

Datenmodellierung

DBS kann vieles, aber nicht alles!

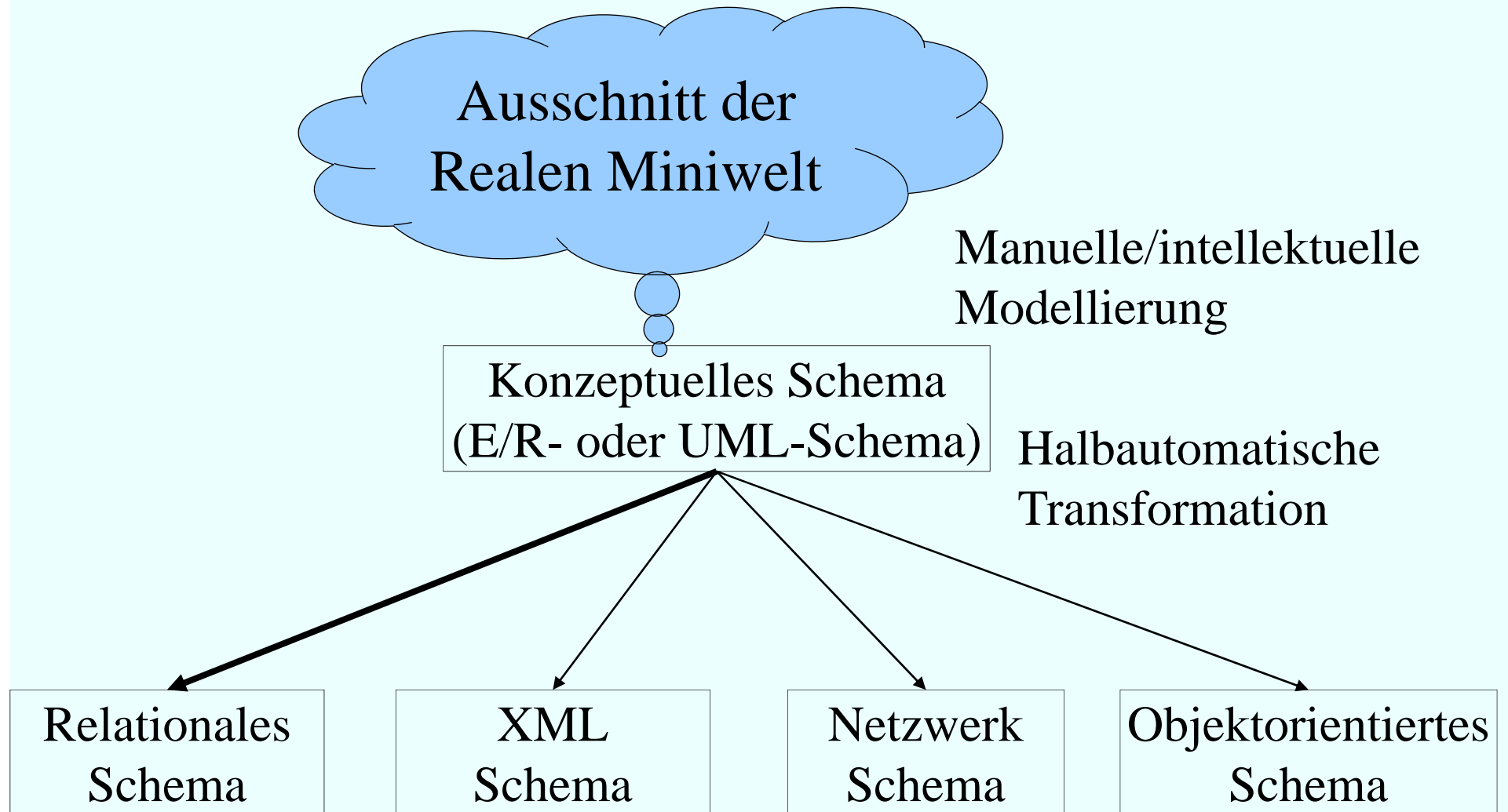
Benutzer muss spezifizieren

- Anforderungen einer Anwendung
- Art von zu speichernden Daten

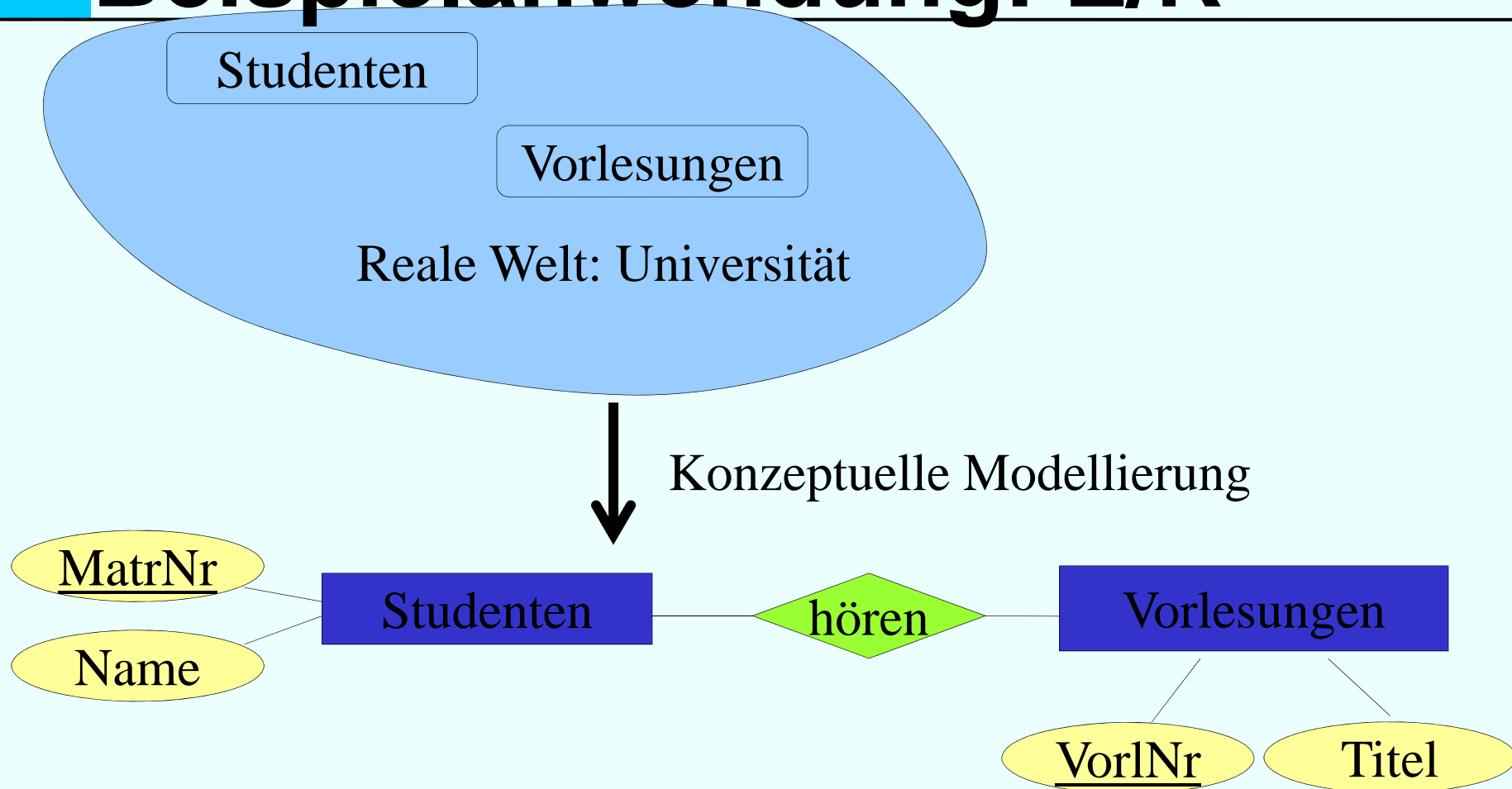
Zwei wichtige Konzepte beim Entwurf:

- Datenmodell: Konstrukte zum Beschreiben der Daten
- Schema: konkrete Beschreibung einer bestimmten Datensammlung
(unter Verwendung eines Datenmodells)

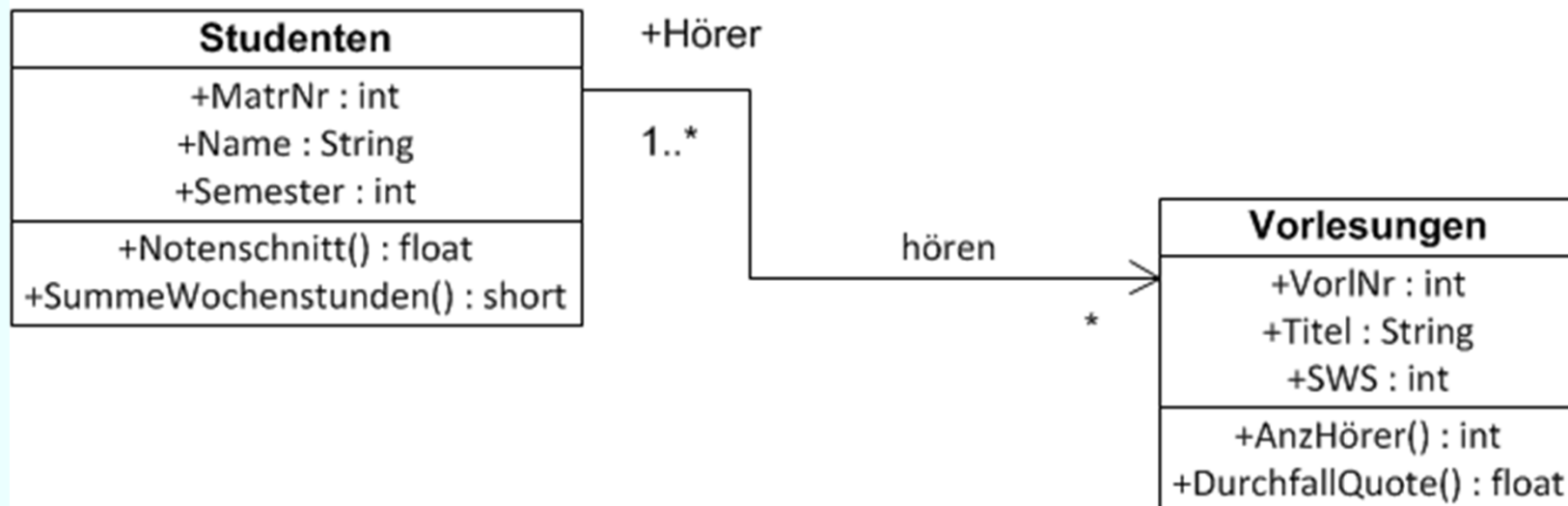
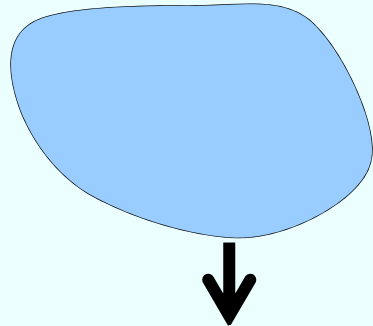
Datenmodellierung



Modellierung einer kleinen Beispielanwendung: E/R



Modellierung einer kleinen Beispielanwendung: UML



Das relationale Datenmodell

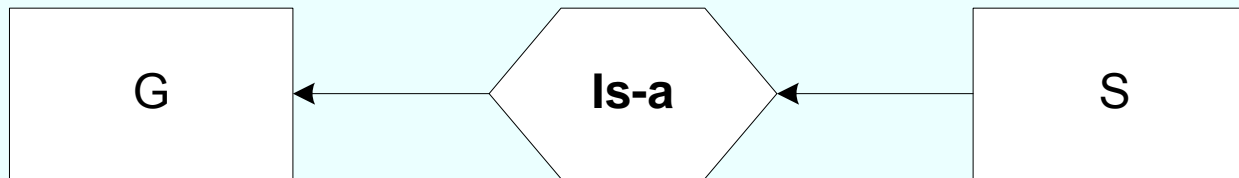
Studenten	
MatrNr	Name
26120	Fichte
25403	Jonas
...	...

hören	
MatrNr	VorlNr
25403	5022
26120	5001
...	...

Vorlesungen	
VorlNr	Titel
5001	Grundzüge
5022	Glaube und Wissen
...	...

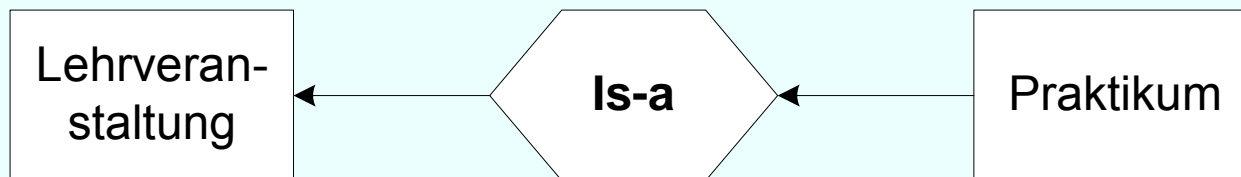
```
Select Name  
From Studenten, hören, Vorlesungen  
Where Studenten.MatrNr = hören.MatrNr and  
        hören.VorlNr = Vorlesungen.VorlNr and  
        Vorlesungen.Titel = `Grundzüge`;
```

Generalisierung / Spezialisierung

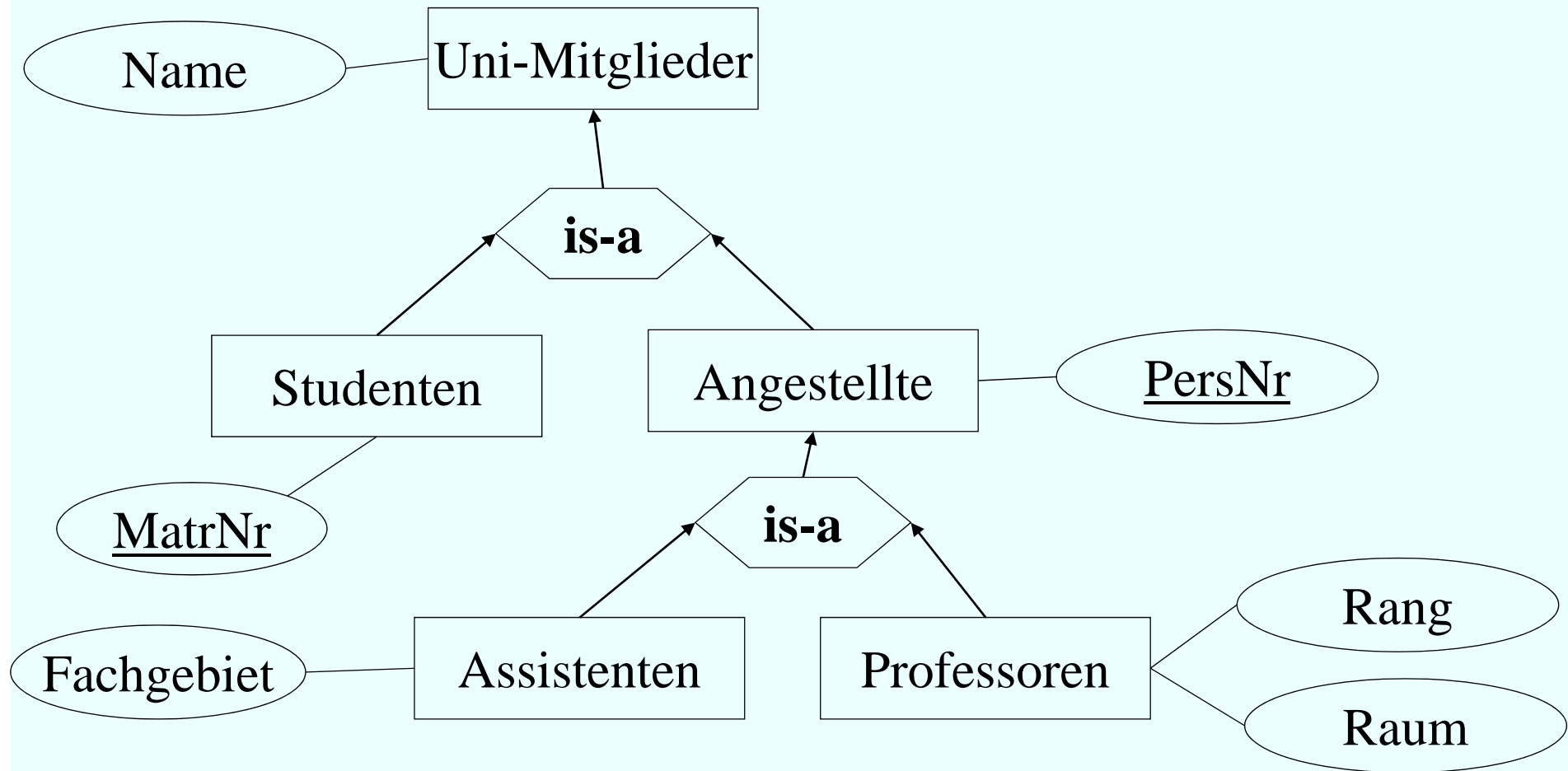


S ist eine Spezialisierung von G

- Beispiel:



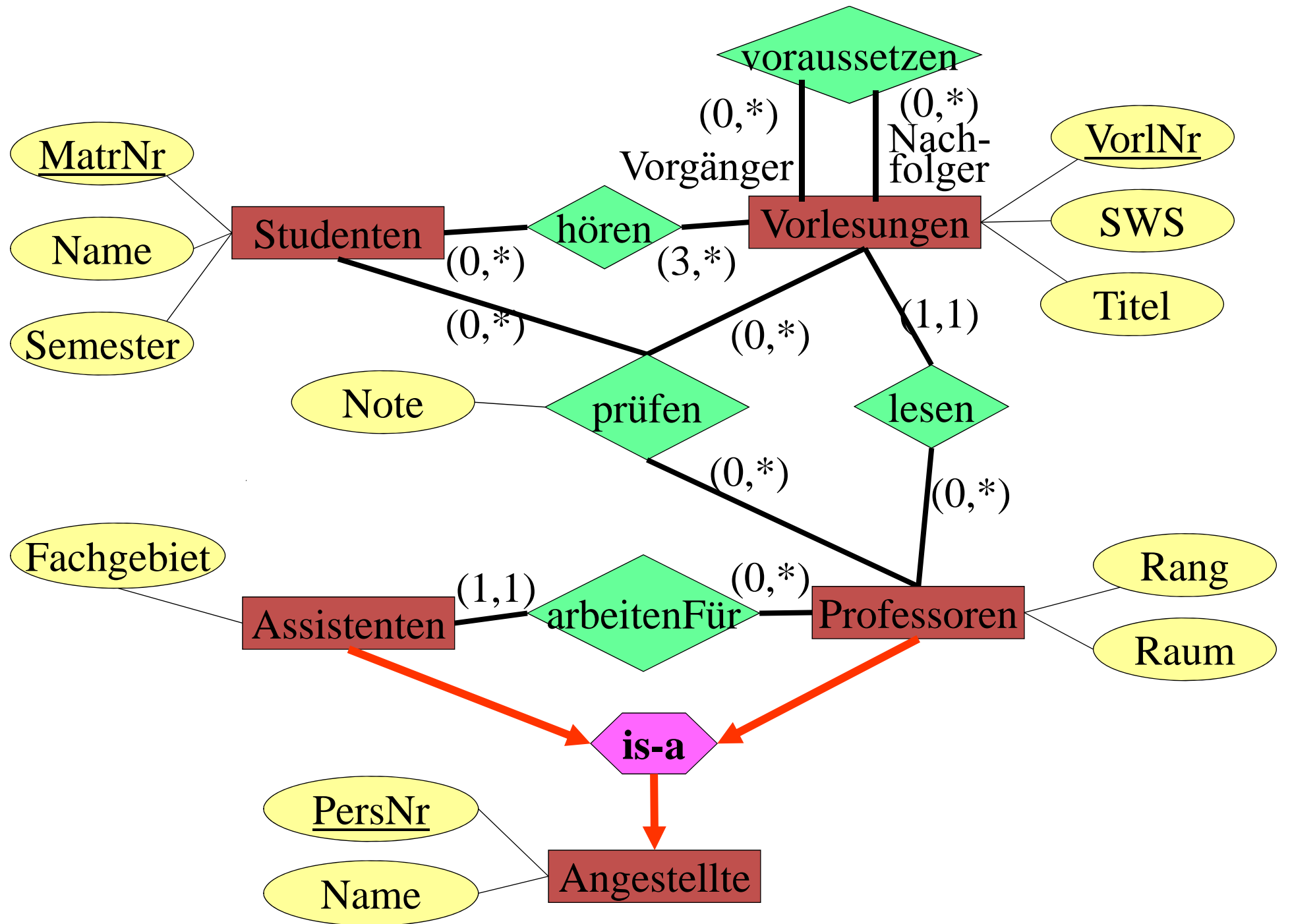
Generalisierung



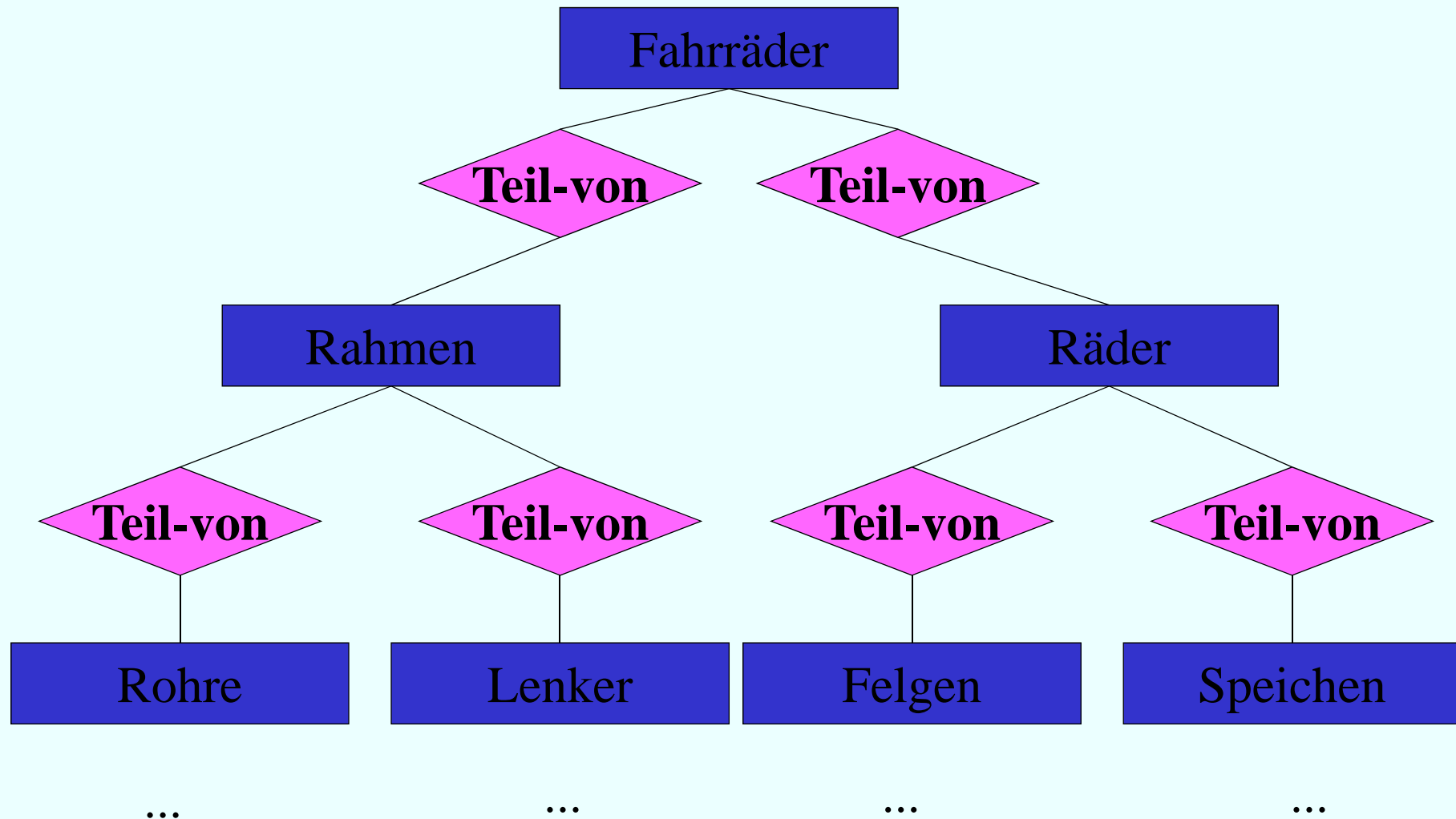
Zusammenfassung

Universitätsschema mit Generalisierung und (min, max)- Markierung

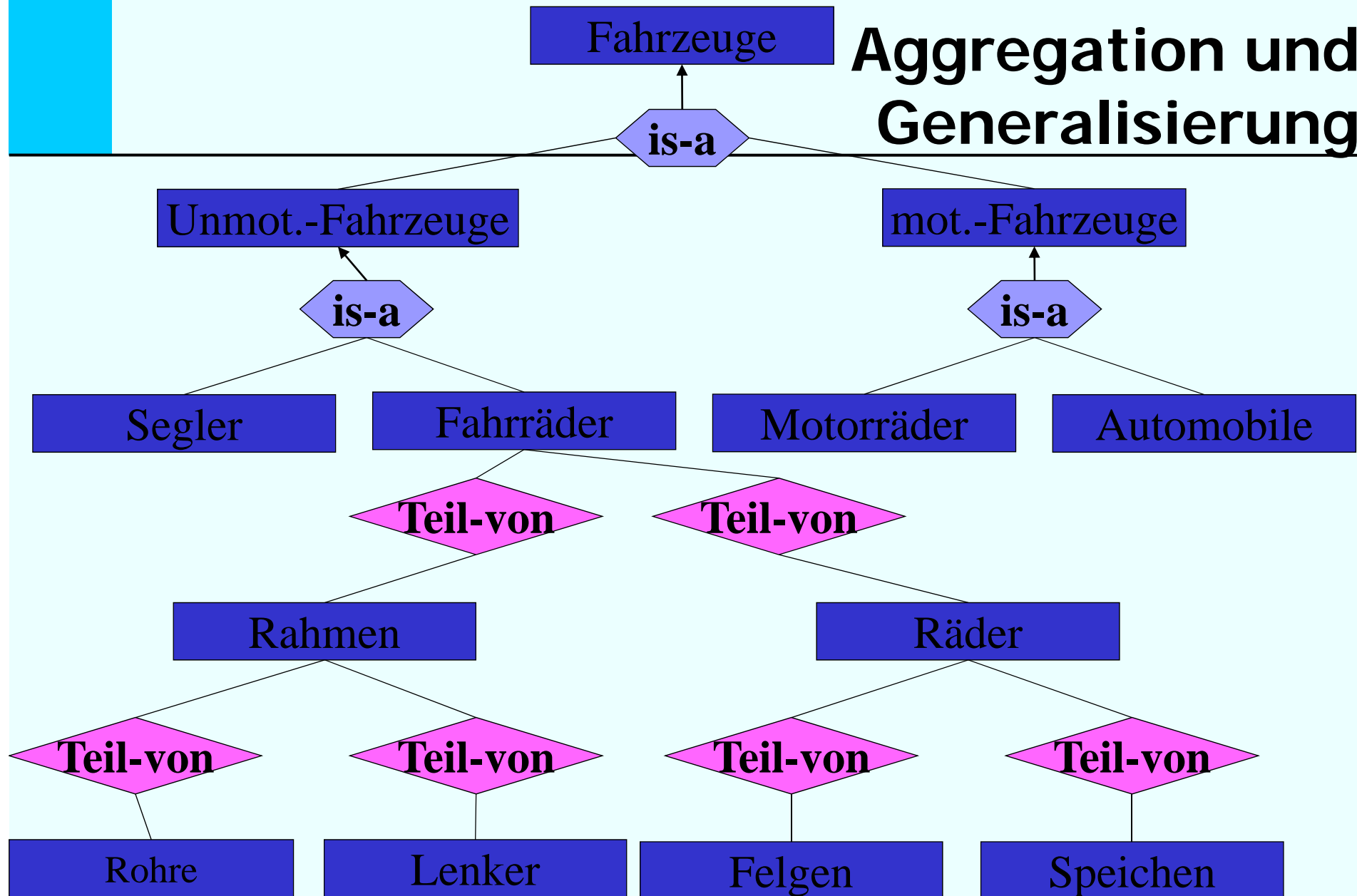
→ Nächste Seite



Aggregation



Aggregation und Generalisierung



ER-Modellierung

Regeln zur Klassifikation von Entities und Attributen:

Entities sollten deskriptive Informationen enthalten.

Mehrwertige Attribute sollten als Entities klassifiziert werden.

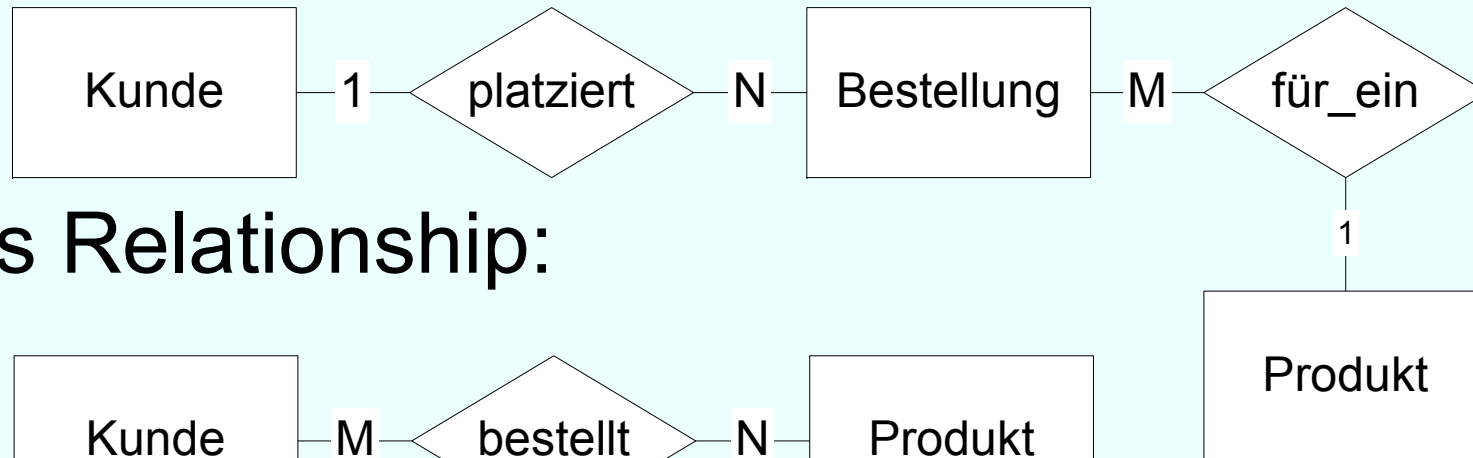
Attribute sollten der Entity zugeordnet werden, die sie am direktesten beschreibt.

Redundante Relationships sollten vermieden werden.

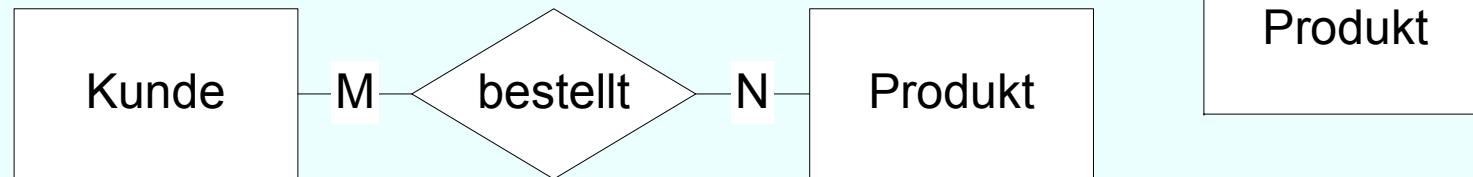
Wie eine Informationseinheit repräsentiert wird, ist *anwenderabhängig*.

Beispiel: Bestellung

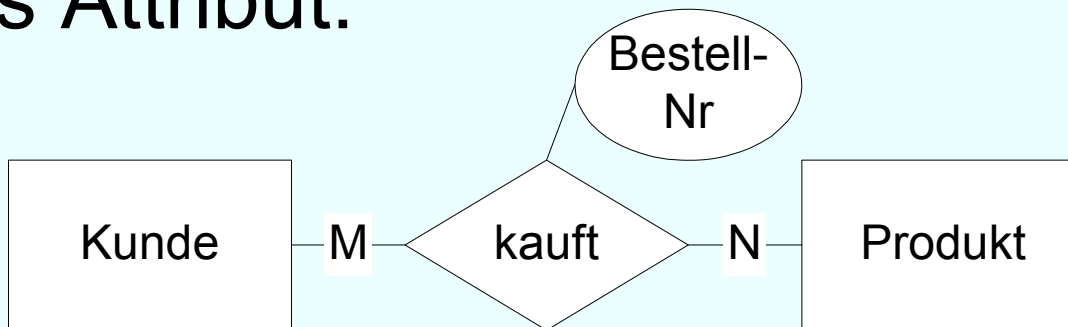
Als Entity:



Als Relationship:



Als Attribut:



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)

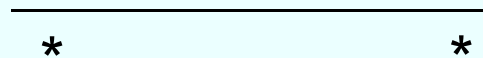
Assoziationen zwischen Klassen entsprechen Beziehungstypen

Generalisierungshierarchien

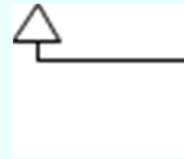
Aggregation

UML Notation

Assoziation:



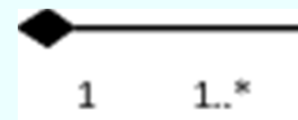
Generalisierung:



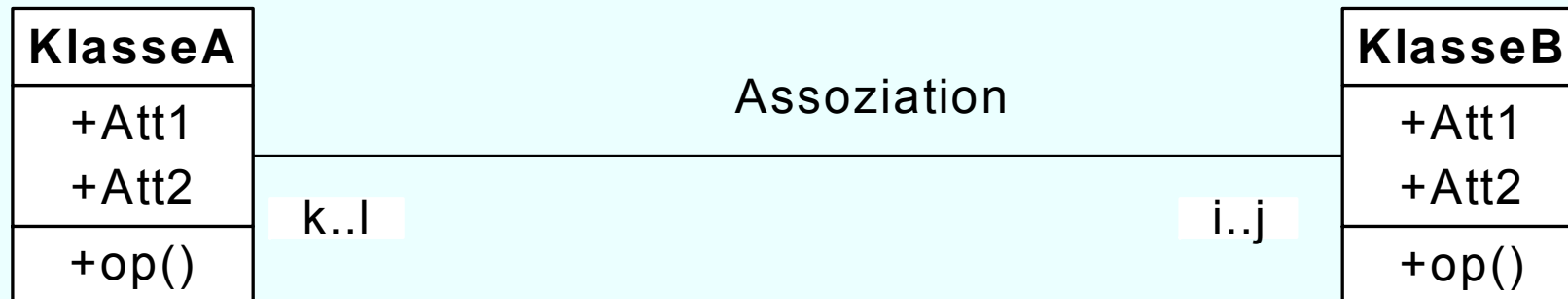
Aggregation:
(Teil-von)



Komposition:
(Spezialfall von Aggregation)



Multiplizität



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

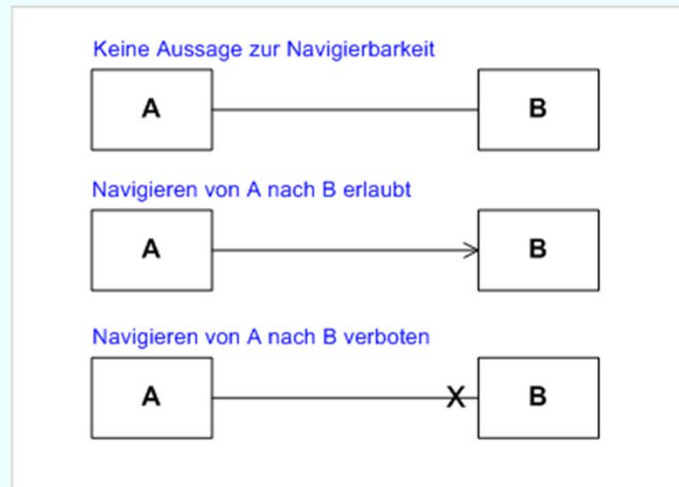
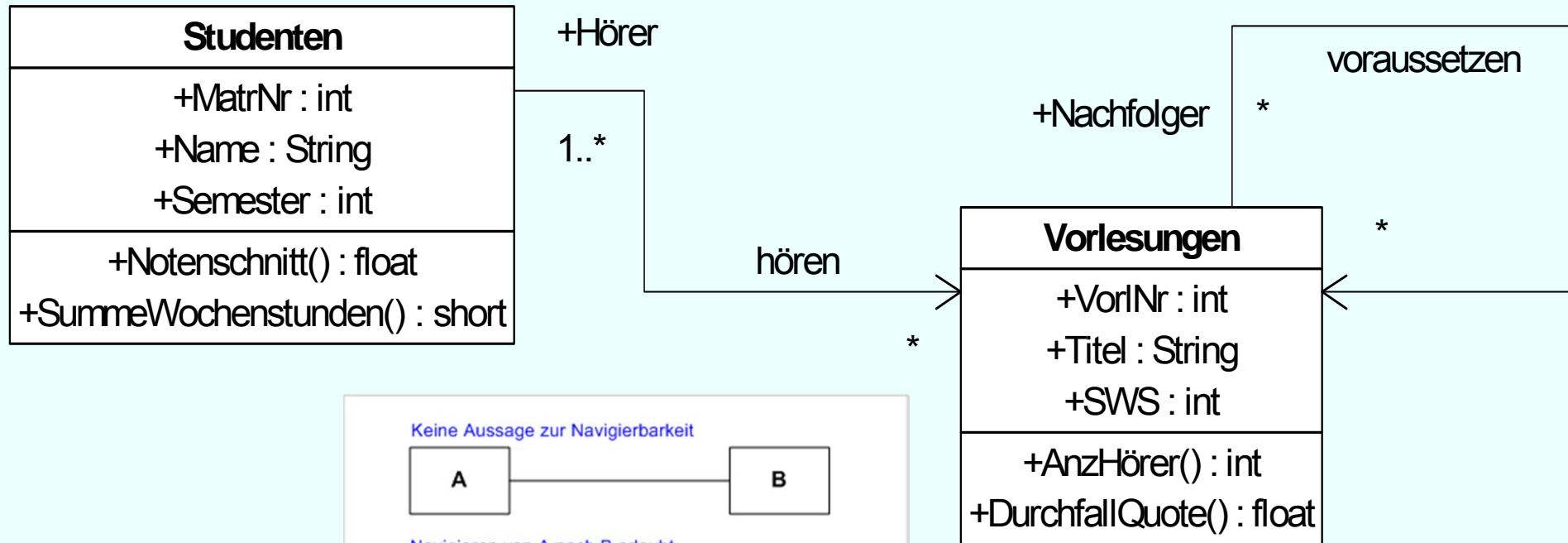
... und mit maximal j vielen KlasseB-Elementen

Analoges gilt für das Intervall k..l

Multiplizitätsangabe ist analog zur Funktionalitätsangabe im ER-Modell

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

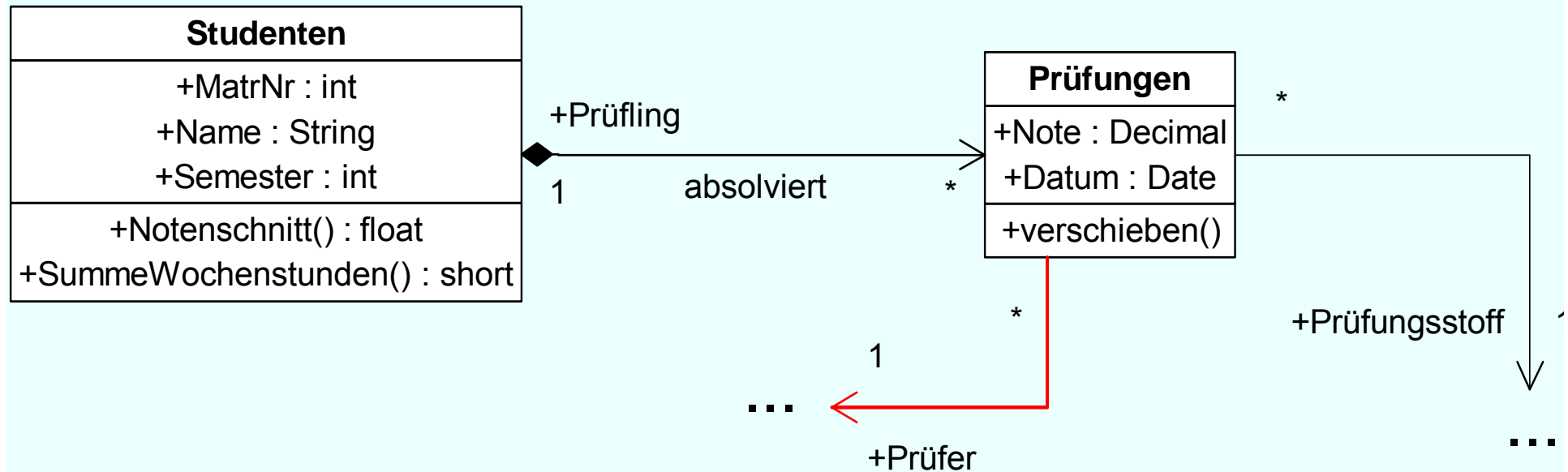
Klassen und Assoziationen



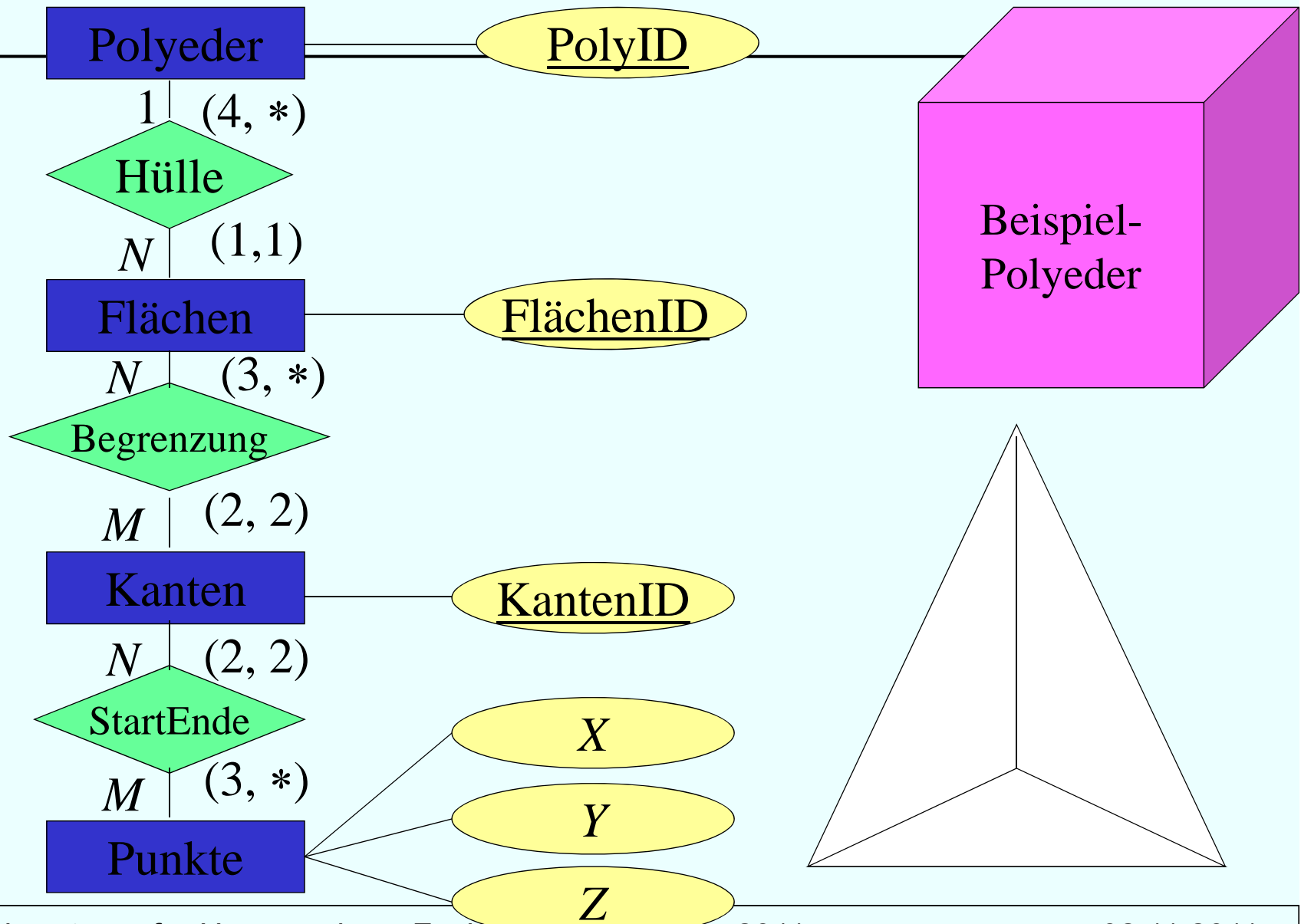
Legende:

+: public

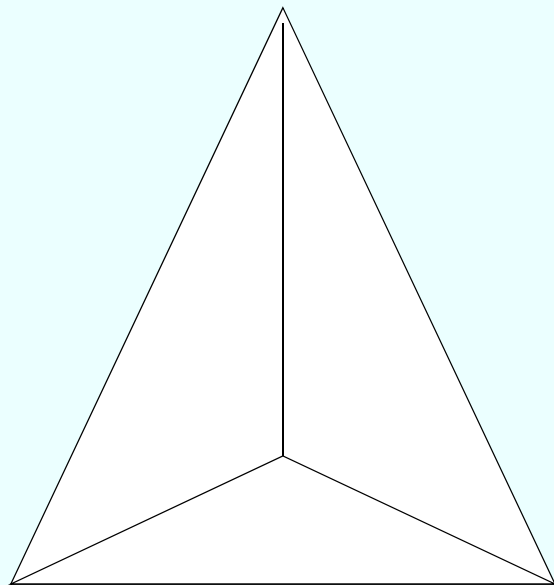
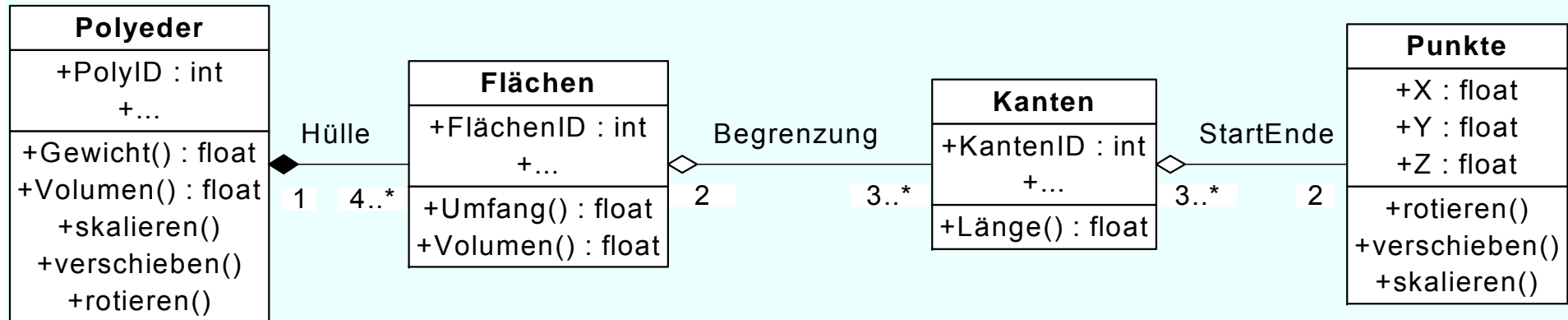
Komposition

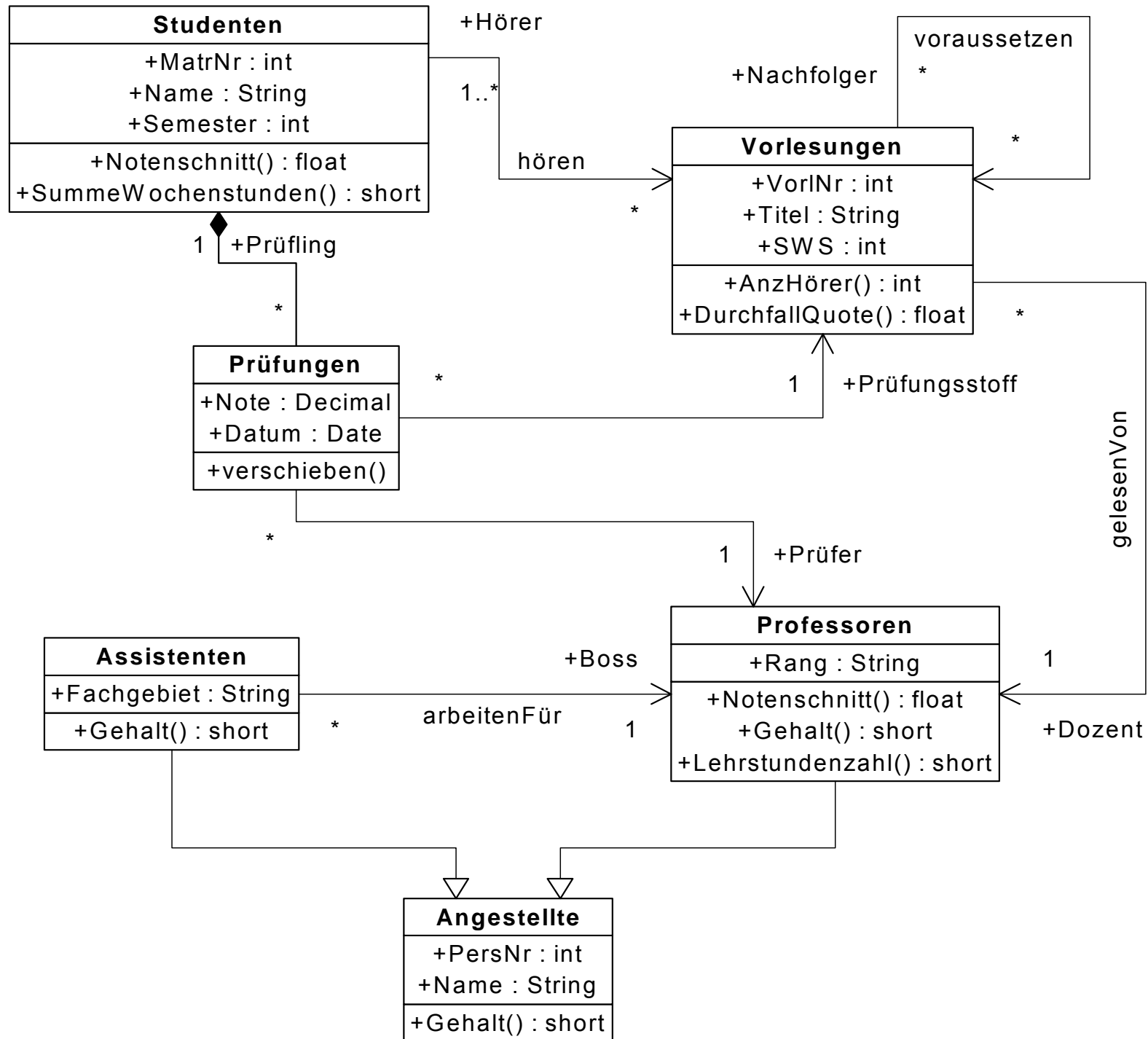


Begrenzungsflächendarstellung

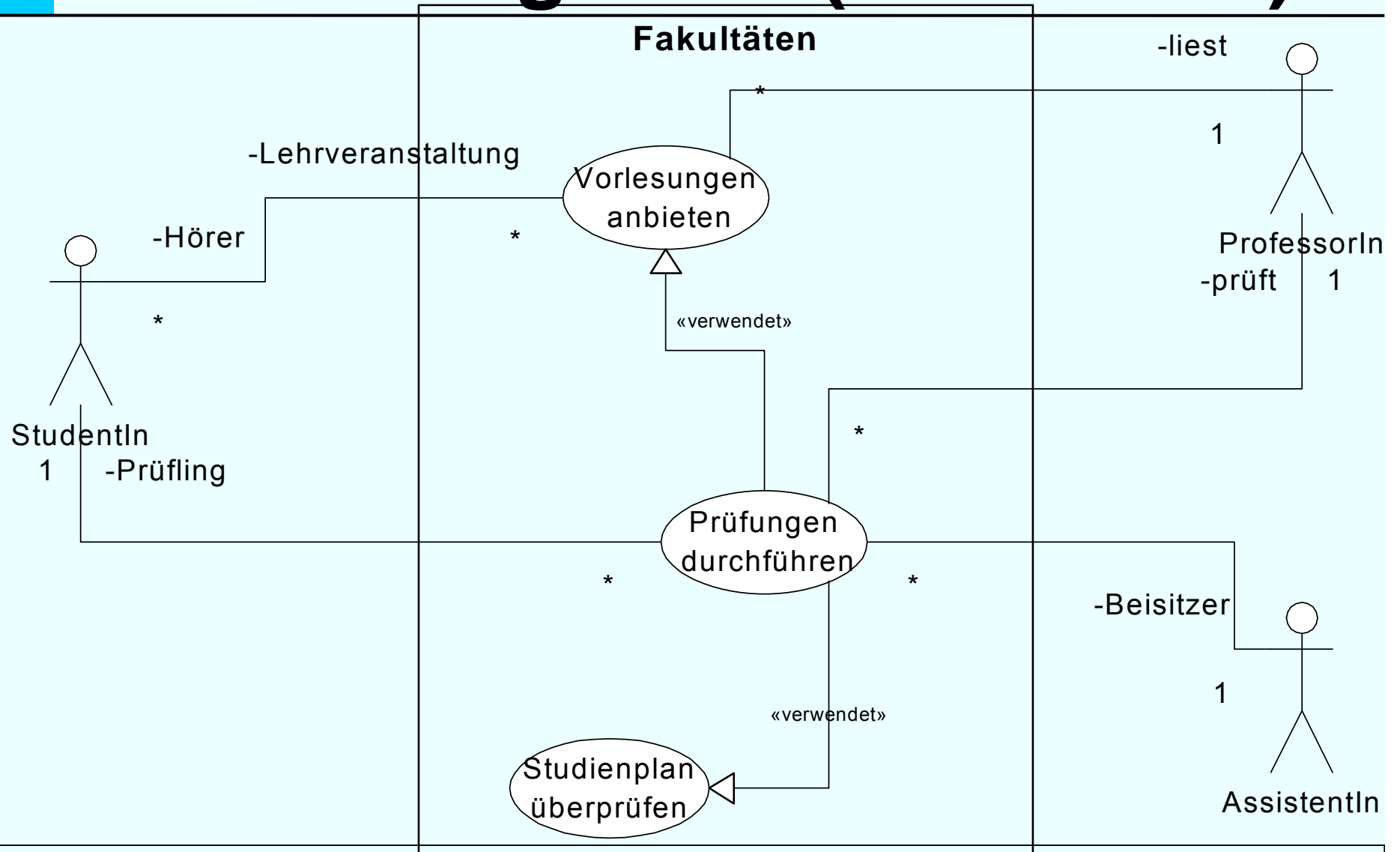


Begrenzungsflächenmodellierung von Polyedern in UML

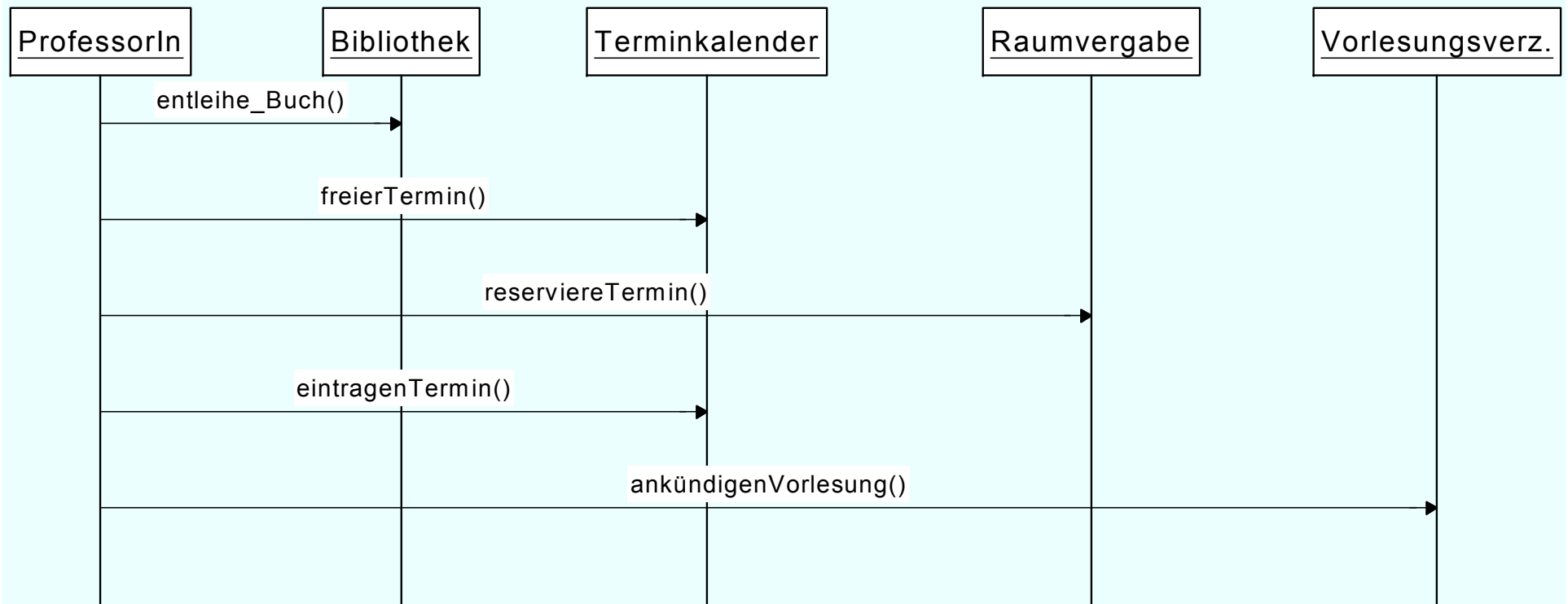




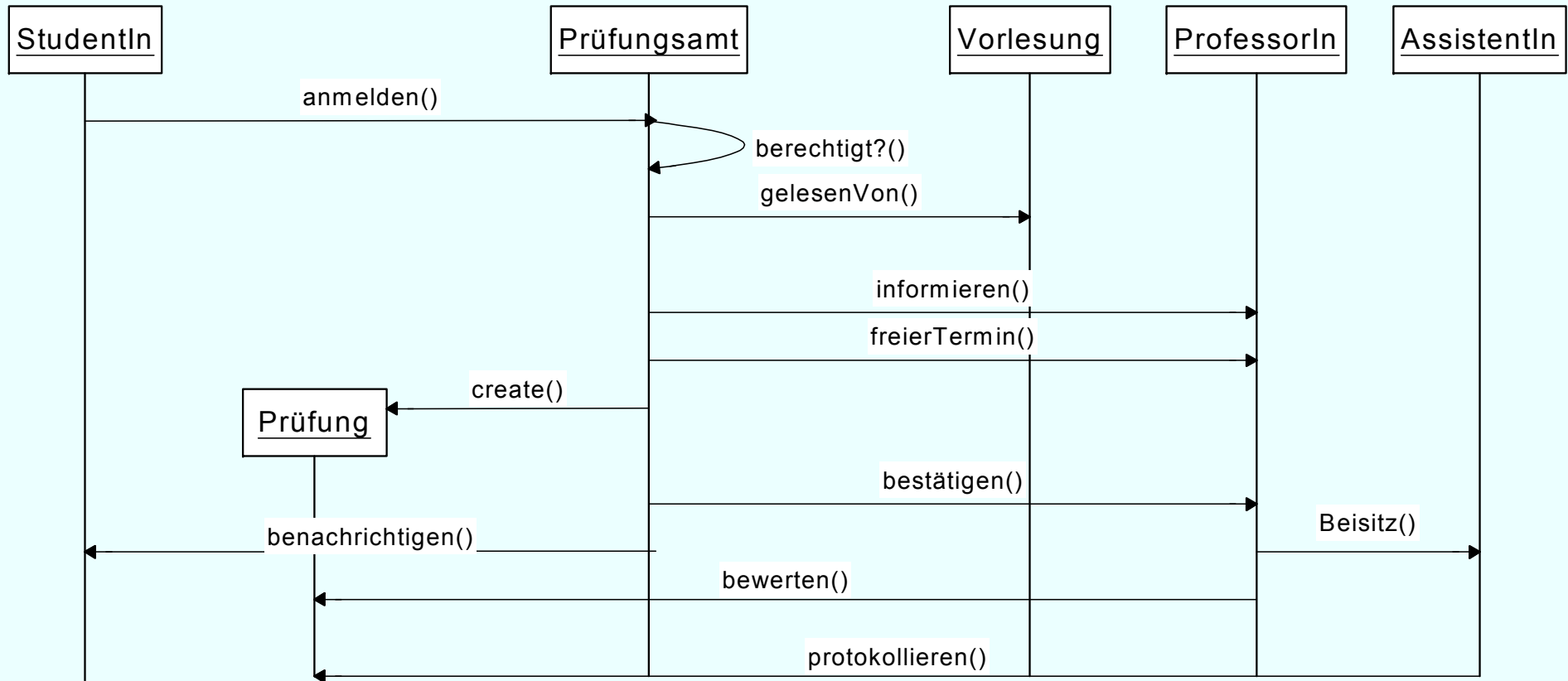
Anwendungsfälle (use cases)



Interaktions-Diagramm: Modellierung komplexer Anwendungen



Interaktions-Diagramm: *Prüfungsdurchführung*



Historische Entwicklung relationaler DBMS

- Codasyl, Anfang 1960: netzwerkartiges Datenmodell
- IMS, Mitte 1960: hierarchisches Datenmodell
- Ted Codd, 1970: Grundlage relationales Datenmodell
- System R, Mitte 1970: relationales Datenbanksystem
(Forschungsprototyp)

Historische Entwicklung relationaler DBMS

Kommerzielle relationale Datenbanksysteme

- Oracle V2, Ende 1970
- Ingres (Berkeley), Ende 1970 → PostgreSQL
- SQL/DS, Anfang 1980: IBM → DB2
- MS SQL Server, 1990 (aus Sybase)
- MySQL, Ende 1990

→ ab Ende 1990: Objektrelationale Erweiterungen

- Objektorientierte DBMS (ab Ende 1980),
XML DBMS (ab Ende 1990)

Grundlagen des relationalen Modells

Seien D_1, D_2, \dots, D_n **Domänen** (\sim Wertebereiche)

Relation: $R \subseteq D_1 \times \dots \times D_n$

Bsp.: Telefonbuch \subseteq string \times string \times integer

Tupel: $t \in R$

Bsp.: $t =$ („Mickey Mouse“, „Main Street“, 4711)

Schema: legt die Struktur der gespeicherten Daten fest

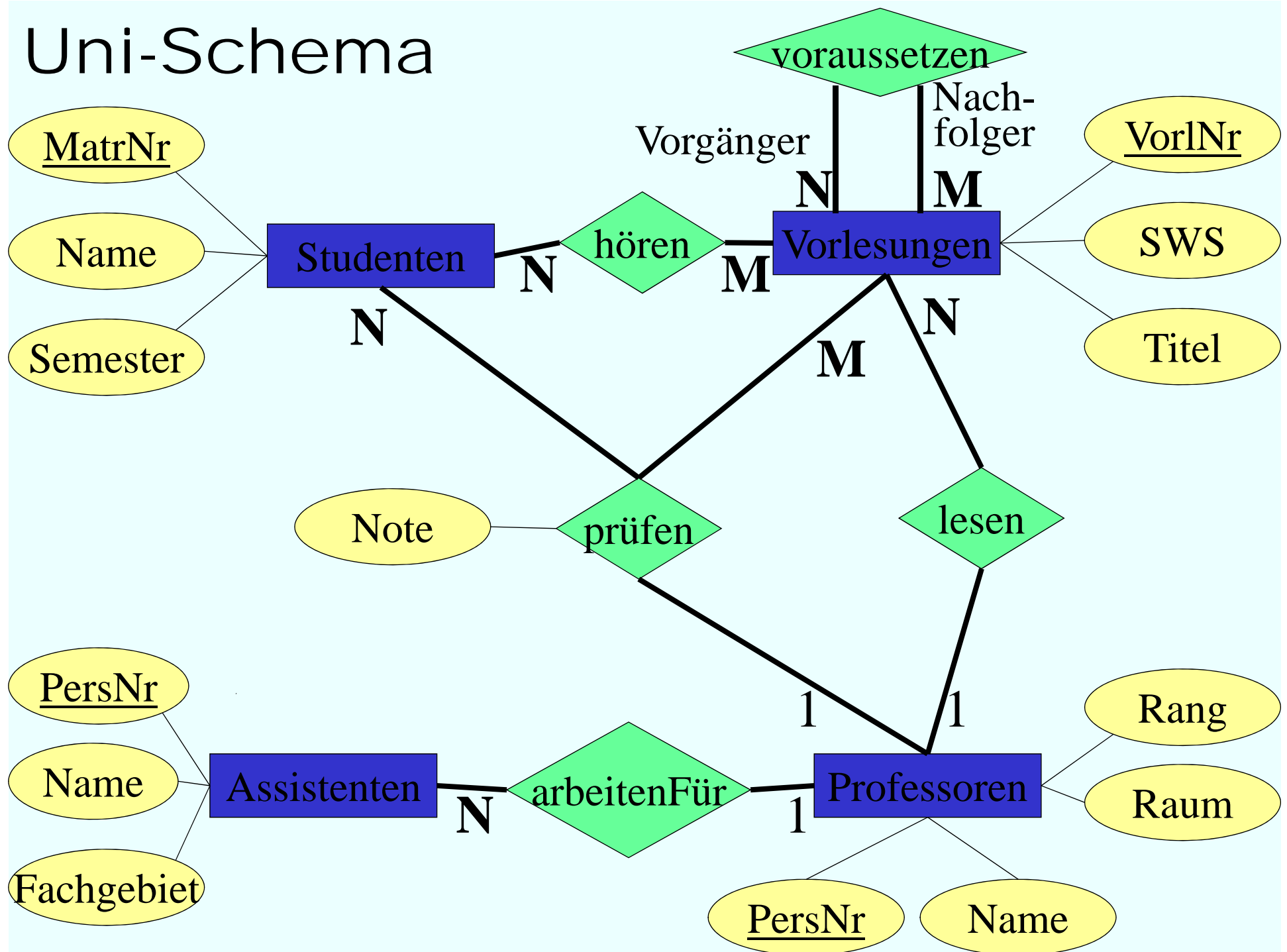
*Bsp.: Telefonbuch: {[Name: string, Adresse: string,
Telefon#: integer]}*

Relation

Telefonbuch		
Name	Straße	<u>Telefon#</u>
Mickey Mouse	Main Street	4711
Minnie Mouse	Broadway	94725
Donald Duck	Broadway	95672
...

- **Ausprägung:** der aktuelle Zustand der Datenbasis
- **Schlüssel:** minimale Menge von Attributen, deren Werte ein Tupel eindeutig identifizieren
- **Primärschlüssel:** wird unterstrichen
 - Einer der Schlüsselkandidaten wird als Primärschlüssel ausgewählt
 - Hat eine besondere Bedeutung bei der Referenzierung von Tupeln

Uni-Schema



Abbildungsregeln (1)

Entitymengen auf Relationen:

Entitymenge E mit Attributen A_i aus Domänen D_i ($1 \leq i \leq k$)
 \Rightarrow k -stellige Relation $E(A_1:D_1, \dots, A_k:D_k)$.

Übernahme der Schlüsselattribute

Relationale Darstellung von Entitymengen

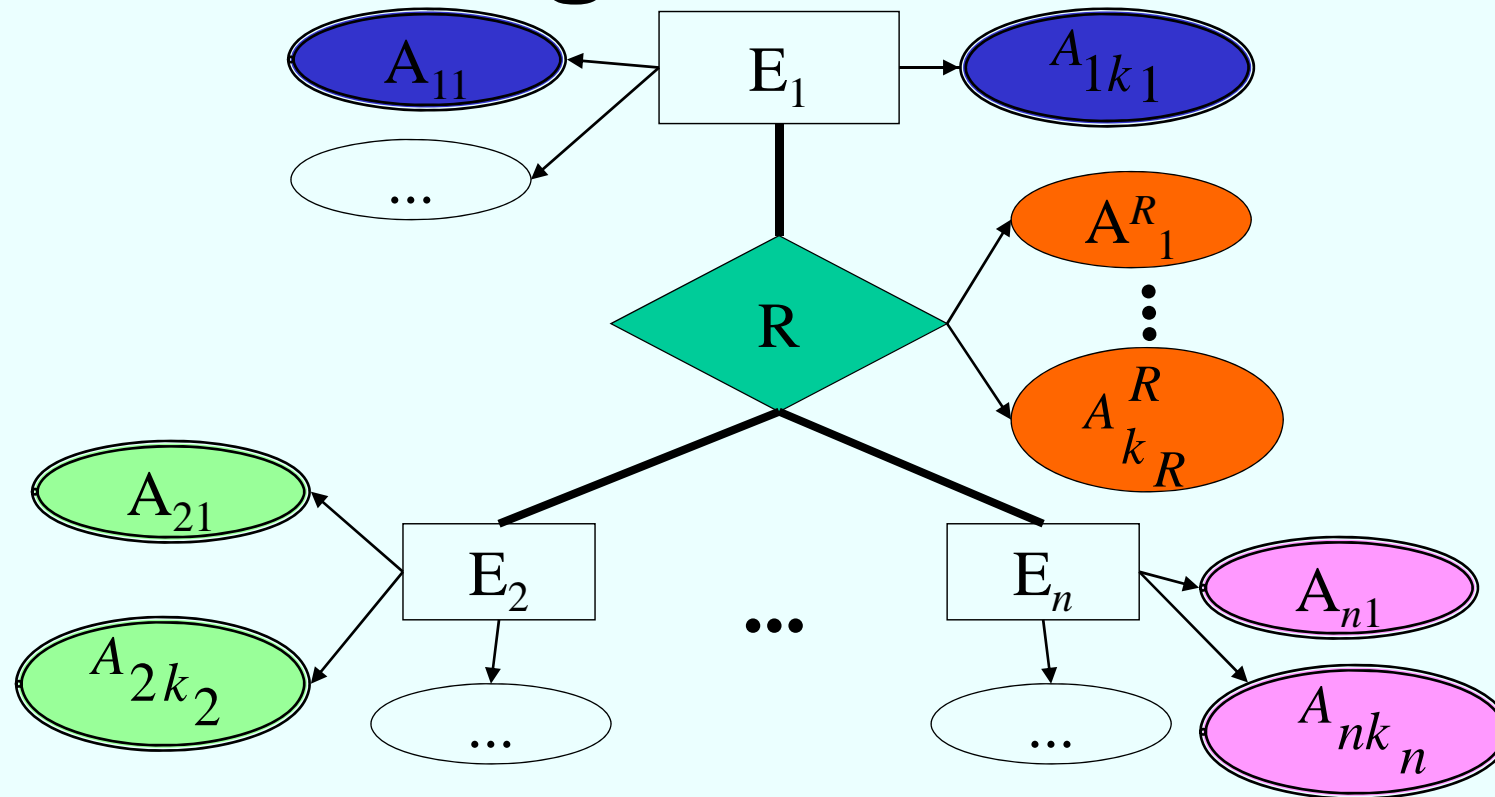
Studenten: {[MatrNr:integer, *Name: string*,
Semester: integer]}

Vorlesungen: {[VorlNr:integer, *Titel: string*,
SWS: integer]}

Professoren: {[PersNr:integer, *Name: string*,
Rang: string, *Raum: integer*]}

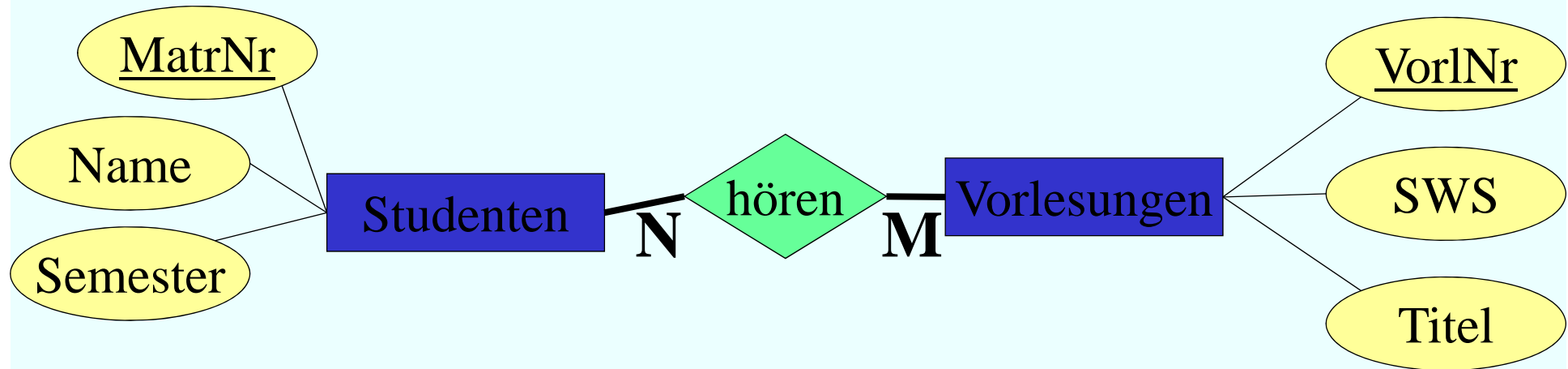
Assistenten: {[PersNr:integer, *Name: string*,
Fachgebiet: string]}

Relationale Darstellung von Beziehungen



$R: \{ \underbrace{[A_{11}, \dots, A_{1k_1}]}_{\text{Schlüssel von } E_1}, \underbrace{[A_{21}, \dots, A_{2k_2}]}_{\text{Schlüssel von } E_2}, \dots, \underbrace{[A_{n1}, \dots, A_{nk_n}]}_{\text{Schlüssel von } E_n}, \underbrace{[A_1^R, \dots, A_{k_R}^R]}_{\text{Attribute von } R} \}$

Beziehungen unseres Beispiel-Schemas



hören (N:M): {[MatrNr: integer, VorlNr: integer]}

lesen (1:N): {[PersNr: integer, VorlNr: integer]}

arbeitenFür (N:1): {[AssistentenPersNr: integer, ProfPersNr: integer]}

voraussetzen (N:M): {[Vorgänger: integer, Nachfolger: integer]}

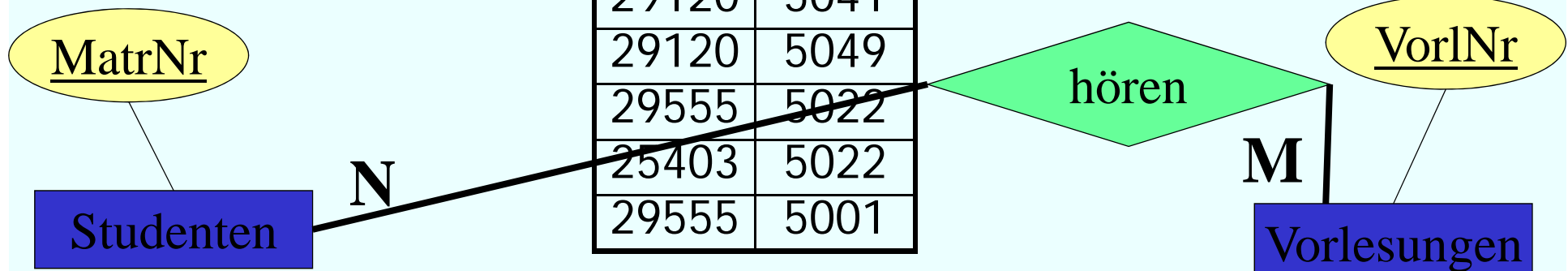
prüfen (N:M:1): {[MatrNr: integer, VorlNr: integer, PersNr: integer, Note: decimal]}

Ausprägung Beziehung *hören*

Studenten	
<i>MatrNr</i>	...
26120	...
27550	...
...	...

hören	
MatrNr	VorlNr
26120	5001
27550	5001
27550	4052
28106	5041
28106	5052
28106	5216
28106	5259
29120	5001
29120	5041
29120	5049
29555	5022
25403	5022
29555	5001

Vorlesungen	
<i>VorlNr</i>	...
5001	...
4052	...
...	...



Verfeinerte Abbildungsregeln

Relationships auf Relationen (cont'd):

1:1-Beziehung:

Relationship R zwischen 2 Entities E und F.

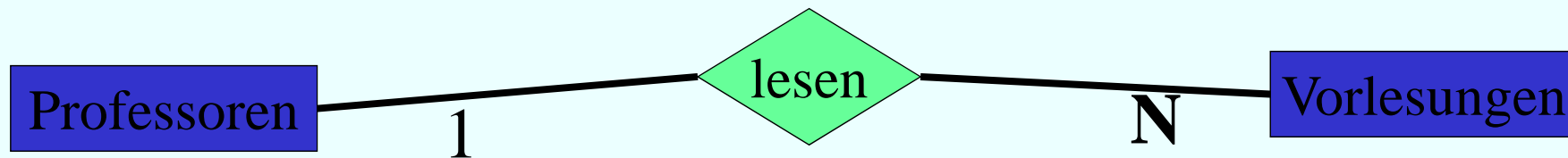
⇒ keine Relation aus R, stattdessen Primärschlüssel von E in Relation in F oder umgekehrt.

1:n-Beziehung:

Relationship R zwischen 2 Entities E und F.

⇒ Keine Relation R, stattdessen Primärschlüssel von E in Relation F als Fremdschlüssel aufnehmen. Fall R eigene Attribute hat, müssen diese auch in F aufgenommen werden.

Verfeinerung des relationalen Schemas



1:N-Beziehung

Initial-Entwurf

Vorlesungen : {[VorlNr, Titel, SWS]}

Professoren : {[PersNr, Name, Rang, Raum]}

lesen: {[VorlNr, PersNr]}

Verfeinerung des relationalen Schemas

1:N-Beziehung

Initial-Entwurf

Vorlesungen : {[VorlNr, Titel, SWS]}

Professoren : {[PersNr, Name, Rang, Raum]}

lesen: {[VorlNr, PersNr]}

Verfeinerung durch Zusammenfassung

Vorlesungen : {[VorlNr, Titel, SWS,
gelesenVon]}

Professoren : {[PersNr, Name, Rang, Raum]}

Regel: Relationen mit gleichem Schlüssel kann man zusammenfassen

... aber nur diese und keine anderen!

Ausprägung von *Professoren* und *Vorlesung*

Professoren			
PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

Vorlesungen			
VorlNr	Titel	SWS	Gelesen Von
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125
4052	Logik	4	2125
5052	Wissenschaftstheorie	3	2126
5216	Bioethik	2	2126
5259	Der Wiener Kreis	2	2133
5022	Glaube und Wissen	2	2134
4630	Die 3 Kritiken	4	2137

Professoren

1

lesen

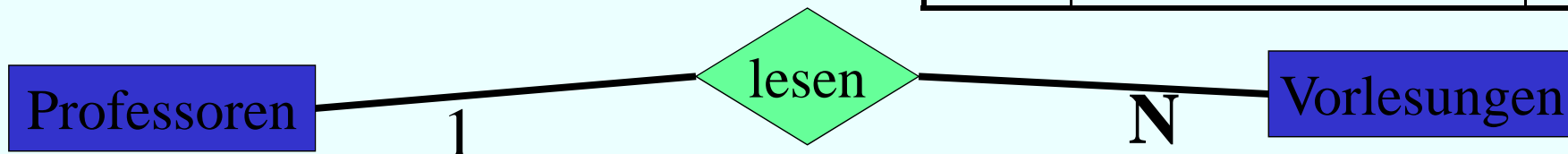
N

Vorlesungen

Vorsicht: So geht es NICHT

Professoren				
PersNr	Name	Rang	Raum	liest
2125	Sokrates	C4	226	5041
2125	Sokrates	C4	226	5049
2125	Sokrates	C4	226	4052
...
2134	Augustinus	C3	309	5022
2136	Curie	C4	36	??

Vorlesungen		
VorlNr	Titel	SWS
5001	Grundzüge	4
5041	Ethik	4
5043	Erkenntnistheorie	3
5049	Mäeutik	2
4052	Logik	4
5052	Wissenschaftstheorie	3
5216	Bioethik	2
5259	Der Wiener Kreis	2
5022	Glaube und Wissen	2
4630	Die 3 Kritiken	4



Vorsicht: So geht es NICHT: Folgen → Anomalien

Professoren				
PersNr	Name	Rang	Raum	liest
2125	Sokrates	C4	226	5041
2125	Sokrates	C4	226	5049
2125	Sokrates	C4	226	4052
...
2134	Augustinus	C3	309	5022
2136	Curie	C4	36	??

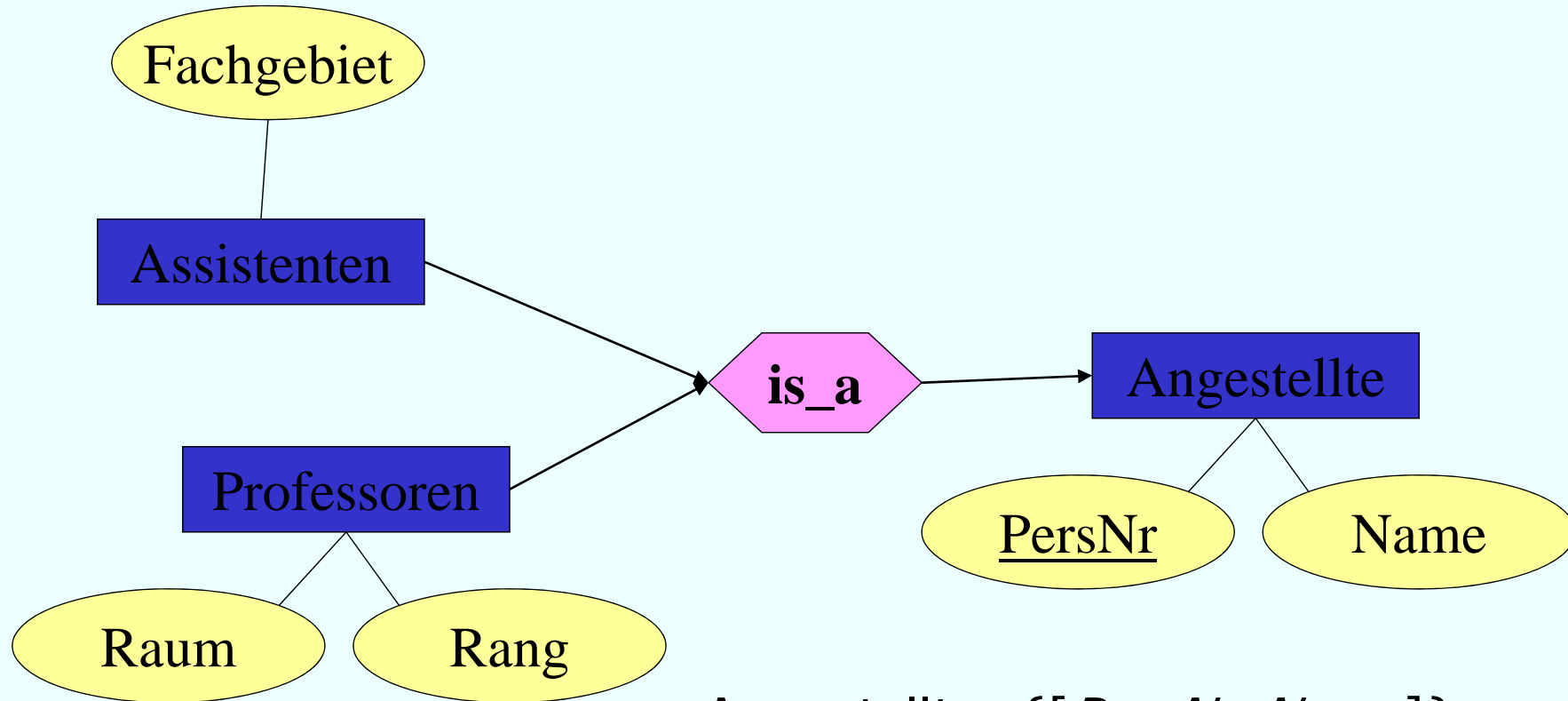
Vorlesungen		
VorINr	Titel	SWS
5001	Grundzüge	4
5041	Ethik	4
5043	Erkenntnistheorie	3
5049	Mäeutik	2
4052	Logik	4
5052	Wissenschaftstheorie	3
5216	Bioethik	2
5259	Der Wiener Kreis	2
5022	Glaube und Wissen	2
4630	Die 3 Kritiken	4

Update-Anomalie: Was passiert wenn Sokrates umzieht

Lösch-Anomalie: Was passiert wenn „Glaube und Wissen“ wegfällt

Einfügeanomalie: Curie ist neu und liest noch keine Vorlesungen

Relationale Modellierung der Generalisierung

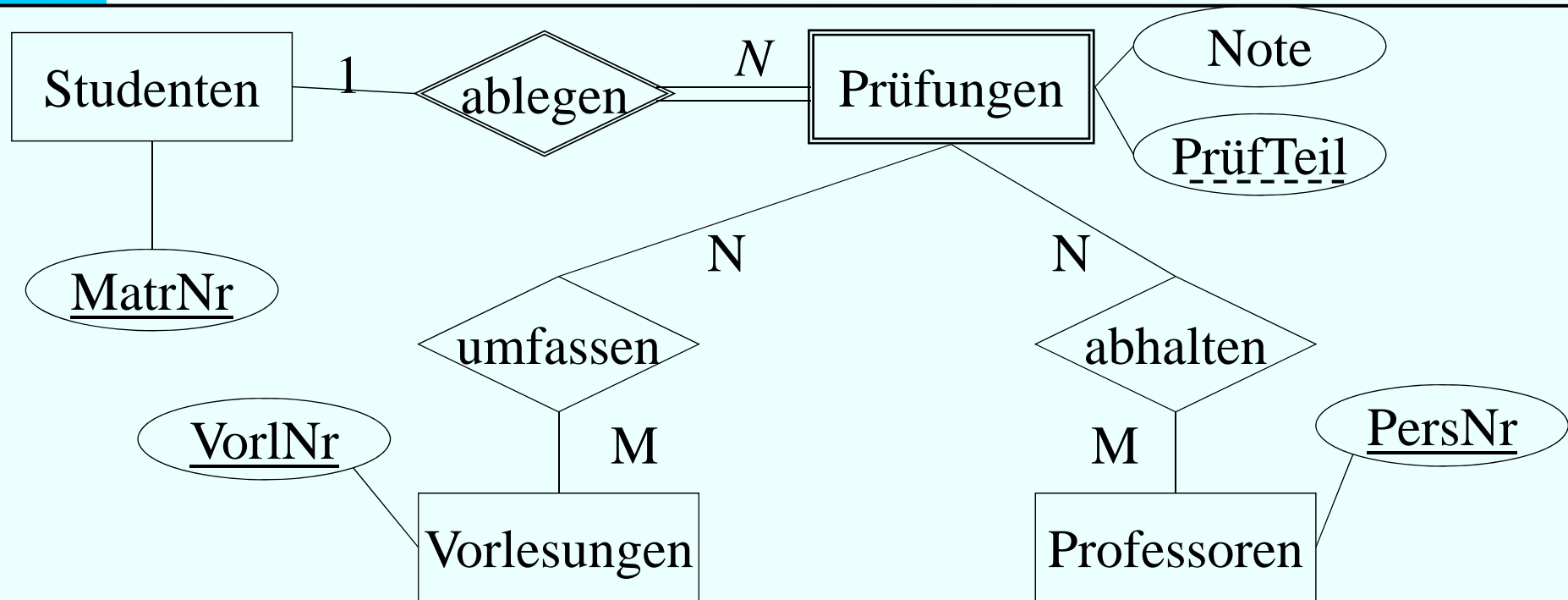


Angestellte: $\{[\underline{PersNr}, Name]\}$

Professoren: $\{[\underline{PersNr}, Rang, Raum]\}$

Assistenten: $\{[\underline{PersNr}, Fachgebiet]\}$

Relationale Modellierung schwacher Entitytypen



Prüfungen: {[MatrNr: integer, PrüfTeil: string, Note: integer]}

umfassen: {[MatrNr: integer, PrüfTeil: string, VorlNr: integer]}

abhalten: {[MatrNr: integer, PrüfTeil: string, PersNr: integer]}

Fremdschlüssel auf ein schwaches Entity

Man beachte, dass in diesem Fall der (global eindeutige) Schlüssel der Relation *Prüfung* nämlich *MatrNr* **und** *PrüfTeil* als Fremdschlüssel in die Relationen *umfassen* und *abhalten* übernommen werden muß.