

# Datenmodellierung

DBS kann vieles, aber nicht alles!

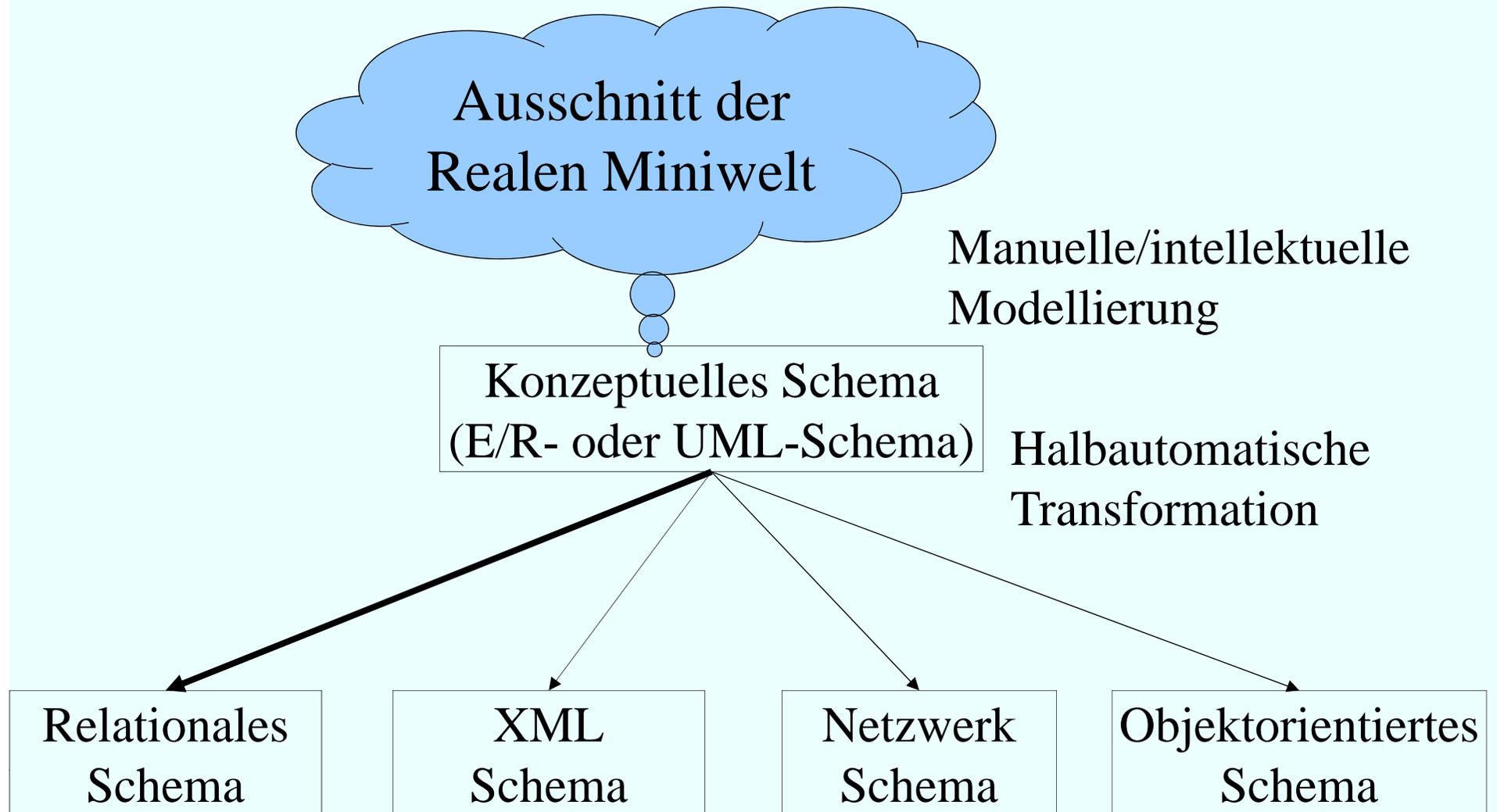
Benutzer muss spezifizieren

- 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)

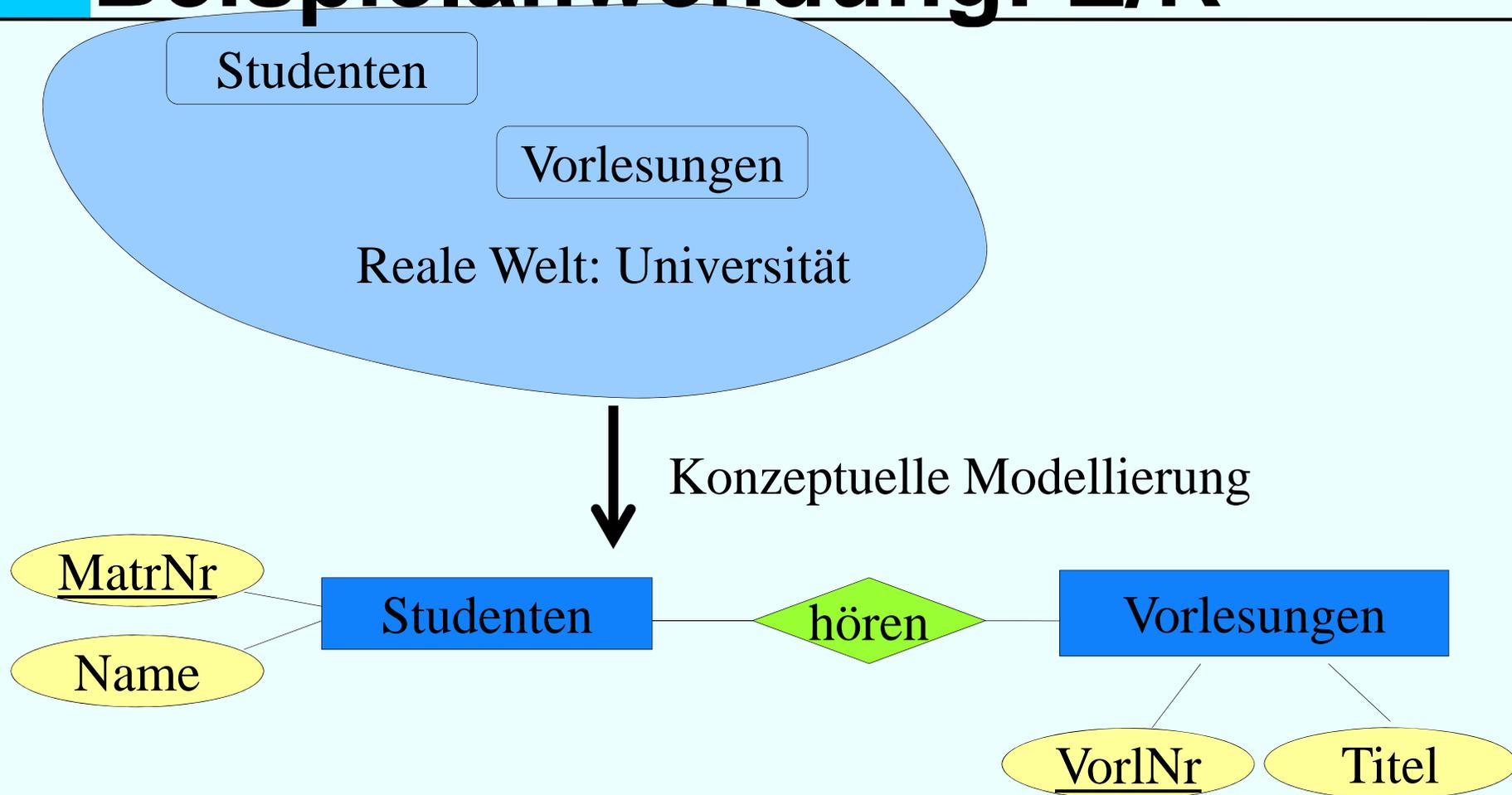
# Datenmodellierung



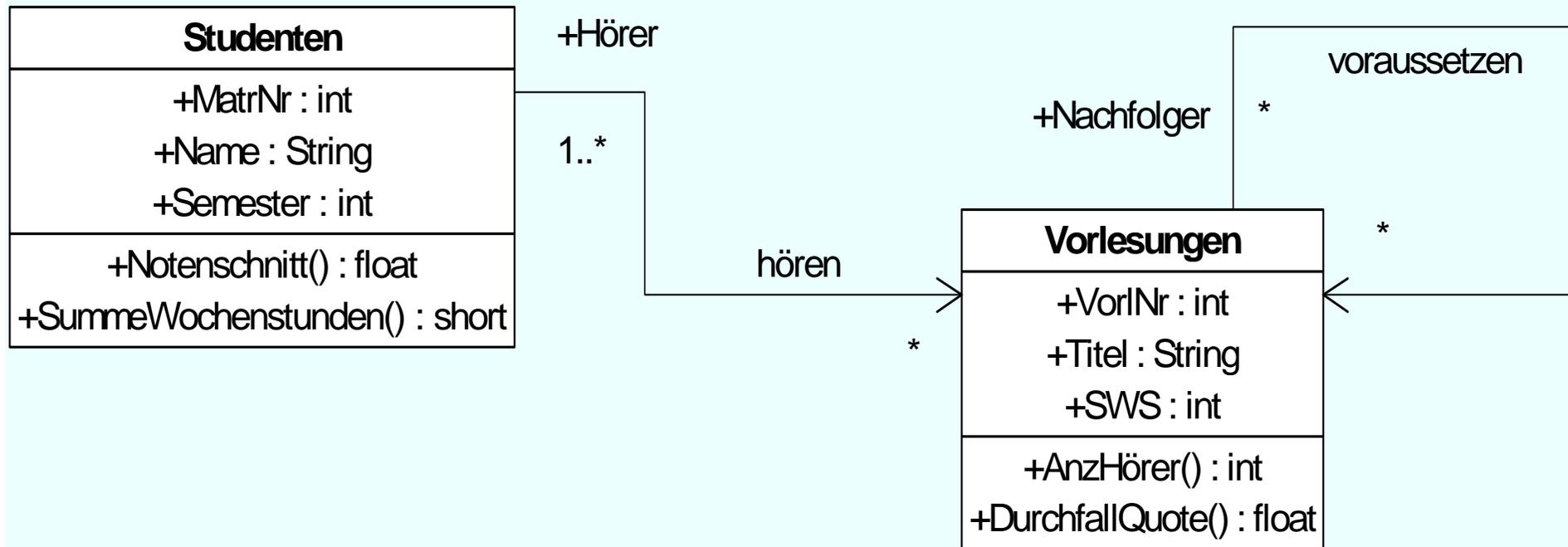
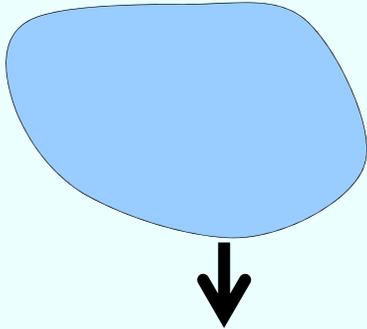
# Logische Datenmodelle

- Netzwerkmodell
- Hierarchisches Datenmodell
- **Relationales Datenmodell**
- XML Schema
- Objektorientiertes Datenmodell
  - Objektrelationales Schema
- Deduktives Datenmodell

# 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

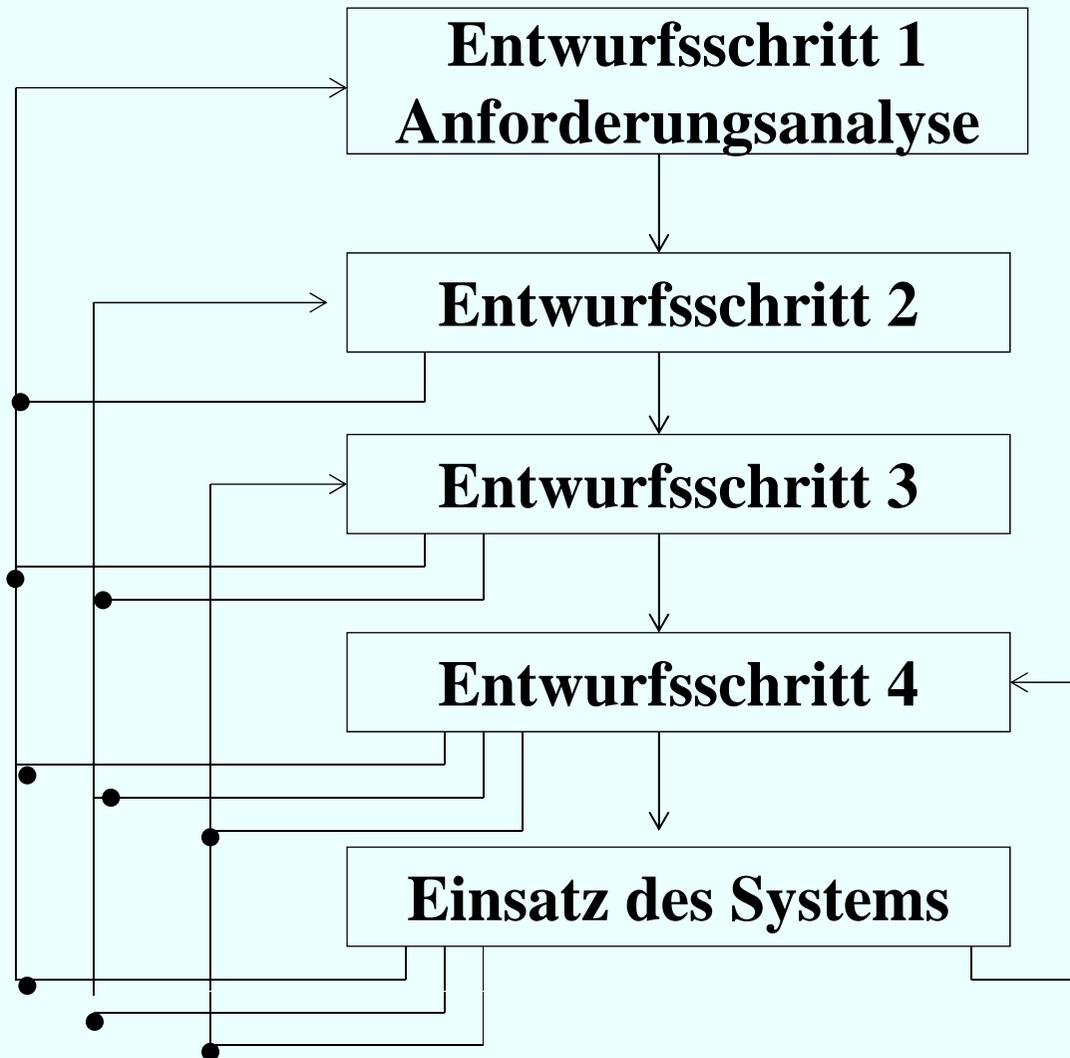
**Where** Studenten.MatrNr = hören.MatrNr **and**  
hören.VorlNr = Vorlesungen.VorlNr **and**  
Vorlesungen.Titel = `Grundzüge`;

# Datenbankentwurf

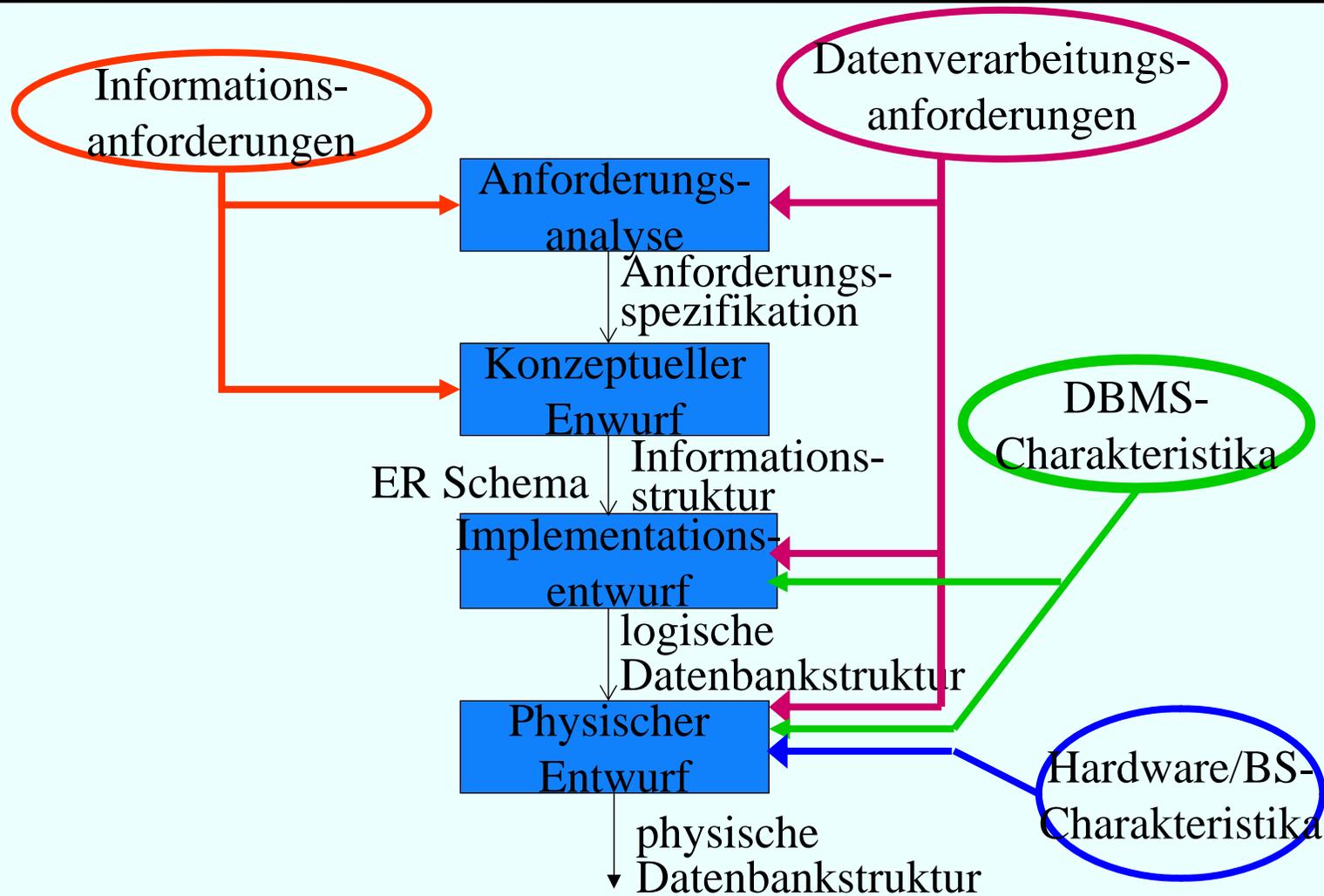
## Abstraktionsebenen des Datenbankentwurfs

1. Konzeptuelle Ebene
2. Implementationsebene
3. Physische Ebene

# Allgemeiner top-down Entwurf



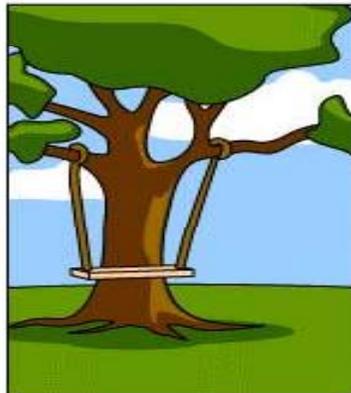
# Phasen des Datenbankentwurfs



# Softwareentwicklung und Kommunikationsfähigkeit



Wie es der Kunde erklärte



Wie es der Projektleiter verstand



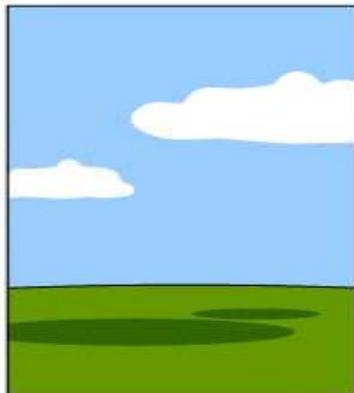
Wie es der Engineer geplant hat



Wie es der Programmierer umsetzte



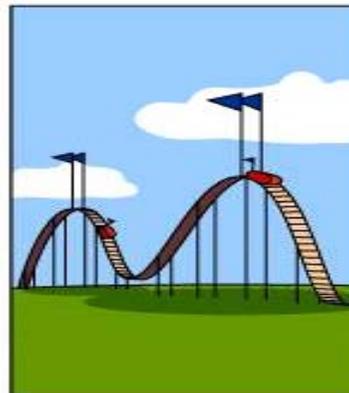
Wie es der Berater verstand



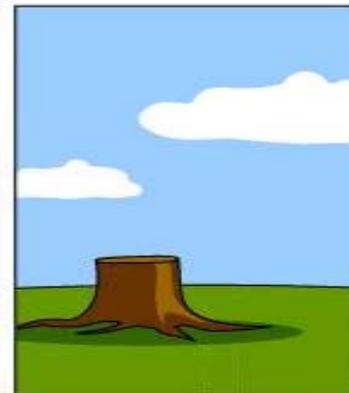
Wie es dokumentiert wurde



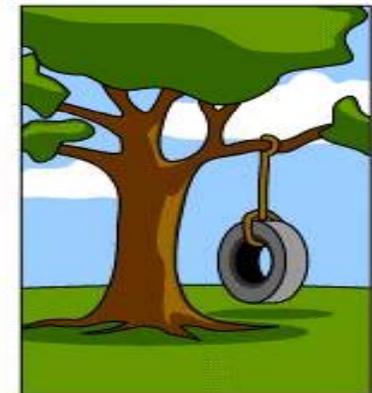
Wie es installiert wurde



Was dem Kunde berechnet wurde



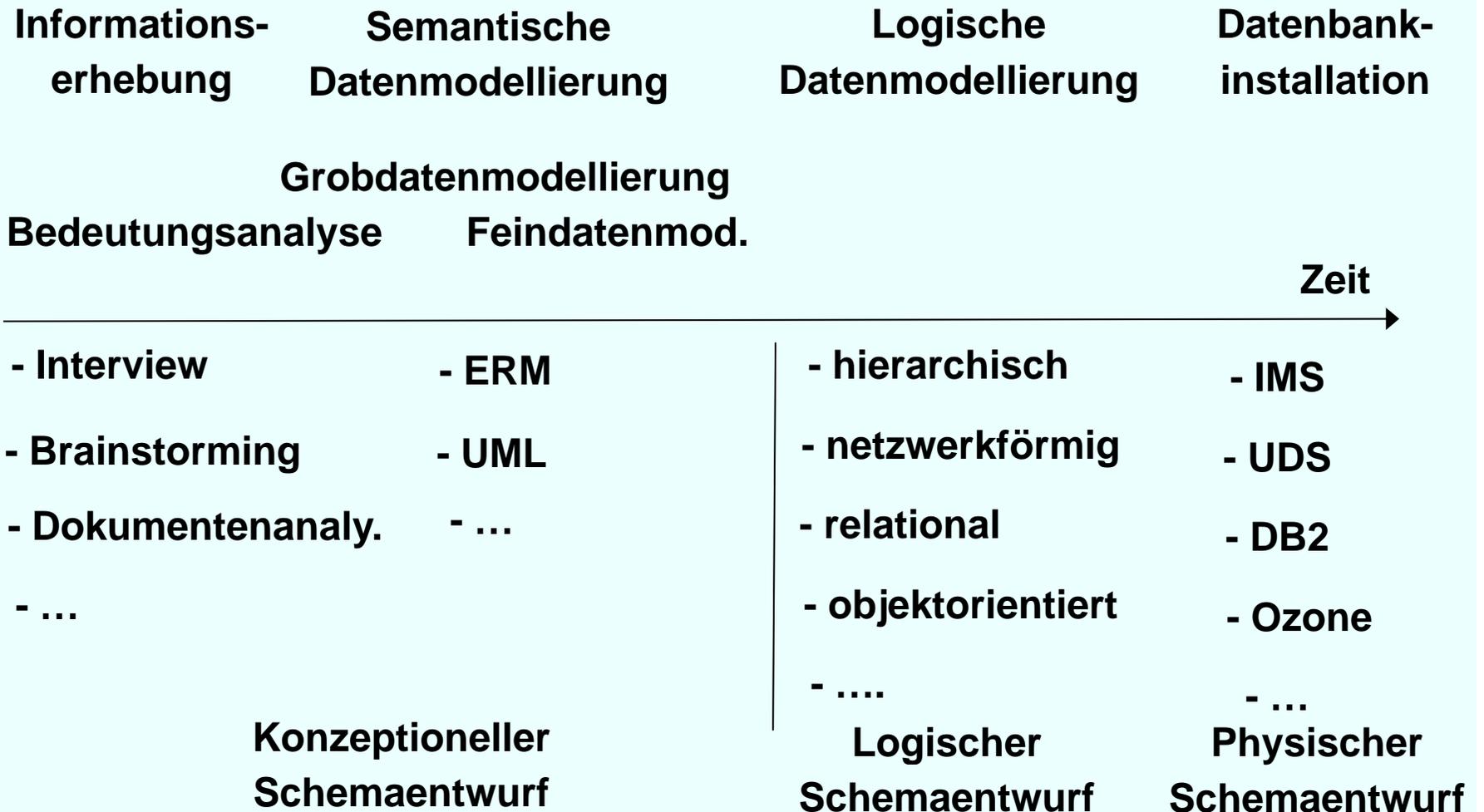
Was der Servicevertrag unterstützt



Was der Kunde wirklich wollte

# Schemaentwurf

## Prinzipielle Vorgehensweise:



# Objektbeschreibung

## Uni-Angestellte

-Anzahl: 1000

-Attribute

### ❖ Personalnummer

- Typ: Zahl
- Wertebereich:  
0...999.999.99
- Definiertheit: 100%
- Identifizierend: ja
- Beispiel: 007

### ❖ Gehalt

- Typ: dezimal
- Länge: (7,2)
- Einheit: Euro pro Monat
- Definiertheit: 10%
- Identifizierend: nein

### ❖ Rang

- Typ: String
- Länge: 2
- Definiertheit: 100%
- Identifizierend: nein
- Beispiel: W2

# Beziehungsbeschreibung: *prüfen*

## Beteiligte Objekte:

- Professor als Prüfer
- Student als Prüfling
- Vorlesung als Prüfungsstoff

## Attribute der Beziehung:

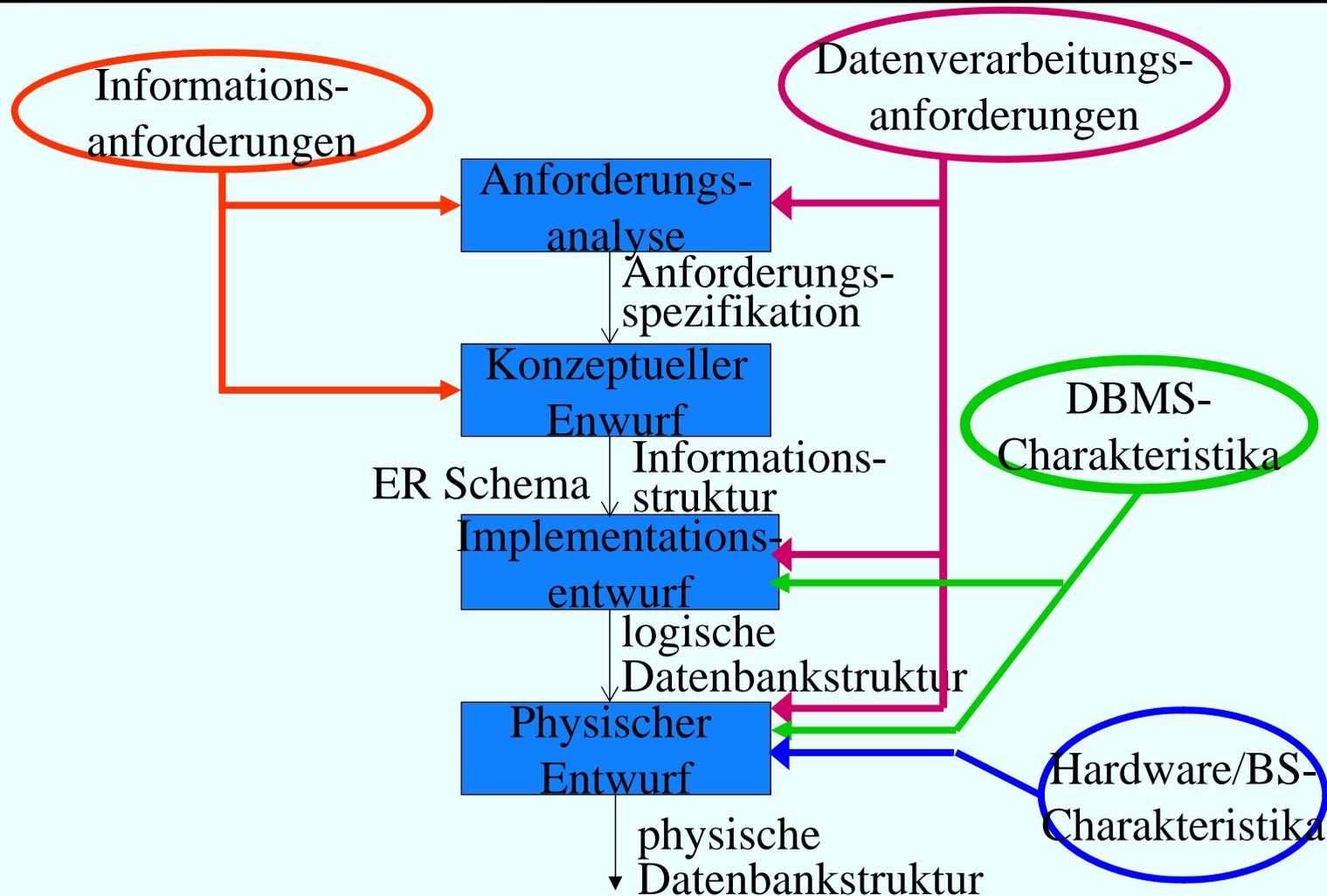
- Datum
- Uhrzeit
- Note

Anzahl: 100 000 pro Jahr

# Prozeßbeschreibung: *Zeugnisausstellung*

- Häufigkeit: halbjährlich
- benötigte Daten
  - \* Prüfungen
  - \* Studienordnungen
  - \* Studenteninformation
  - \* ...
- Priorität: hoch
- Zu verarbeitende Datenmenge
  - \* 500 Studenten
  - \* 3000 Prüfungen
  - \* 10 Studienordnungen

# Phasen des Datenbankentwurfs



# Konzeptueller Entwurf

Der ideale Entwurf (die ideale Spezifikation) ist

- eindeutig
- vollständig
- verständlich (für alle Beteiligten)
- redundanzfrei
- . . . und in der Realität nicht zu erreichen

# Erstellung einer Spezifikation

Die eigentliche Analyse ist ein iterativer Prozess:

- Anwender erzählt Entwickler was er gern hätte
- Entwickler schreibt alles (was er verstanden hat) in seiner "Sprache" auf . . .
- . . . und übersetzt es in die "Sprache" des Anwenders
- dies wird dem Anwender gezeigt, mit dem Ergebnis, dass vieles noch nicht stimmt
- Änderungswünsche werden aufgenommen
- zurück zum zweiten Schritt

# Entity/Relationship-Modellierung

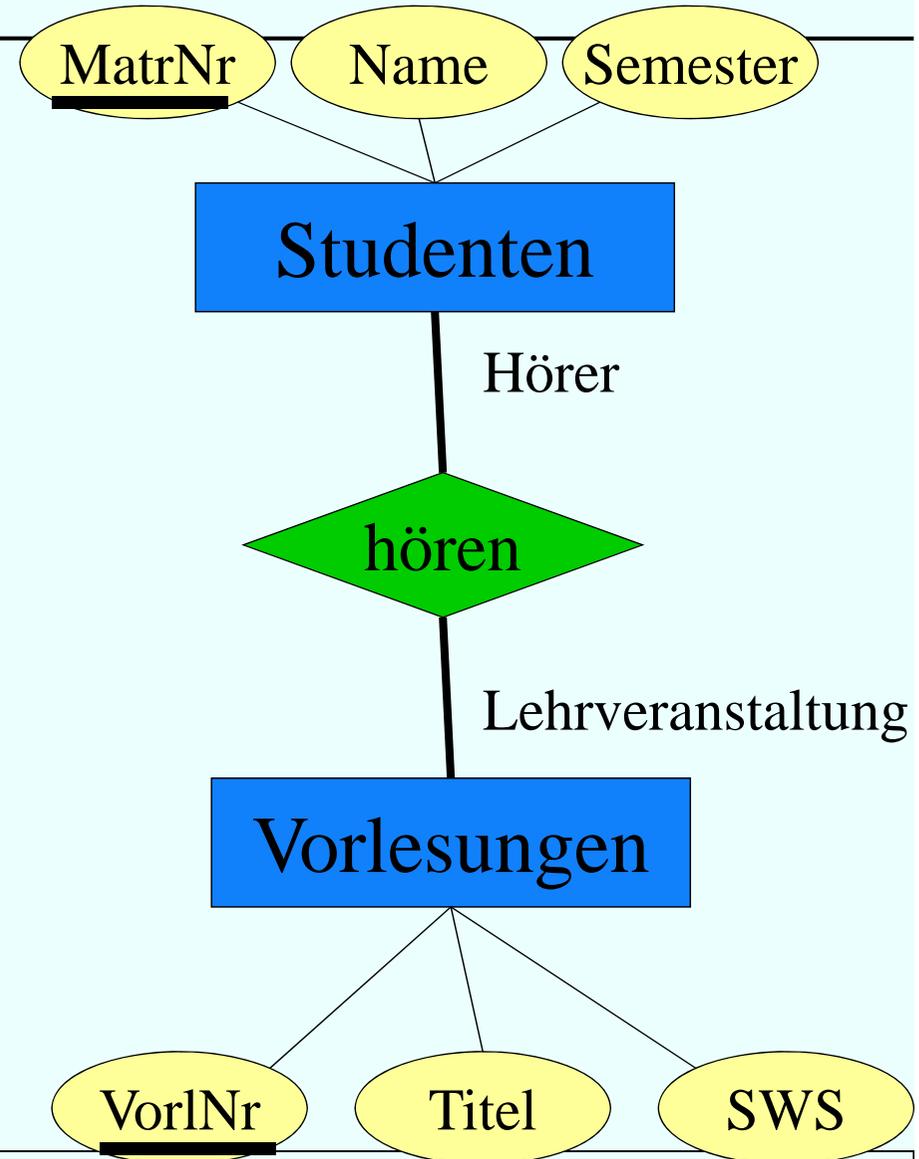
Entity (Gegenstandstyp)

Relationship (Beziehungstyp)

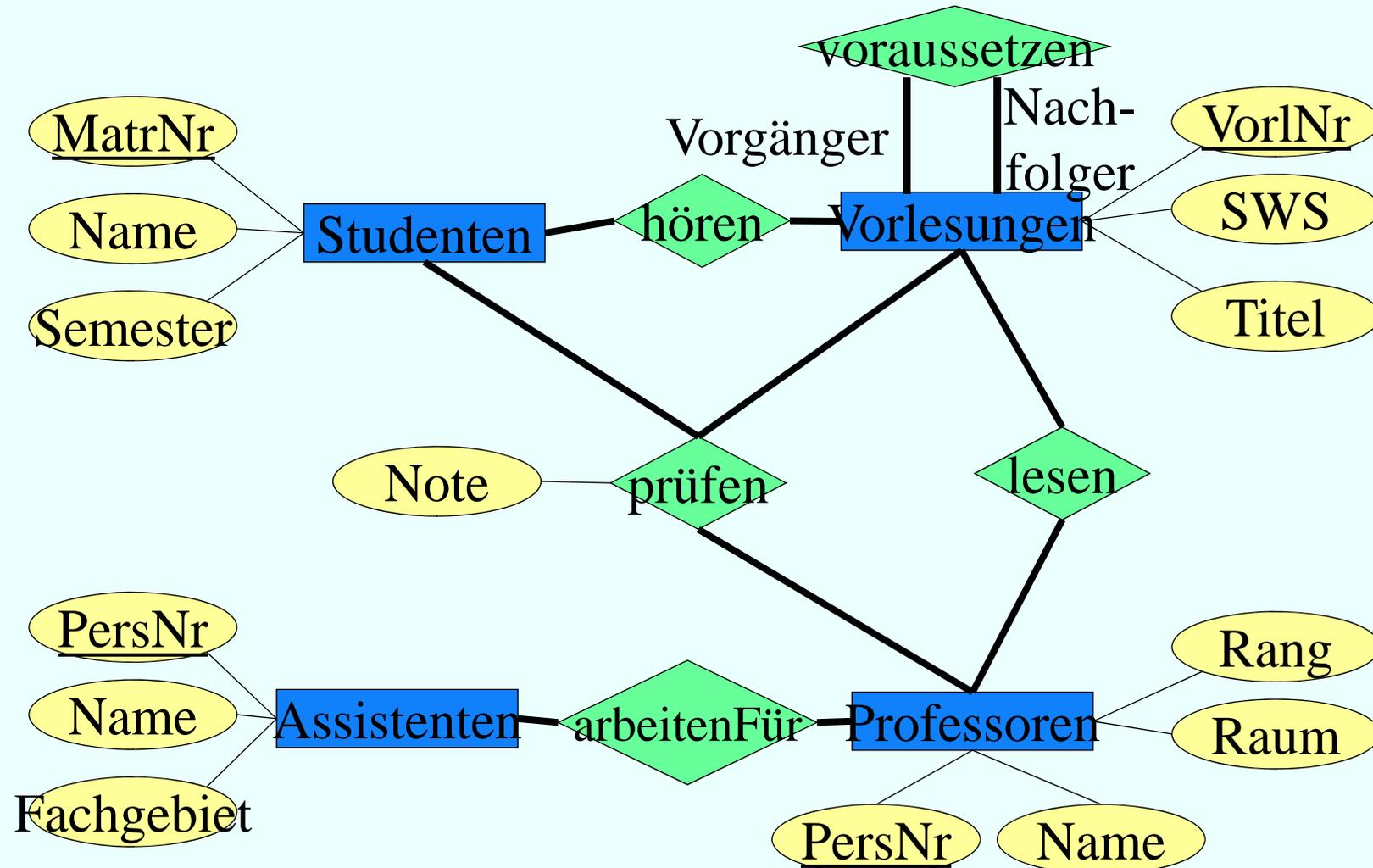
Attribut (Eigenschaft)

Schlüssel (Identifikation)

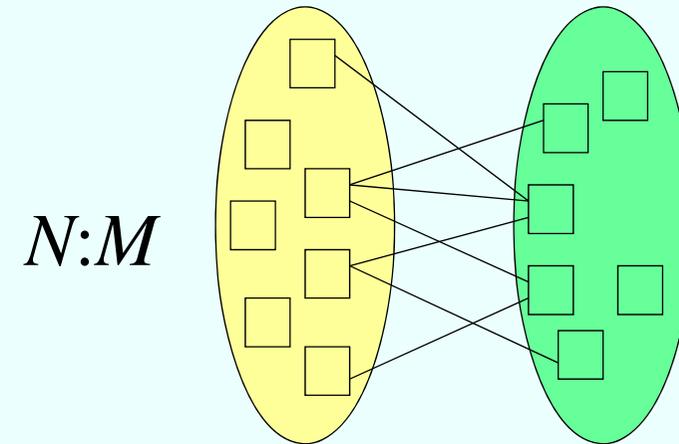
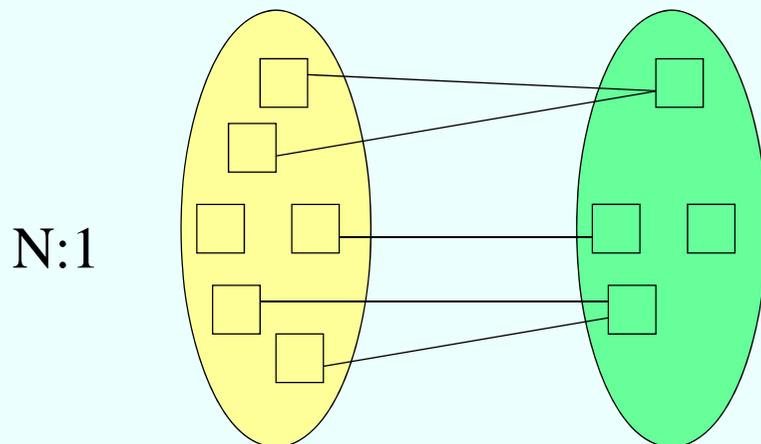
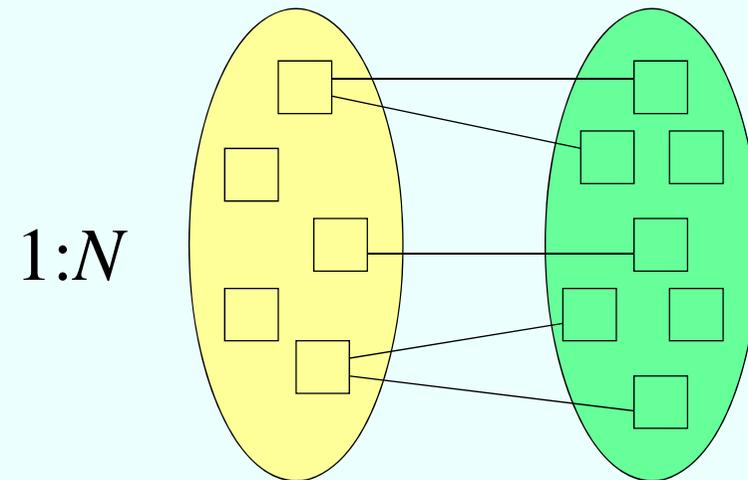
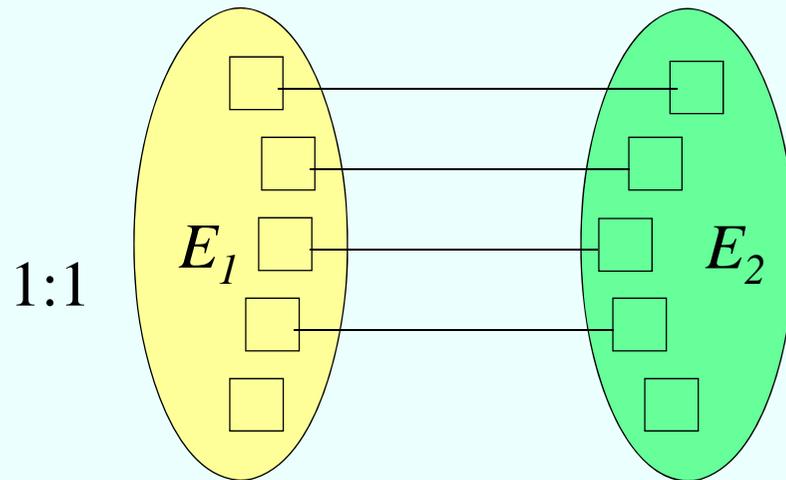
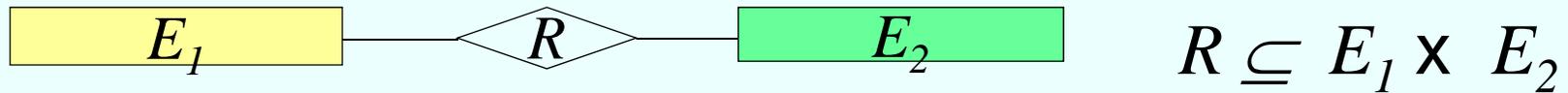
Rolle



# Universitätsschema

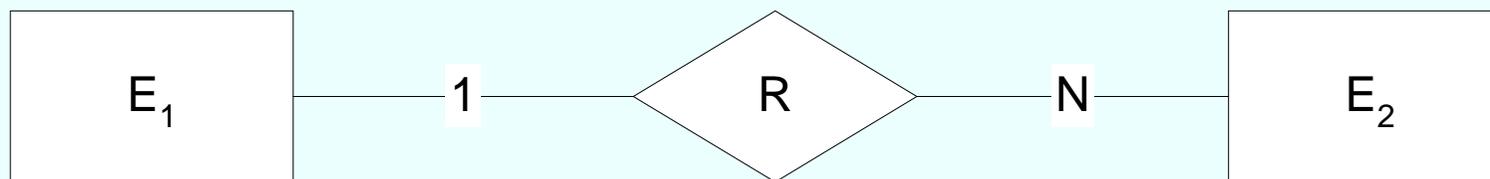


# Funktionalitäten



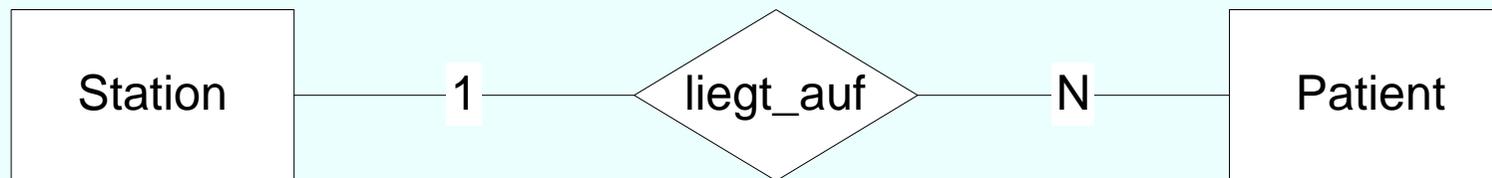
# Beziehungstyp 1:N

## Beziehungstyp 1:N



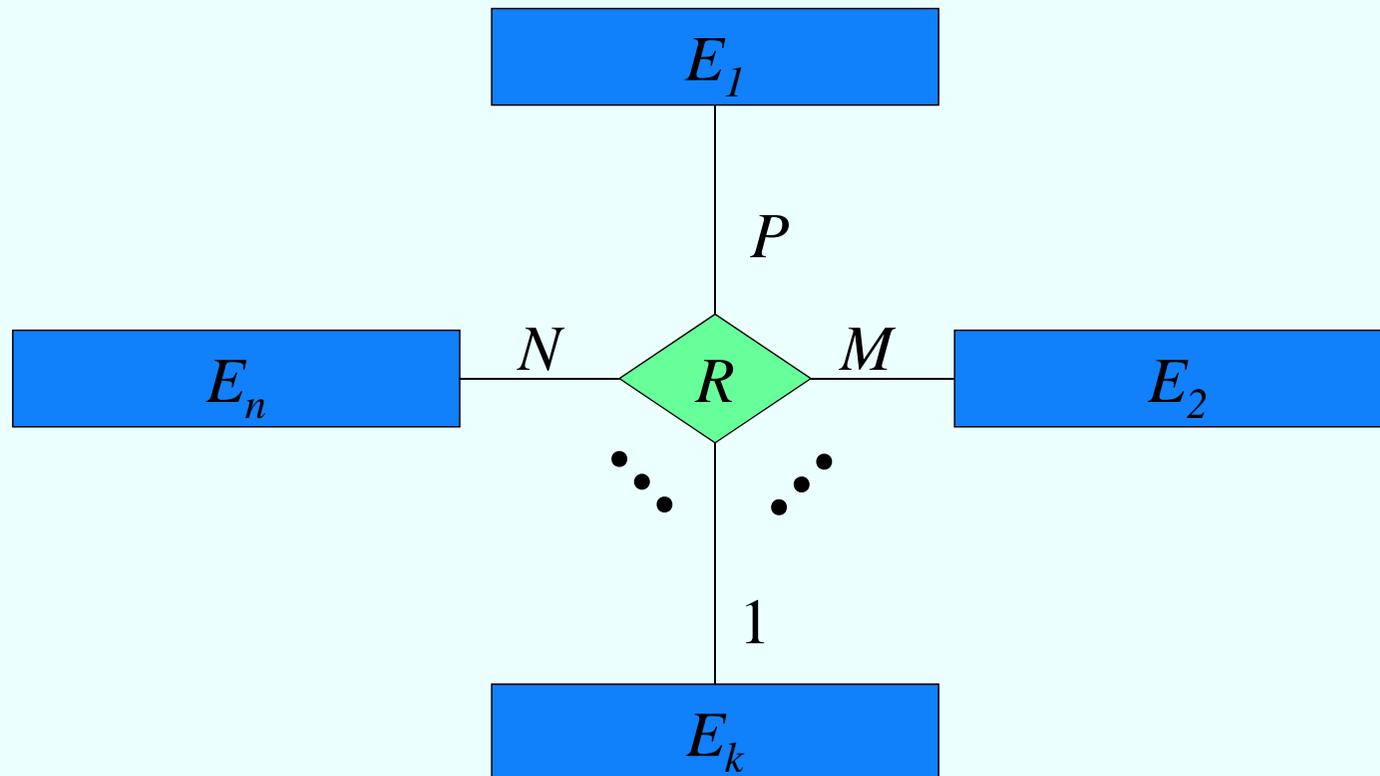
$e_1$  aus  $E_1$  nimmt an  $N$  Beziehungen vom Typ  $R$  teil  
 $e_2$  aus  $E_2$  nimmt an  $1$  Beziehung vom Typ  $R$  teil

## Beispiel:



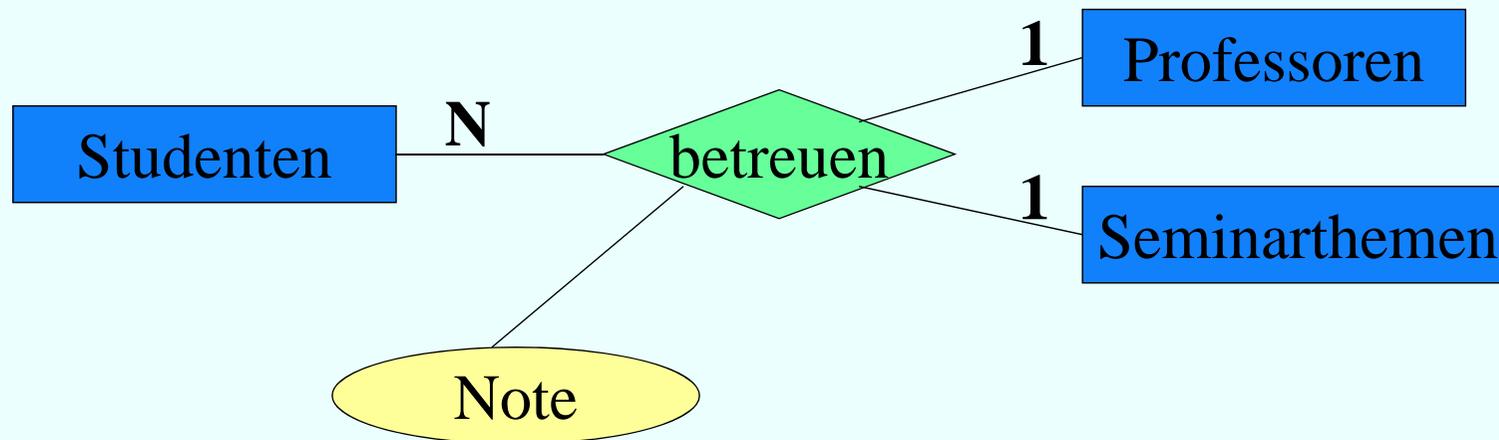
eine Station beherbergt mehrere Patienten  
ein Patient liegt auf einer Station

# Funktionalitäten bei $n$ -stelligen Beziehungen



$$R : E_1 \times \dots \times E_{k-1} \times E_{k+1} \times \dots \times E_n \rightarrow E_k$$

# Beispiel-Beziehung: *betreuen*



betreuen : Professoren x Studenten  $\rightarrow$  Seminarthemen

betreuen : Seminarthemen x Studenten  $\rightarrow$  Professoren

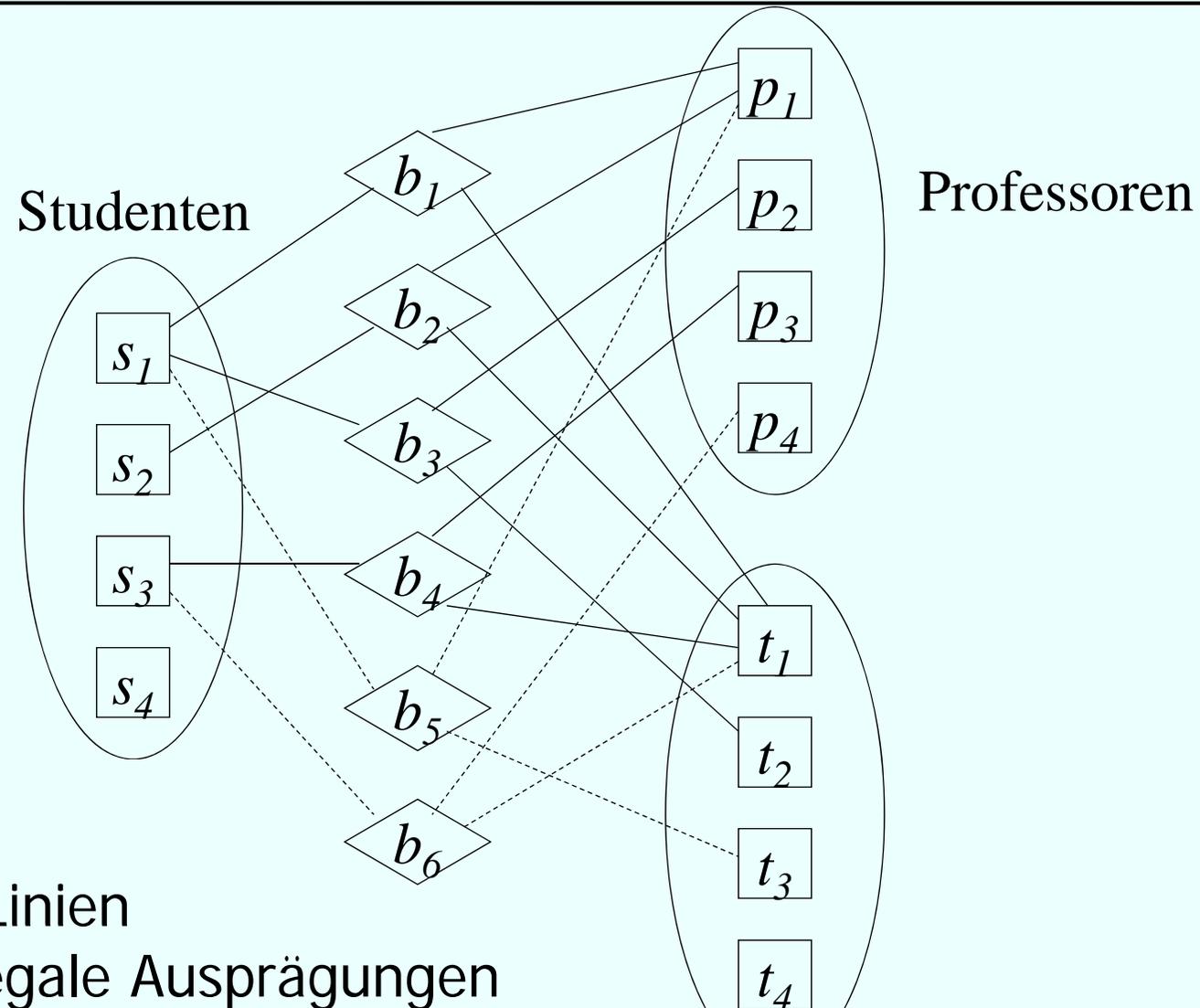
# Dadurch erzwungene Konsistenzbedingungen

1. Studenten dürfen bei demselben Professor bzw. derselben Professorin nur ein Seminarthema "ableisten" (damit ein breites Spektrum abgedeckt wird).
2. Studenten dürfen dasselbe Seminarthema nur einmal bearbeiten – sie dürfen also nicht bei anderen Professoren ein schon einmal erteiltes Seminarthema nochmals bearbeiten.

Professoren können dasselbe Seminarthema „wiederverwenden“ – also dasselbe Thema auch mehreren Studenten erteilen.

Ein Thema kann von mehreren Professoren vergeben werden – aber an unterschiedliche Studenten.

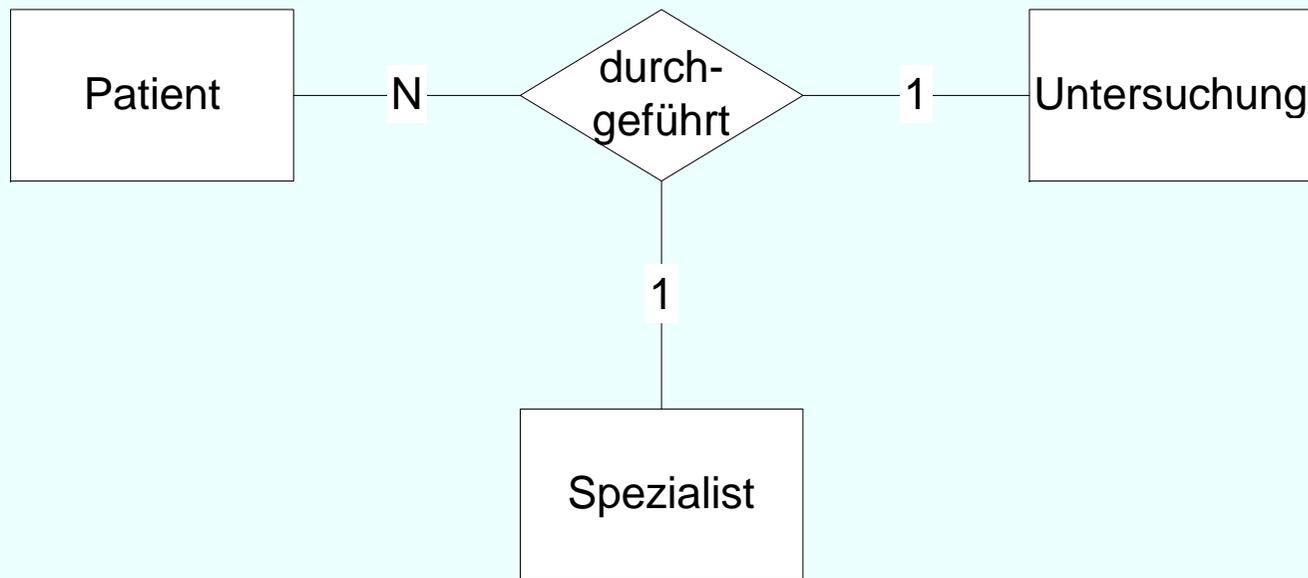
# Ausprägung der Beziehung *betreuen*



Gestrichelte Linien  
markieren illegale Ausprägungen

# Noch ein Beispiel

n-stellige Beziehung:

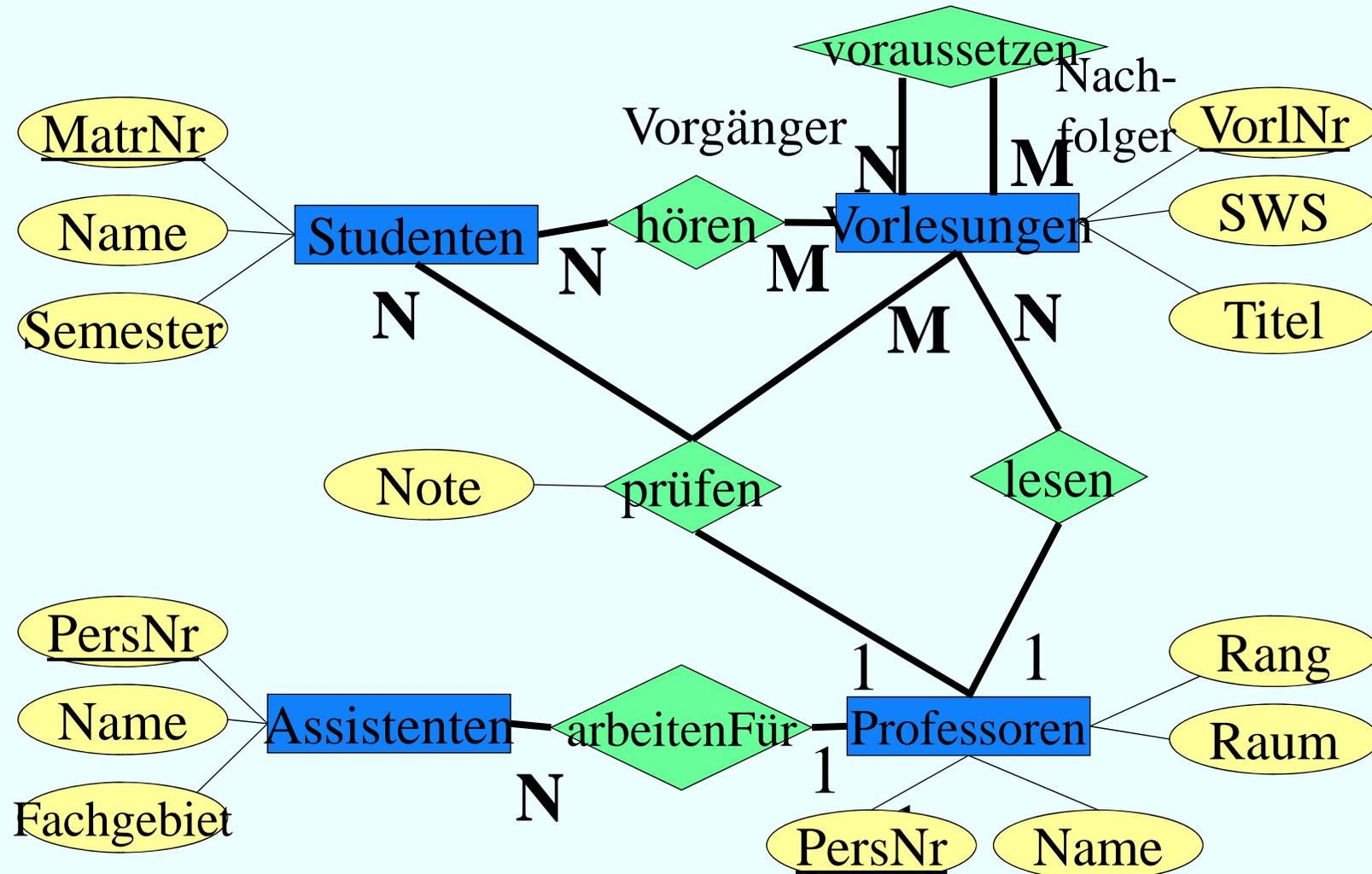


Eine Untersuchung wird von einem Spezialisten an mehreren Patienten durchgeführt

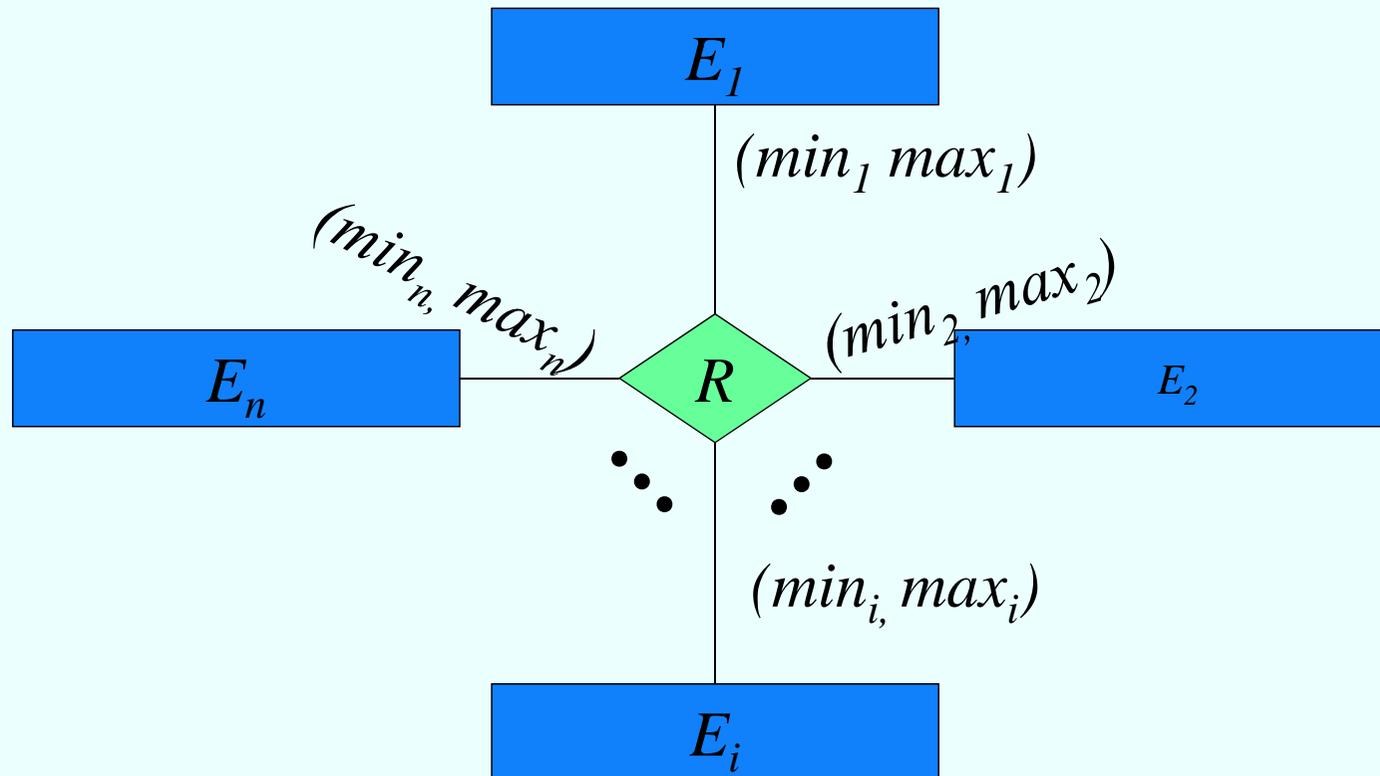
Ein Patient bekommt von einem Spezialisten nur eine Untersuchung

Eine Untersuchung wird an einem Patienten nur von einem Spezialisten durchgeführt

# Universitätsschema



# (min, max)-Notation



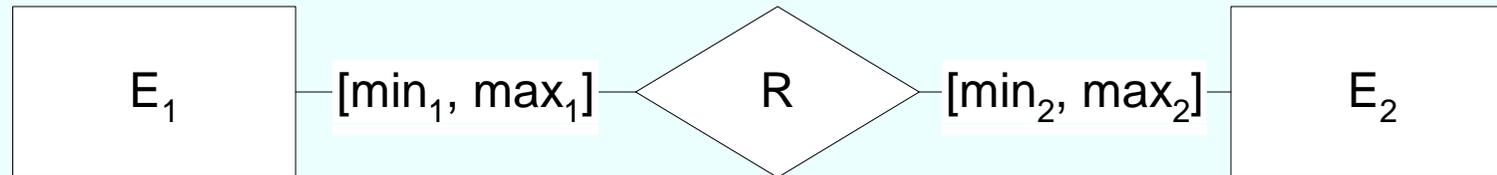
$$R \subseteq E_1 \times \dots \times E_i \times \dots \times E_n$$

Für jedes  $e_i \in E_i$  gibt es

- Mindestens  $min_i$  Tupel der Art  $(\dots, e_i, \dots)$  und
- Höchstens  $max_i$  viele Tupel der Art  $(\dots, e_i, \dots) \in R$

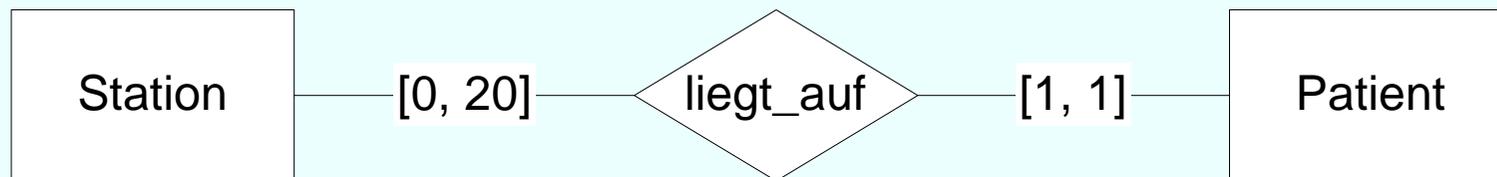
# Beispiel (min, max)

- Kardinalitätsrestriktionen:



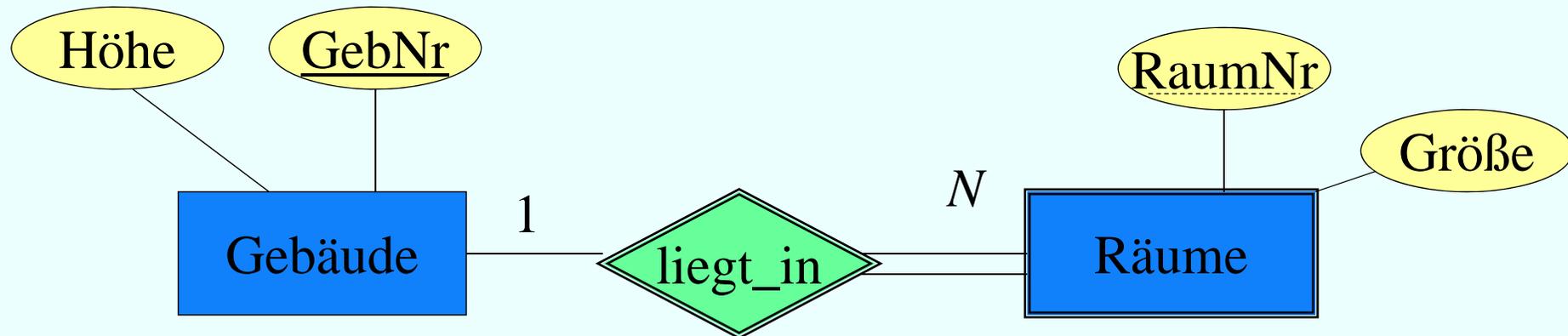
$e_1$  nimmt an  $[min_1, max_1]$  Beziehungen vom Typ  $R$  teil  
 $e_2$  nimmt an  $[min_2, max_2]$  Beziehungen vom Typ  $R$  teil

## Beispiel:



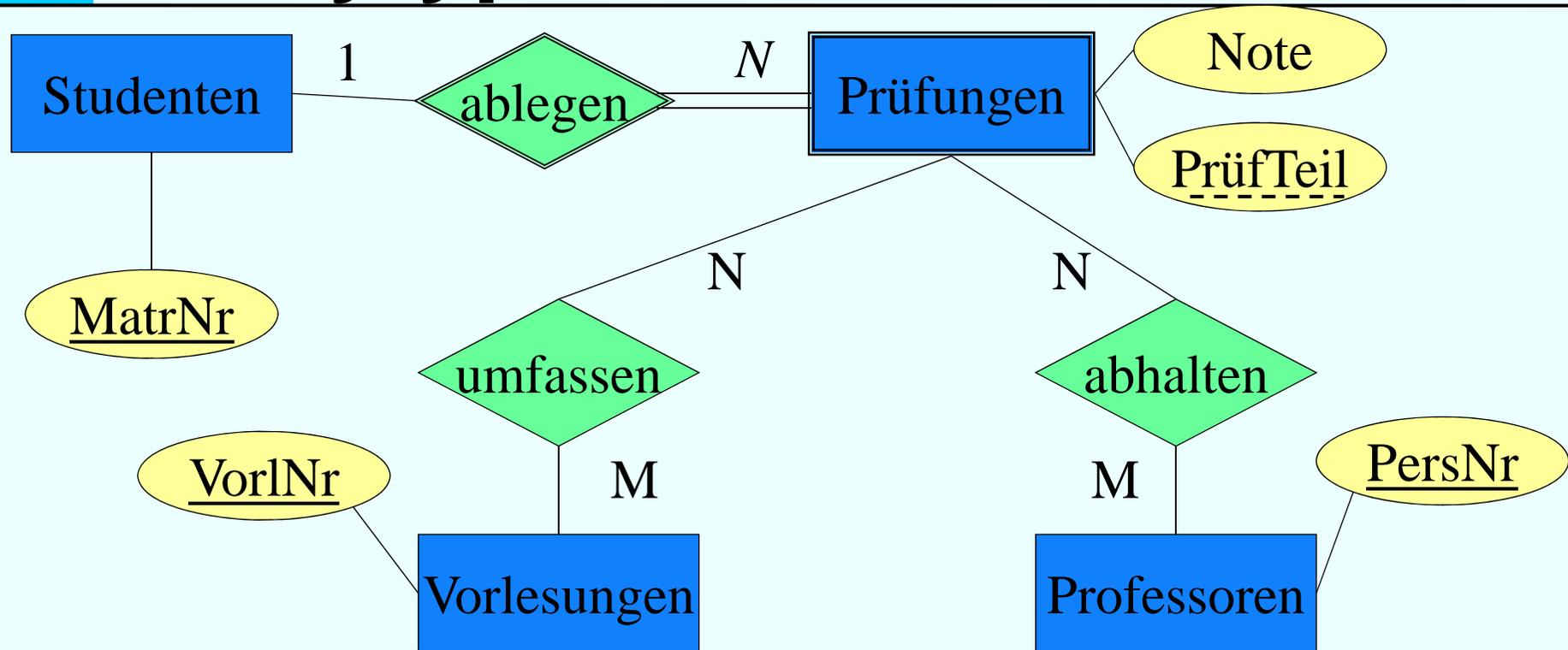
auf einer Station liegen 0 – 20 Patienten  
ein Patient liegt auf genau einer Station

# Schwache, existenzabhängige Entities



- Beziehung zwischen "starken" und schwachem Typ ist immer 1:N (oder 1:1 in seltenen Fällen)
- Warum kann das keine N:M-Beziehung sein?
- RaumNr ist nur innerhalb eines Gebäudes eindeutig
- Schlüssel ist: GebNr **und** RaumNr

# Prüfungen als schwacher Entitytyp



- Mehrere Prüfer in einer Prüfung
- Mehrere Vorlesungen werden in einer Prüfung abgefragt