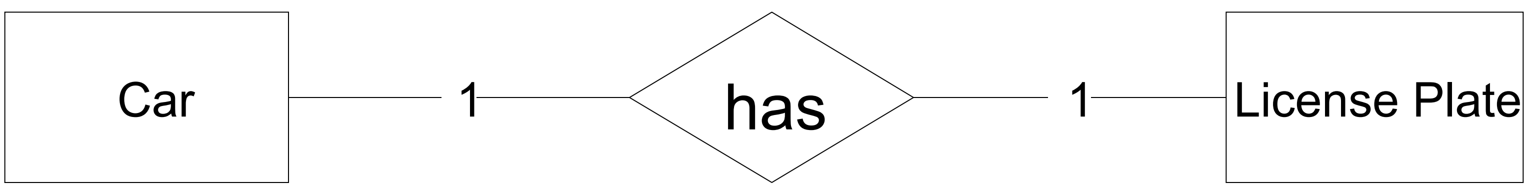
Relationship 1:1



One $e_1 \in E_1$ has 0 or 1 partners in E_2

One $e_2 \in E_2$ has 0 or 1 partners in E_1

Example:

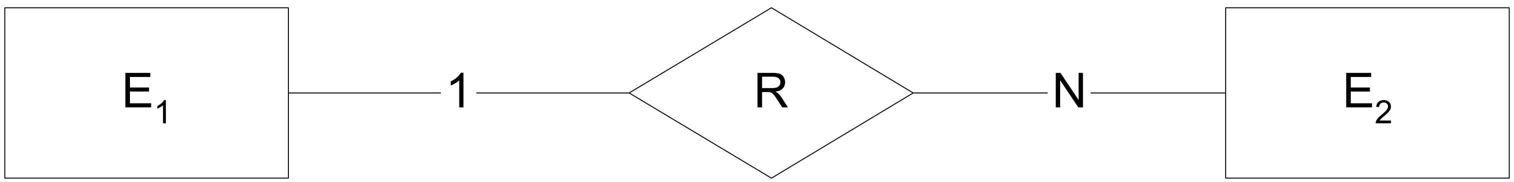


one car has one license plate

one license plate belongs to one car

Relationship 1:N

Relationship 1:N



One $e_1 \in E_1$ has N partners in E_2

One $e_2 \in E_2$ has 0 or 1 partners in E_1

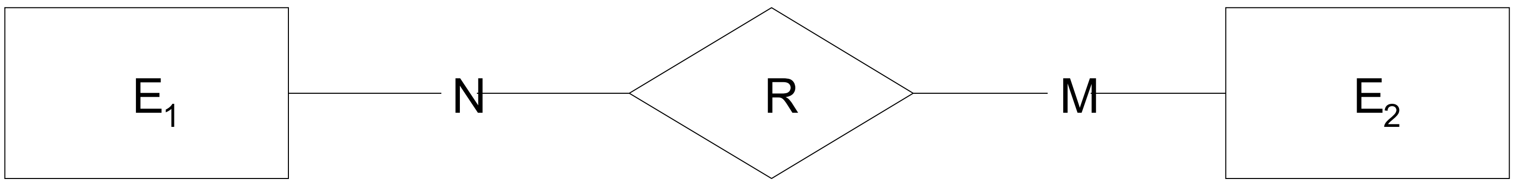
Example:

one mentor advises several students

one student is advised by one mentor

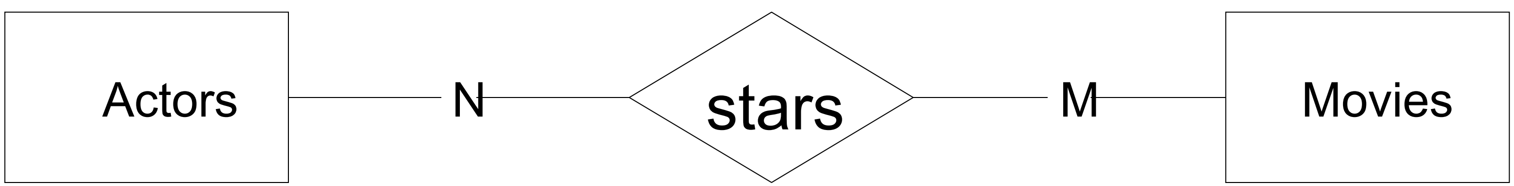
Relationship N:M

Relationship N:M



One $e_1 \in E_1$ has N partners in E_2
One $e_2 \in E_2$ has N partners in E_1

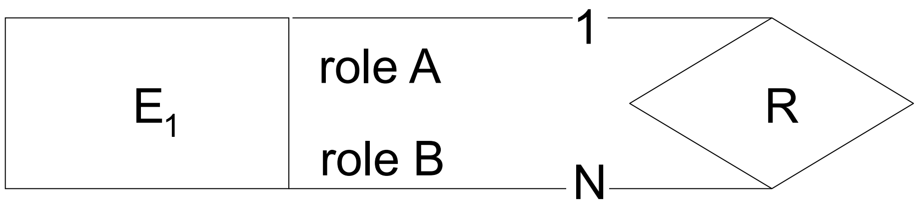
Example:



one actor stars in several movies
one movie has several actors

Recursive Relationship 1:N

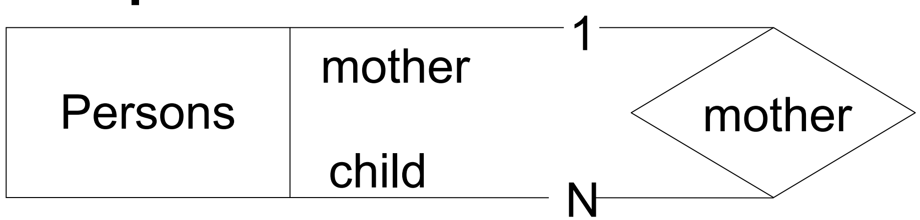
Relationship 1:N



One $e_1 \in E_1$ is called 'roleA' and has N partners in E_1

One $e_2 \in E_1$ is called 'roleB' and has 0 or 1 partners in E_1

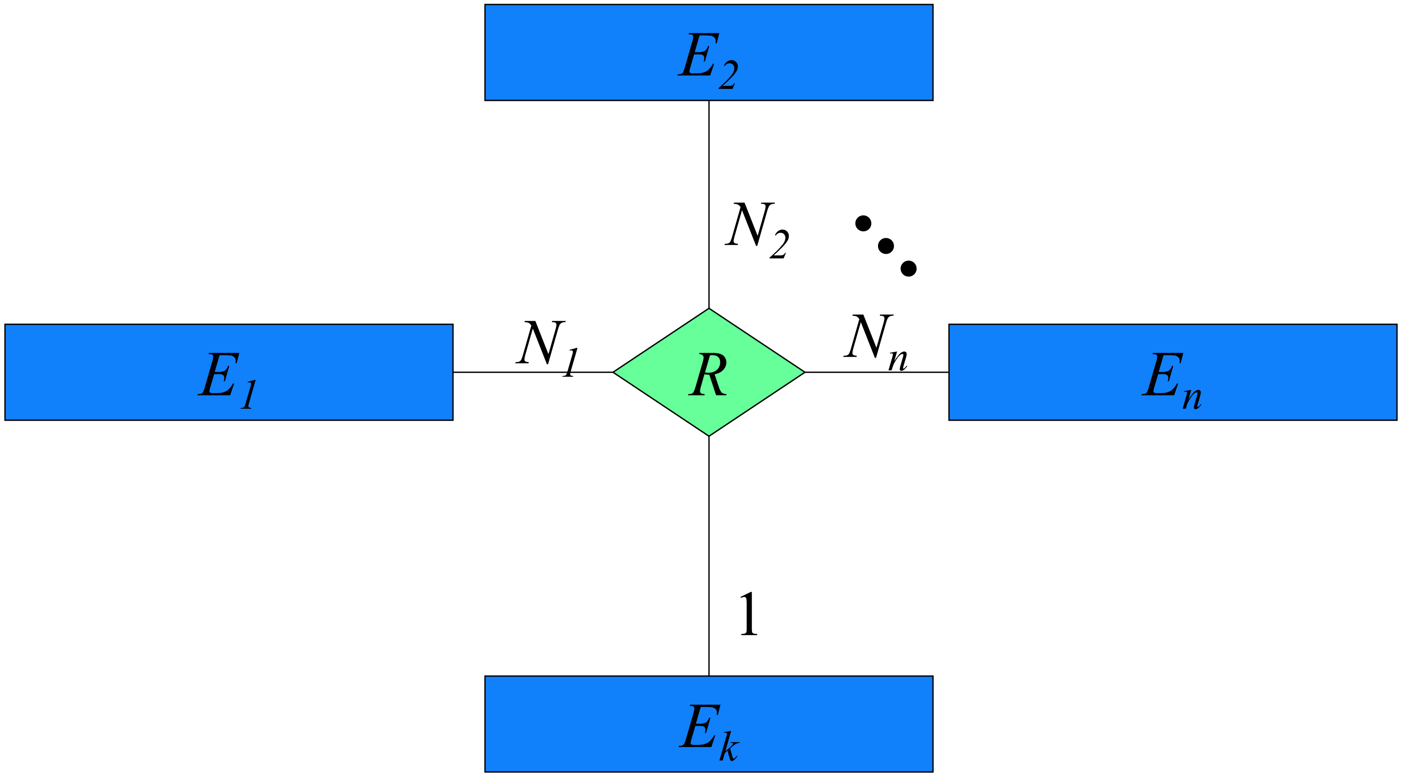
Example:



One person is the mother of several persons (children)

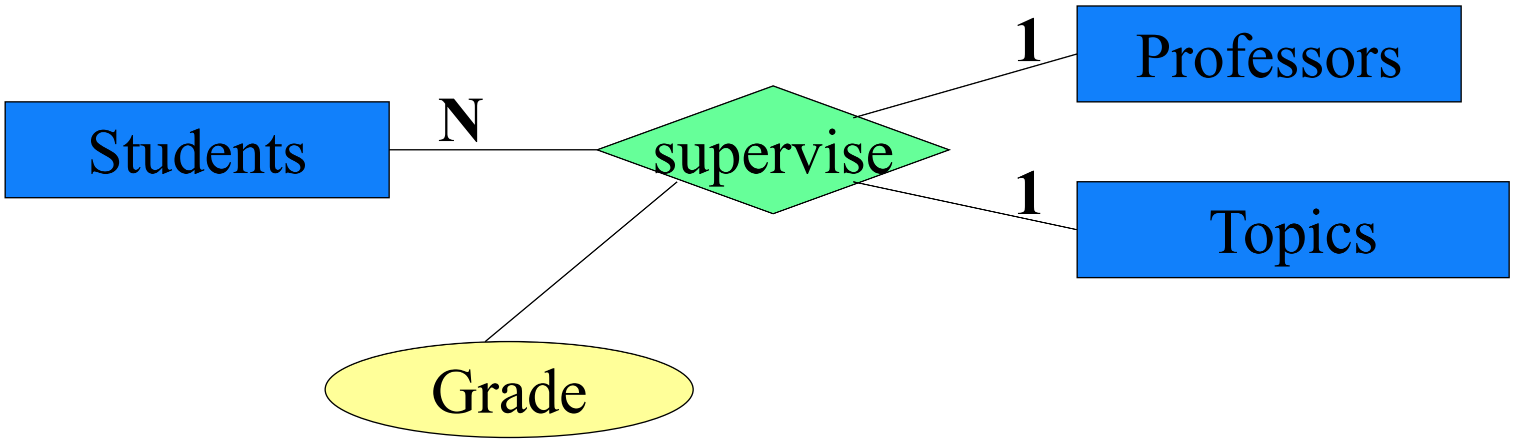
One person is the child of one person (mother)

Functionalities in *n*-ary Relationships



$$R : E_1 \times E_2 \times \ldots \times E_n \rightarrow E_k$$

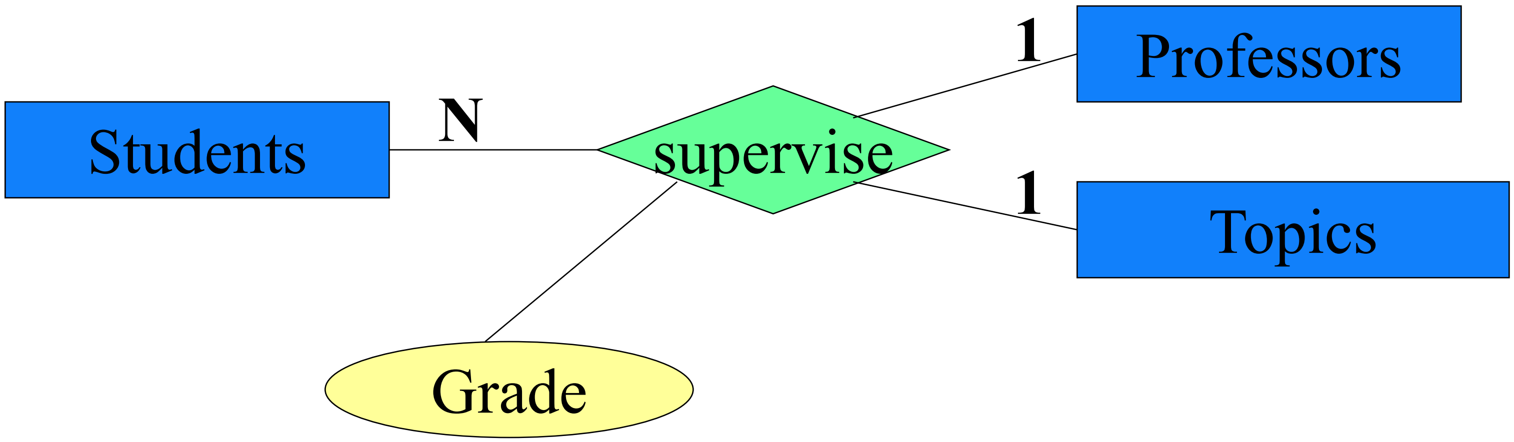
Example Seminar



supervise : Topics x Students → Professors

supervise : Professors x Students → Topics

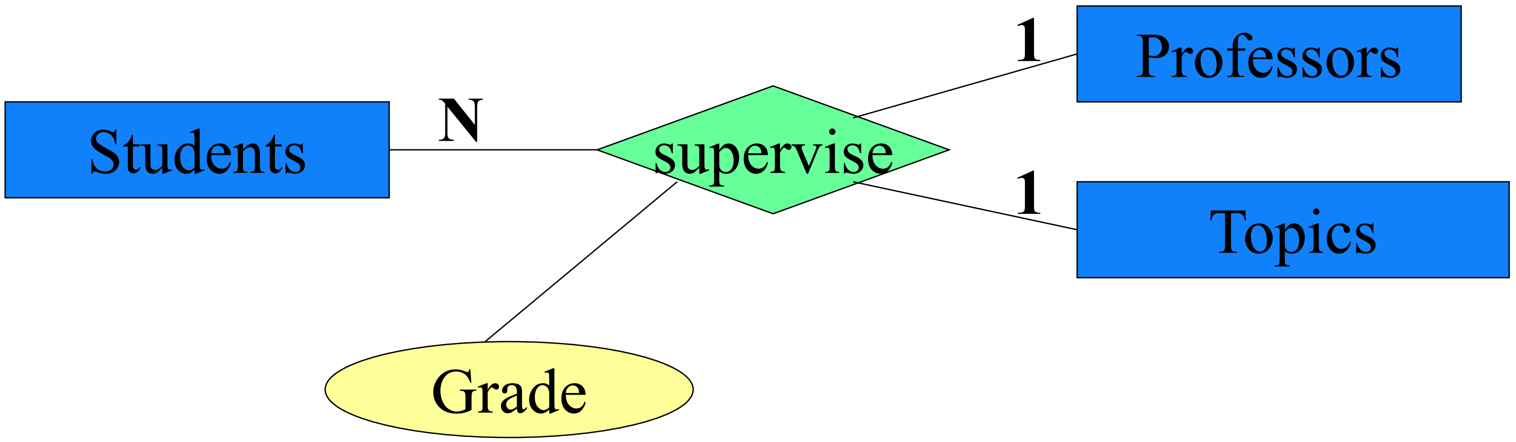
Example Seminar



Student x Professor -> Topic

1. A Student is only allowed to work on one topic with a given professor.

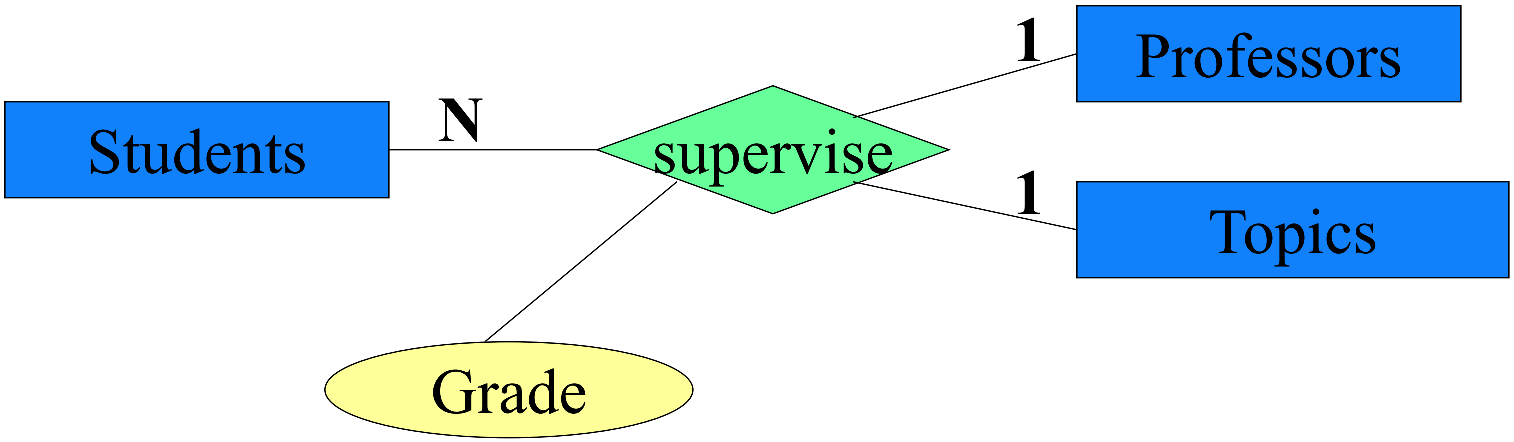
Example Seminar



Student x Topic -> Professor

2. Students may work on the same topic only once – thus they may not work on the same topic again with another professor.

Example Seminar



Not: Professor x Topic -> Student

3. Professors can reuse the same topic – i.e., one professor can give the same topic to different students. (absence of: PXT -> S)

Thereby induced Consistency Constraints

1. A Student is only allowed to work on one topic with a given professor. (SxP -> T)

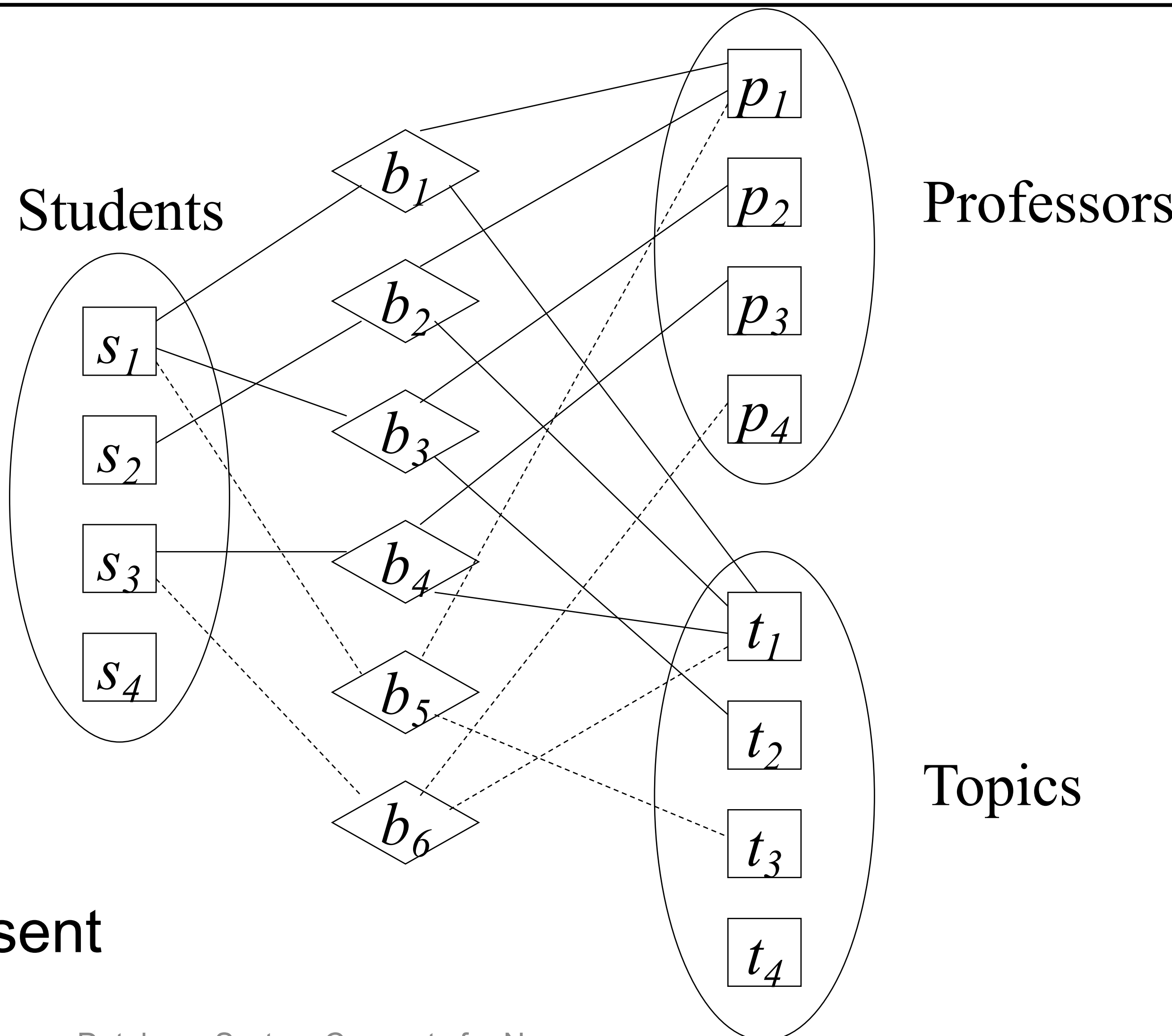
2. Students may work on the same topic only once – thus they may not work on the same topic again with another professor. (SxT -> P)

3. Professors can reuse the same topic – i.e., one professor can give the same topic to different students. (absence of: PxT -> S)

4. The same topic can be given by different professors – but to different students. (absence of: PxT -> S)

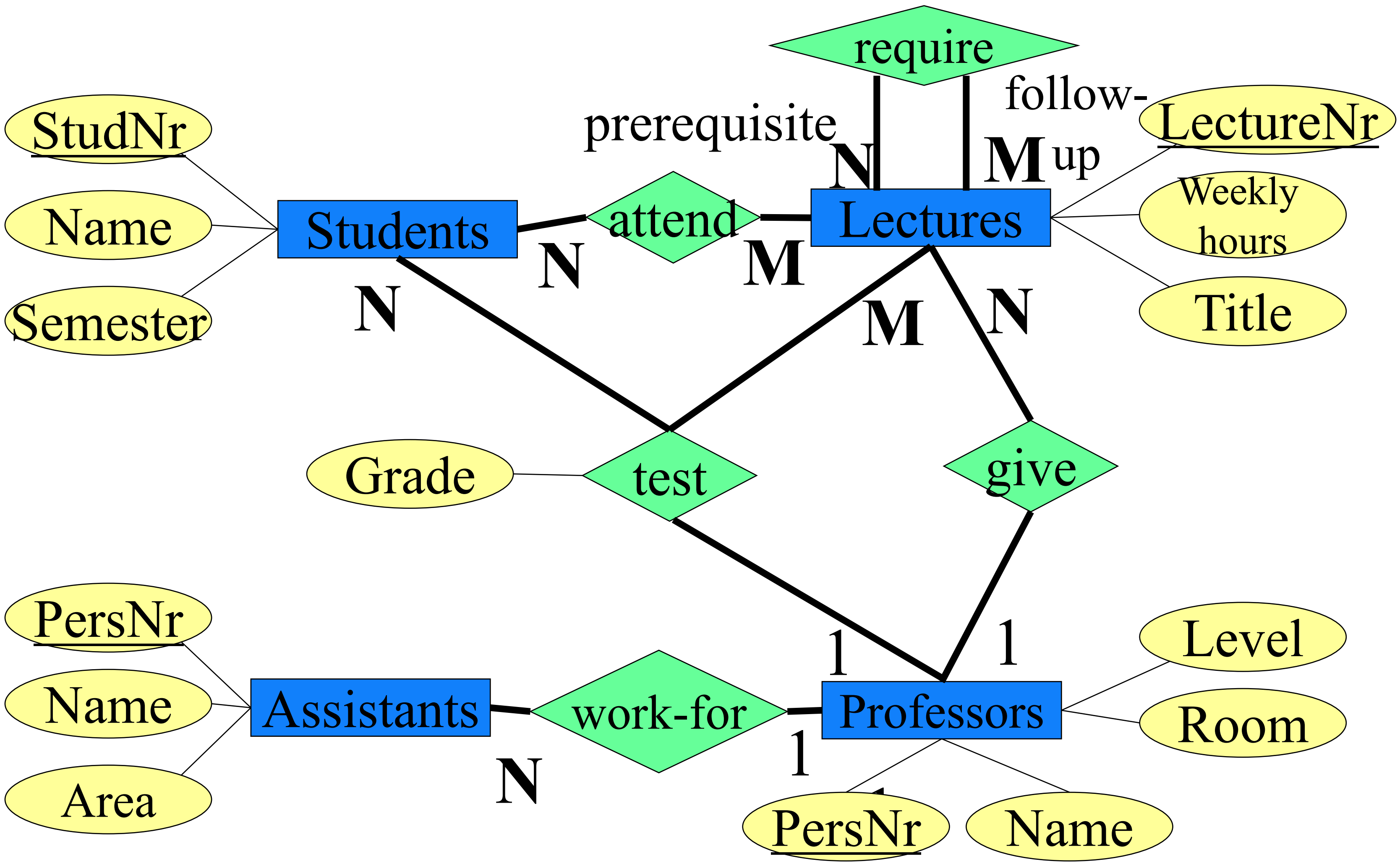
5. The same professor can give different topics – but to different students. (absence of: PxT -> S)

Occurrence of the Relationship *supervise*

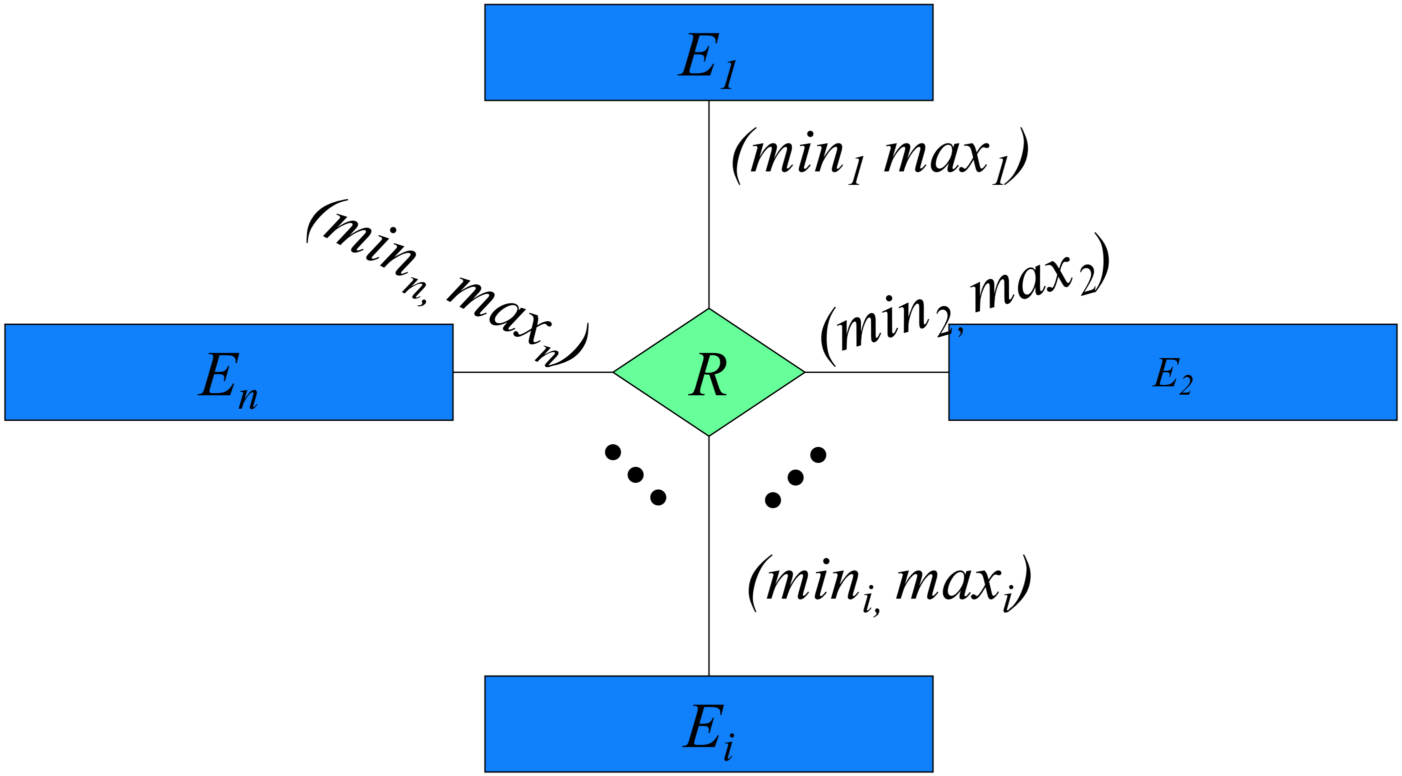


Dashed lines represent illegal occurrences

University Schema



(min, max)-Notation



$R \subseteq E_1 \times \ldots \times E_i \times \ldots \times E_n$

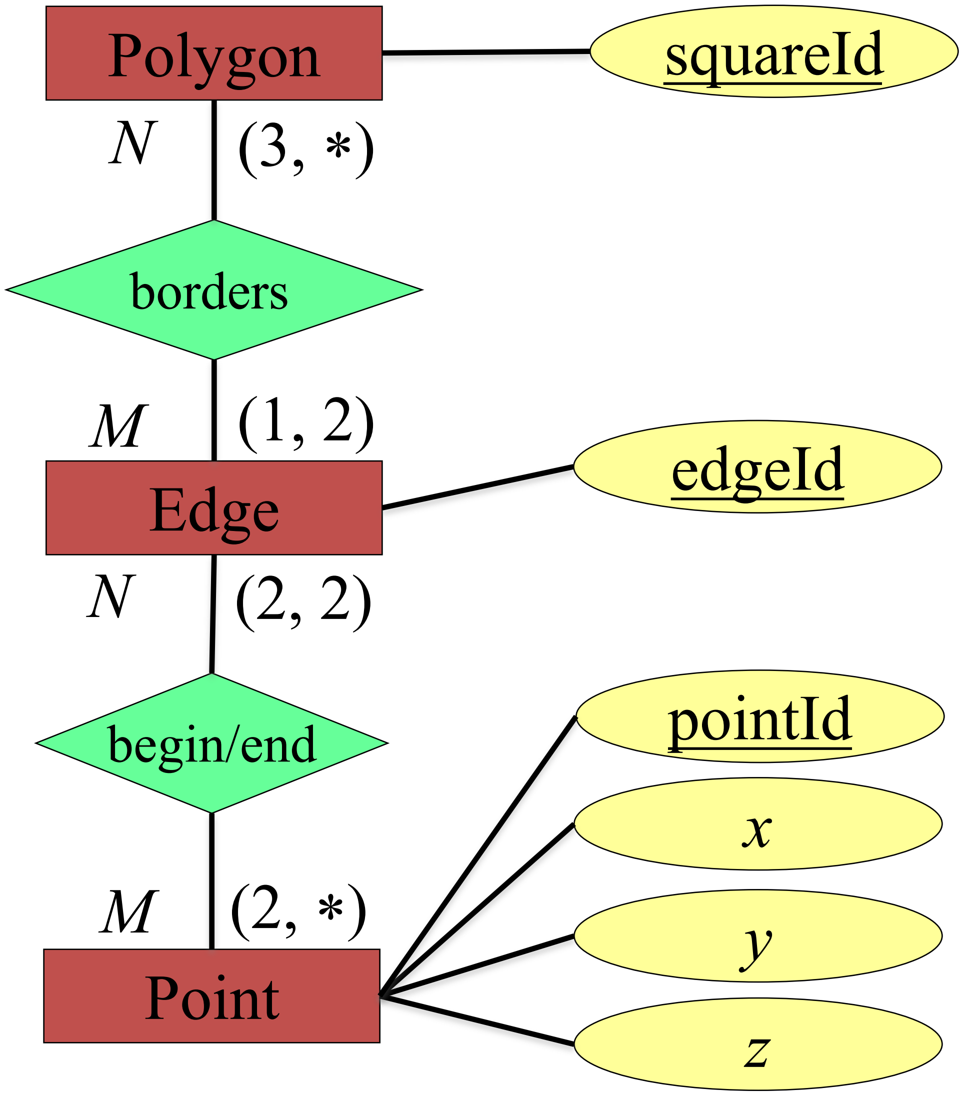
For every $e_i \in E_i$ *there are*

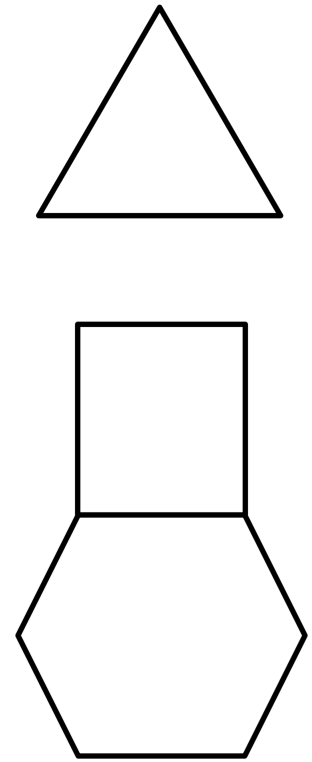
- *at least* min_i tuples $(\ldots, e_i, \ldots) \in R$ and
- at most max_i tuples $(\ldots, e_i, \ldots) \in R$

Example (min, max)

one mentor advises up to 20 students
one student is advised by exactly one mentor

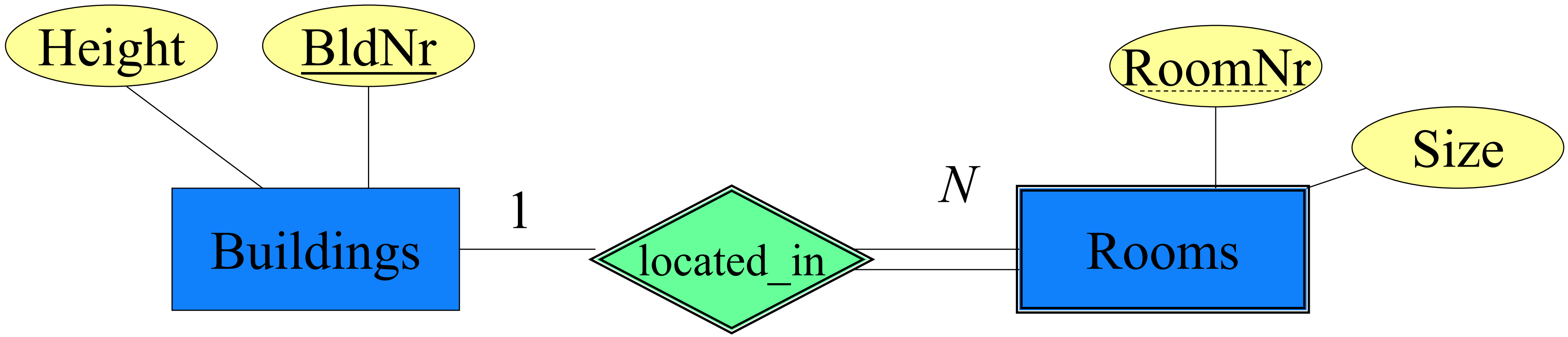
Min,max Notation and Functionalities





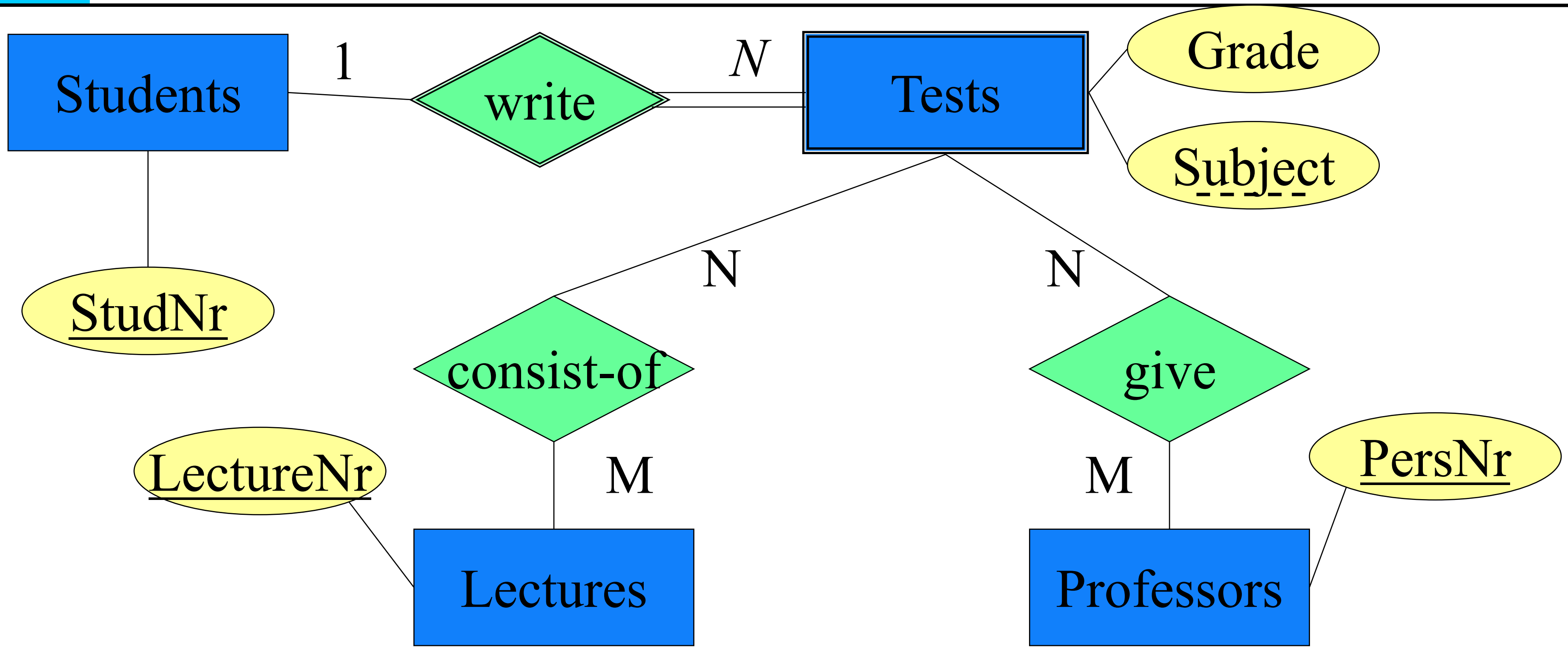
Min-max:
A Polygon has at least 3 Edges.
An Edge has 1 or 2 Polygons.

Weak Entities



- Relationship between "strong" and "weak " type is 1:*N* (or 1:1 in rare cases) - why not *N:M*?
- The existence of a room depends on the existence of the associated building
- RoomNr is unique only within the building
- Key of Rooms is: RoomNr **and** BldNr

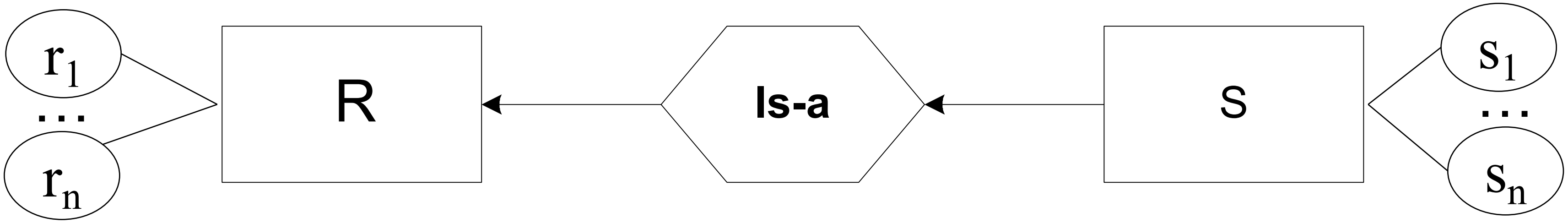
Tests as weak entity type



- Several professors design one test
- Several lectures are inquired in one test

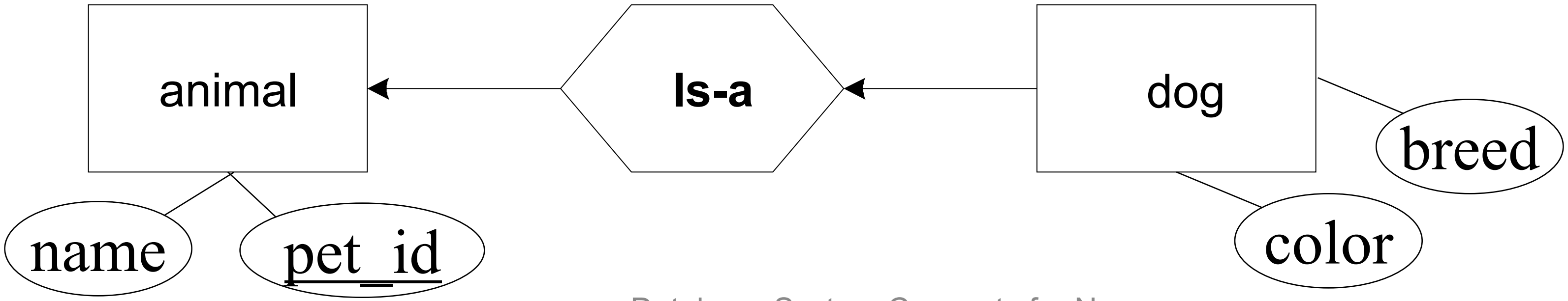
Generalization

Generalization / Specialization:

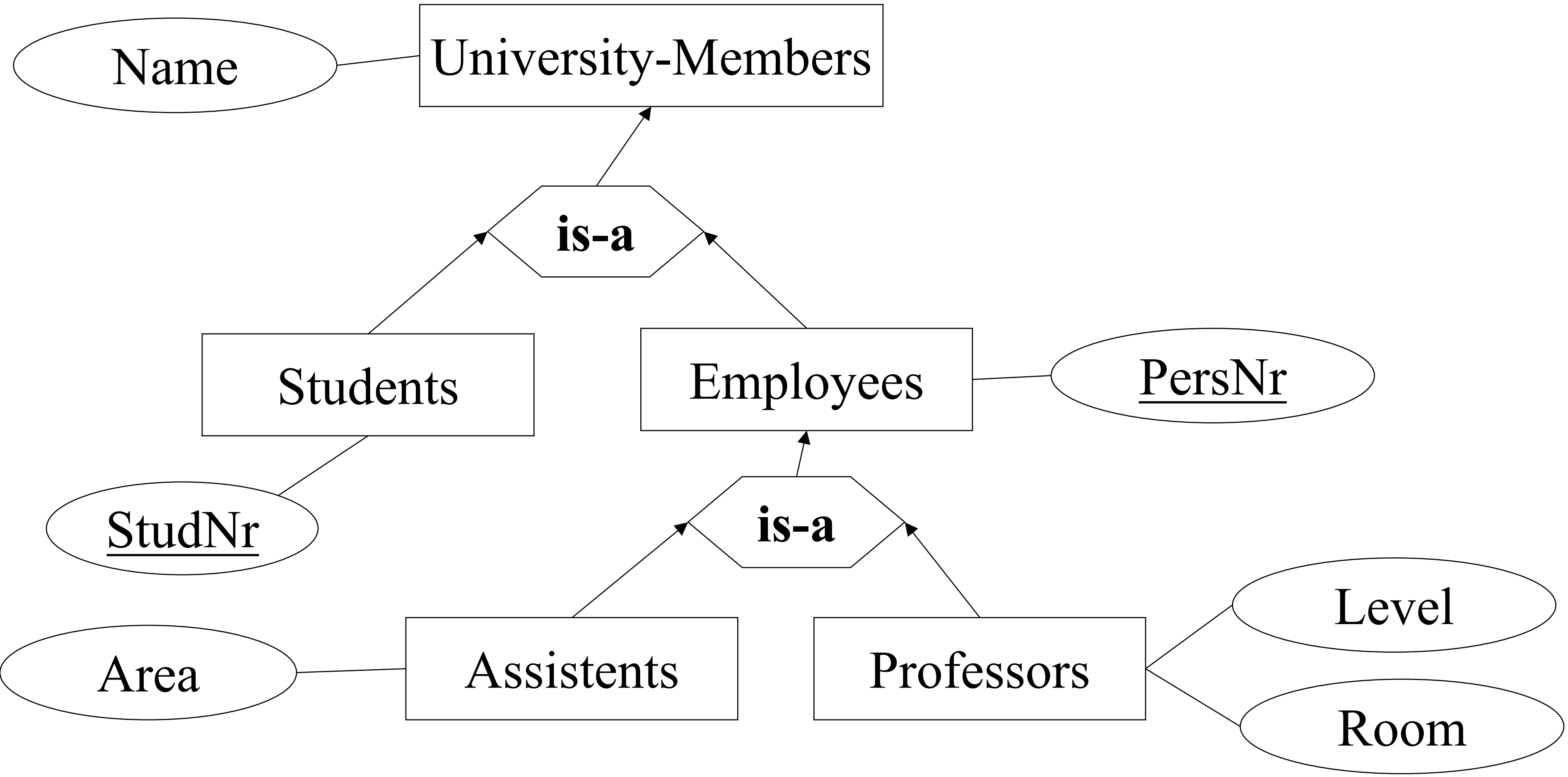


S is a specialization of R

Example:



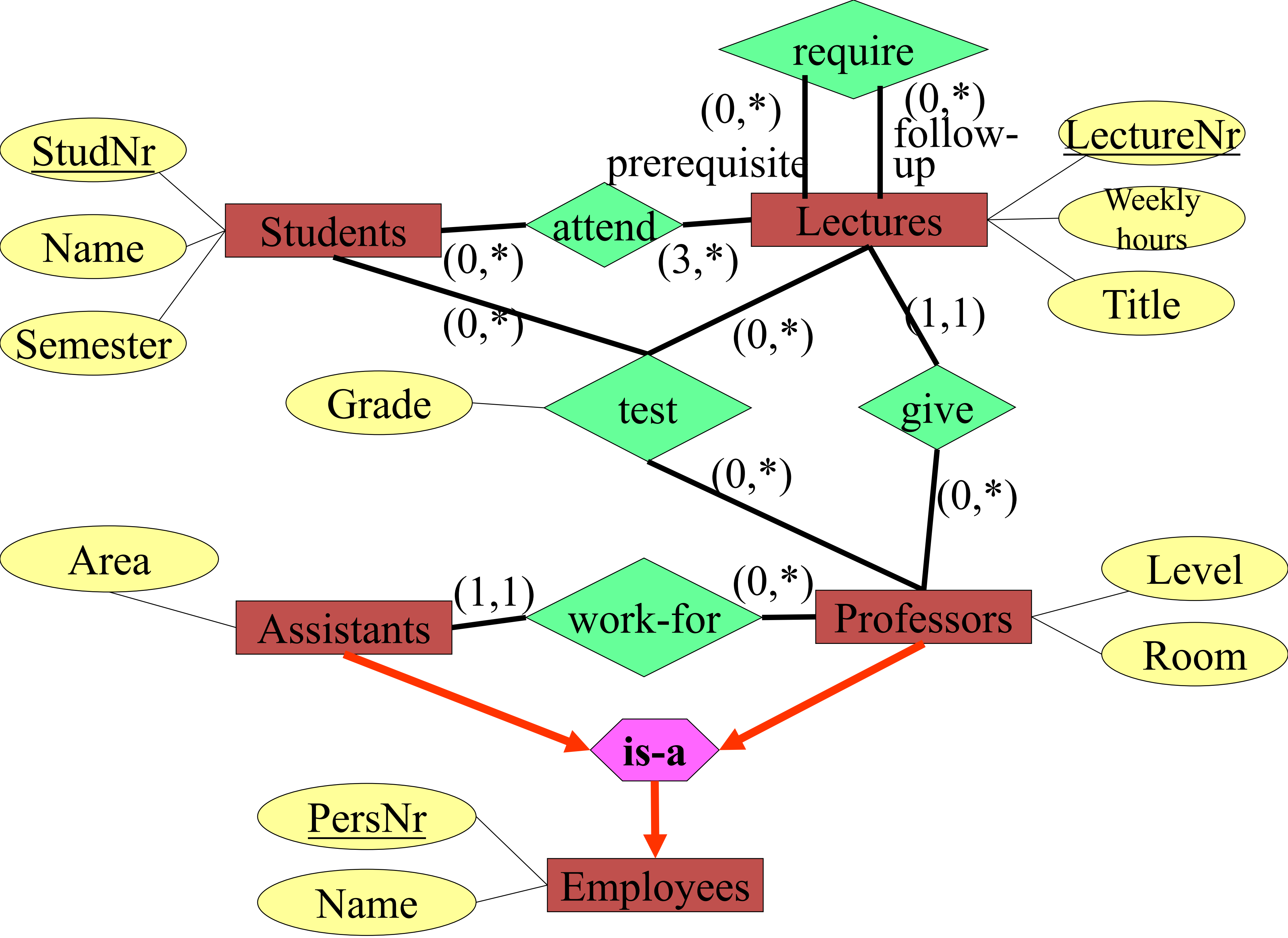
Generalization University



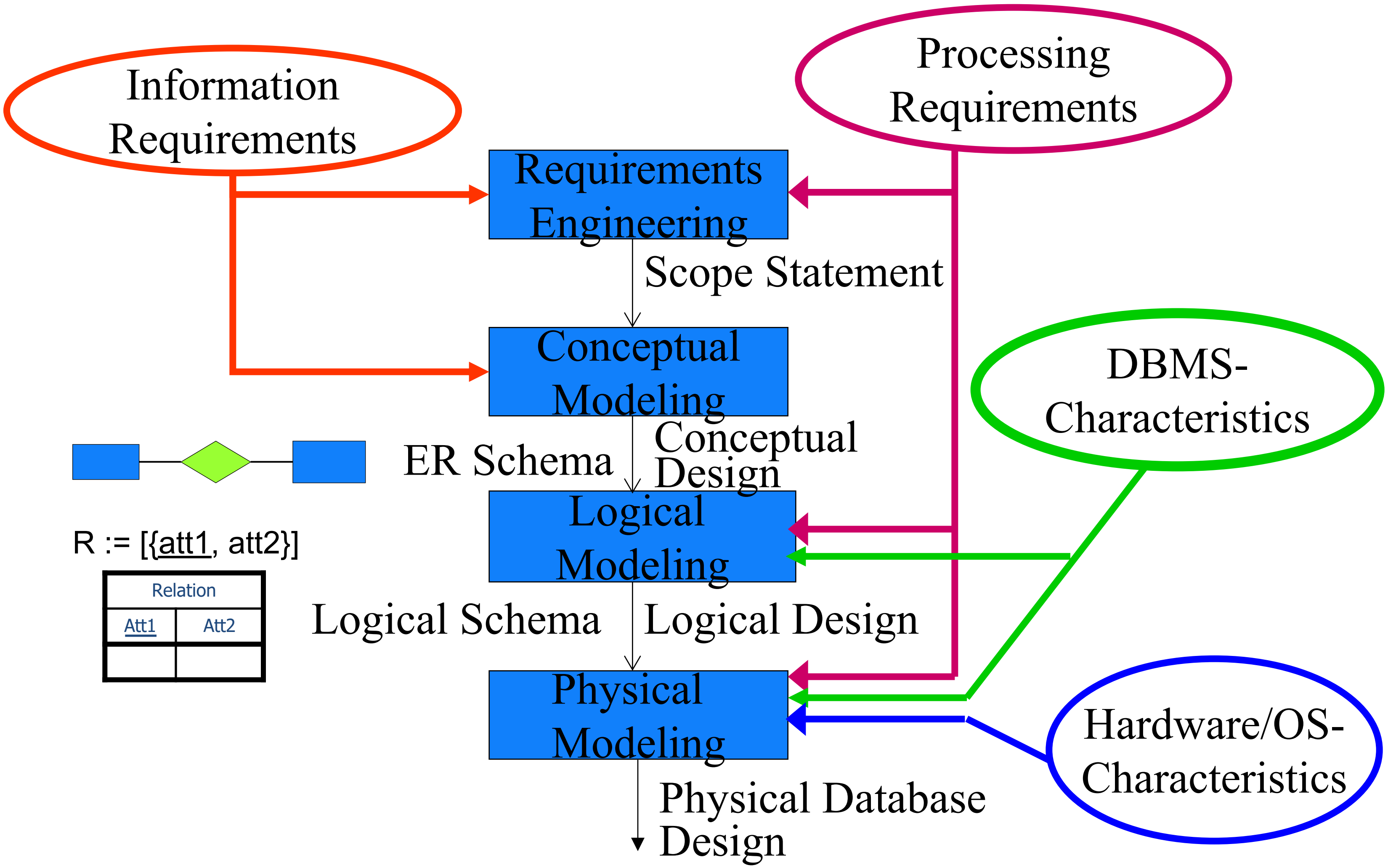
Conclusion

University schema with generalization and (min, max)-notation

➔ Nextpage


require
(0,*)
prerequisite
(0,*)
follow-
up
StudNr
Name
Semester
Students
attend
(0,*)
(3,*)
Lectures
LectureNr
Weekly
hours
Title
(0,*)
(0,*)
(1,1)
Grade
test
give
(0,*)
(0,*)
Area
(1,1)
work-for
(0,*)
Assistants
Professors
Level
Room
is-a
PersNr
Name
Employees

Where are we?



Information Requirements
Processing Requirements
Requirements Engineering
Scope Statement
Conceptual Modeling
DBMS-Characteristics
ER Schema
Conceptual Design
Logical Modeling
R := [{att1, att2}]
Relation
Att1
Att2
Logical Schema
Logical Design
Physical Modeling
Hardware/OS-Characteristics
Physical Database Design