

# Datenbankentwurf

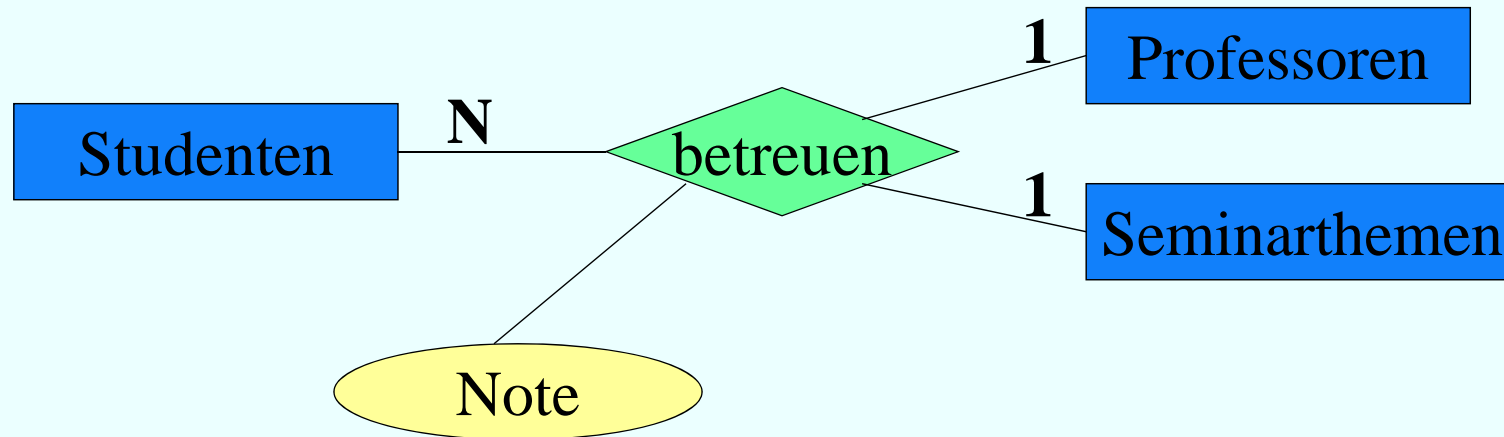
...bisher:

- konzeptueller Entwurf mittels E/R-Modell
- einfache Funktionalitäten
- (min, max)-Notation
- n-stellige Relationships ( $n > 2$ ) (siehe nächste zwei Folien)
- schwache Entities

...nun:

Generalisierung, Aggregation, UML-Modell

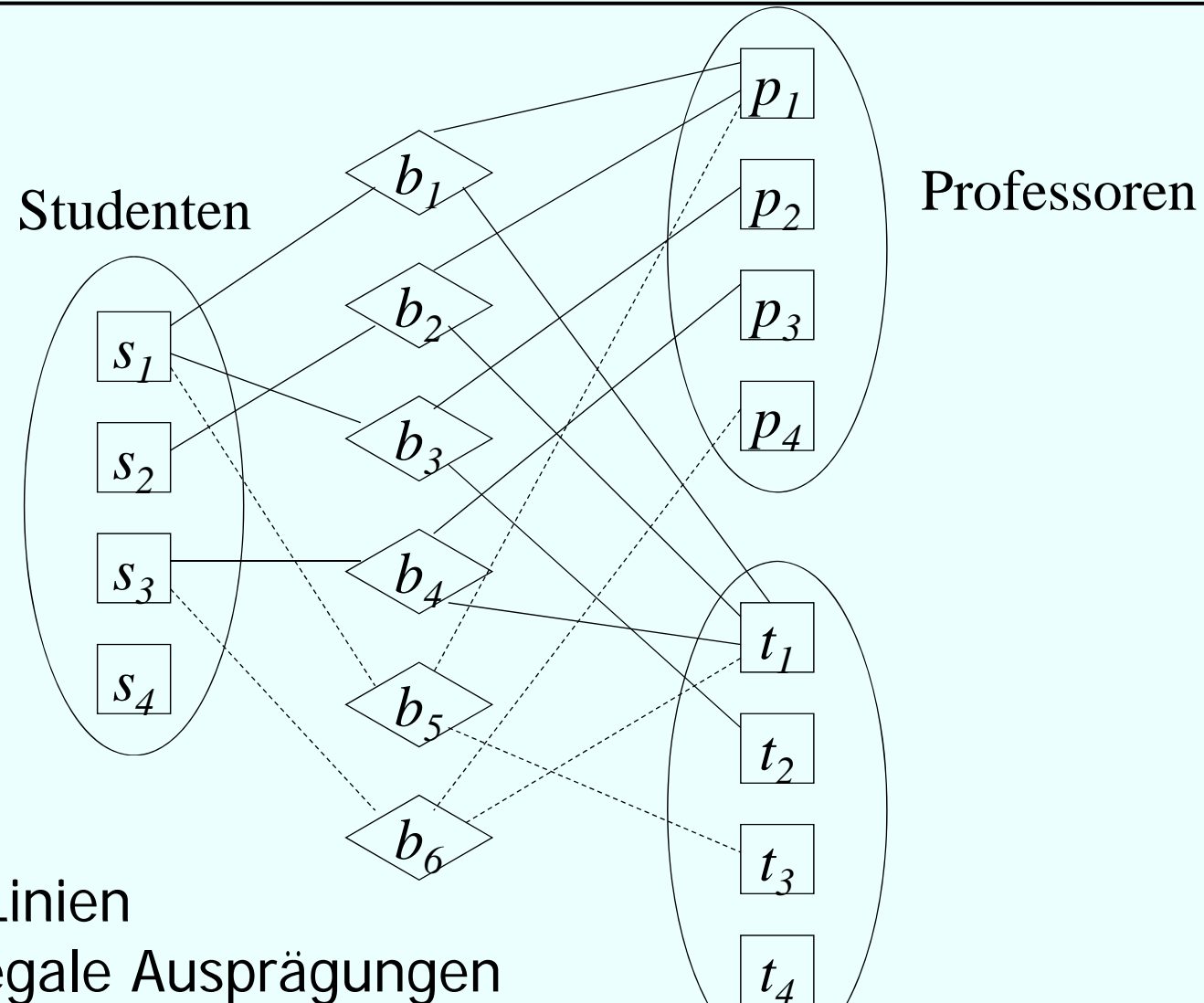
# Beispiel-Beziehung: *betreuen*



betreuen : Professoren x Studenten  $\rightarrow$  Seminarthemen

betreuen : Seminarthemen x Studenten  $\rightarrow$  Professoren

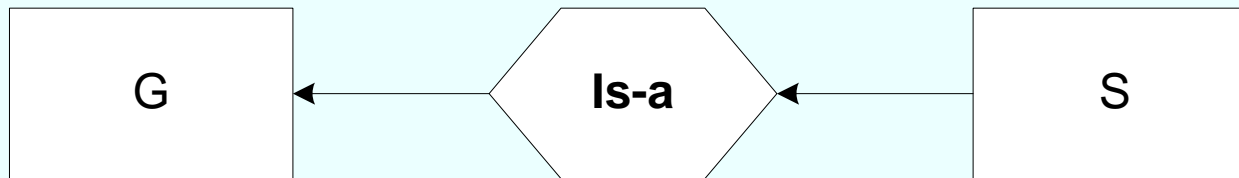
# Ausprägung der Beziehung *betreuen*



Gestrichelte Linien  
markieren illegale Ausprägungen

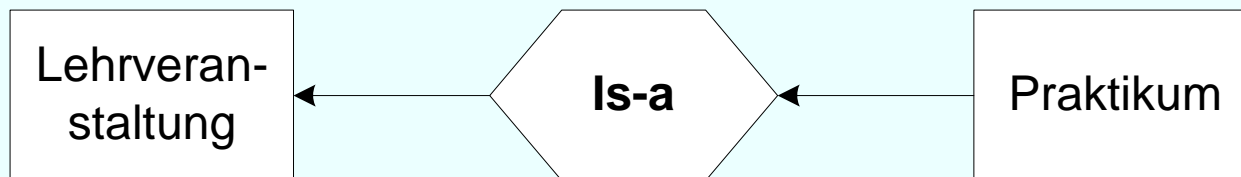
# Generalisierung

Generalisierung / Spezialisierung:

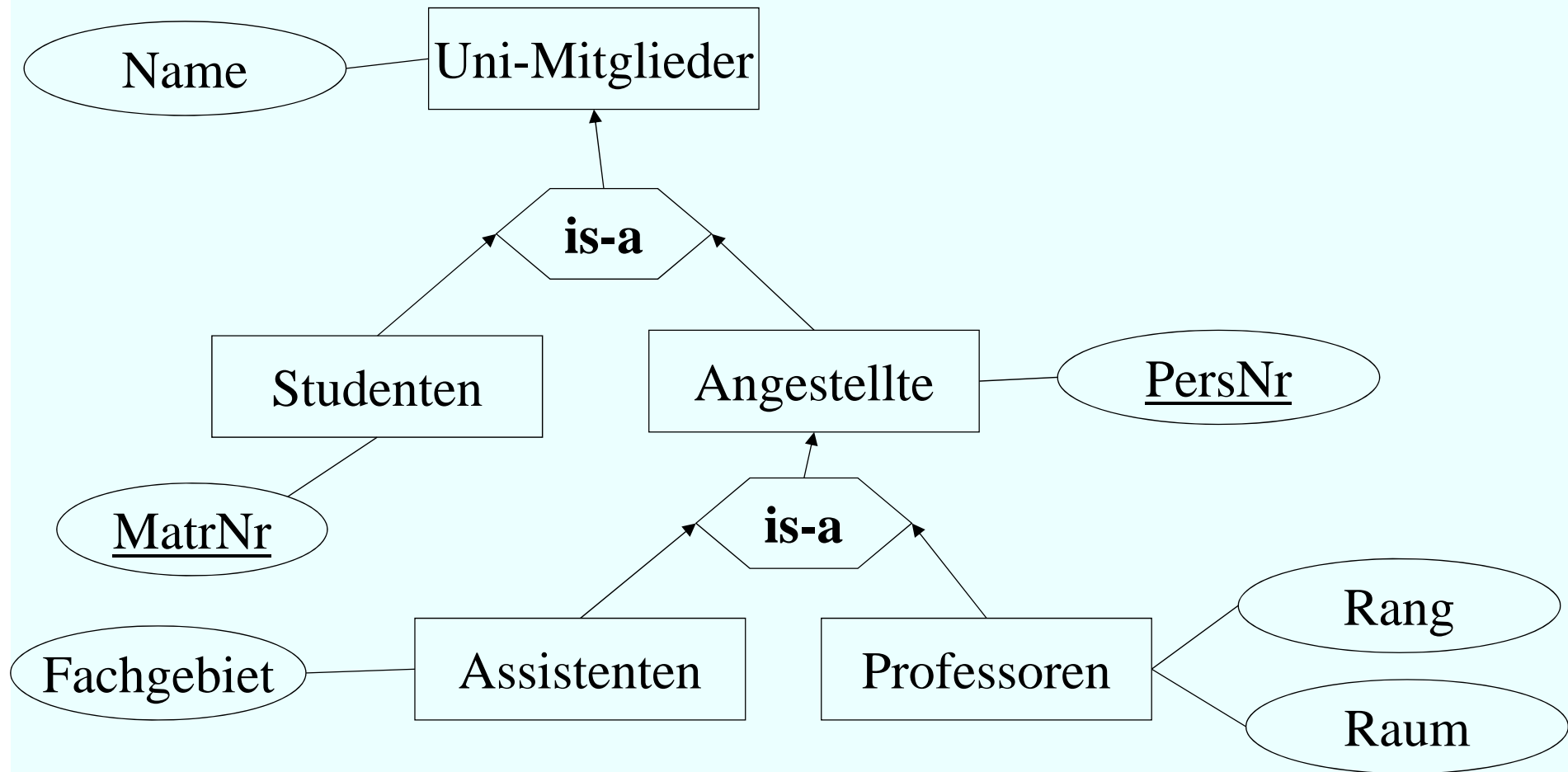


S ist eine Spezialisierung von G

Beispiel:



# Generalisierung Uni-Beispiel

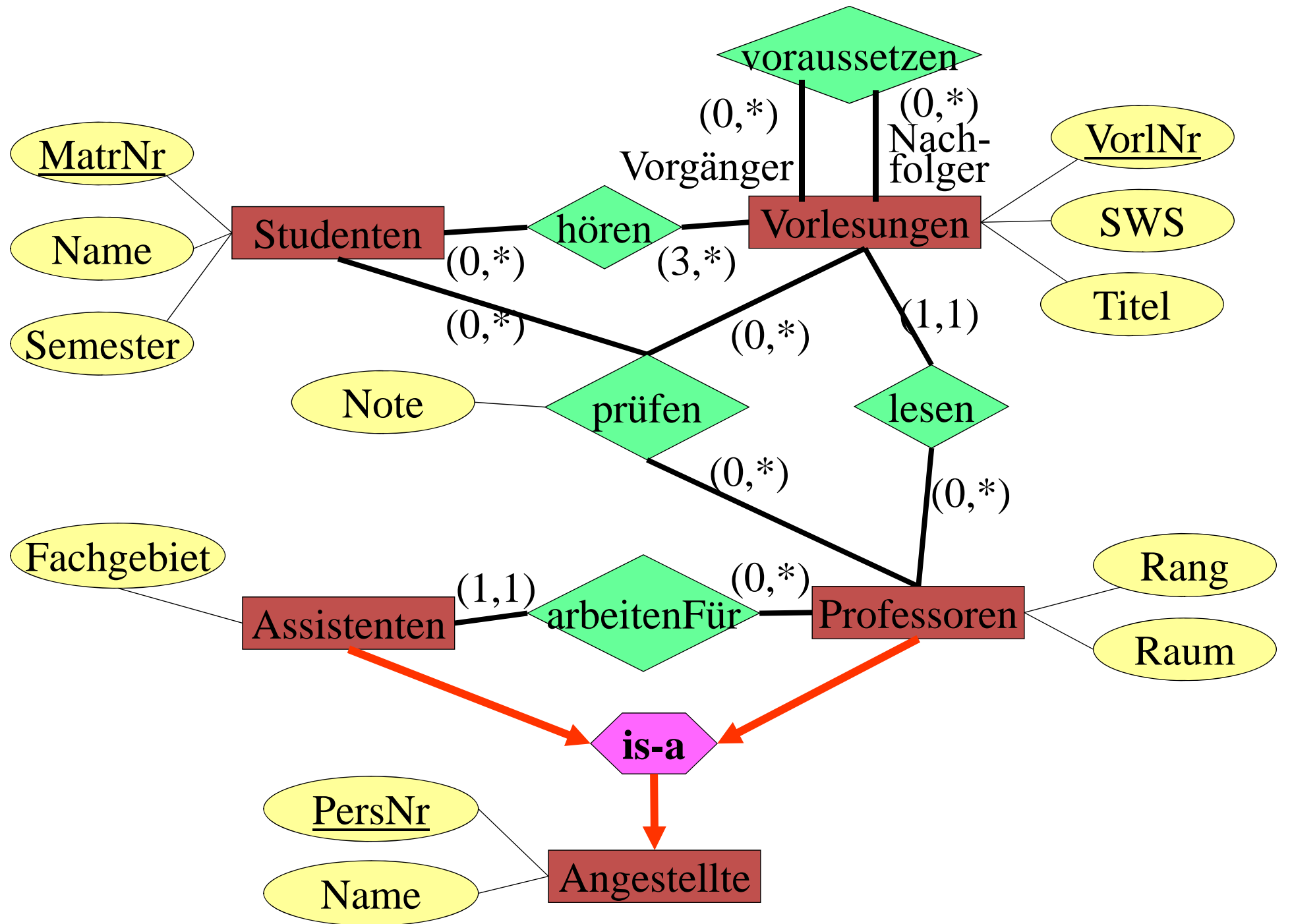


# Zusammenfassung

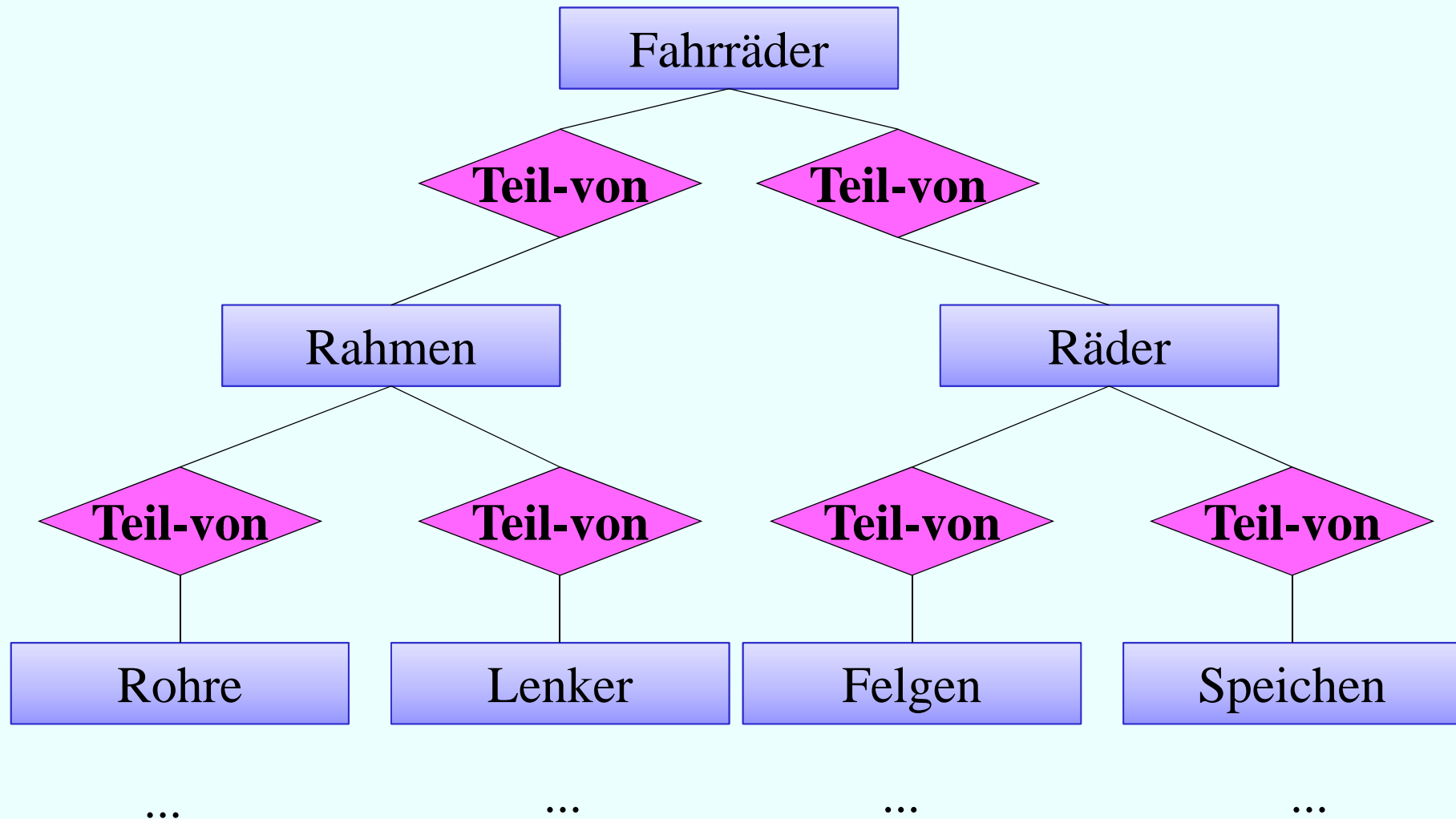
---

## Universitätsschema mit Generalisierung und (min, max)- Markierung

[→ Nächste Seite](#)

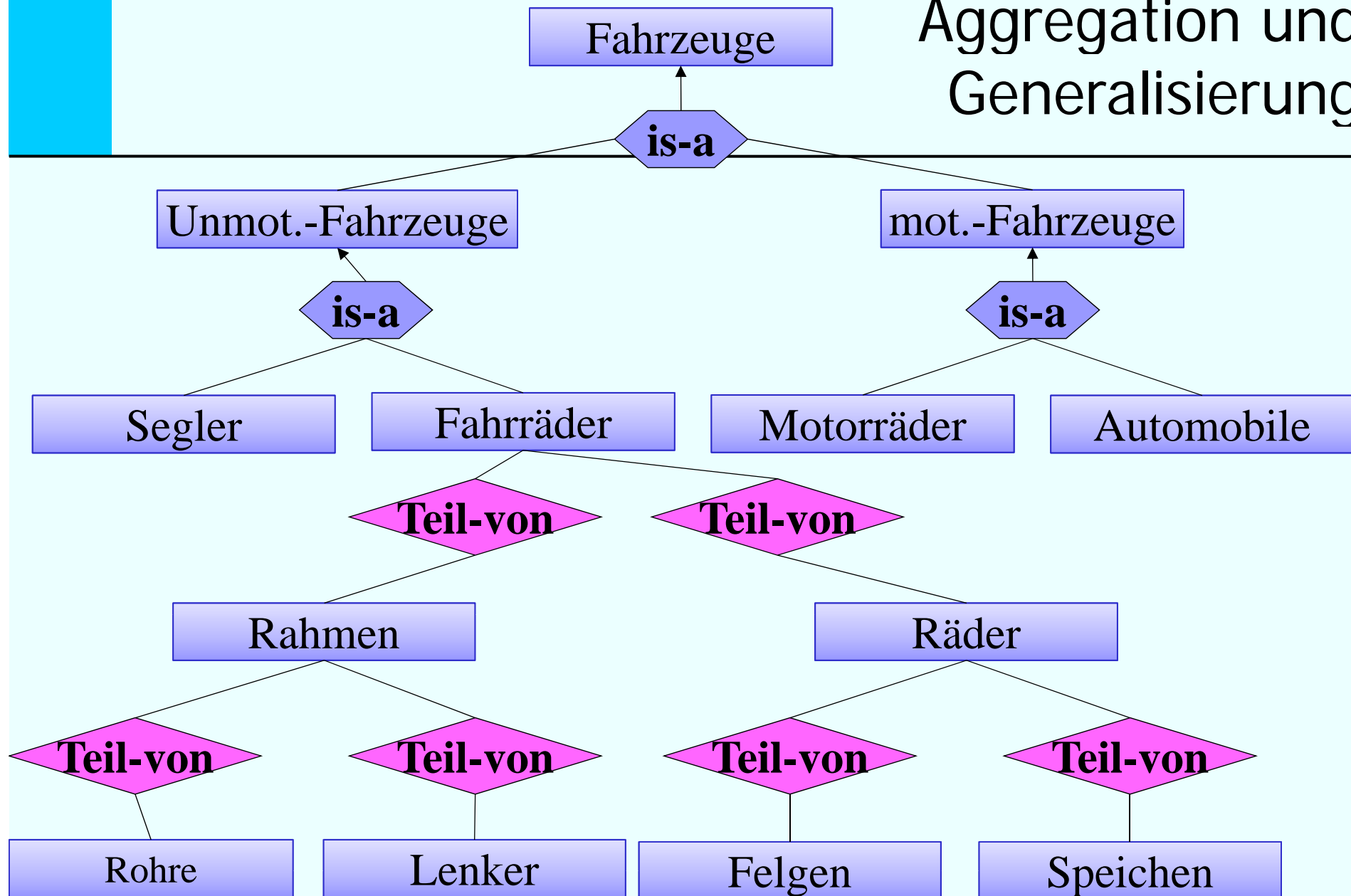


# Aggregation





# Aggregation und Generalisierung



# Entwurfskriterien

Regeln zur Klassifikation von Entities und Attributen:

Entities sollten deskriptive Informationen enthalten.

Mehrwertige Attribute sollten als Entities klassifiziert werden.

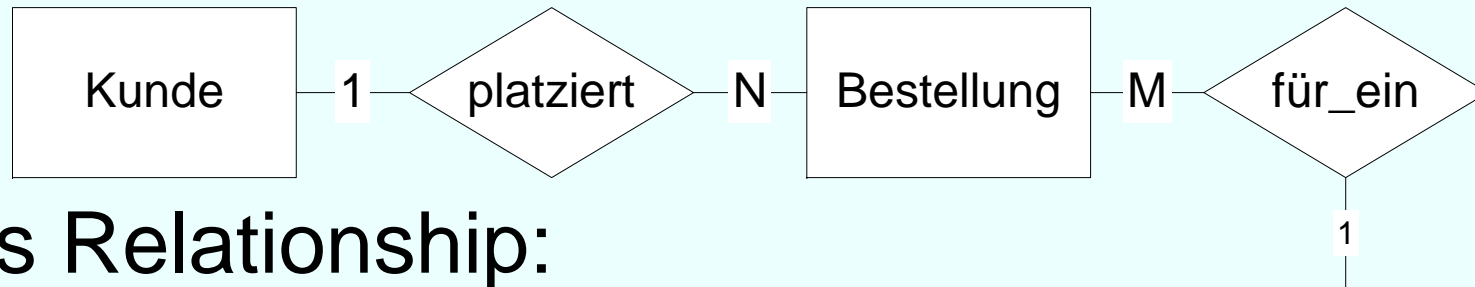
Attribute sollten der Entity zugeordnet werden, die sie am direktesten beschreibt.

Redundante Relationships sollten vermieden werden.

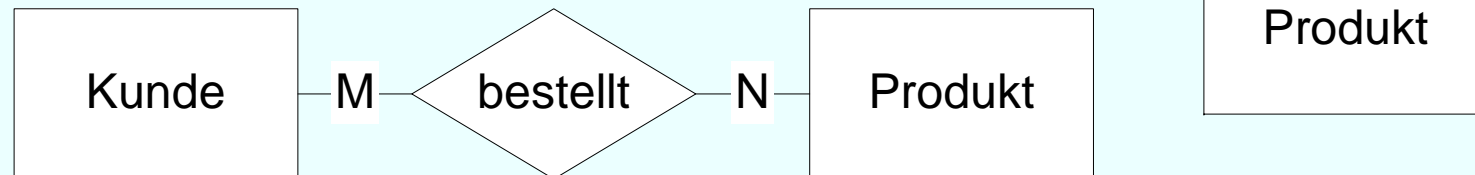
Wie eine Informationseinheit repräsentiert wird, ist *anwendungsabhängig*.

# Beispiel: Bestellung

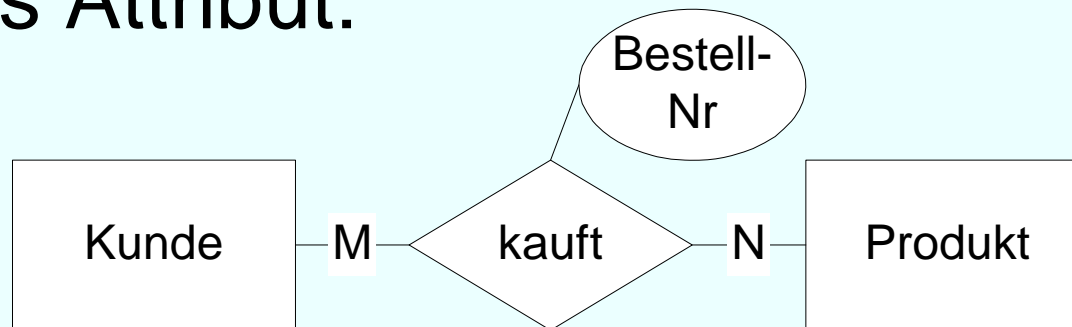
Als Entity:



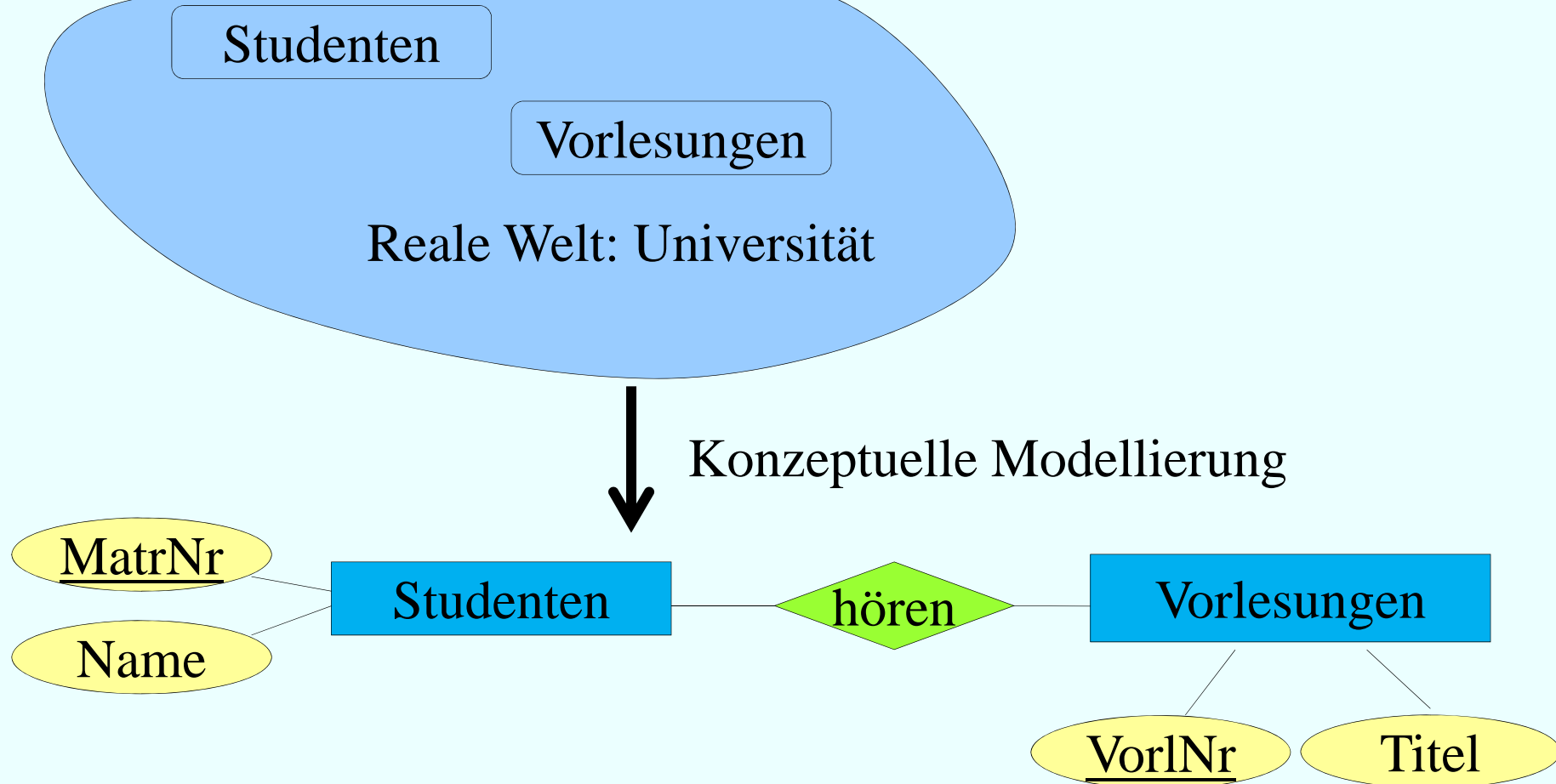
Als Relationship:



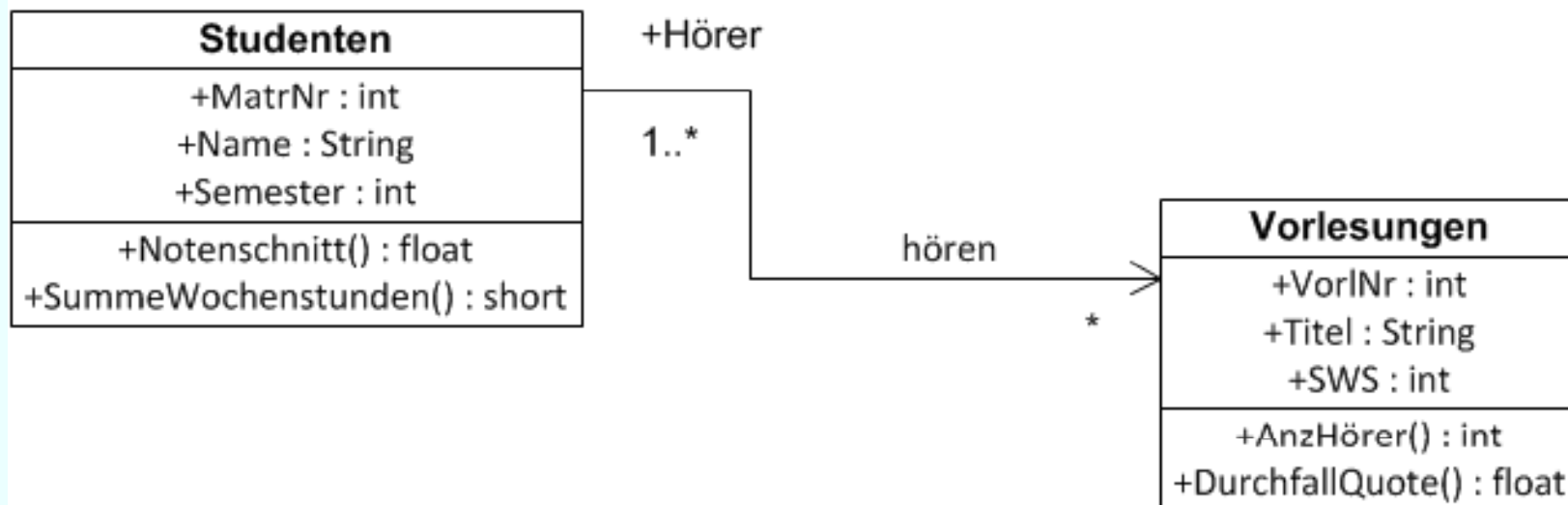
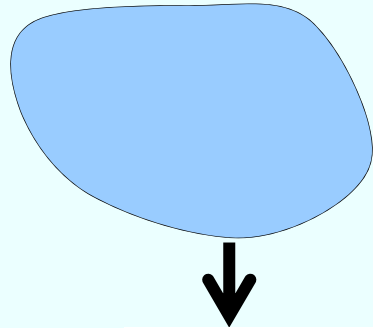
Als Attribut:



# Modellierung einer kleinen Beispielanwendung: E/R



# Modellierung einer kleinen Beispielanwendung: UML



# Datenmodellierung mit UML

UML: Unified Modelling Language

De-facto Standard für den objekt-orientierten Software-Entwurf

Zentrales Konstrukt: Klasse (class),  
modelliert gleichartige Objekte hinsichtlich

- Struktur (~Attribute)
- Verhalten (~Operationen/Methoden)

Assoziationen zwischen Klassen entsprechen Beziehungstypen

Generalisierungshierarchien

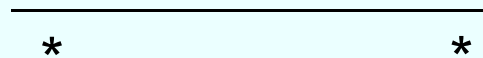
Aggregation

Cheat sheet Class Diagram:

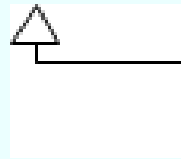
<http://www.code-meets-design.de/wp-content/uploads/2013/07/uml-classdiagram-cheat-sheet.pdf>

# UML Notation

Assoziation:



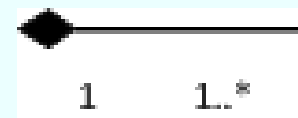
Generalisierung:



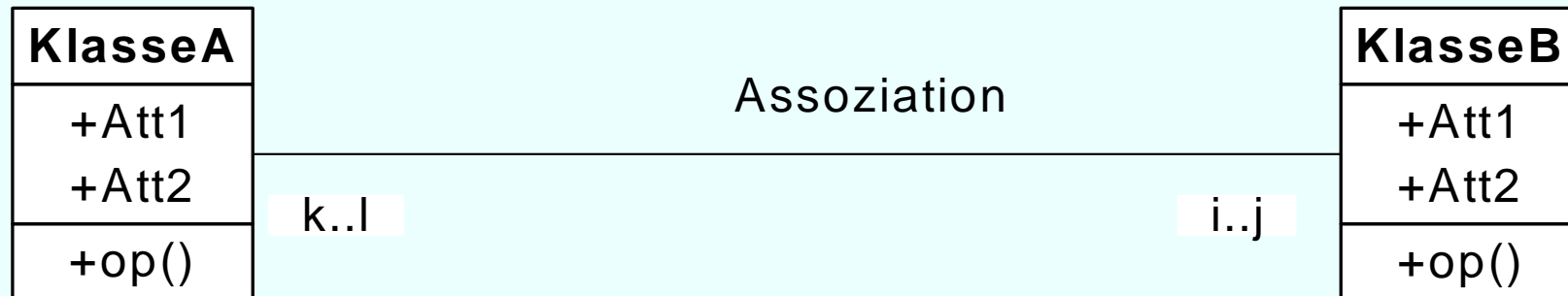
Aggregation:  
(Teil-von)



Komposition:  
(Spezialfall von Aggregation)



# Multiplizität



Jedes Element von KlasseA steht mit mindestens i Elementen der KlasseB in Beziehung

... und mit maximal j vielen KlasseB-Elementen

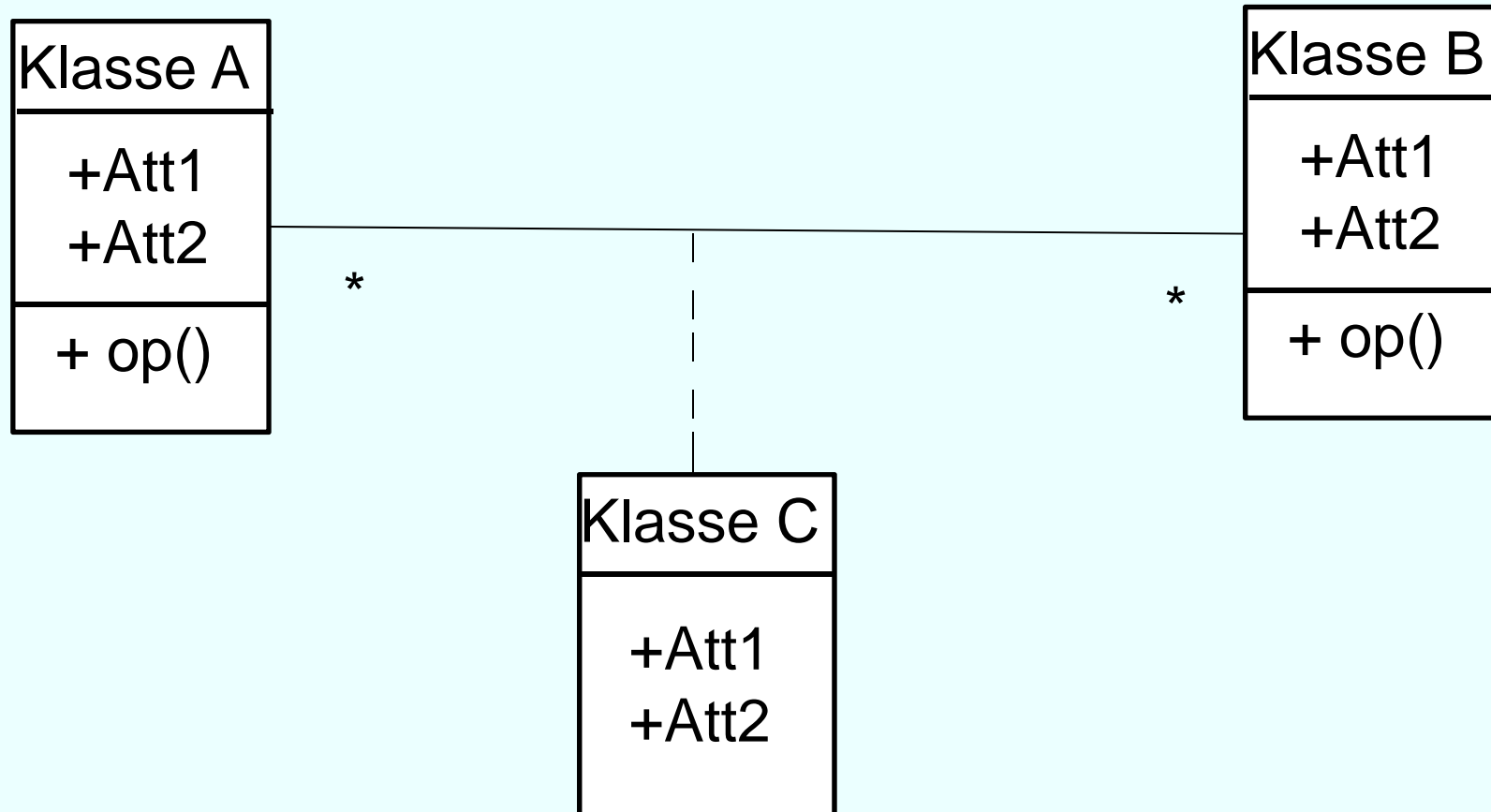
Analoges gilt für das Intervall k..l

Multiplizitätsangabe ist analog zur einfachen Funktionalitätsangabe im ER-Modell

**Nicht** zur (min,max)-Angabe: **Vorsicht!**

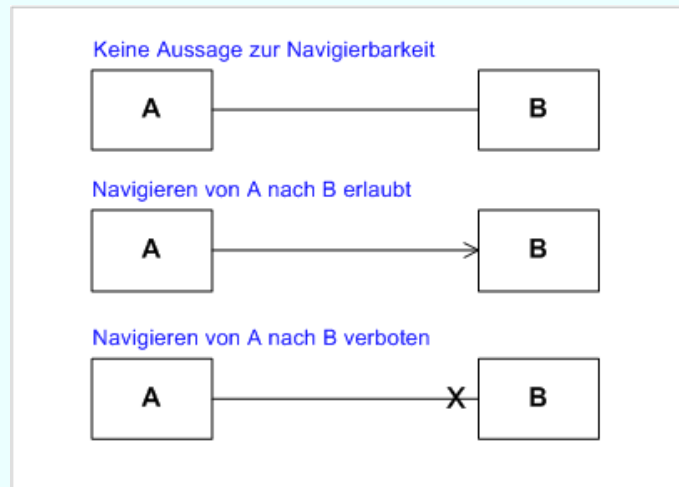
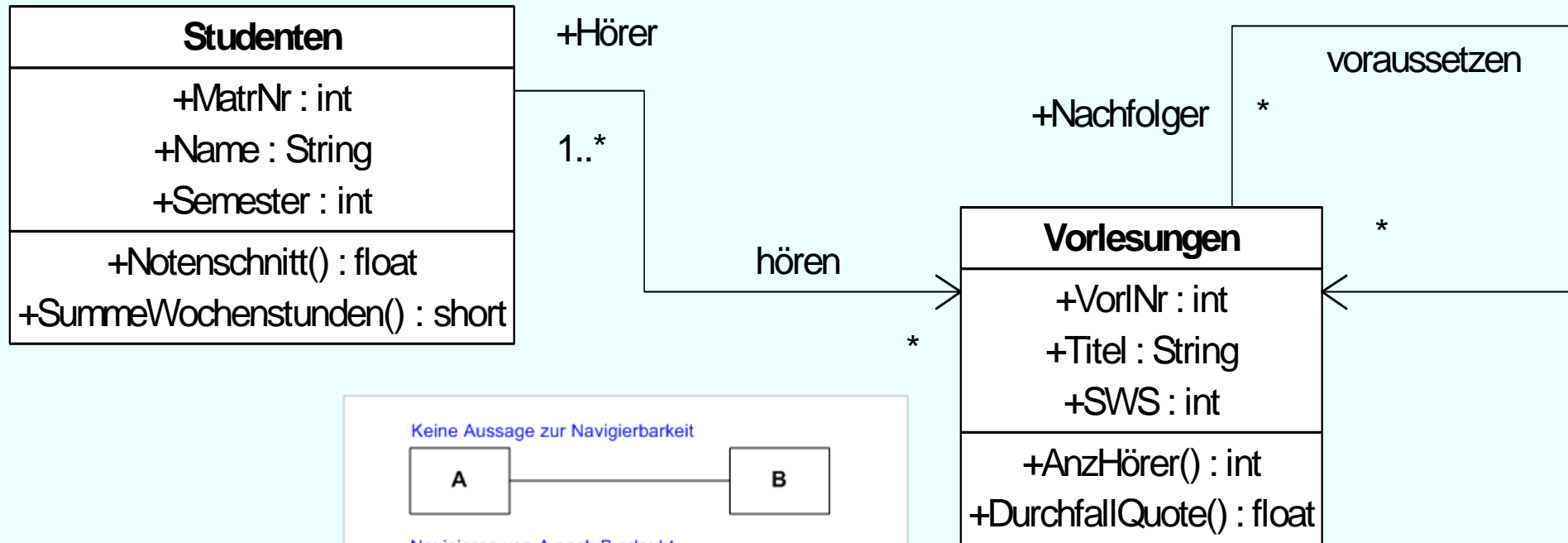


# Assoziationsklasse



... für Attribute der Assoziation

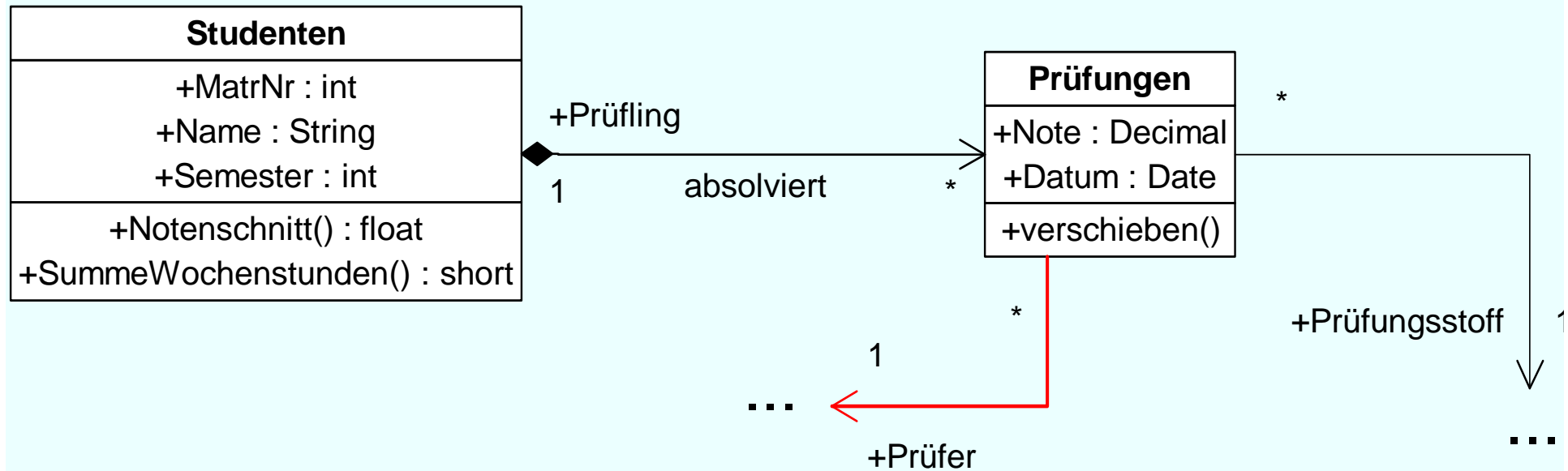
# Klassen und Assoziationen



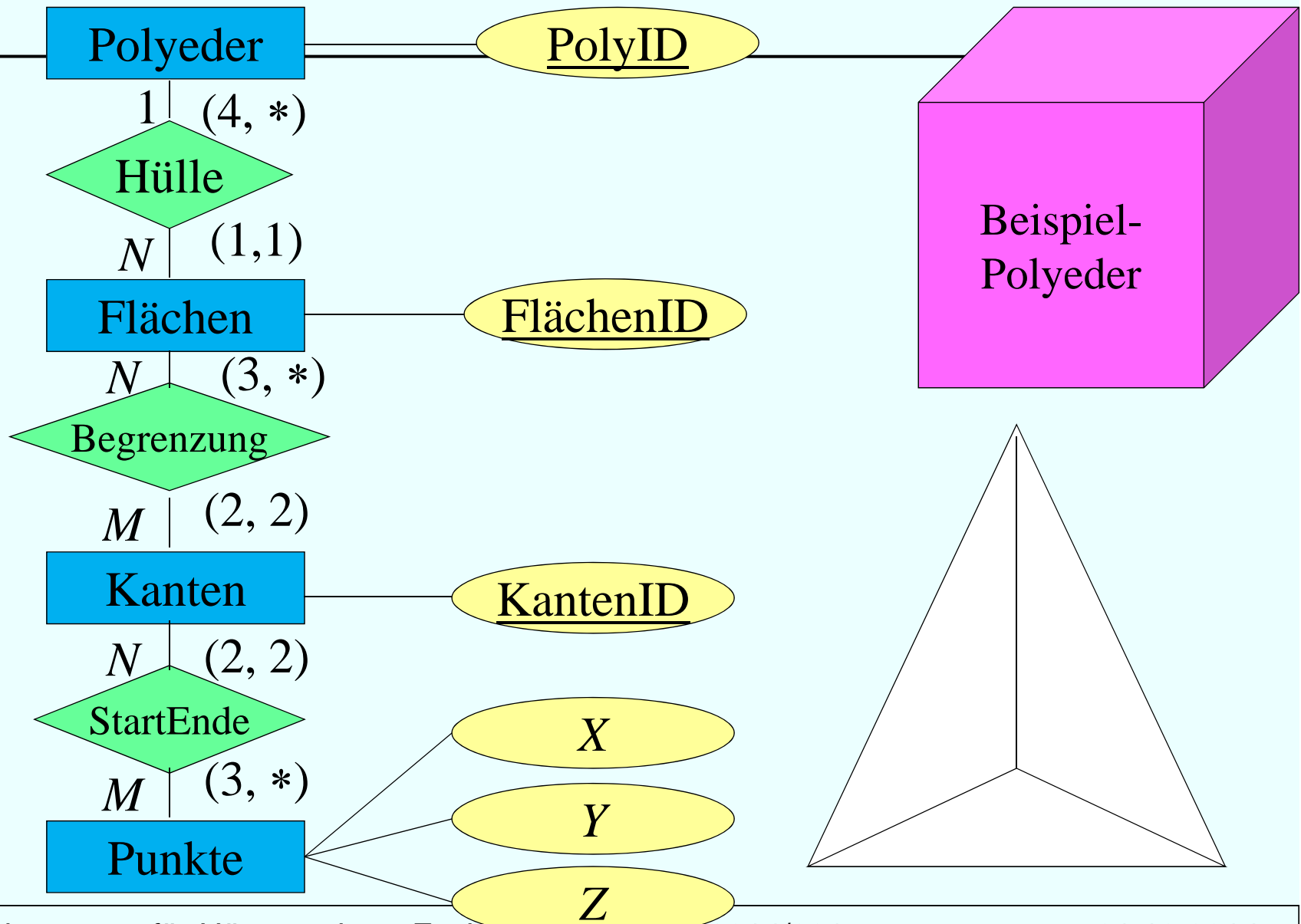
Legende:

+: public

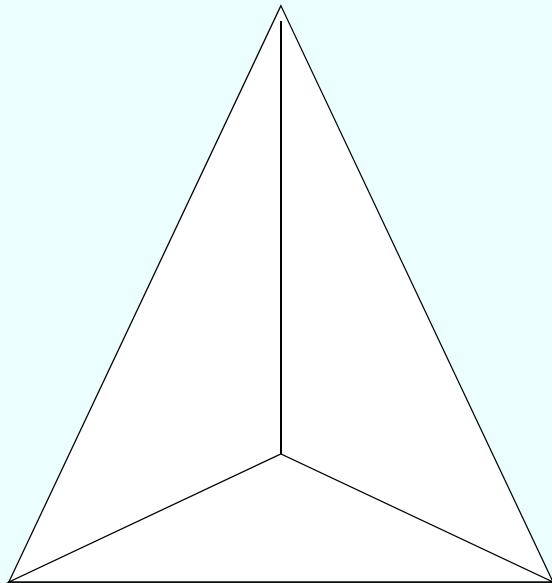
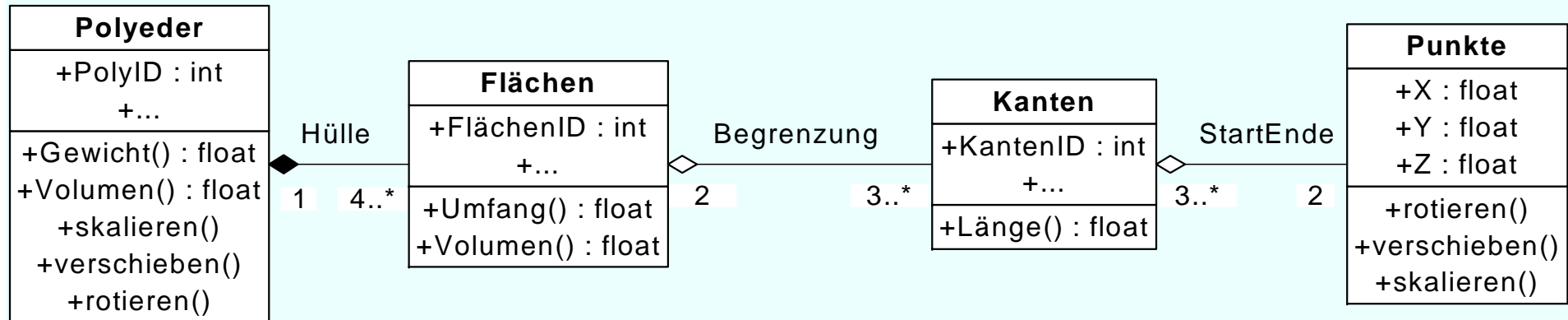
# Komposition

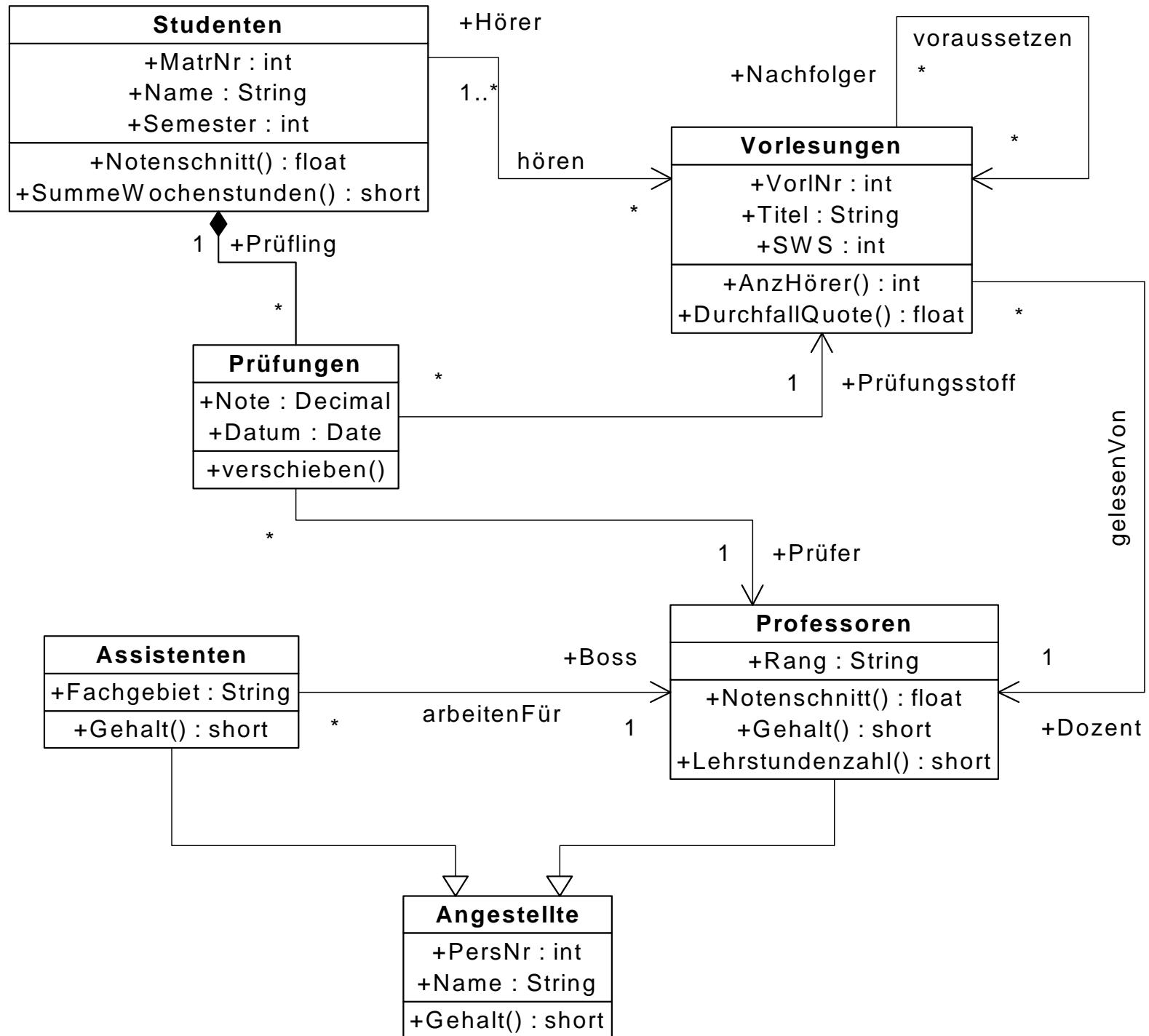


# Begrenzungsflächendarstellung

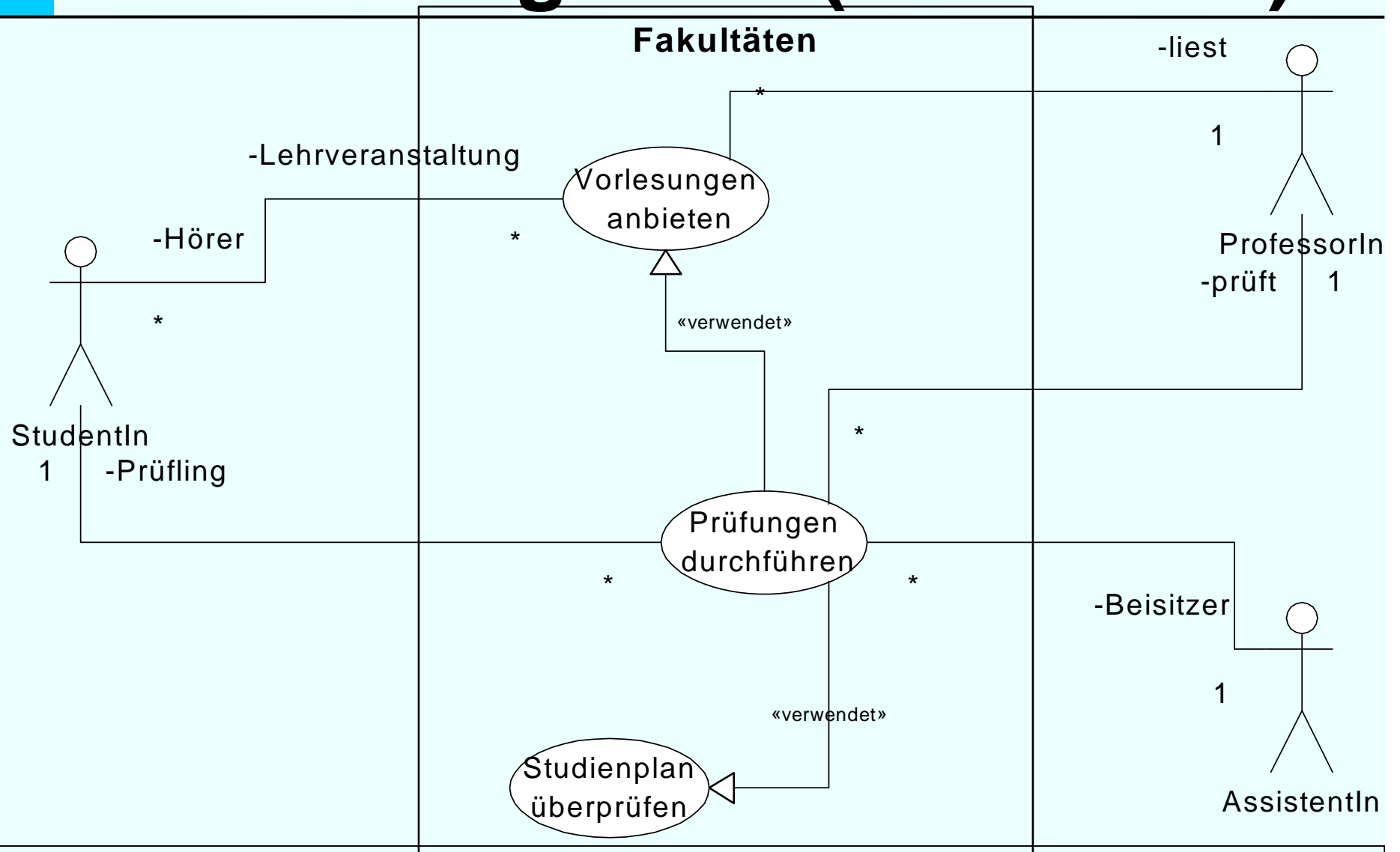


# Begrenzungsflächenmodellierung von Polyedern in UML

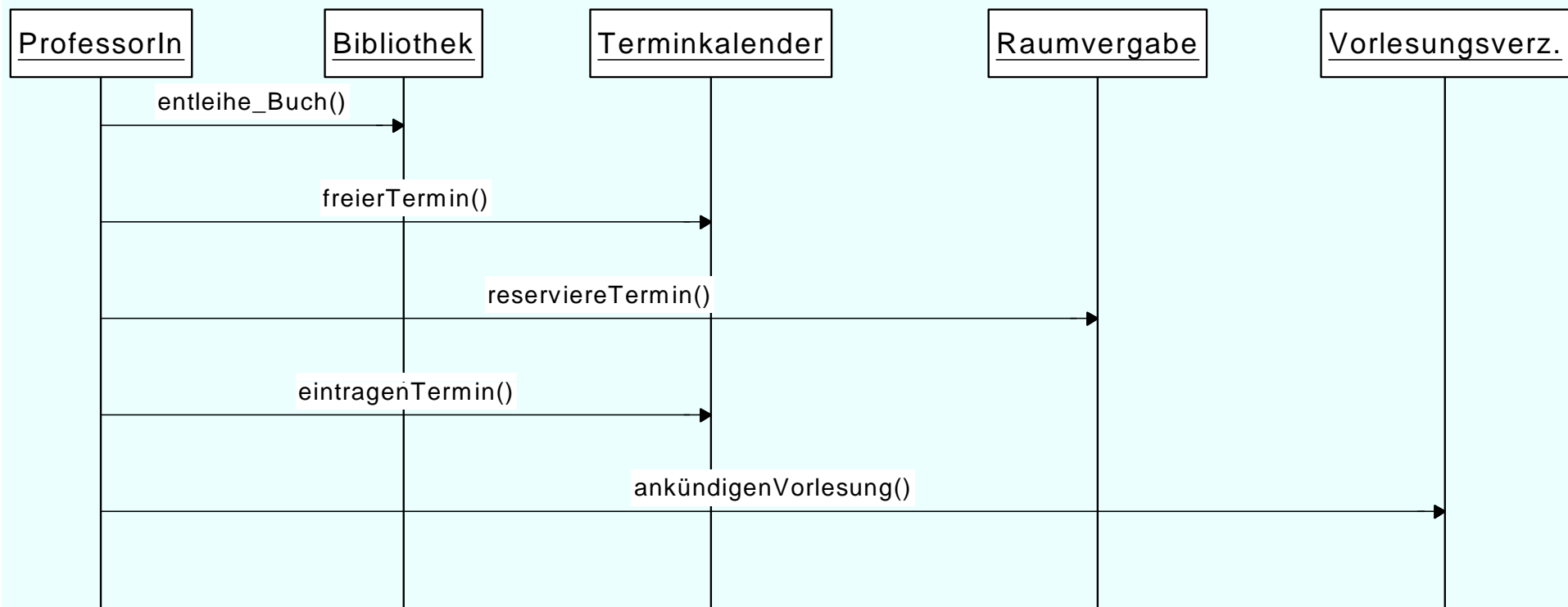




# Anwendungsfälle (use cases)

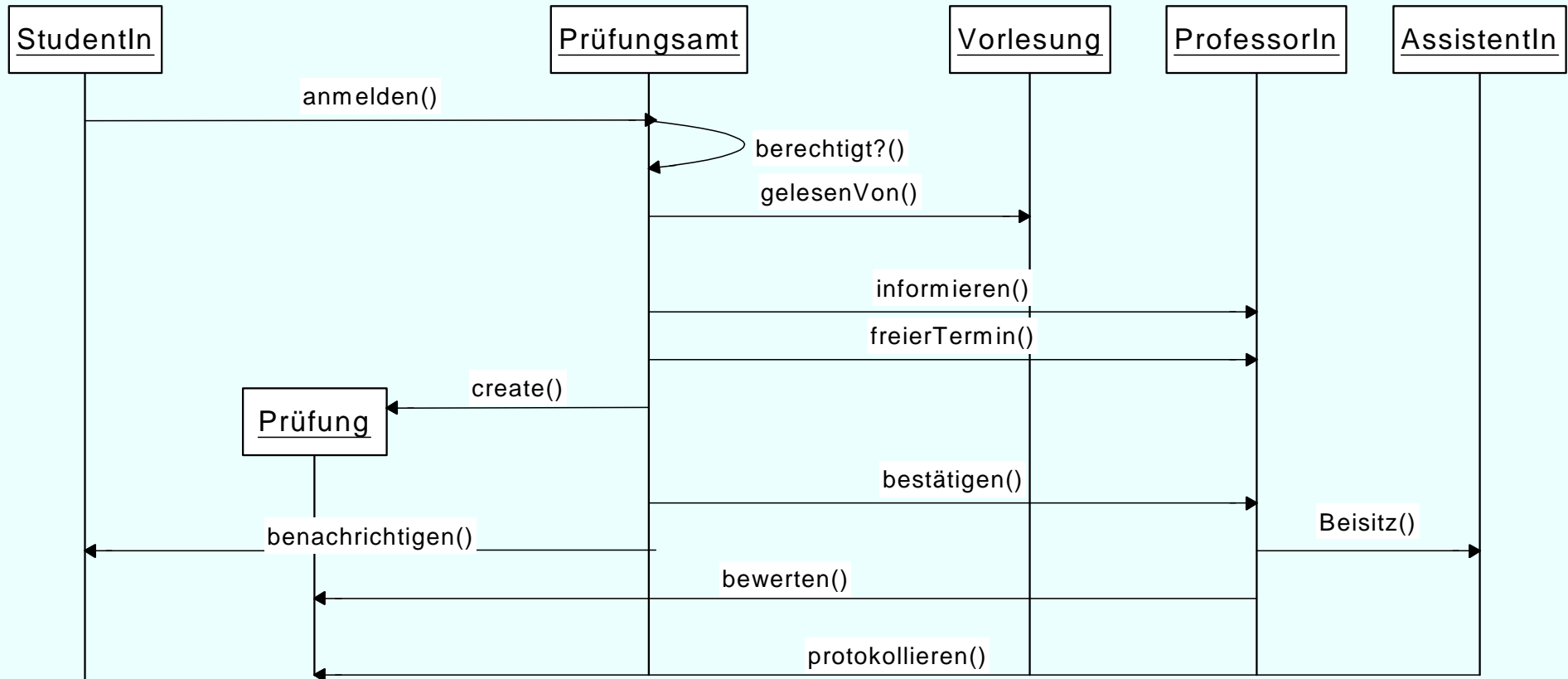


# Interaktions-Diagramm: Modellierung komplexer Anwendungen





# Interaktions-Diagramm: *Prüfungsdurchführung*



UML Notationsübersicht:

<http://www.oose.de/nuetzliches/fachliches/uml/uml-notationsubersicht/>