



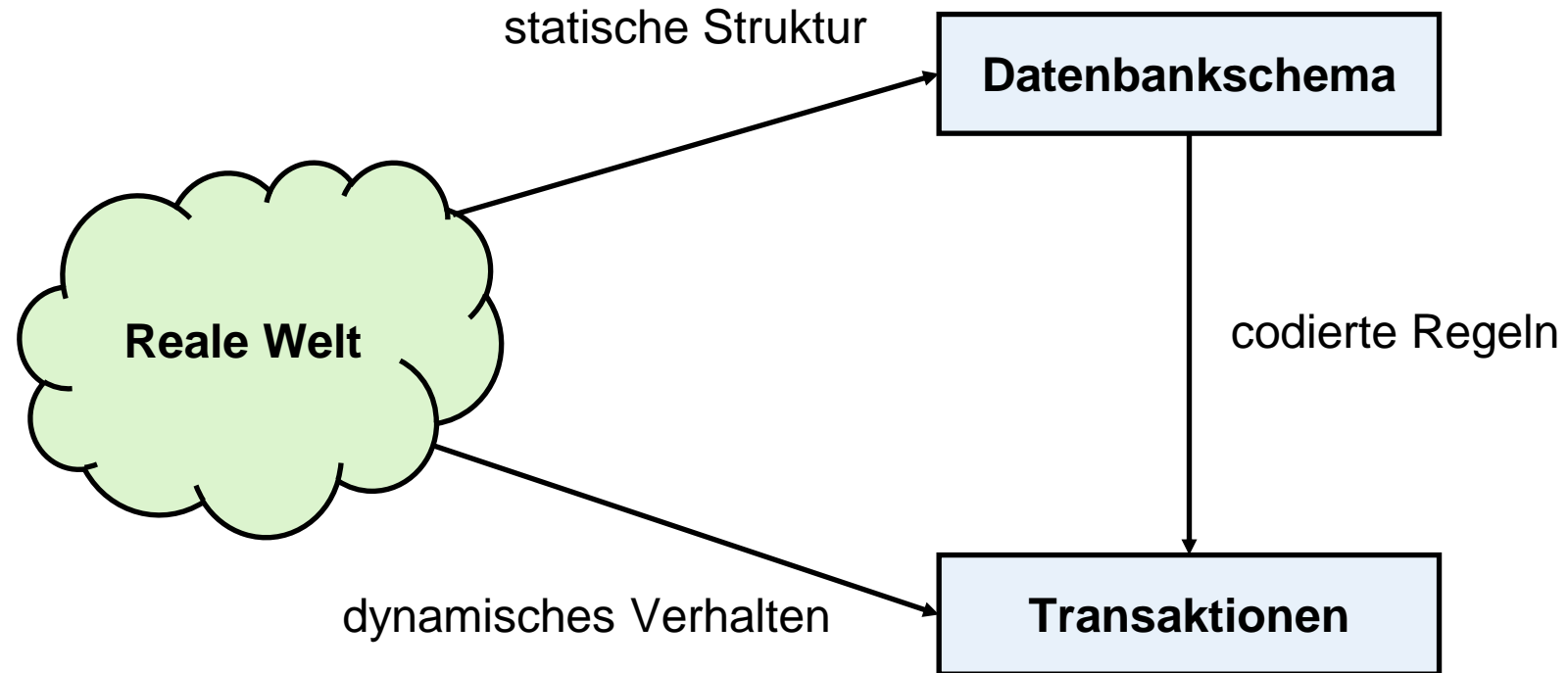
## 2. Das Entity-Relationship-Modell

1. Der Datenbankentwurf
2. Entity-Relationship-Modell
3. Konzeptueller Entwurf



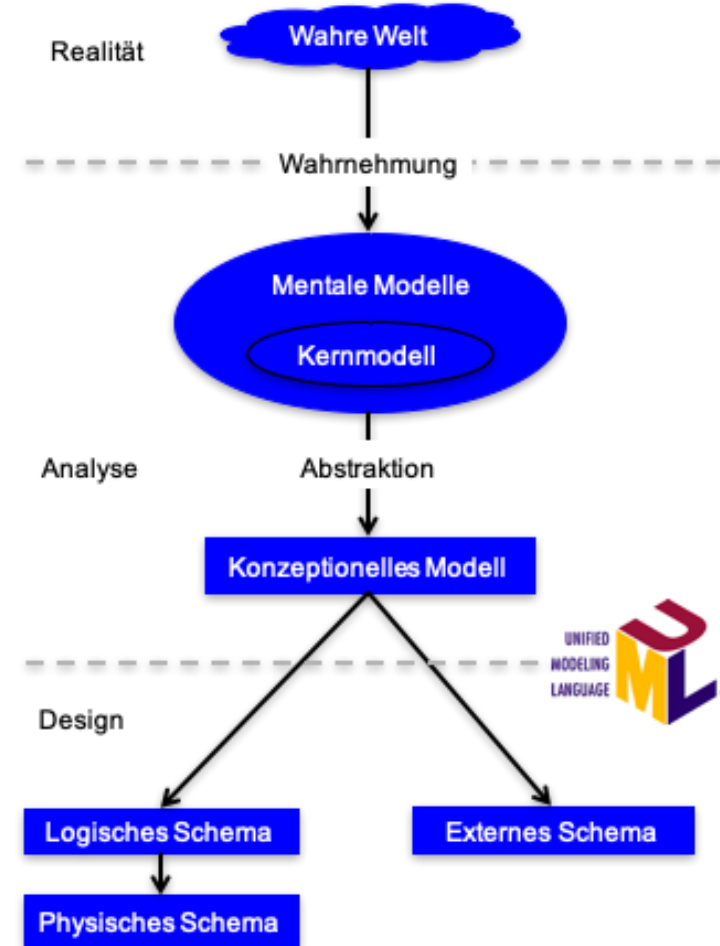
## 2. Das Entity-Relationship-Modell

1. **Der Datenbankentwurf**
2. Entity-Relationship-Modell
3. Konzeptueller Entwurf



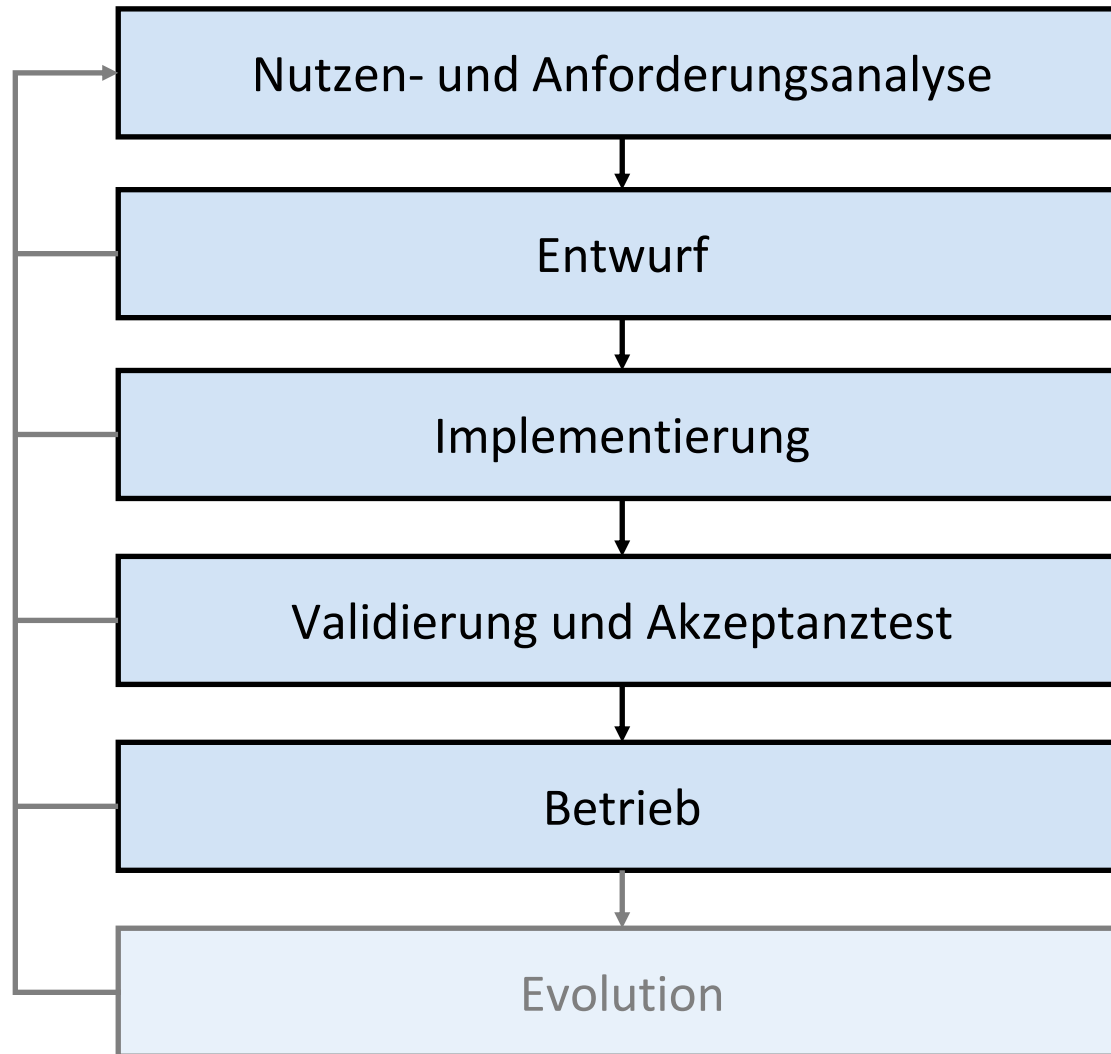


# Modellierung von Datenbanken



- **Definition: *Datenbankentwurf* [Vossen]**
  - „Die Aufgabe des Datenbankentwurfs ist der Entwurf der *logischen (konzeptuellen)* und *physischen Struktur* einer Datenbank so, dass die *Informationsbedürfnisse* der Benutzer in einer Organisation für *bestimmte Anwendungen adäquat* befriedigt werden können.“
  
- **Datenbankentwurf**
  1. Entwurf der logischen (konzeptuellen) Struktur der Datenbank.
  2. Entwurf der physischen Struktur der Datenbank.
  3. Angemessen für bestimmte, wohl umrissene Anwendungen der Datenbank

# Informationssystem-Lebenszyklus



- **Nutzenanalyse**

- Ermittlung potentieller Anwendungsgebiete des Systems
- Kosten-Nutzen-Rechnungen für die Anwendungen
- Es werden Prioritäten für die Anwendungsbereiche gesetzt

- **Anforderungsanalyse**

- Wichtigste Phase
  - Beeinflusst alle anderen Phasen
  - läuft parallel zur Nutzenanalyse
- Im Allgemeinen in *Zusammenarbeit mit potentiellen Benutzern*, um Probleme und Anwendungen umfassend verstehen zu können
- Anforderungen der gewählten Anwendungen werden ermittelt
  - Informationsanforderungen
  - Bearbeitungsanforderungen
    - Möglichkeiten (quantitativ, qualitativ)
  - Benutzeranforderungen
    - Gute Schnittstelle, Ähnlichkeit zu anderen bekannten (Papier-)Formaten, mögliche/unterstützte Operationen, Dokumentation, Erklärungs-/Hilfskomponenten

- **Entwurf**

- Entwurf der Anwendungen, die die Datenbank verwenden sollen
- Entwurf der Datenbank als zentrales Informations-Reservoir
- Berücksichtigung von physikalischen Randbedingungen wie Latenz, Bandbreite
- Konzeptueller Entwurf
  - Unabhängig von Zieldatenmodell
- Logischer Entwurf
  - Übersetze Konzepte in konkretes Datenbankschema

- **Implementierung**

- Implementierung anhand des Entwurfs
- Transformierung der Modelle in physische Strukturen
  - Datenstrukturen, Zugriffsstrukturen



- **Validierung und Akzeptanztest**
  - Schlüsselkriterium für den Erfolg
  - Validierung der Implementierung anhand der Anforderungsanalyse
    - Das System muss die Benutzer-Anforderungen erfüllen (funktionale Eigenschaften)
    - Das System muss gegebene Performance-Vorgaben erfüllen (nicht-funktionale Eigenschaften)
- **Betrieb**
  - Das System kann in Betrieb genommen werden
  - Eventuelle Konvertierung alter Anwendungen auf das neue System
  - Überwachung und Verbesserung der System-Performance
- **Evolution**
  - Fortentwicklung des Systems
  - Rückgriffe auf frühere Phasen können notwendig werden

- **Korrektheit**

- Korrekte Verwendung der Konzepte des Datenmodells
  - Sprache zur Schemabeschreibung wird *syntaktisch* korrekt verwendet
  - Beim Entity-Relationship-Modell kann zwischen *syntaktischer* und *semantischer* Korrektheit unterschieden werden

- **Vollständigkeit**

- Ein Datenbankenwurf bzw. ein konzeptionelles Datenbankschema soll vollständig sein
  - Alle relevanten Aspekte und Eigenschaften des Anwendungsbereichs sind erfasst

- **Minimalität**

- Jeder Aspekt der Anforderungen sollte lediglich einmal im Schema vorkommen
- Redundanzvermeidung
  - Gewünschte und unvermeidbare Redundanzen werden dokumentiert

- **Lesbarkeit**

- Datenbankschema sollte selbsterklärend sein, d.h. wichtige Elemente und Konzepte sollten durch das Schema verstanden werden können
- Zusätzlich sollte es eine umfassende Dokumentation geben

- **Modifizierbarkeit/Anpassbarkeit**

- Anpassung an neue/andere Anforderungen
- Modifikation der Datenbank ist vergleichsweise leicht wenn die Datenbank modular aufgebaut ist und gut dokumentiert wurde

- **Normalisierung**

- Datenbankschema sollte gewissen Normen genügen
- Erstellung einer „übersichtlichen“ Struktur und Vermeidung von Redundanzen



## 2. Das Entity-Relationship-Modell

1. Der Datenbankentwurf
- 2. Entity-Relationship-Modell**
3. Konzeptueller Entwurf

# Entity-Relationship-Modell

---

- Das Entity-Relationship-Modell dient dazu, für einen zu modellierenden Ausschnitt der realen Welt ein konzeptionelles Schema zu erstellen und auf hohem Abstraktionsniveau graphisch darzustellen.
- Das Ergebnis ist ein *Entity-Relationship-Diagramm (ER Diagramm oder ER-Modell)*.
  - Das ER-Modell ist ein abstraktes, maschinenfernes Datenmodell
  - Überlegungen zur Effizienz und zur physischen Umsetzung spielen noch keine Rolle.
- Für die konkrete Implementierung eines Datenbankschemas muss das ER-Diagramm auf eines der maschinennäheren Modelle abgebildet werden (z.B. hierarchisches Modell, Netzwerkmodell, relationales Modell).
  - Für eine rudimentäre Transformation können einfache Regeln angegeben werden
  - Die Gewinnung eines *effizienten* Schemas erfordert ein tieferes Verständnis des Zielmodells

# Schemaentwurf

---

- Die generelle Aufgabe des Schemaentwurfs besteht darin, für den zu modellierenden Teil der “realen Welt” eine formale Beschreibung zu entwickeln. Dabei gibt es verschiedene Zwischenstufen:
  - Beschreibung durch natürliche Sprache (z.B. *Pflichtenheft, ...*)
  - Beschreibung durch abstrakte graphische Darstellungen (z.B. *ER-Diagramm, ...*)
  - Beschreibung durch konkrete, implementierbare Sprache (z.B. *im Relationalen Modell*)
- *Beispielanwendung*: Lehrbetrieb an der Fachgruppe Informatik
- Folgende Objekte spielen eine Rolle:
  - *Personen (genauer: Studierende, Professor/Professorinnen, Mitarbeiter/Mitarbeiterinnen, ...), Räume, Zeiten, Lehrveranstaltungen, Bücher, usw.*
- Diese Objekte der realen Welt sind nie voneinander isoliert, vielmehr stehen sie in verschiedenen Beziehungen zueinander, z.B.:
  - *veranstaltet, nimmt teil an, findet statt in, wohnt in*




# Entity-Relationship-Diagramme

---

- Das Entity-Relationship-Modell (ER-Modell) beschreibt die Welt durch:
  - **Objekttypen** (*Entity-Typen*) mit
  - **Eigenschaften** (*Attributes*) und
  - **Beziehungen** (*Relationships*) zueinander.
- Beim Schemaentwurf entstehen sogenannte *ER-Diagramme*, d.h. graphische Darstellungen des betrachteten Realitätsausschnittes.
- Eine entscheidende Aufgabe beim Entwurf eines Datenbankschemas besteht darin, *geeignete* Objekttypen, Attribute und Beziehungen zu bestimmen.
- Ein *ER-Diagramm* ist die graphische Darstellung eines konkreten Datenbankentwurfs im ER-Modell.
  - Innerhalb von Rechtecke, Ellipsen und Rauten werden im ER-Diagramm die Namen der jeweiligen Entity-Typen, Attribute bzw. Beziehungen notiert.

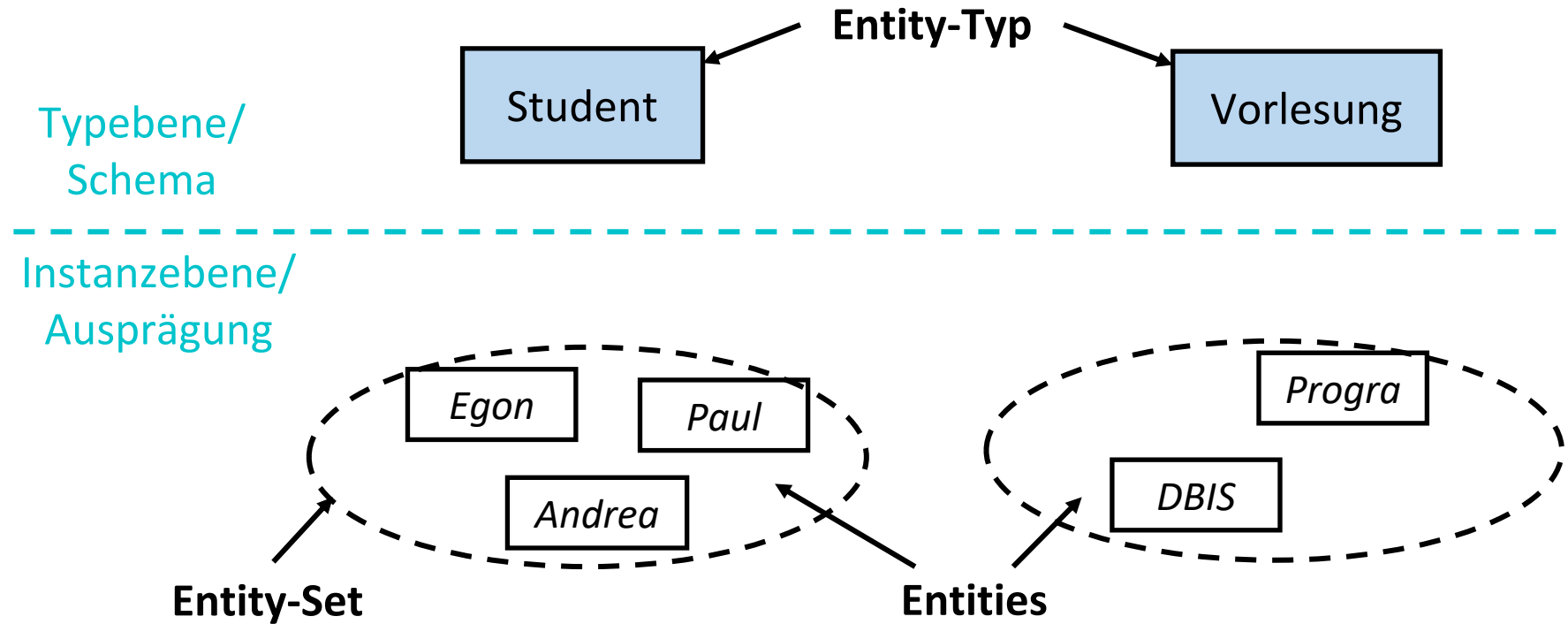
## Elemente des ER-Modells (Entity-Typen)

---

- **Entities:** Als *Entity* (= “Seiendes”, Objekt) bezeichnet man etwas, das existiert und unterscheidbar ist von anderen Entities. Beispiele für Entities sind *Person, Auto, Kunde, Buch, etc.*
  - Einzelne Entities des gleichen Typs sind unterscheidbar, d.h. sie haben eine *Objektidentität*.
- Diese Eigenschaften zeigen, dass das ER-Modell eine Urform des objektorientierten Modellierens ist.
- **Entity-Typ** (Objekttyp) ist eine Menge von Entities mit den selben *Attributen* z.B: *Personen, Autos, Kunden, etc.*
  - Entity-Typen werden im ER-Diagramm durch *Rechtecke* dargestellt: 
- **Entity-Set** (Objektmenge) ist eine Menge von Entities eines bestimmten Entity-Typ zu einem Zeitpunkt. Ein Entity-Set wird normalerweise mit dem gleichen Namen wie der Entity-Typ referenziert.

# Elemente des ER-Modells (Entity-Typen)

Beispiel:



## Elemente des ER-Modells (Attribute und Schlüssel)

---

- **Attribute**

- Alle Mitglieder eines Entity-Typs werden durch eine Menge charakterisierender Eigenschaften (*Attribute*) beschrieben. Beispiele dafür sind “Farbe”, “Gewicht” und “Preis” beim Entity-Typ <Teile>. Die Werte eines Attributes stammen normalerweise aus Wertebereichen wie INTEGER, REAL, STRING, etc. Aber auch strukturierte Werte wie Listen, Bäume, usw. sind vorstellbar.

- Attribute werden im ER-Diagramm durch *Ellipsen* dargestellt:



- Attribute sind über ungerichtete Kanten mit dem zugehörigen Entity-Typ verbunden.
- Schlüssel-Attribute werden (in der Regel) unterstrichen.
- Entity-Typen, an die nur ein Attribut gebunden ist, werden oft nur über die Attribut-Ellipse repräsentiert und erhalten den Namen des Attributes

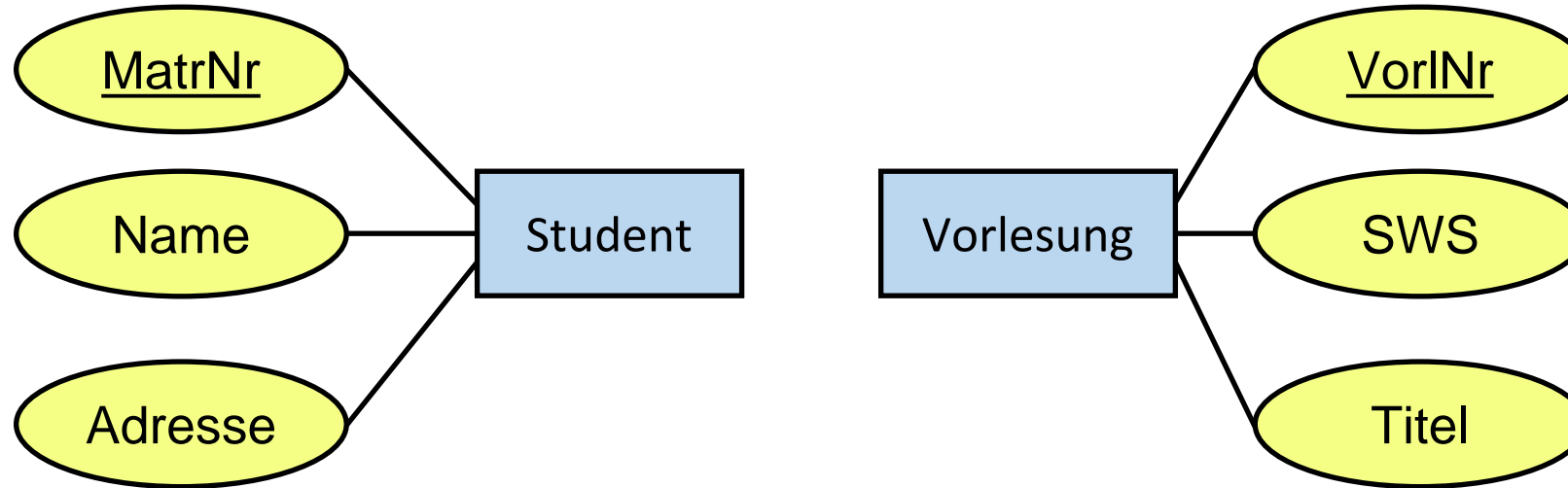
# Elemente des ER-Modells (Attribute und Schlüssel)

---

- **Schlüssel**
- Eine minimale Menge von Attributen, deren Werte das zugeordnete Entity eindeutig innerhalb aller Entities seines Typs identifizieren, nennt man *Schlüssel*.
  - Gibt es verschiedene minimale und identifizierende Attributmengen als Schlüsselkandidaten, dann wählt man einen dieser Kandidaten-Schlüssel als *Primärschlüssel* aus.
  - Häufig wird einem Typ ein zusätzliches Attribute als Schlüssel hinzugefügt (*Kunstschlüssel*), z.B. Personalnummer, Vorlesungsnummer, etc.
  - Attribute, die den Primärschlüssel bilden, werden durch Unterstreichung gekennzeichnet.
- *Beispiele:*
- Typ Student mit Attributen (MatrNr, Name, Adresse)
- Typ Vorlesung mit Attributen (VorlNr, SWS, Titel)
- Typ Leistungsnachweis mit Attributen (MatrNr, VorlNr, Note)

# Elemente des ER-Modells (Attribute und Schlüssel)

Beispiele:



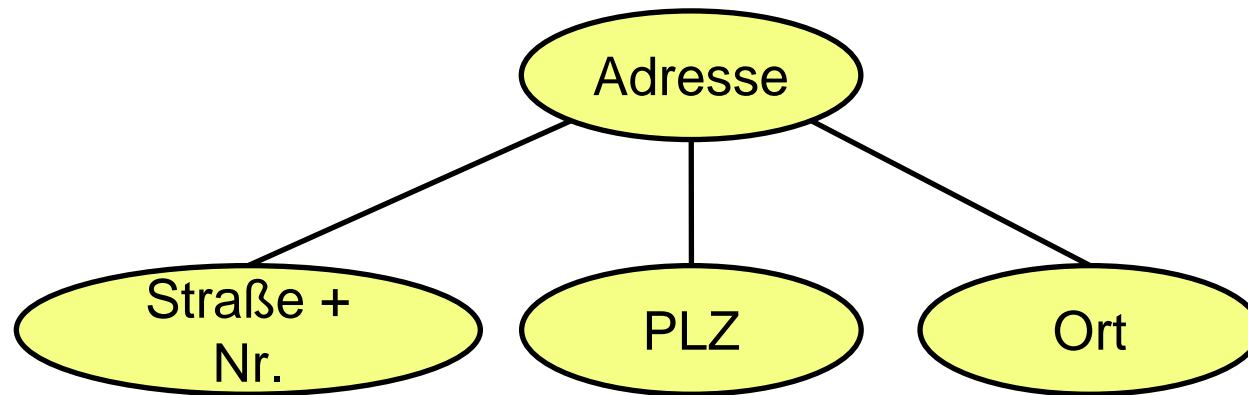
- Entity-Typ *Student* mit Attributen MatrNr (Schlüsselattribut), Name und Adresse
- Entity-Typ *Vorlesung* mit Attributen VorlNr (Schlüsselattribut), SWS und Titel



## Elemente des ER-Modells (Zusammengesetzte Attribute)

---

- Ein Attribut kann aus anderen Attributen bestehen
  - Diese Unterattribute werden mit Linien zu dem Attribut verknüpft

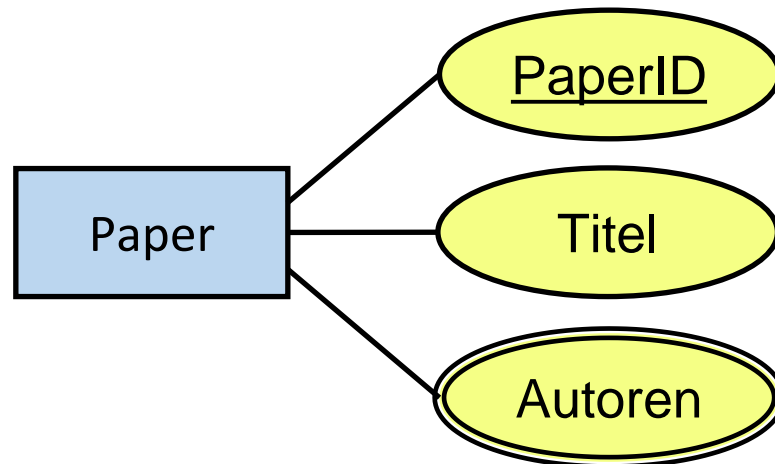


- Das Attribut *Adresse* besteht aus den Unterattributen *Straße + Nr.*, *PLZ* und *Ort*

## Elemente des ER-Modells (Mehrwertige Attribute)

---

- Ein (mehrwertiges) Attribut kann eine Menge von Werten enthalten
  - Die bisherigen Attribute waren einwertig
  - Ein mehrwertiges Attribut wird als *Ellipse* mit *doppelter Umrandung* dargestellt



- Ein *Paper* hat eine *PaperID* (Schlüsselattribut) und einen *Titel*, aber kann mehrere *Autoren* haben

## Elemente des ER-Modells (Relationship)

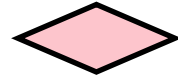
---

- **Relationships (Beziehungen)**
- Über *Relationships* (Beziehungen) lassen sich Zusammenhänge zwischen Entity-Typen darstellen.
  
- *Beispiele:*
  - <Vorlesung> *setzt voraus* < Vorlesung >
  - <Vorlesung > *findet statt in* <Raum>

## Elemente des ER-Modells (Relationship)

---

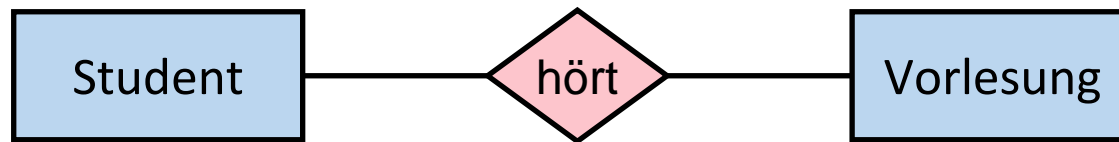
- Beziehungen werden im ER-Diagramm durch *Rauten* dargestellt:



- Beziehungen werden mit den entsprechenden Entity-Typen durch Kanten verbunden.
- Eine Beziehung kann auch Attribute haben

*Beispiel:*

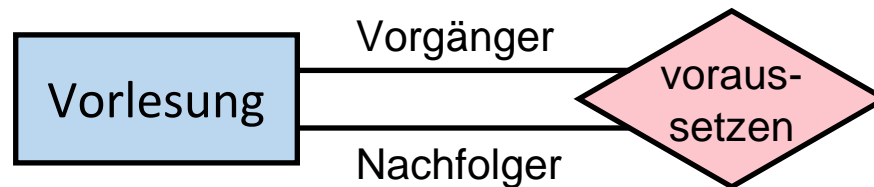
- *Student* (Entity-Typ) *hört* (Beziehung) *Vorlesung* (Entity-Typ)



## Elemente des ER-Modells (Rekursive Beziehung)

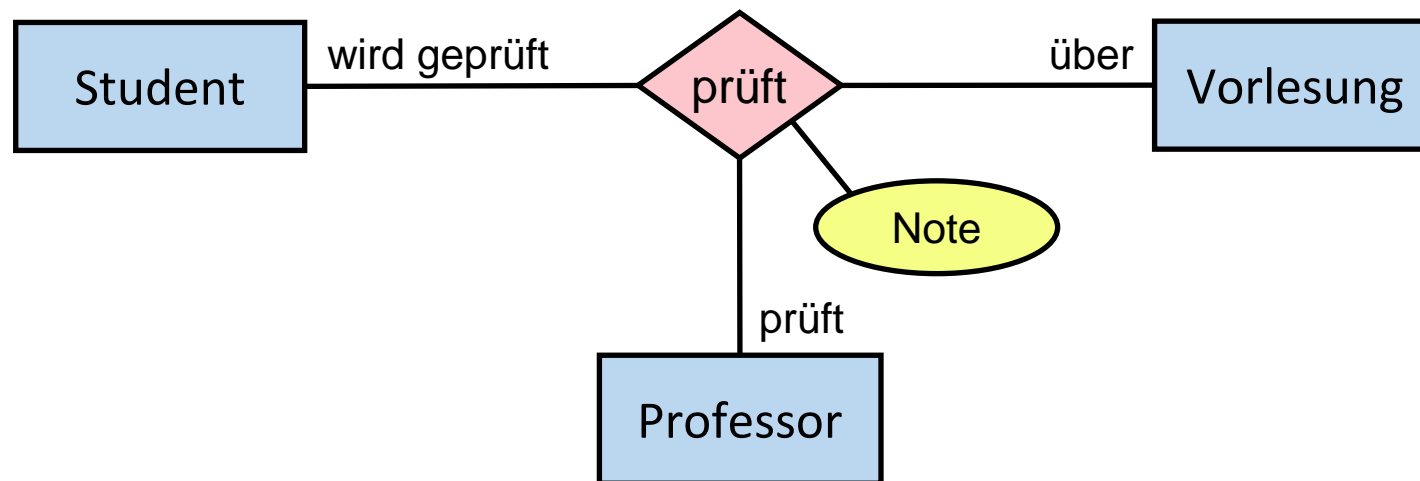
---

- Eine *Rekursive Beziehung* verbindet einen gleichen Entity-Typen mehrfach
  - Tritt ein Entity-Typ in einer Beziehung mehrfach auf, so hat es dort unterschiedliche *Rollen*
- *Beispiel: <Vorlesung> setzt voraus <Vorlesung>*
  - Der Objekttyp *Vorlesung* tritt in den beiden Rollen ‘Vorgänger’ und ‘Nachfolger’ auf.



## Elemente des ER-Modells (n-stellige Beziehung)

- Zusätzlich zu zweistelligen Beziehungen gibt es auch mehrstellige Beziehungen
  - Der Spezialfall der mehrstelligen Beziehung, die drei Entity-Typen verbindet nennt man auch *ternäre* Beziehung
- *Beispiel (ternäre Beziehung):*
- <Student> wird von <Professor> über <Vorlesung> *geprüft* und erhält *Note*





## Elemente des ER-Modells (Relationship)

---

- *Formal:*

- Eine Beziehung  $R(E_1, E_2, \dots, E_n)$  für  $n \geq 1$  Entity-Typ  $E_i$  kann als Relation im mathematischen Sinne aufgefasst werden, d.h.  $R \subseteq E_1 \times E_2 \times \dots \times E_n$ . Ein bestimmter Entity-Typ darf mehrfach vorkommen, es sind zwei- und mehrstellige Beziehungen möglich.
- Eine Ausprägung einer Beziehung ist eine endliche Menge von  $n$ -Tupeln  $(e_1, \dots, e_n) \in R$ , d.h.  $e_i \in E_i$  für  $i = 1, \dots, n$ .

## Elemente des ER-Modells (Ausprägung einer ternären Beziehung)

---

- Beispiel Beziehung:  
    <Produkt> *wird geliefert von* <Lieferant> an <Filiale>
- Ternäre Beziehung zwischen <Produkt>, <Lieferant>, <Filiale>
- Ausprägung:
  - *wird geliefert von an* (Milch, Huber, Lehel)
  - *wird geliefert von an* (Milch, Meier, Lehel)
  - *wird geliefert von an* (Milch, Meier, Altstadt)

# Beispiel: ER-Diagramm (Universität)

## Entity Typen

---

Student

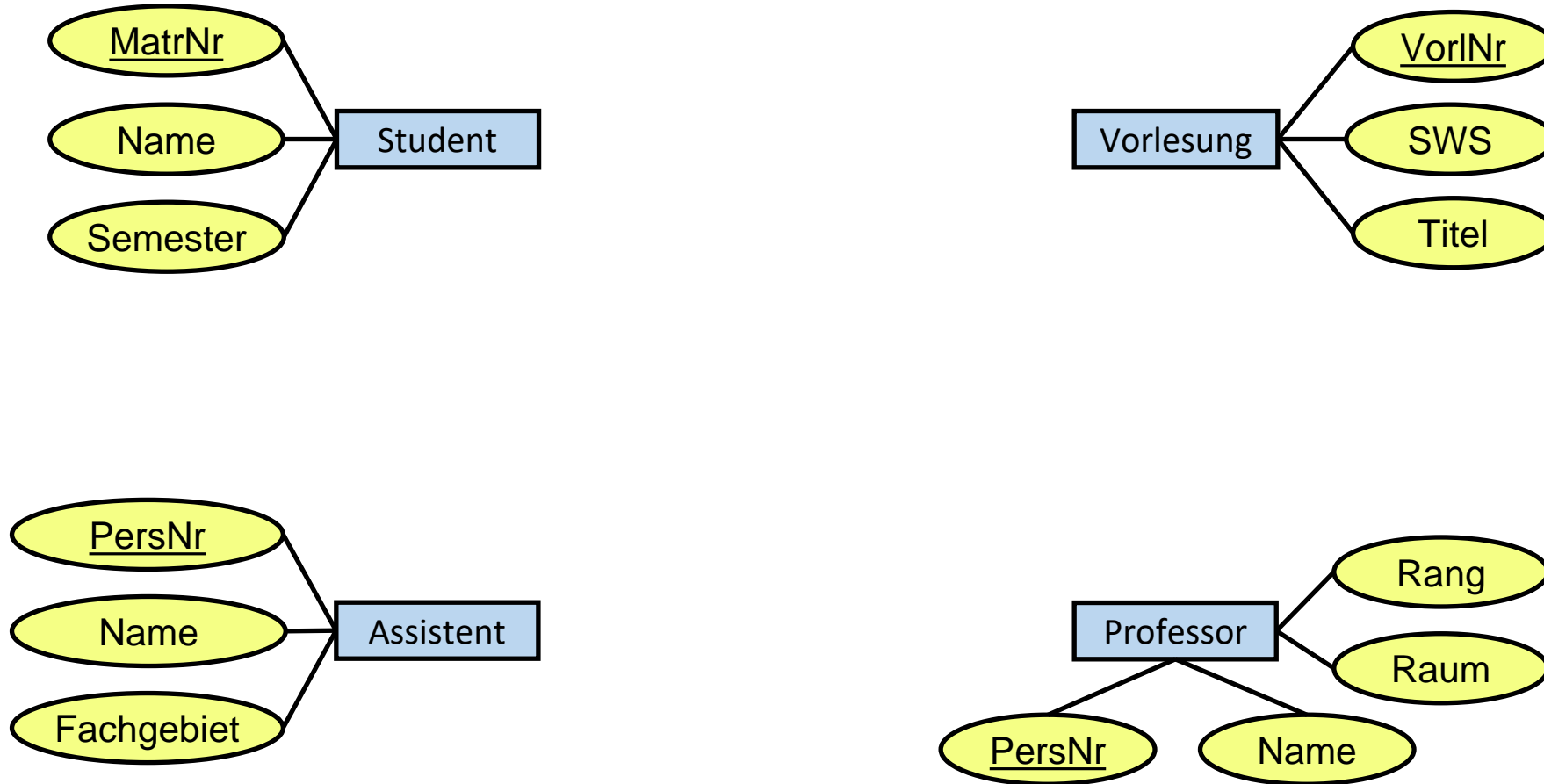
Vorlesung

Assistent

Professor

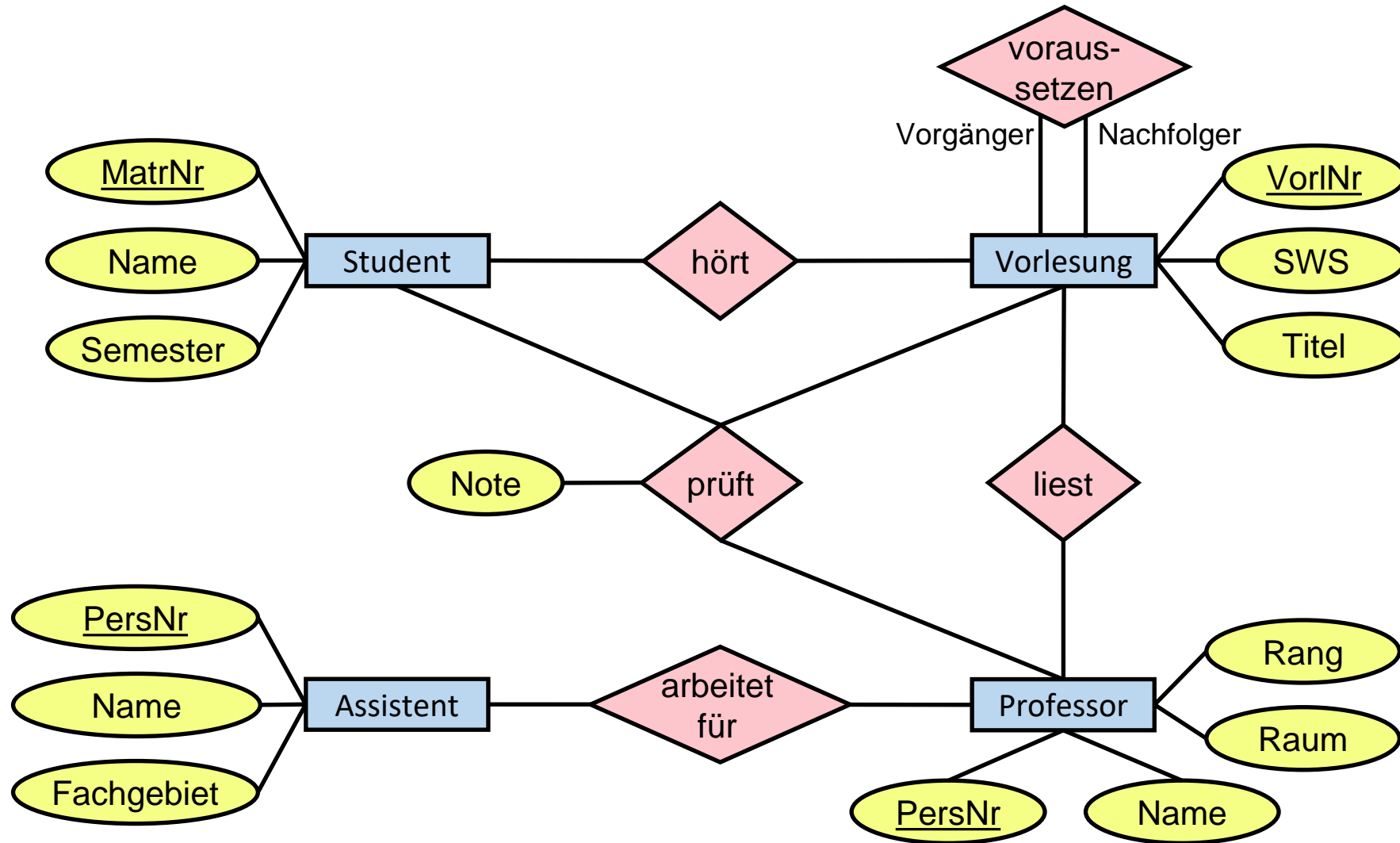
# Beispiel: ER-Diagramm (Universität)

## Entity Typen + Attribute



# Beispiel: ER-Diagramm (Universität)

## Entity Typen + Attribute + Beziehungen



## Elemente des ER-Modells (Kardinalitäten)

---

- Fragestellung: “Wie viele Entitäten können über eine Beziehung einer Entität zugeordnet werden?”
- Beispiele:
  - Wie viele Assistenten-Entities können über die *arbeitet für* Beziehung mit einer Professor-Entity in Beziehung stehen?
  - Wie viele Professoren-Entities kann die *liest* Beziehung einer einzelnen Vorlesungs-Entity zuordnen?
- Die Antwort hängt davon ab, was modelliert wird.
- Am einfachsten mit binären Beziehungen zu modellieren.

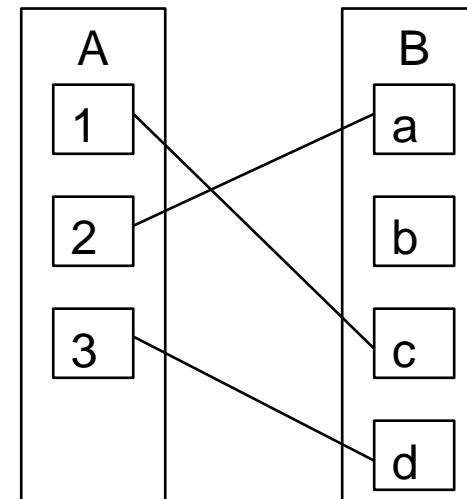
# Elemente des ER-Modells (Kardinalitäten)

Gegeben:

- Entity-Typen A and B
- Binäre Beziehung R zwischen A und B

## 1:1-Beziehung (one-to-one)

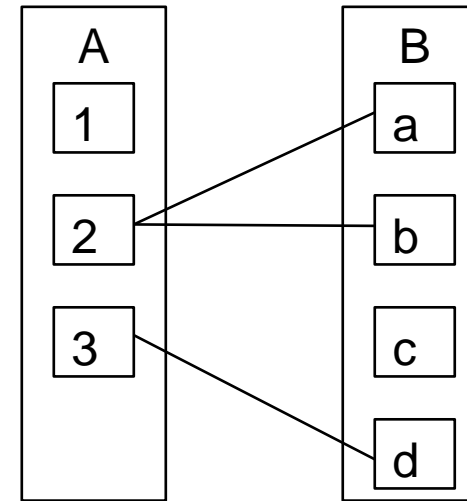
- Jede Entity in A ist höchstens mit einer Entity aus B in Beziehung
- Jede Entity in B ist höchstens mit einer Entity aus A in Beziehung



## Elemente des ER-Modells (Kardinalitäten)

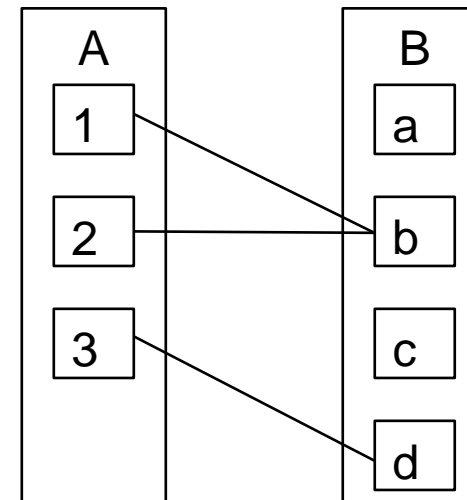
### 1:m-Beziehung (one-to-many)

- Jede Entity aus A steht mit null oder mehr Entities aus B in Beziehung.
- Jede Entity aus B steht mit höchstens einer Entity aus A in Beziehung



### m:1-Beziehung (many-to-one)

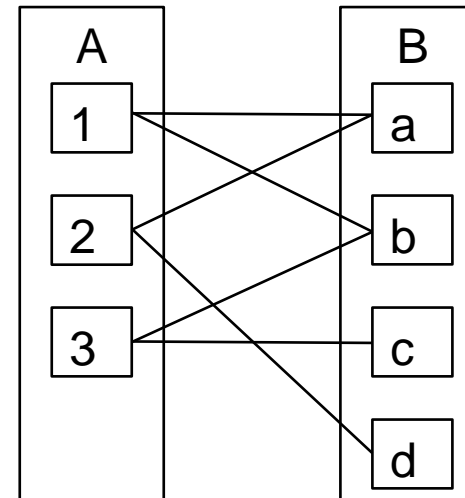
- Analog zu 1:m
- Jede Entity aus A steht mit höchstens einer Entity aus B in Beziehung
- Jede Entity aus B steht mit null oder mehr Entities aus A in Beziehung.





## m:n-Beziehung (many-to-many)

- Jede Entity aus A steht mit null oder mehr Entities aus B in Beziehung
- Jede Entity aus B steht mit null oder mehr Entities aus A in Beziehung



Erweiterung auf n-stellige Beziehungen:

- **m:1-Beziehung (many-to-one)**

- Binäre Beziehungen: Eine Beziehung  $R(E_1, E_2)$  “von  $E_1$  nach  $E_2$ ”, bei der jedes Entity aus  $E_1$  zu höchstens einem Entity aus  $E_2$  in Beziehung steht (nicht notwendig umgekehrt), heißt *m:1-Beziehung*.
- N-stellige Beziehungen: Eine Beziehung  $R(E_1, \dots, E_j, \dots, E_n)$  heißt “*m:1* von  $E_1, \dots, E_{j-1}, E_{j+1}, \dots, E_n$  nach  $E_j$ ”, falls jede Auswahl von Entities aus  $E_1, \dots, E_{j-1}, E_{j+1}, \dots, E_n$  höchstens ein Entity in  $E_j$  bestimmt.

- Es besteht eine funktionale Abhängigkeit.

Deswegen wird die Kardinalität auch als Funktionalität bezeichnet.

## Elemente des ER-Modells (Kardinalitäten)

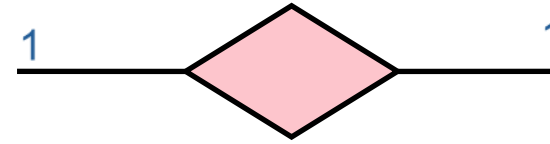
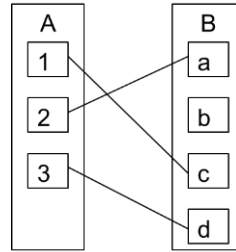
---

- Welche Kardinalität ist für eine Beziehung am besten geeignet?
  - Die Antwort hängt davon ab, was modelliert werden soll!
  - Man könnte einfach überall m:n Beziehungen verwenden, aber das wäre nicht klug.
- Ziel:
  - Die Kardinalität sollte reflektieren, was zulässig sein sollte.
  - Die Datenbank kann diese Einschränkungen automatisch erzwingen.
  - Ein gutes Datenbankdesign reduziert oder eliminiert die Möglichkeit, falsche Daten zu speichern.

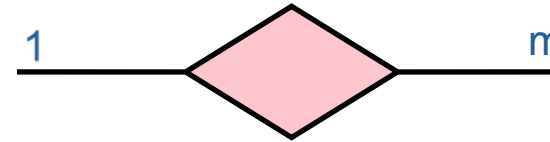
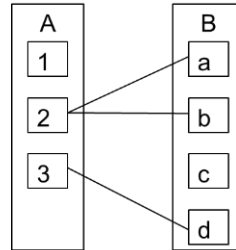
# Elemente des ER-Modells (Kardinalitäten)

## m:n Notation

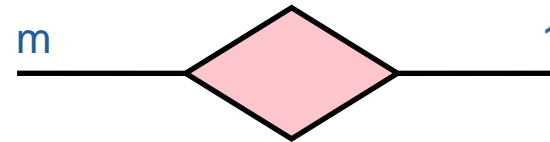
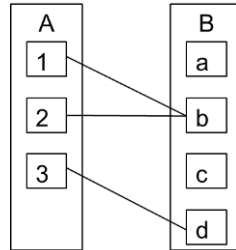
1:1-Beziehung (one-to-one)



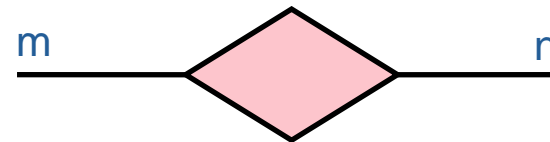
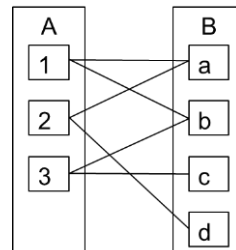
1:m-Beziehung (one-to-many)



m:1-Beziehung (many-to-one)



m:n-Beziehung (many-to-many)

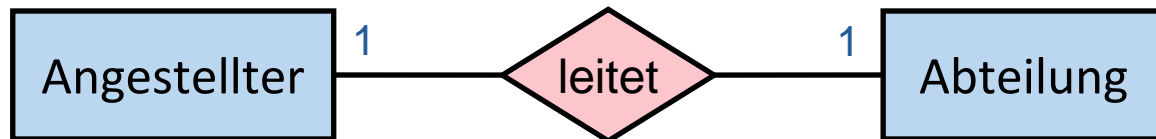


# Elemente des ER-Modells (Kardinalitäten)

---

## Beispiele

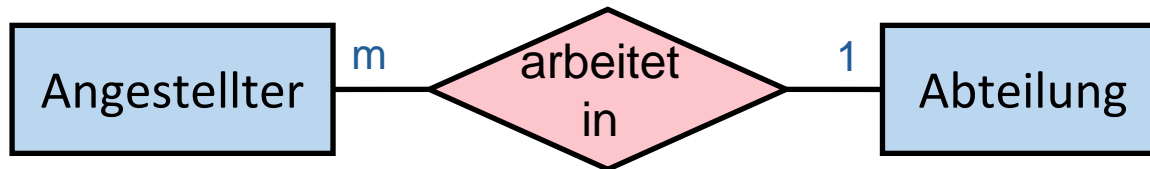
- <Abteilung> *wird geleitet von* <Angestellter>
  - 1:1-Beziehung unter der Annahme, dass jede Abteilung genau einen Leiter hat und kein Angestellter mehr als eine Abteilung leitet.



## Elemente des ER-Modells (Kardinalitäten)

*Beispiele:*

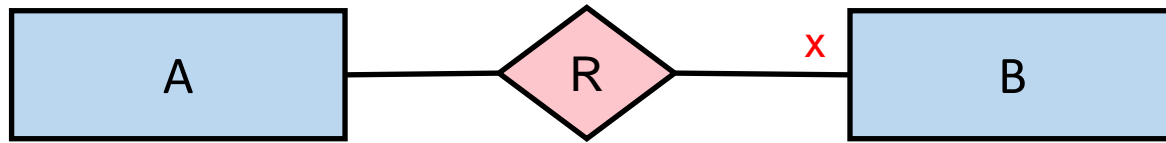
- <Angestellter> *arbeitet in* <Abteilung>
  - m:1-Beziehung unter der Annahme, dass jeder Angestellte in genau einer Abteilung arbeitet.



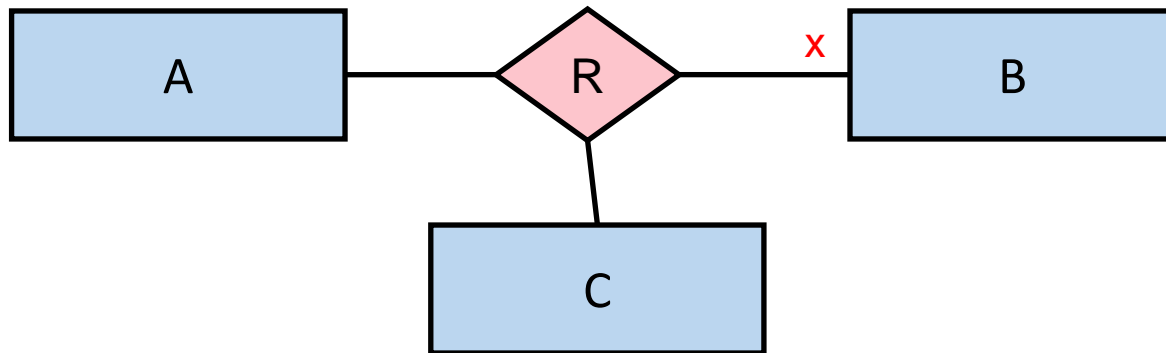
- <Student> *hört* <Vorlesung>
  - m:n-Beziehung, weil Studierende im allgemeinen mehrere Vorlesungen hören und Vorlesungen in der Regel von mehreren Studierenden besucht werden.



## Kardinalitäten für zwei- und mehrstellige Beziehungen: 1:n-Notation



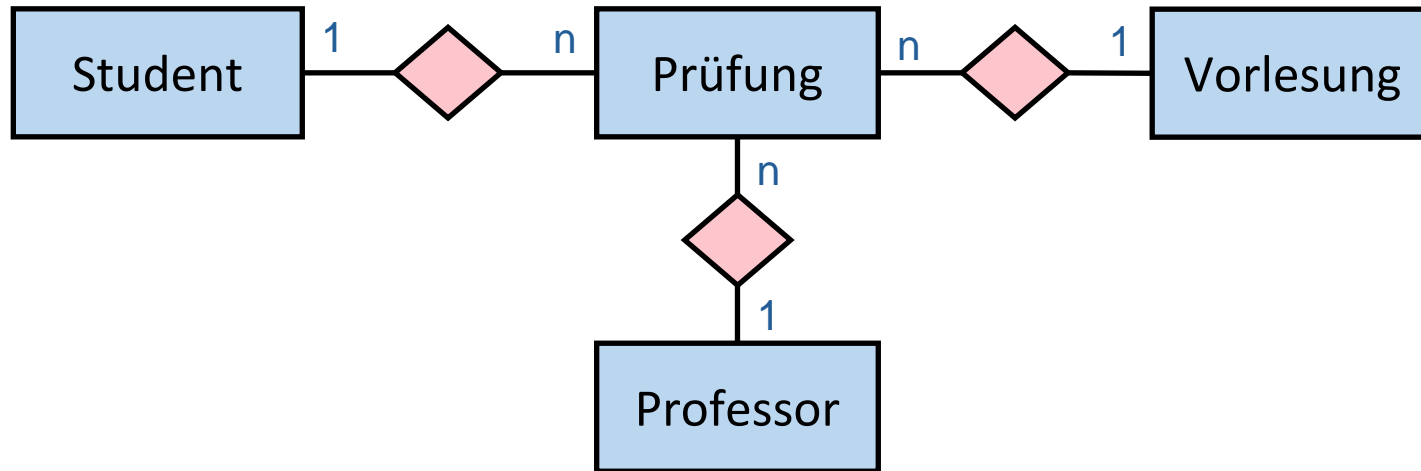
- **x**: Wie viele Entities des Entity Typ B stehen in Beziehung zu einer Entity des Entity Typ A?



- **x**: Wie viele Entities des Entity Typ B stehen in Beziehung zu einem (a,c)-Paar mit  $a \in A$  und  $c \in C$ ?

## Elemente des ER-Modells (Kardinalitäten)

- Eine ternäre Beziehung kann als binäre Beziehungen dargestellt werden
  - Damit die Semantik weitestgehend erhalten bleibt ist ein neuer Entity-Typ notwendig



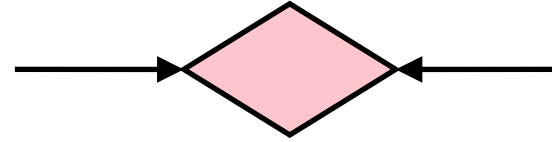
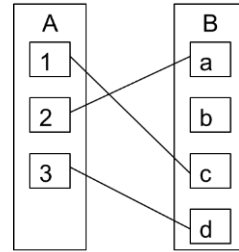
- Aber: Die Einschränkung, dass Studenten eine Vorlesung nur bei einem Professor prüfen lassen können, ist nicht darstellbar



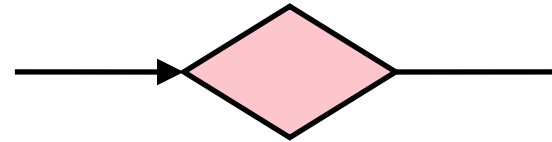
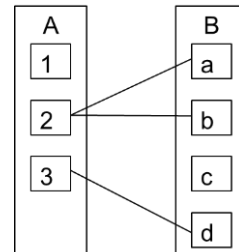
# Elemente des ER-Modells (Kardinalitäten)

## Pfeilnotation

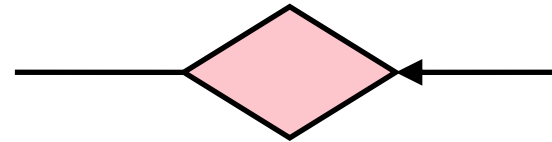
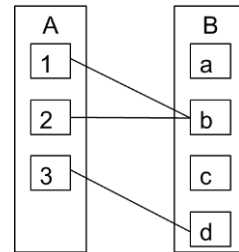
1:1-Beziehung (one-to-one)



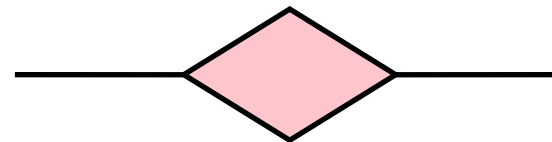
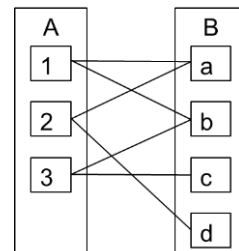
1:m-Beziehung (one-to-many)



m:1-Beziehung (many-to-one)

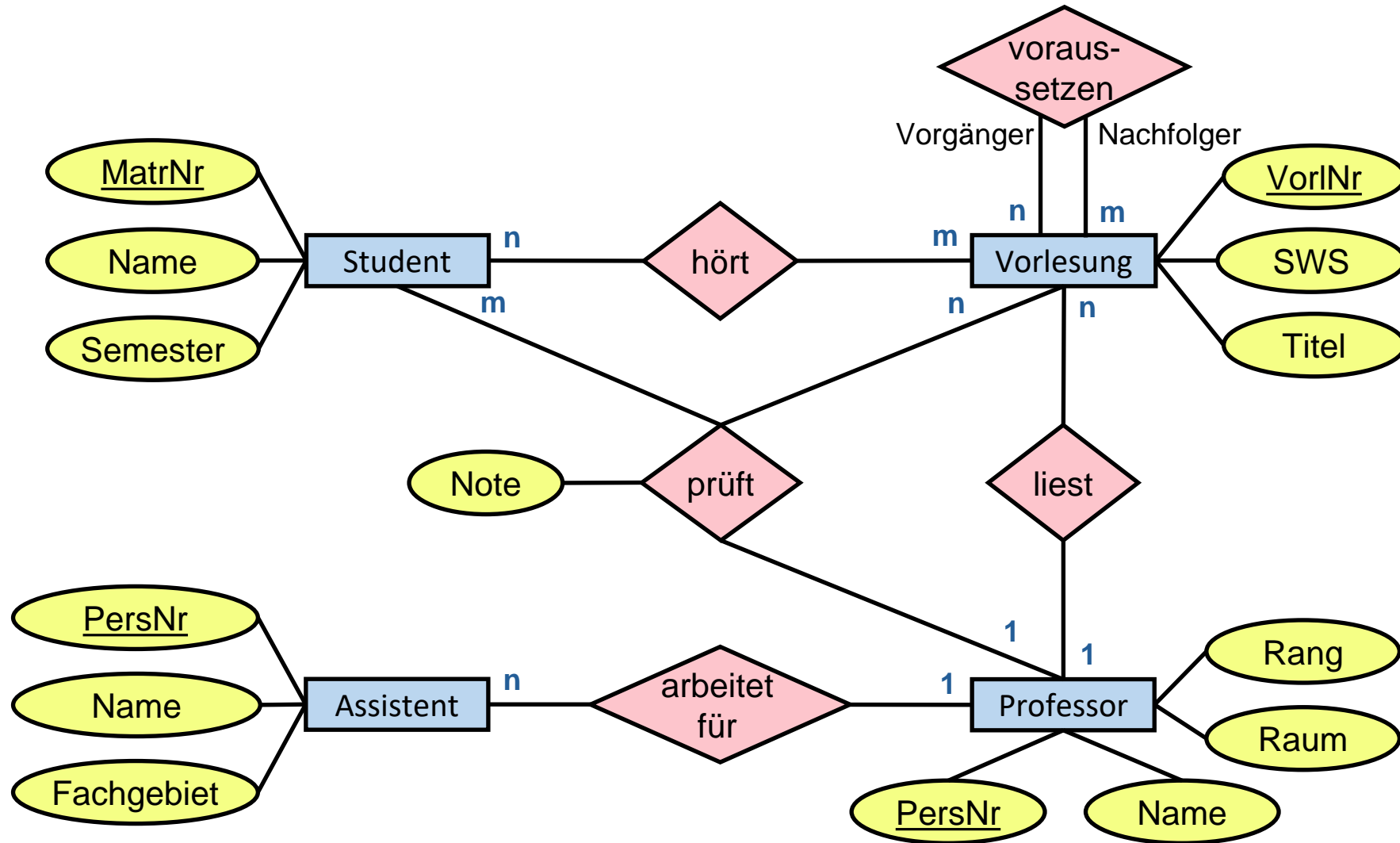


m:n-Beziehung (many-to-many)



# Beispiel: ER-Diagramm (Universität)

## Entity Typen + Attribute + Beziehungen + Kardinalitäten



## Elemente des ER-Modells (Kardinalitäten)

---

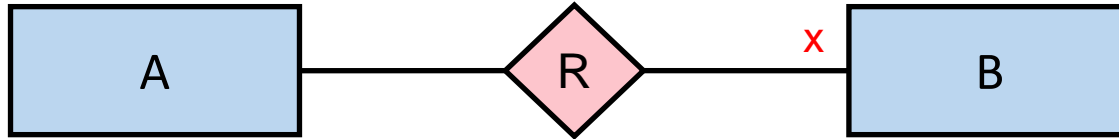
- Alternative Notation für Kardinalitäten: (min,max)-Notation
  - Erlaubt eine Beschränkung der minimalen Anzahl korrespondierender Entities
  - (min,max)-Notation und 1:n-Notation haben unterschiedliche Ausdruckstärke



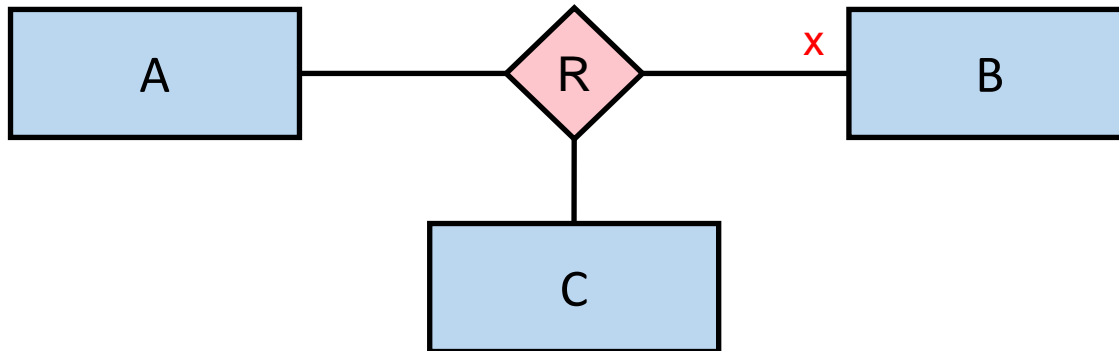
- **Achtung:** Positionsvertauschung in (min,max)-Notation

## Elemente des ER-Modells (Kardinalitäten)

Kardinalitäten für zwei- und mehrstellige Beziehungen: **(min,max)-Notation**



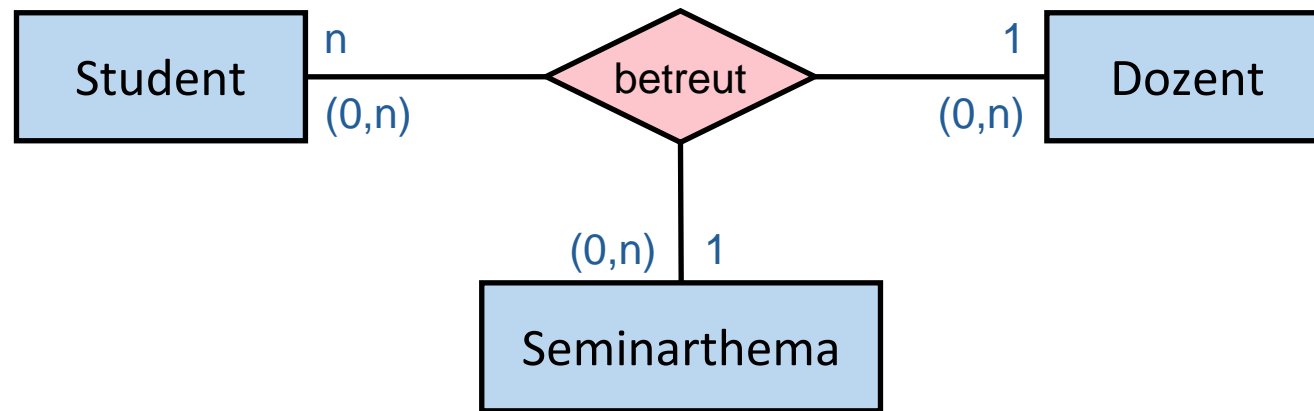
- **x**: An wie vielen R-Ausprägungen ist ein Element des Entity Typs B beteiligt?  
= Wie viele Elemente des Entity Typs A stehen in Beziehung zu einem Element des Entity Typs B?



- **x**: An wie vielen R-Ausprägungen ist ein Element des Entity Typs B beteiligt?  
= Mit wievielen Paare  $(a,c)$  mit  $a \in A$  und  $c \in C$  steht ein Element des Entity Typs B in Beziehung?

# Elemente des ER-Modells (Kardinalitäten)

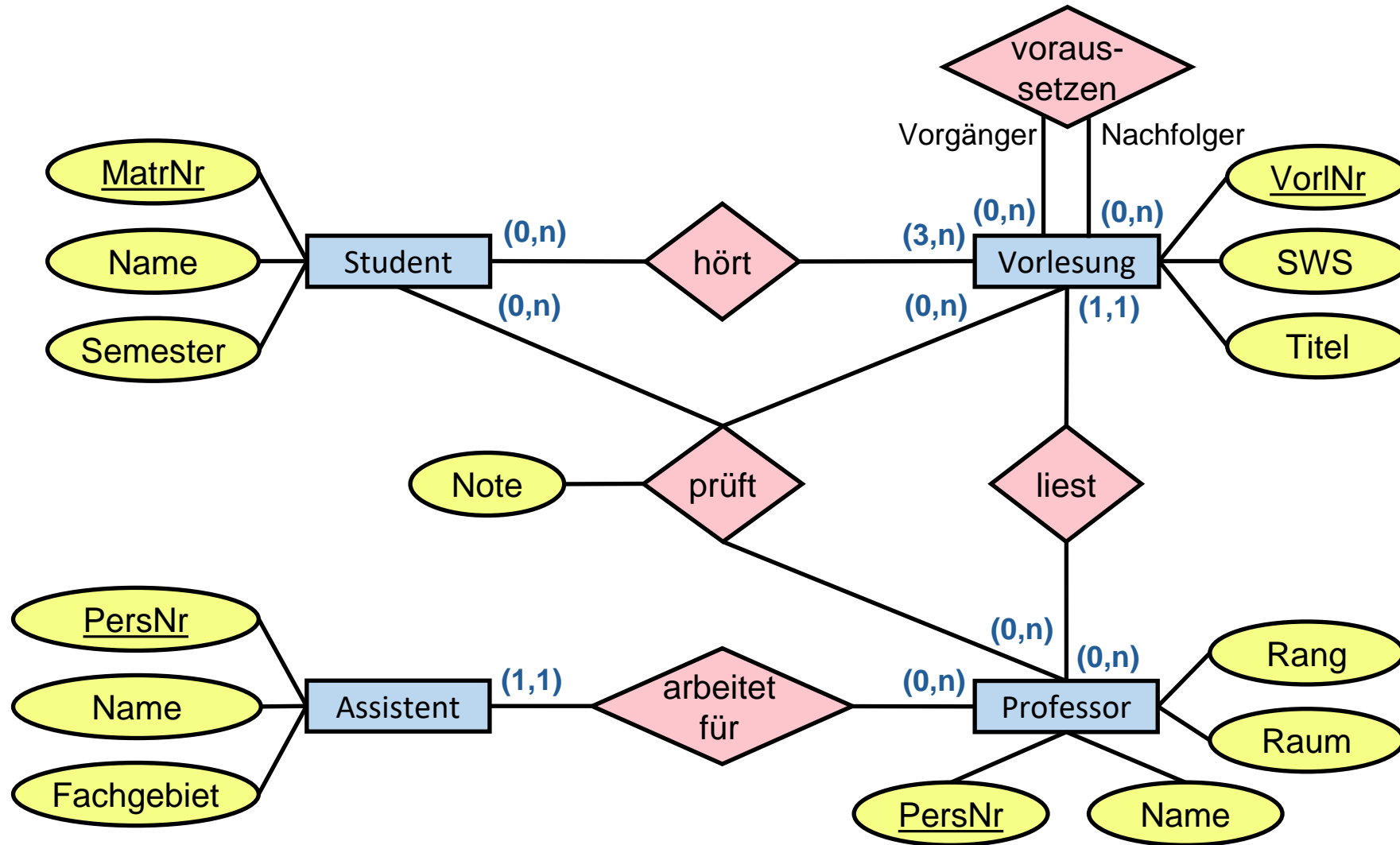
Beispiel:



- Annahmen über die Domäne
  - Studenten dürfen pro Dozent nur ein Seminarthema belegen
  - Studenten dürfen Seminarthema nur einmal belegen
  - Dozenten können Seminarthemen wiederverwenden
  - Das gleiche Thema kann von verschiedenen Dozenten vergeben werden

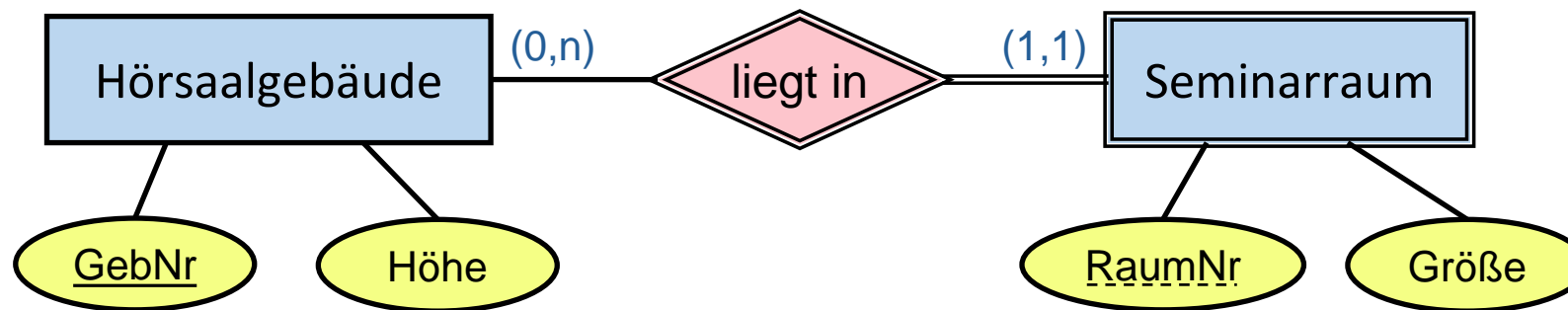
# Beispiel: ER-Diagramm (Universität)

## Entity Typen + Attribute + Beziehungen + Kardinalitäten (min,max)



## Elemente des ER-Modells (Schwache Entity-Typen)

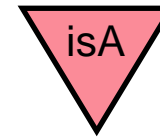
- Ein schwacher Entity-Typ ist von der Existenz des übergeordneten Entity-Typs abhängig.
- *Beispiel:* Ohne Hörsaalgebäude kann es keine Seminarräume geben
  - Kardinalitätsrestriktion zu übergeordneten Entity-Typ in (min,max): (1,1)
  - Der schwache Entity-Typ (*Seminarraum*) und seine Relation (*liegt in*) zum übergeordneten Entity (*Hörsaalgebäude*) werden durch doppelte Umrandung des Entity-Typ, der Verbindung zur Beziehung und der Beziehung selbst dargestellt



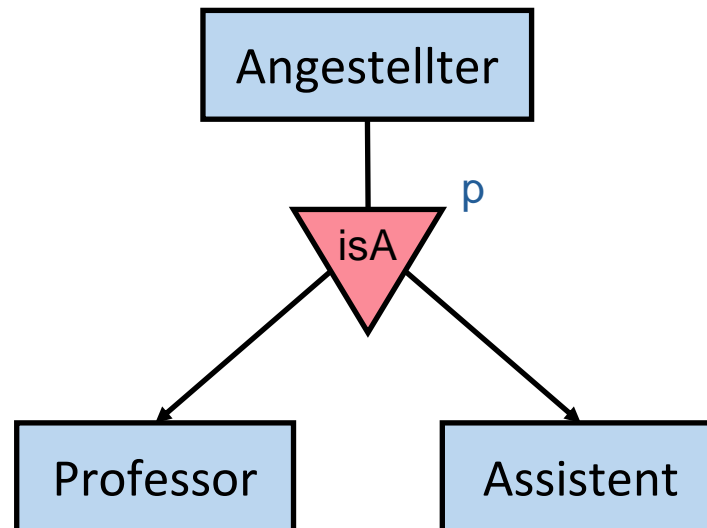
Schlüssel von schwachen Entity-Typen: gestrichelt unterstrichen

## Elemente des ER-Modells (Generalisierung/Spezialisierung)

- Vererbungsbeziehung (*isA*) zwischen Entity-Typ und spezialisierten Entity-Typ
  - Spezialisierter Entity-Typ erbt von dem allgemeinen Entity-Typ
  - Eine *isA* Beziehung wird durch ein umgedrehtes Dreieck dargestellt  
Ein Unterscheidungsmerkmal bestimmt die Zugehörigkeit und kann explizit mitmodelliert werden.



*Beispiel:*





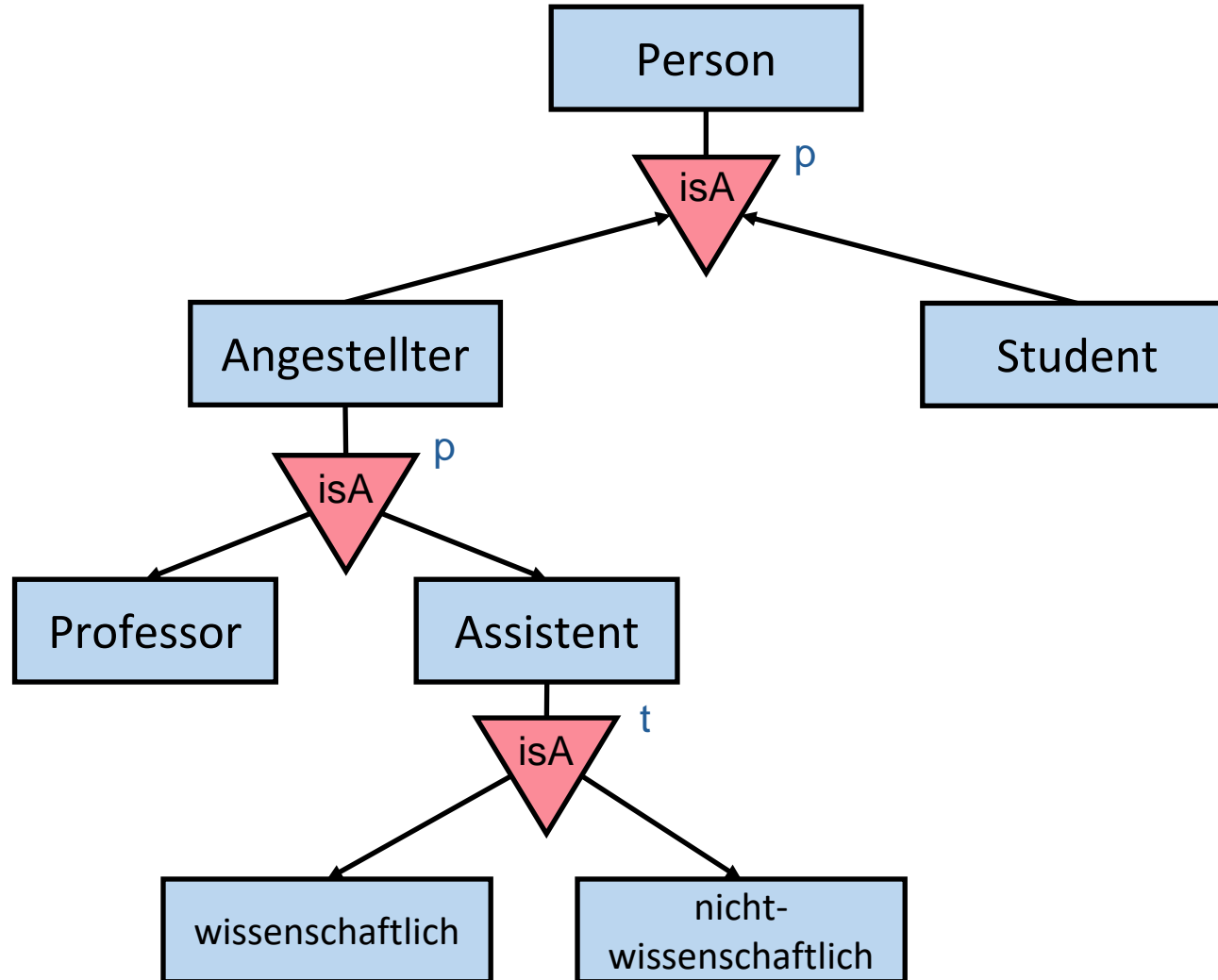
# Elemente des ER-Modells (Generalisierung/Spezialisierung)

---

- Merkmale (Generalisierung/Spezialisierung)
  - **Disjunkt**
    - Spezialisierungen sind disjunkt (Ein Angestellter kann nicht Assistent und Professor sein)
    - *Pfeile zeigen auf die Spezialisierung*
  - **Nicht disjunkt**
    - Spezialisierung sind nicht disjunkt (Eine Person kann Angestellter und Student sein)
    - Pfeile Zeigen auf in Richtung der Generalisierung
  - **Total (t)**
    - Die Dekomposition der Generalisierung ist vollständig (Es gibt entweder wissenschaftliche oder nicht wissenschaftliche Mitarbeiter)
    - Wird durch “*t*” neben der isA Beziehung dargestellt
  - **Partiell (p)**
    - Die Vereinigung der Spezialisierung ist eine echte Untermenge der Generalisierung
    - Wird durch “*p*” neben der isA Beziehung dargestellt

# Elemente des ER-Modells (Generalisierung/Spezialisierung)

Beispiel:



## Elemente des ER-Modells (Generalisierung/Spezialisierung)

Beispiele	Disjunkt	Nicht disjunkt
Total	<ul style="list-style-type: none"><li>• Raum unterteilt in: Hörsaal, Büro, Seminarraum,...</li><li>• Studenten dieser Vorlesung: Klausur teilgenommen vs. Nicht teilgenommen</li></ul>	<ul style="list-style-type: none"><li>• Student mit Sub-Entity-Typ für <b>jeden</b> Studiengang</li><li>• Student unterteilt in Master, Bachelor, Promotion (man kann in mehreren Studiengängen eingeschrieben sein)</li></ul>
Partiell	<ul style="list-style-type: none"><li>• Studenten dieser Vorlesung: Klausur bestanden vs. Nicht bestanden (es gibt auch welche, die nicht teilgenommen haben)</li></ul>	<ul style="list-style-type: none"><li>• Student mit Sub-Entity-Typ <b>nur</b> für Studiengänge der Fakultät 1</li></ul>



## 2. Das Entity-Relationship-Modell

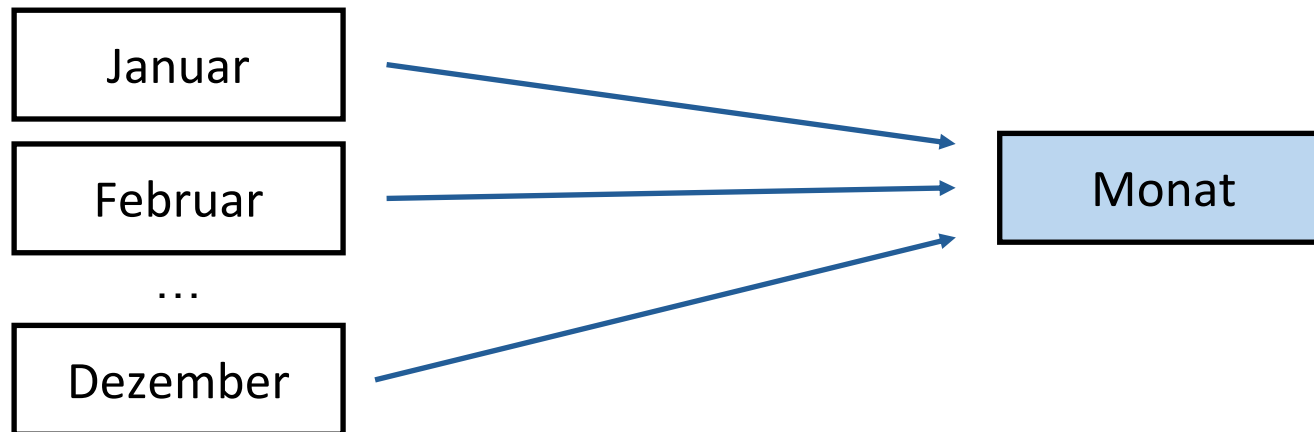
1. Der Datenbankentwurf
2. Entity-Relationship-Modell
3. **Konzeptueller Entwurf**

# Konzeptueller Entwurf (Abstraktionskonzepte)

---

- **Klassifikation**

- Abstraktion von einer Menge von Objekten mit ähnlichen Eigenschaften auf eine Klasse
- *Beispiel:* Klassifikation von Monatsobjekten (Januar, ..., Dezember) zum Entity-Typ *Monat*



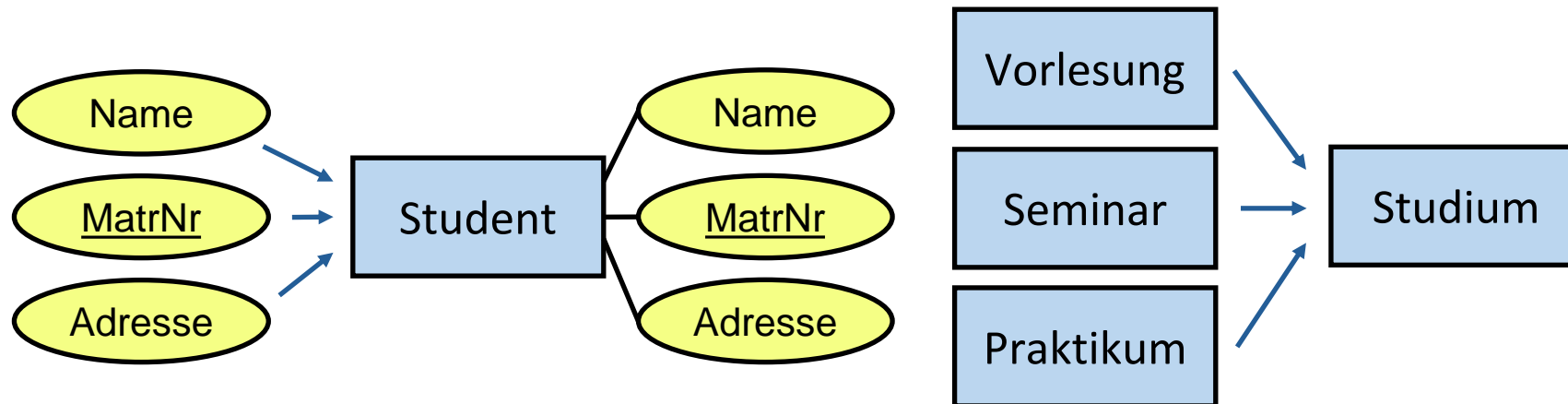
- **Identifikation**

- Hinzufügen von Identifikators (Beispiel: Schlüsselattribute)

# Konzeptueller Entwurf (Abstraktionskonzepte)

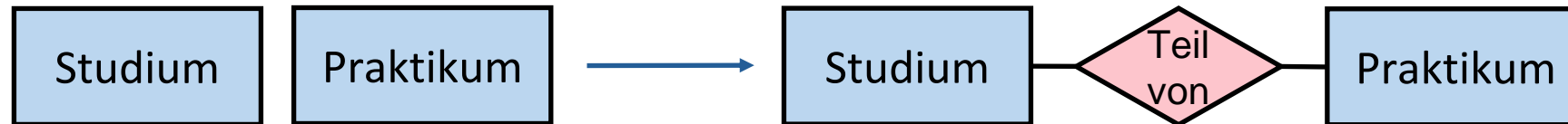
- **Aggregation**

- Abstraktion von einer Menge von Komponenten oder Teilen auf das Ganze
  - Verschiedene Attribute werden zu einem Entity-Typ aggregiert
  - Verschiedene Entity-Typen werden zu einem neuen Entity-Typ aggregiert



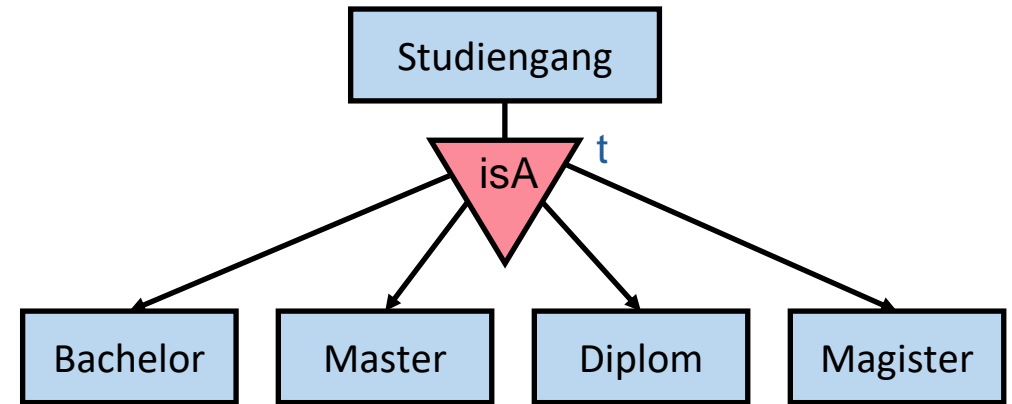
- **Assoziation**

- Unabhängige Klassen werden in Beziehung gestellt



# Konzeptueller Entwurf (Abstraktionskonzepte)

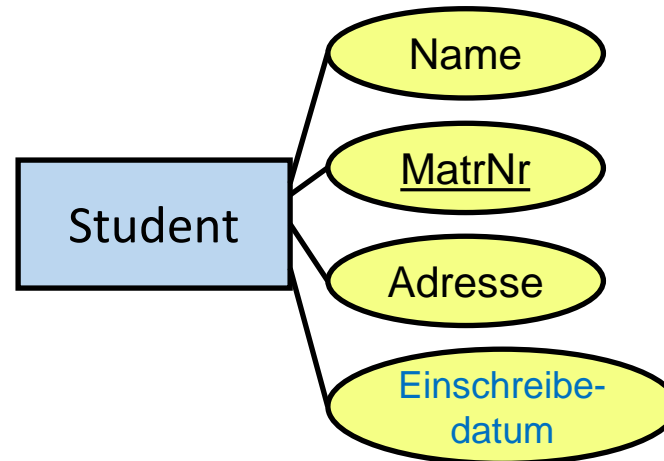
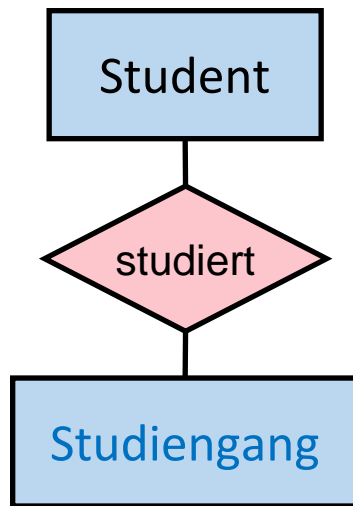
- **Generalisierung/Spezialisierung**
- Zwei Möglichkeiten der Modellierung
- Top Down (Spezialisierung)
  - Es werden zuerst sehr große Informationsblöcke modelliert
  - Anschließend werden die Blöcke schrittweise spezialisiert
- Bottom Up (Generalisierung)
  - *Es werden zuerst sehr detaillierte Informationen modelliert*
  - *Schrittweise werden diese Informationen verallgemeinert*



# Konzeptueller Entwurf (Richtlinien)

- **Entity-Typ oder Attribut**

- Entity-Typ, falls das Konzept selber Eigenschaften hat oder mehrfach im Modell auftaucht (z.B.: in Beziehung zu anderen Entities)
- Andernfalls wird das Konzept als Attribut modelliert
- *Beispiele:*
  - Studiengang eines Studenten □ Entity-Typ
  - Einschreibedatum eines Studenten □ Attribut

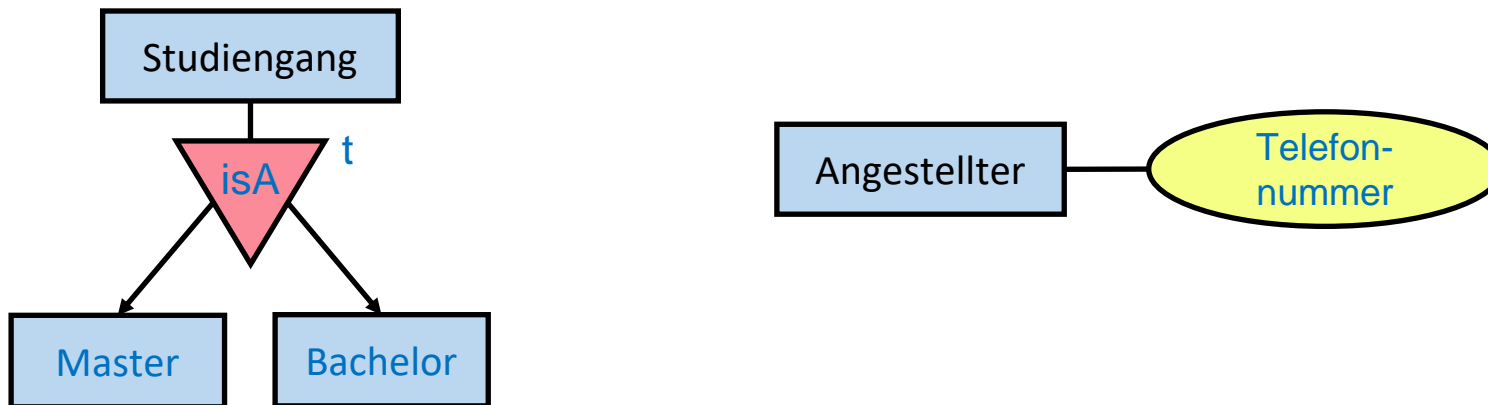




# Konzeptueller Entwurf (Richtlinien)

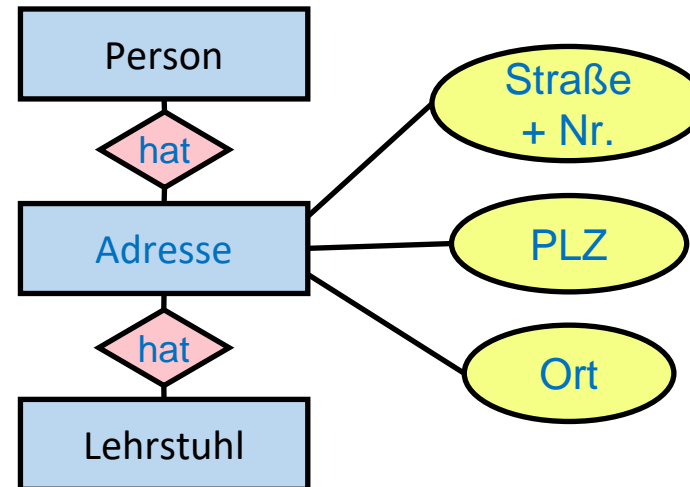
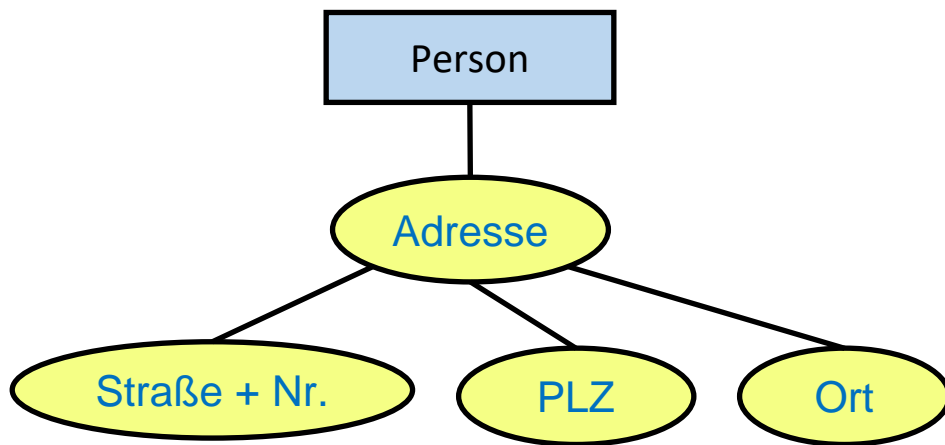
- **Generalisierung/Spezialisierung oder Attribut**

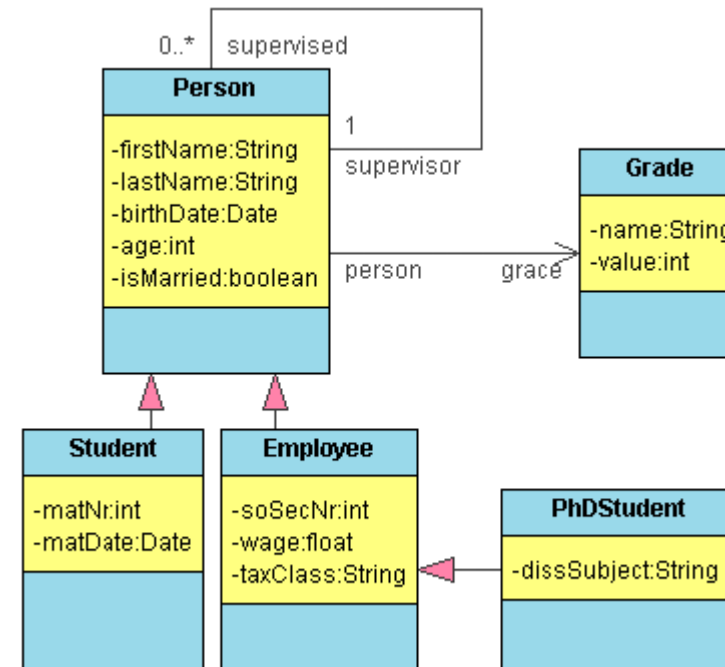
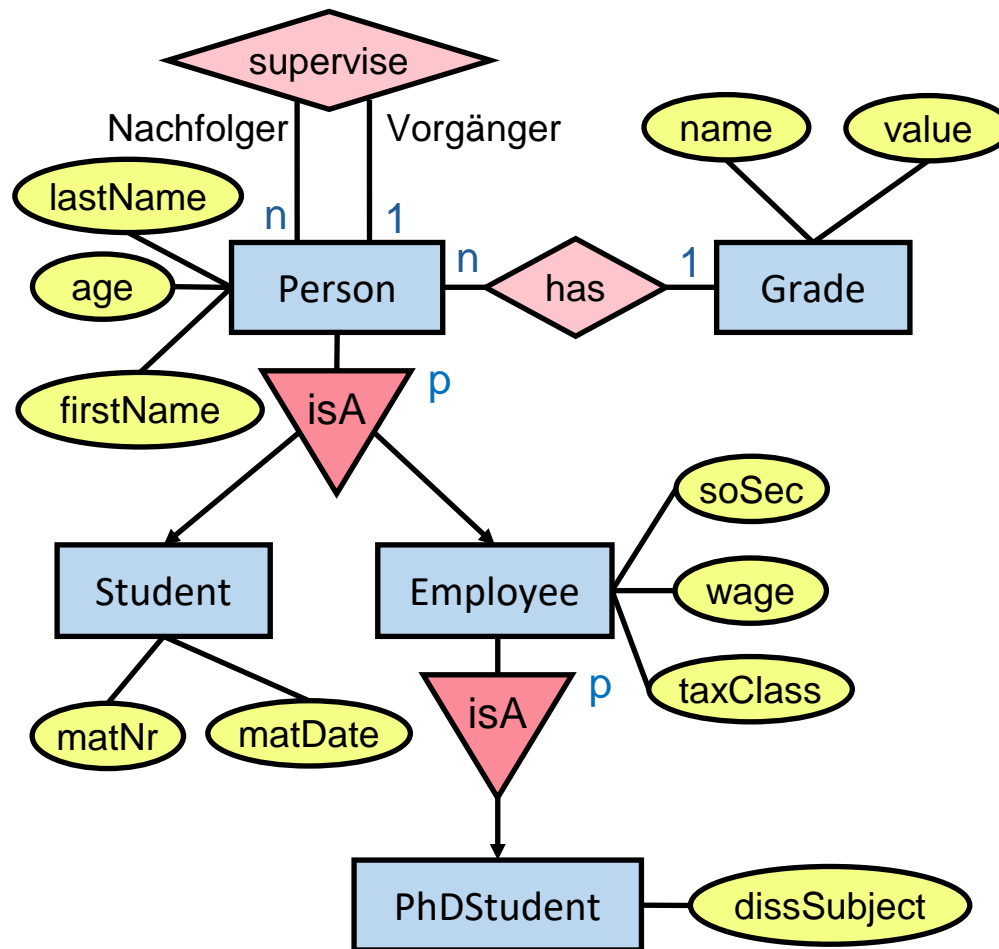
- Generalisierung/Spezialisierung, falls das Unterkonzept zusätzliche Attribute oder Beziehungen hat, oder nun spezielle semantische Constraints ausdrückt
- Andernfalls wird das Konzept als Attribut modelliert
- *Beispiele:*
  - Bachelor und Master Studiengänge □ Spezialisierung
  - Telefonnummer eines Angestellten □ Attribut



# Konzeptueller Entwurf (Richtlinien)

- **Zusammengesetztes Attribut, einfaches Attribut oder Entity-Typ**
  - Zusammengesetztes Attribut, falls der Name des Attributs eine spezielle Bedeutung hat
  - Entity-Typ, falls es mehrfach verwendet wird und die Information gemeinsam genutzt werden soll
  - Andernfalls wird das Konzept als Attribut modelliert
  - *Beispiel:*
    - Adresse von Personen □ Zusammengesetztes Attribut
    - Adresse von Personen und Lehrstühlen □ Entity-Typ





- **Der Datenbankentwurf**
  - Datenbank-Lebenszyklus
  - Qualitätskriterien für den Datenbankentwurf
  
- **Entity-Relationship-Modell**
  - Elemente des ER-Modell
    - Entities, Entity-Typen
    - Attribute
    - Relationships (Beziehungen)
    - Generalisierung/Spezialisierung
  - Nicht standardisiert □ unterschiedliche Notationen (1:n, (min,max))
  
- **Konzeptueller Entwurf**
  - Abstraktionskonzepte
  - Richtlinien