

Informatik ist eine  
Strukturwissenschaft

## Teil II

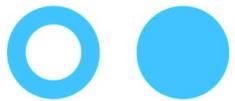


Von Daten und ihren Modellen

Robert Hartmann (SoSe 2024)

basierend auf Folien von  
Prof. Dr. Harm Knolle

Fachbereich Informatik  
Hochschule Bonn-Rhein-Sieg



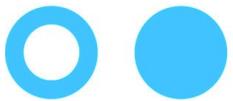
## - Kapitel 3 - Semantische Datenmodelle -

### Inhalt

- 0 - Vorbemerkungen
- Teil I** - Von EDV-Anwendungen und Ihren Anforderungen
- 1 - Einführung
- Teil II** - Von Daten und ihren Modellen
- 2 - Prozess des Datenbankentwurfs
- 3 - Semantische Datenmodelle**
- 4 - Logische Datenmodelle
- 5 - Datenbankmodelle
- 6 - Datenanfrage und Datenänderung
- Teil III** - Von Datenbanken und ihren Systemen
- 7 - Datenbanksysteme
- 8 - Speicherstrukturen
- 9 - Ausblick

### Überblick

- ◆ Einführung
- ◆ Entity Relationship Modell (ERM)
- ◆ Erweiterungen des ERM
- ◆ Zeit und Versionen
- ◆ Objekt-orientierte Modelle



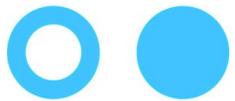
## - Attribut -

### Inhalt

- ◆ Einführung
- ◆ Entity Relationship Modell (ERM)
  - Schema der Miniwelt (Diskursbereich)
  - Ziel: Schema in Diagrammform
  - Historie und Notation
  - Integritätsbedingungen
  - Entity
  - Relationship
  - Kardinalität
  - Attribut
- ◆ Erweiterungen des ERM
- ◆ Zeit und Versionen
- ◆ Objekt-orientierte Modelle

### Überblick

- ◆ Spezifikation
- ◆ Graphische Darstellung
- ◆ Wertebereich / Domäne
- ◆ Schlüssel / Primärschlüssel



## - Spezifikation (I) -

### Attribut

- ♦ Eigenschaften von Entity-/Beziehungs-Typen
- ♦ jeder Entity-/Beziehungs-Typ wird durch die Gesamtheit seiner Attribute A<sub>1</sub>, … , A<sub>n</sub> beschrieben

### Wertebereich (Domäne)

- ♦ Zusammenfassung zulässiger Werte einer Eigenschaft
- ♦ jedes Attribut A kann nur Werte aus seinem Wertebereich D annehmen

### Semantik

- ♦ ein Attribut steht für die Menge der möglichen Werte seines Wertebereichs

### Anmerkung

- ♦ im Rahmen der ERD-Modellierung sollen keine künstlichen (impliziten) Attribute zur Sicherung der Identität wie z.B. „Nr“ modelliert werden

### Bezeichnung

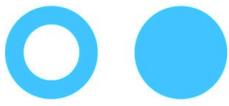
- ♦ Attribut A besitzt den Wertebereich D
- ♦ A : D
- ♦ Entity: E(A<sub>1</sub>:D<sub>1</sub>, … , A<sub>m</sub>:D<sub>m</sub>)
- ♦ Kurzform: E(A<sub>1</sub>, … , A<sub>m</sub>)
- ♦ Beziehung: R(E<sub>1</sub>, … , E<sub>n</sub>; A<sub>1</sub>:D<sub>1</sub>, … , A<sub>p</sub>:D<sub>p</sub>)
- ♦ Kurzform: R(E<sub>1</sub>, … , E<sub>n</sub>; A<sub>1</sub>, … , A<sub>p</sub>)

### Beispiel für Attribute

- ♦ Name, Lehrgebiet, Titel, SWS

### Beispiel für Wertebereiche

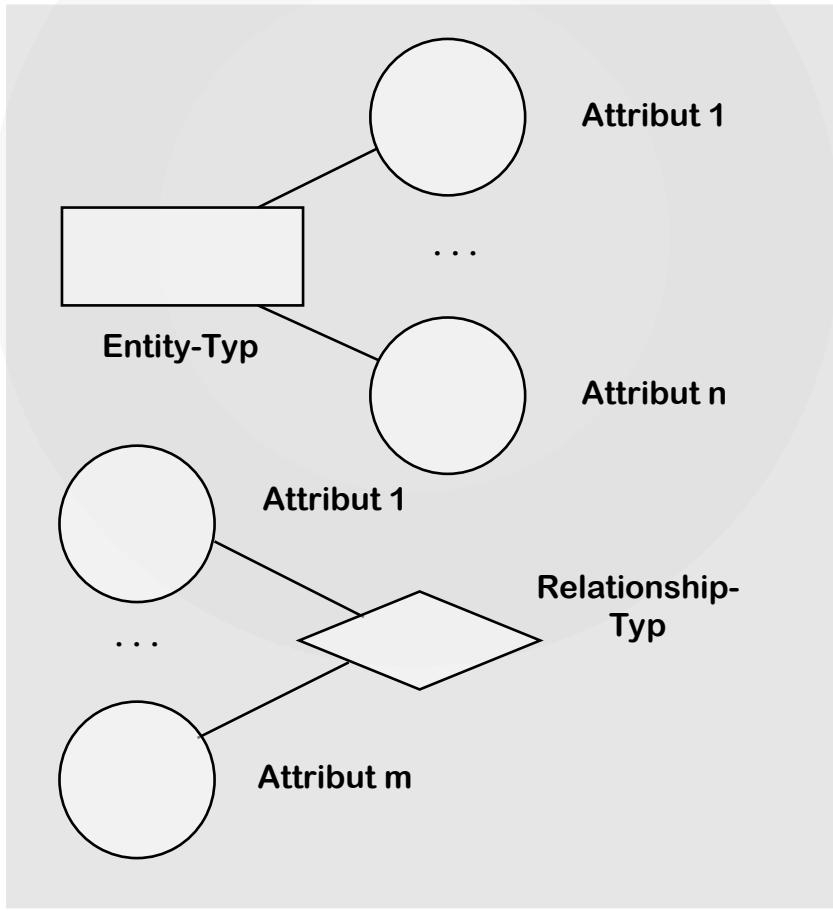
- ♦ int, numeric(3,2)
- ♦ string, char(3)
- ♦ {"true", "false"}, {0, 1}
- ♦ date



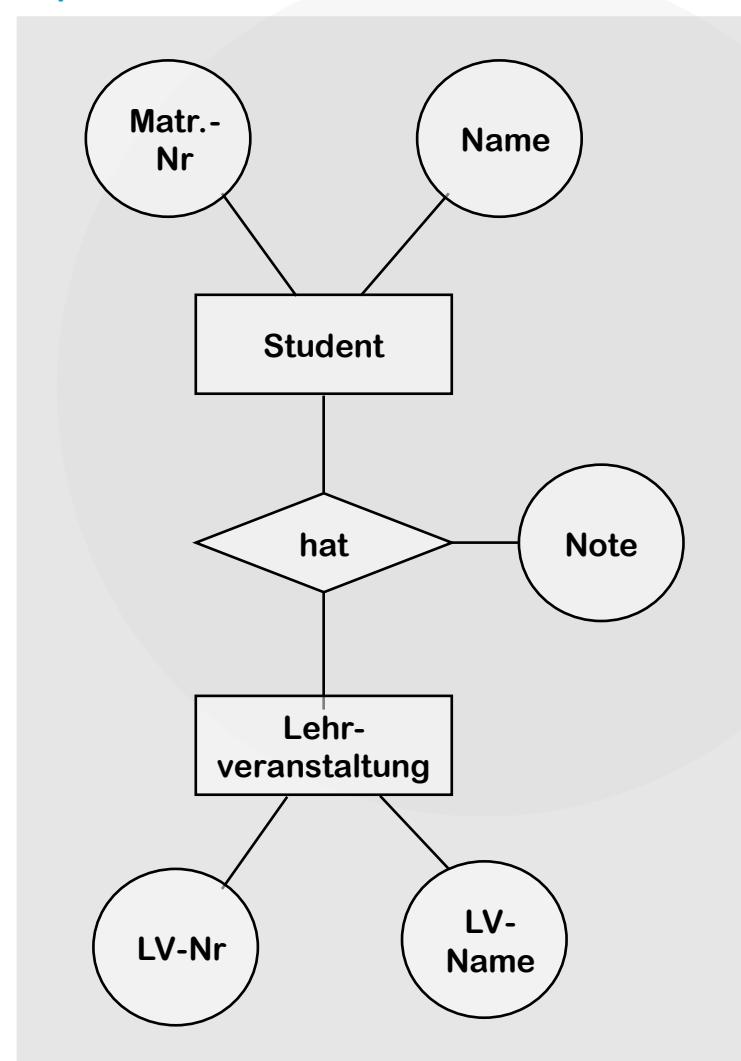
## - Spezifikation (II) -

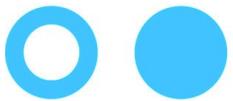
### Graphische Darstellung nach Chen

- CASE-Tools stellen die Attribute häufig nicht kreisförmig dar (siehe z.B. DB-Main)

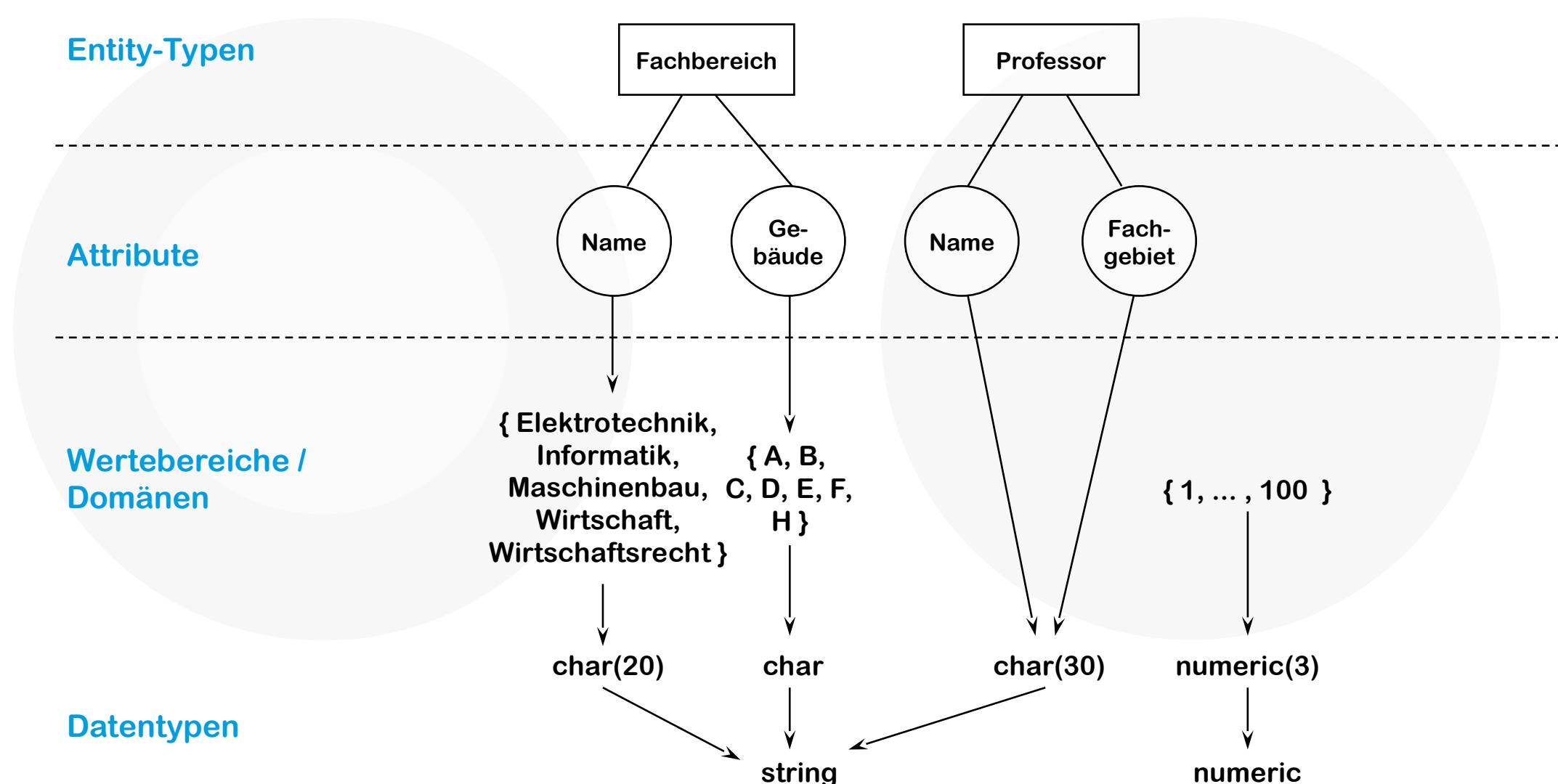


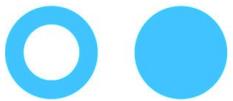
### Beispiel





## - Wertebereich / Domäne -





## - Schlüssel / Primärschlüssel -

### Schlüssel = Schlüsselkandidaten

- ◆ Attribut oder (kleinste!) Attributmenge, die jedes Entity eines Entity-Typs eindeutig identifiziert
- ◆ eignen sich die Attribute eines Entity-Typen nicht, kann auch ein künstlicher Schlüssel vergeben werden : **In der semantischen Darstellung sollen keine künstliche Schlüssel vergeben werden.**

### Primärschlüssel

- ◆ bei mehreren gültigen Schlüsseln wird einer als Primärschlüssel bezeichnet

### Semantik

- ◆ die Werte der Schlüsselattribute identifizieren eindeutig Instanzen seines Entity-Typs

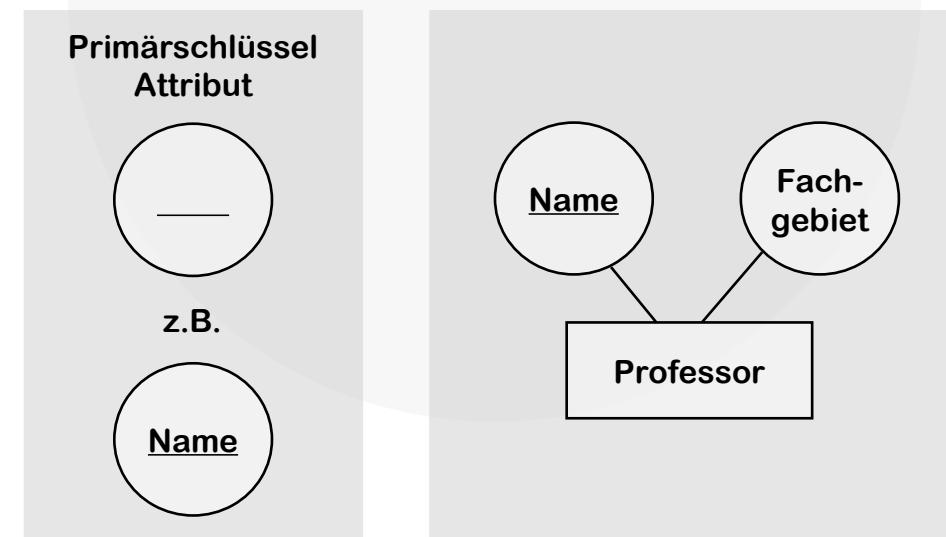
### Anmerkung

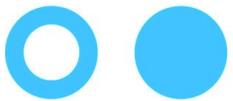
- ◆ im Rahmen der ERD-Modellierung existieren häufig keine Attribute mit künstlichen Werten
- ◆ Schlüssel stellen im ERD, wenn überhaupt, daher häufig nur natürliche Attribute dar

### Bezeichnung

- ◆ Entity  $E(A_1, \dots, A_n)$  besitzt den Schlüssel  $\{S_1, \dots, S_m\}$ :
- ◆  $\{S_1, \dots, S_m\} \subseteq \{A_1, \dots, A_n\}$
- ◆ Kurzform:  $E(\dots, \underline{S_1}, \dots, \underline{S_m}, \dots)$

### Graphische Darstellung nach Chen





## - Erweiterungen des ERM -

### Inhalt

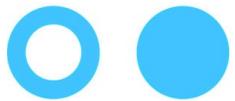
- ◆ Einführung
- ◆ Entity Relationship Modell (ERM)
- ◆ **Erweiterungen des ERM**
- ◆ Zeit und Versionen
- ◆ Objekt-orientierte Modelle

### Extended Entity-Relationship-Modell (EERM)

- ◆ Mengenwertige & strukturierte Attribute
- ◆ n-äre Beziehungstypen
- ◆ Schwache Entitätstypen
- ◆ Abhängigkeiten (Komposition & Aggregation)
- ◆ Generalisierung / Spezialisierung
  - Klassifizierung von Generalisierung / Spezialisierung
  - Beispiel zur Generalisierung / Spezialisierung

ERD = Entity-Relationship-Diagramm

EERD = Extended Entity-Relationship-Diagramm



## - Zusammengesetztes Attribut -

### Komplexe Entities

- ◆ Attribute können zusammengesetzt sein
  - Menge
  - Liste
  - Teilattribute

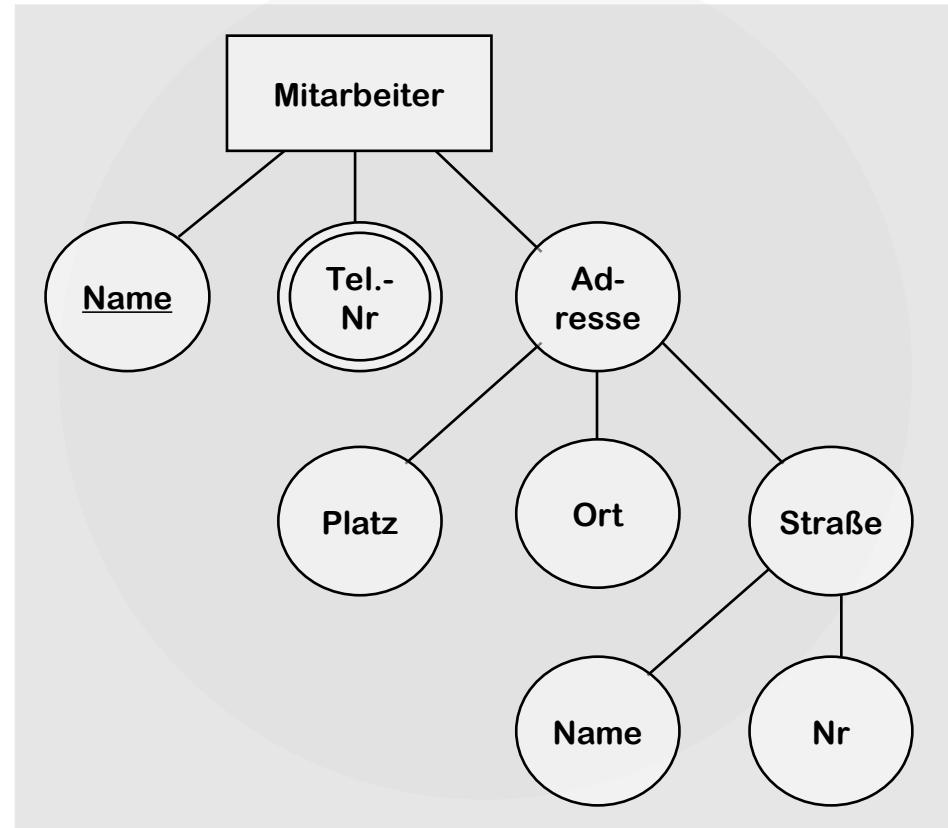
### Semantik

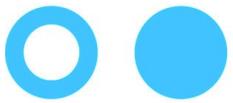
- ◆ Aufhebung der Atomarität von Attributen

### Beispiele

- ◆ Adresse
- ◆ Liste der Autoren eines Buchs
- ◆ Menge von Telefonnummern

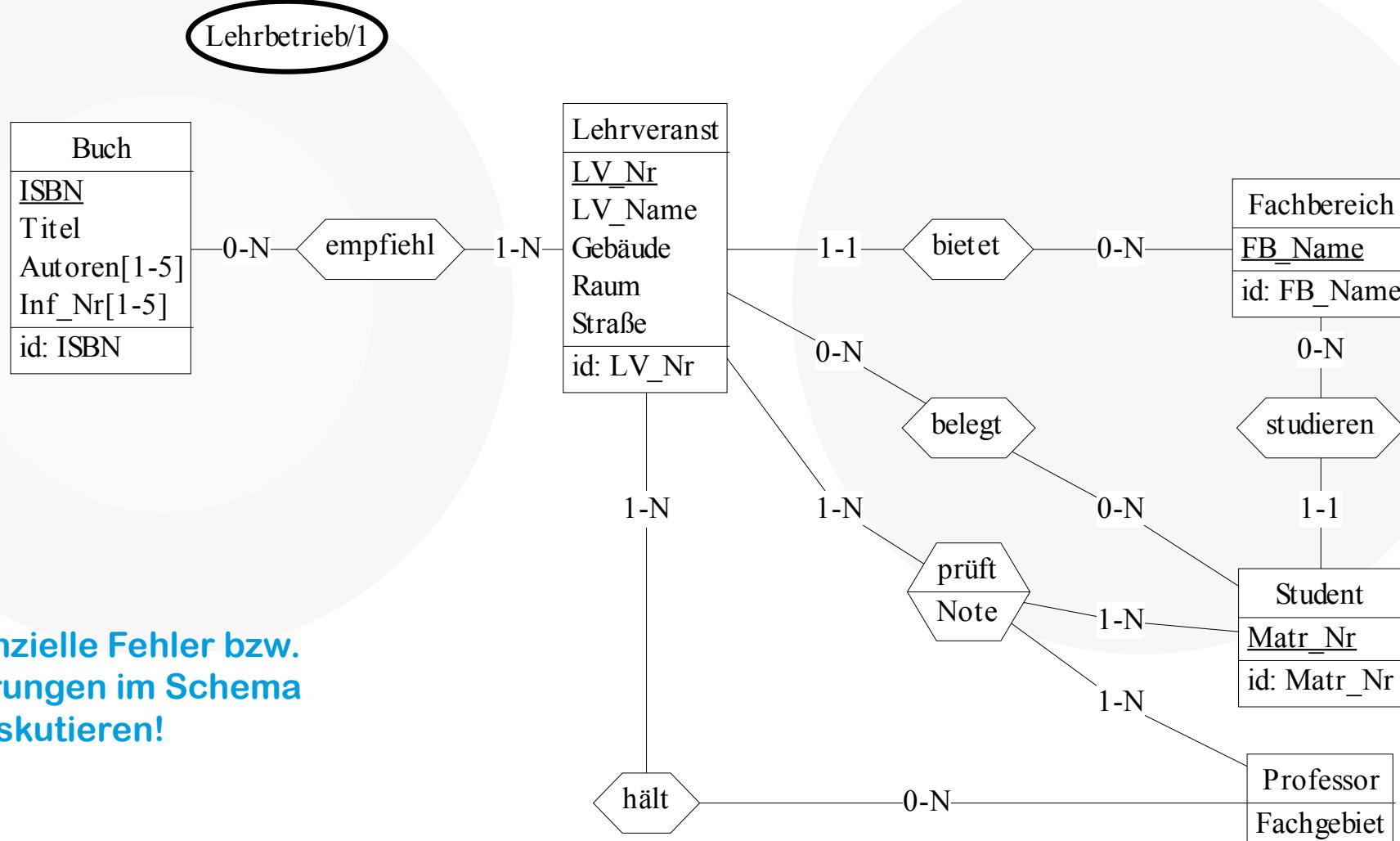
### Graphische Darstellung

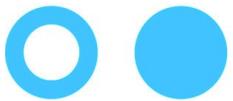




## - Beispiel mit Attributen -

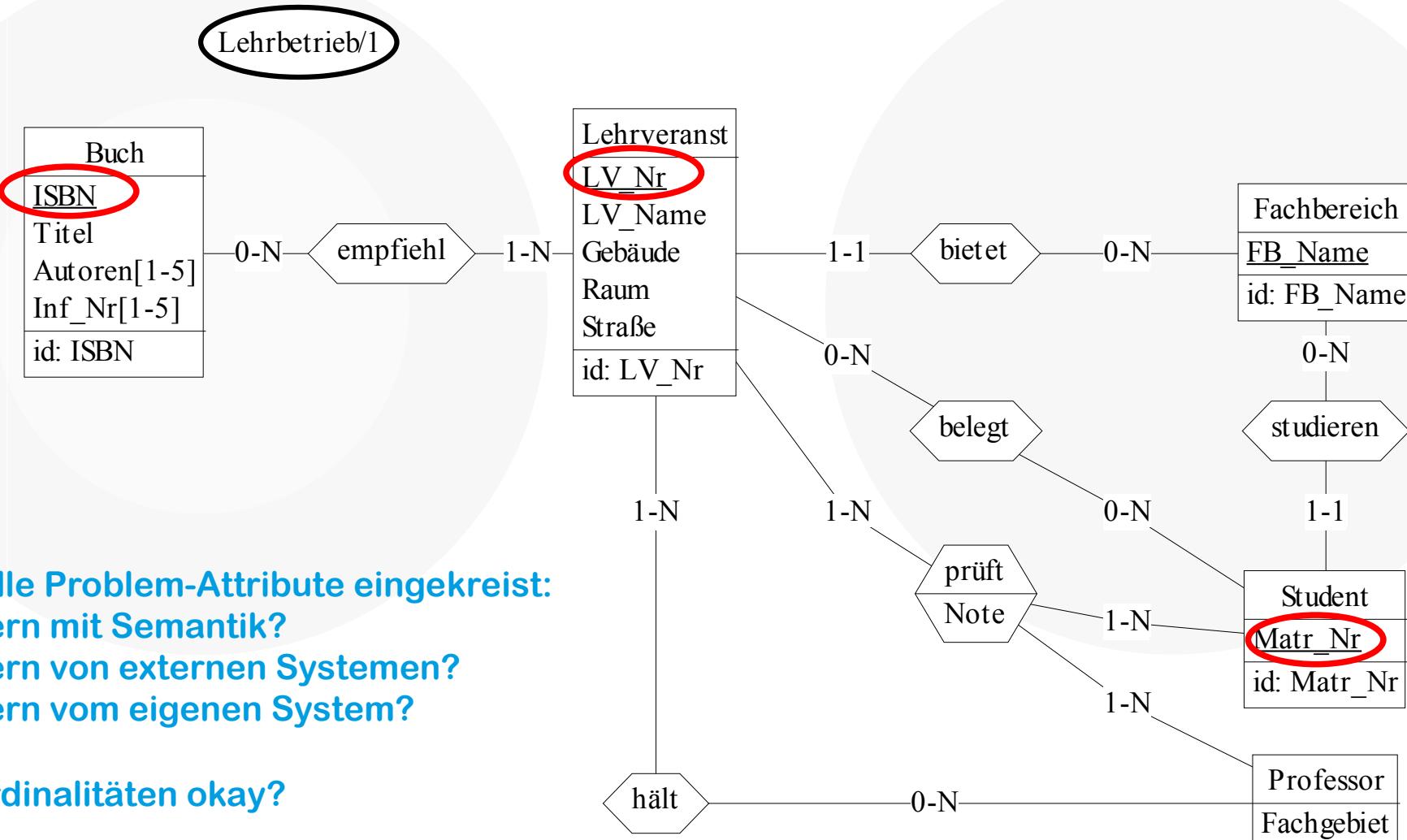
Erstellt mit dem CASE-Werkzeug DB-MAIN





## - Beispiel mit Attributen -

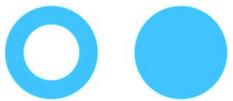
Erstellt mit dem CASE-Werkzeug DB-MAIN



Potenzielle Problem-Attribute eingekreist:

- Nummern mit Semantik?
- Nummern von externen Systemen?
- Nummern vom eigenen System?

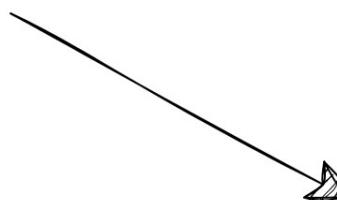
Sind Kardinalitäten okay?



id: Expliziter Primärschlüssel  
aus Attribut(en)

Module
titel: varchar (30)

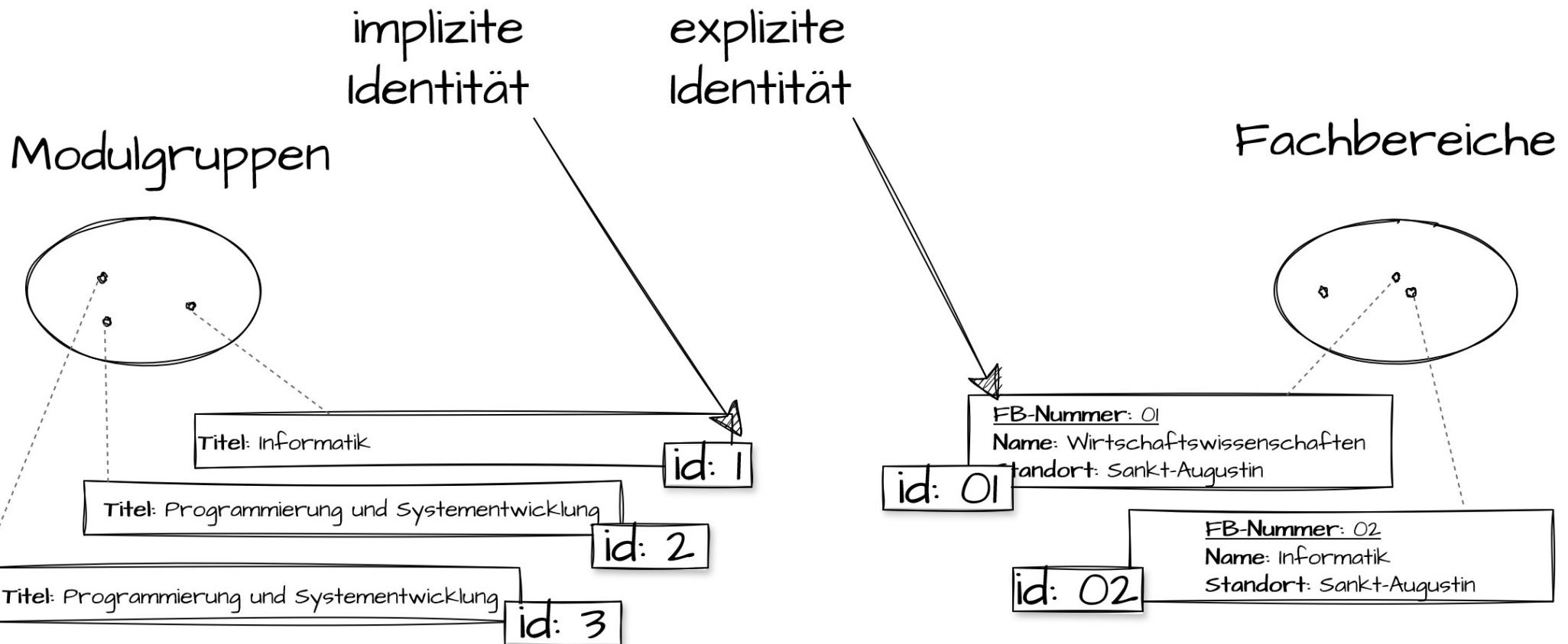
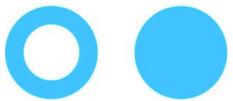
Fachbereiche
fb_nr: char (2)
fb_name: varchar (20)
standort: varchar (30)
id: fb_nr
id': fb_name
standort



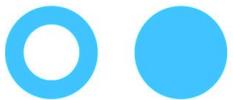
Kein id: Impliziter Primärschlüssel  
(versteckter Zähler)

... grundsätzlich hat jede Entität eine Identität bzw. Primärschlüssel

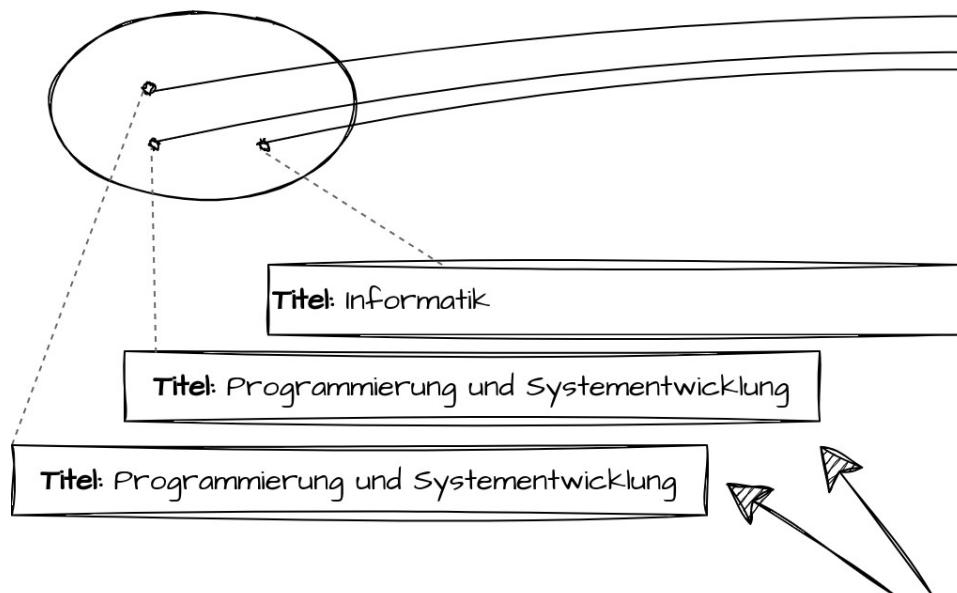
... ist kein Schlüssel definiert, wird ein impliziter Primärschlüssel angenommen



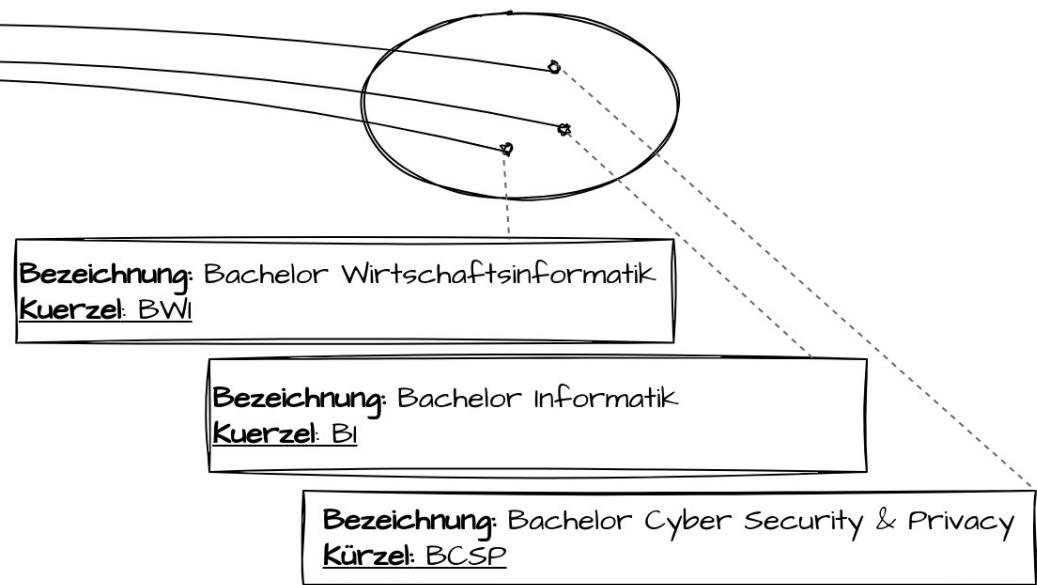
bei impliziter Identität werden Primärschlüssel  
erst später im logischen Schema angelegt



## Modulgruppen

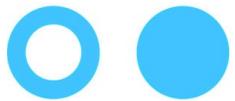


## Studiengaenge

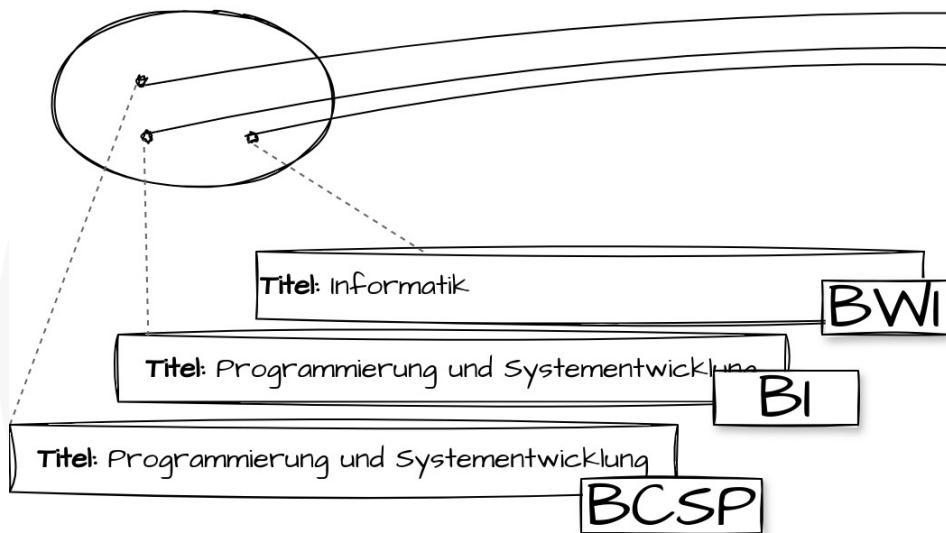


enthalten unterschiedliche Module

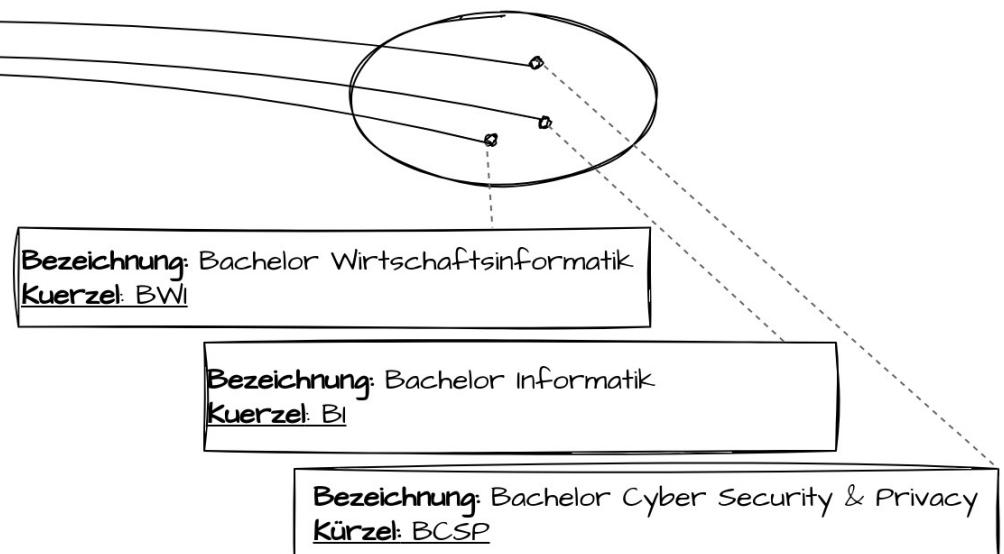
Haben Modulgruppen wirklich keine identifizierendes Attribut?



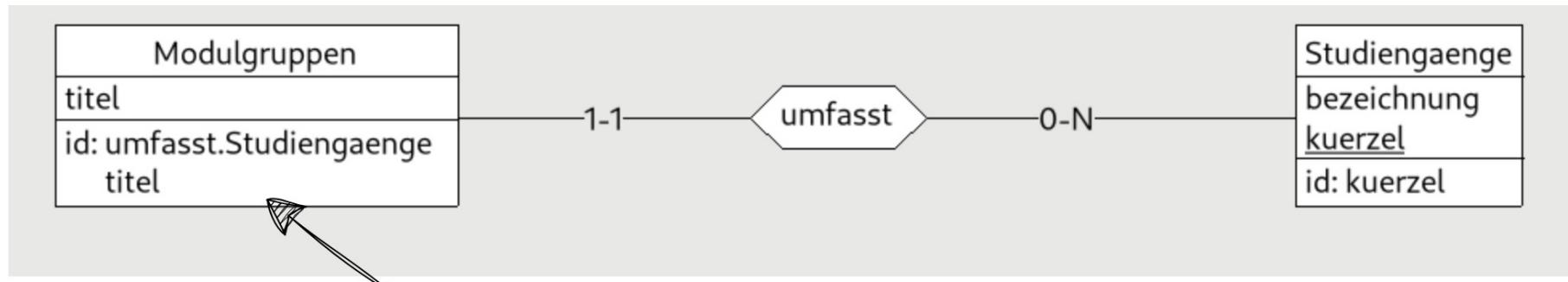
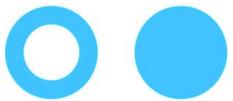
## Modulgruppen



## Studiengaenge

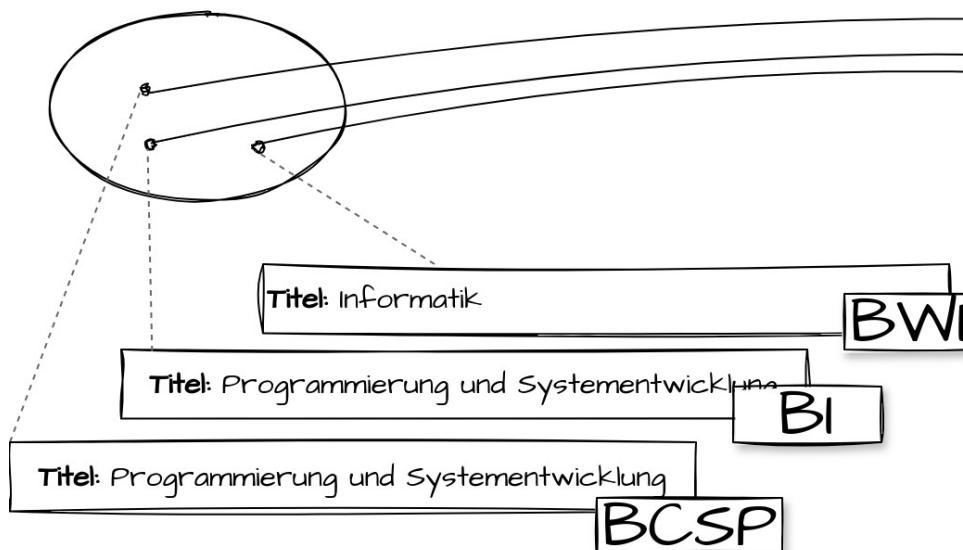


Wie kann man das formalisieren?

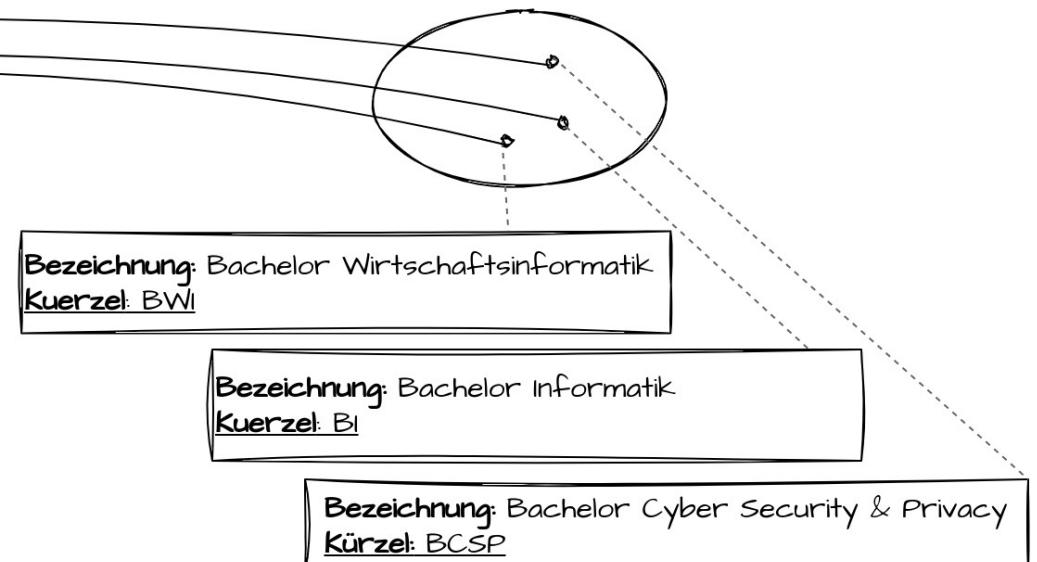


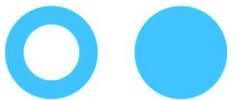
Titel der Modulgruppe + Identität des Studiengangs

Modulgruppen

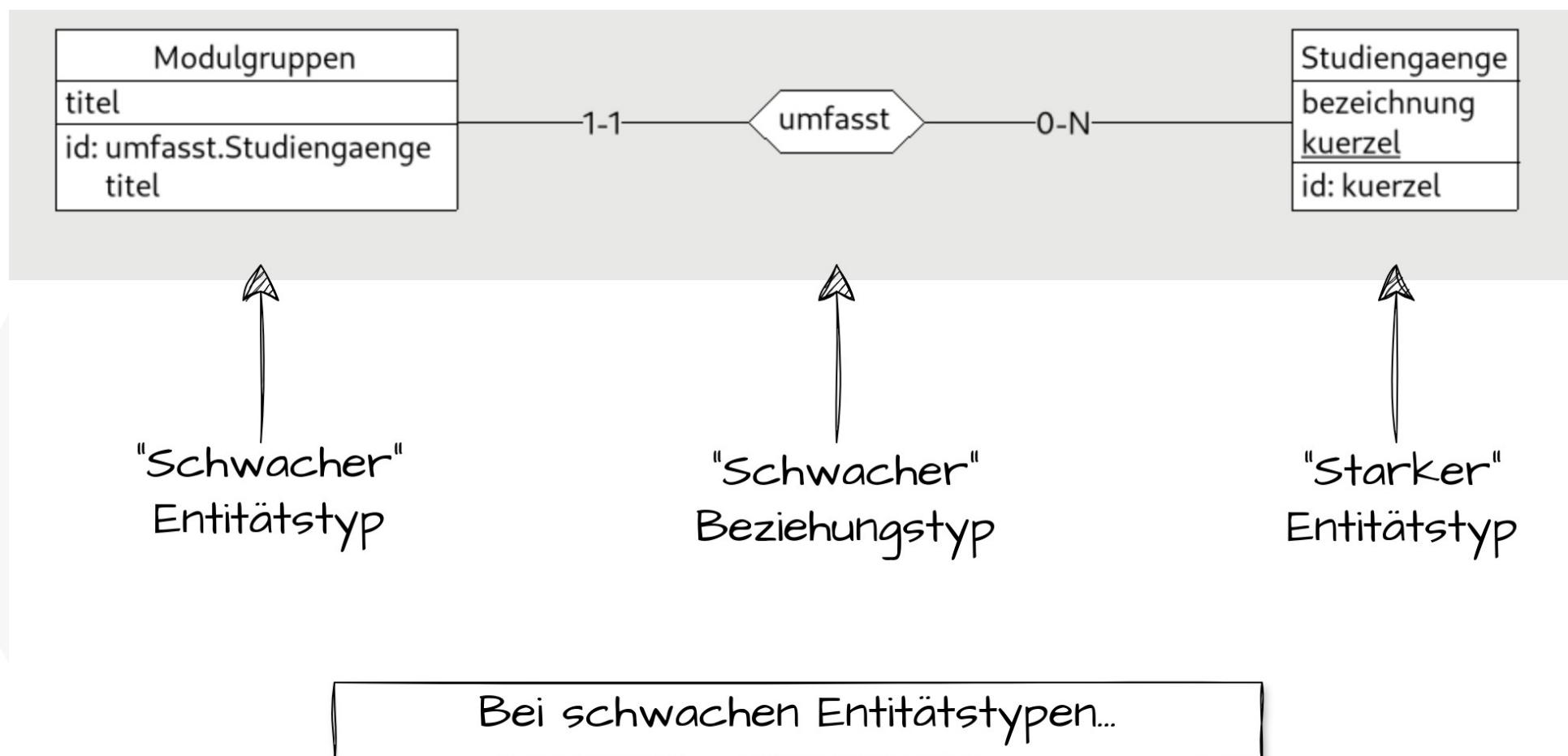


Studiengaenge

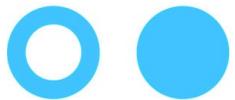




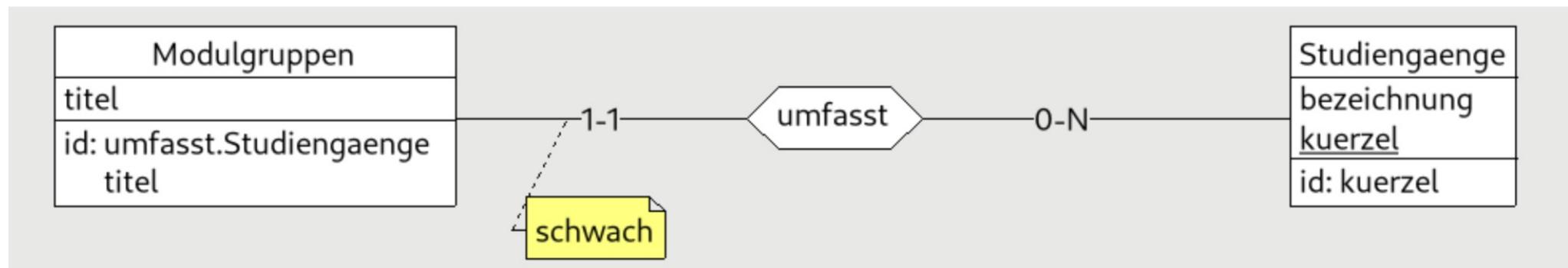
# "Schwache" Entitätstypen

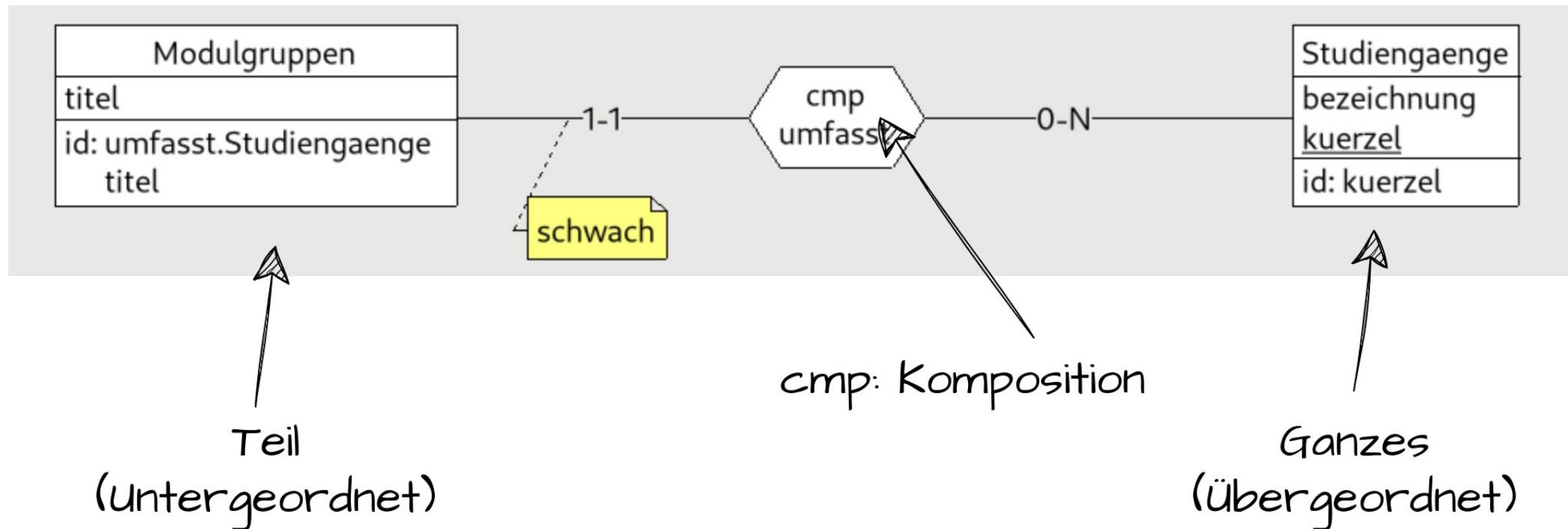
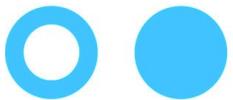


- ... haben die Entitäten keine vollständige eigene Identität
- ... ergibt sich ein Teil der Identität über die verbundene Entität
- ... können Entitäten nicht ohne die übergeordnete Entität existieren



DB-Main kennt die doppelt-umrandeten Symbole nicht,  
daher Auszeichnung mit Notiz

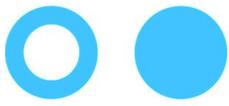




Schwache Beziehungstypen sind "Kompositionen":

... sollten entsprechend markiert werden ("cmp")

... stellen Teil-Ganzes-Beziehungen dar



## - Abhängigkeiten -

### Abhängiger (schwacher) Entity-Typ (Weak Entity)

- Entity-Typ, dessen Entities nicht durch eigene Schlüsselattribute eindeutig zu identifizieren sind
- die Identifizierung erfolgt unter Ausnutzung einer abhängigen Beziehung zu einem anderen Entity und dessen Schlüsselattributen

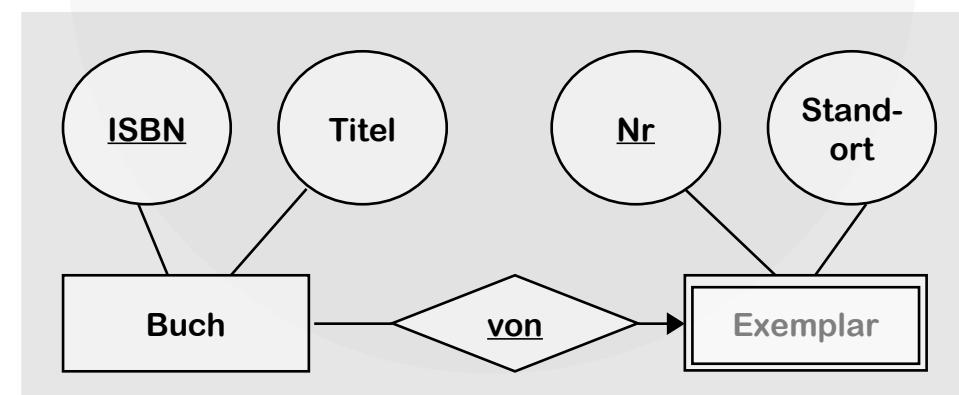
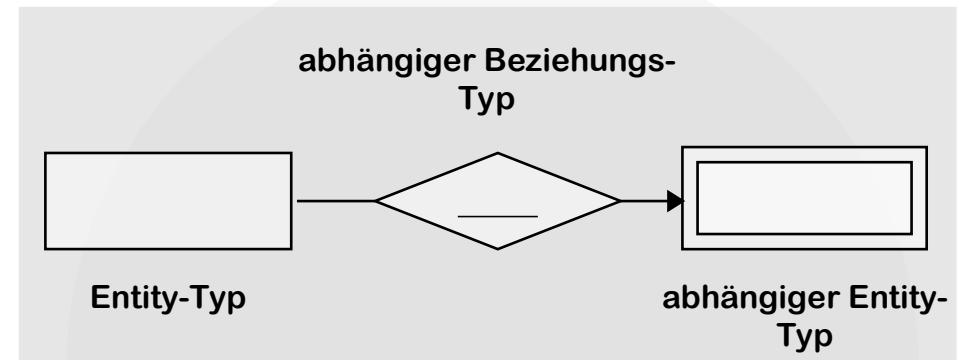
### Abhängiger (schwacher) Beziehungs-Typ

- Beziehung, an der ein abhängiger Entity-Typ hängt

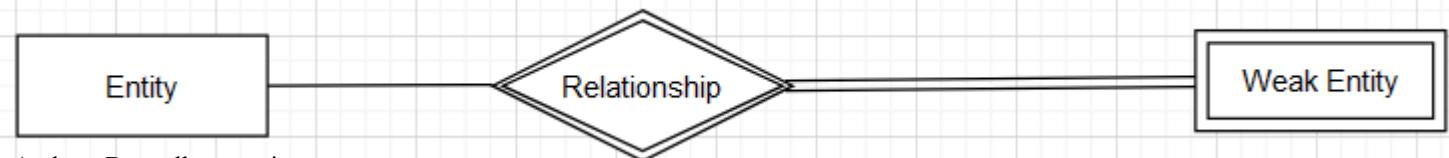
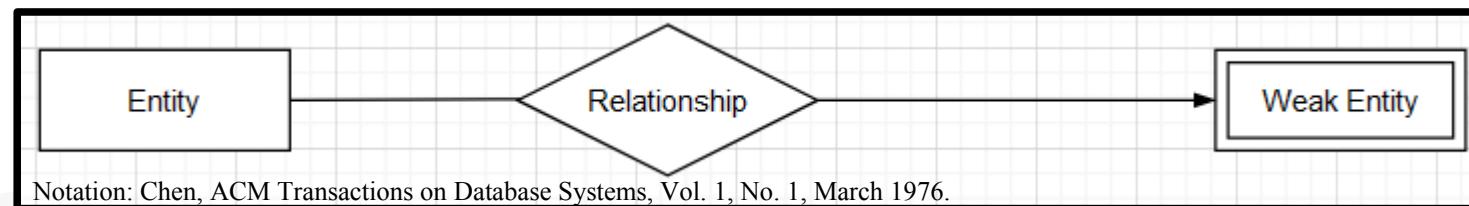
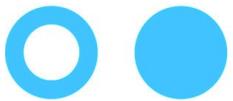
### Semantik

- abhängige Entities können nicht ohne abhängige Beziehungen existieren
- Zur Identifizierung von Exemplaren der abhängigen Entitäten ist zusätzlich die Kenntnis der identifizierenden Merkmale der zugehörigen starken Entität nötig. Die starke Entität ist Schlüssellieferant für die schwache Entität

### Graphische Darstellung nach Chen

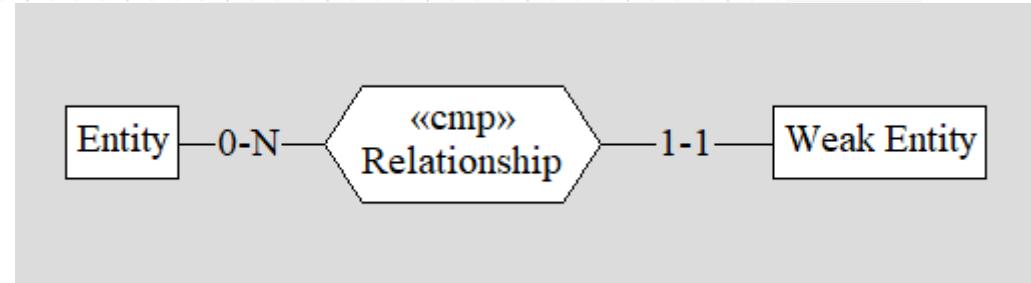


Pfeilspitze zeigt in Richtung der Schlüsselweitergabe.



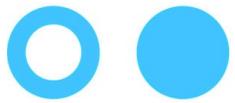
## Komposition (Teil-Ganzes-Beziehung , Teil-von-Beziehung)

- Relationstyp als funktionale Einheit bestehend aus übergeordneter Entität (strong Entity) und untergeordneten Entitäten (weak Entities).
- Wird die übergeordnete Entität gelöscht oder entfernt, müssen auch die untergeordneten Entitäten, also die Bestandteile, gelöscht bzw. entfernt werden.
  - Existentielle Abhängigkeit der Teile vom Ganzen.
  - Übergeordnete Entität ist Schlüssellieferant für untergeordnete Entitäten.

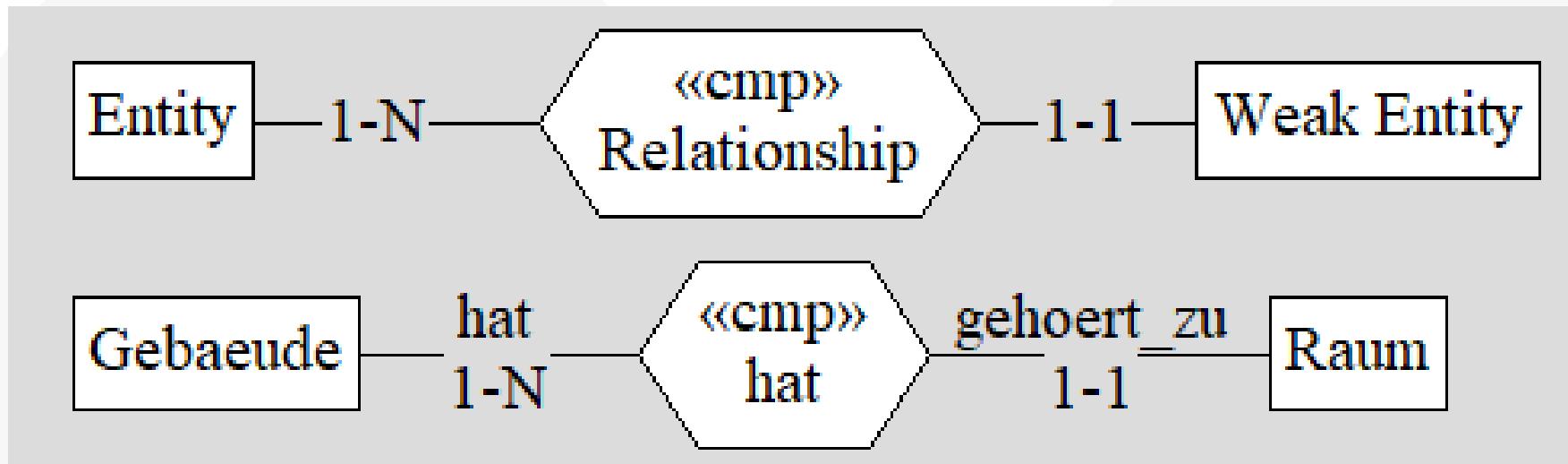


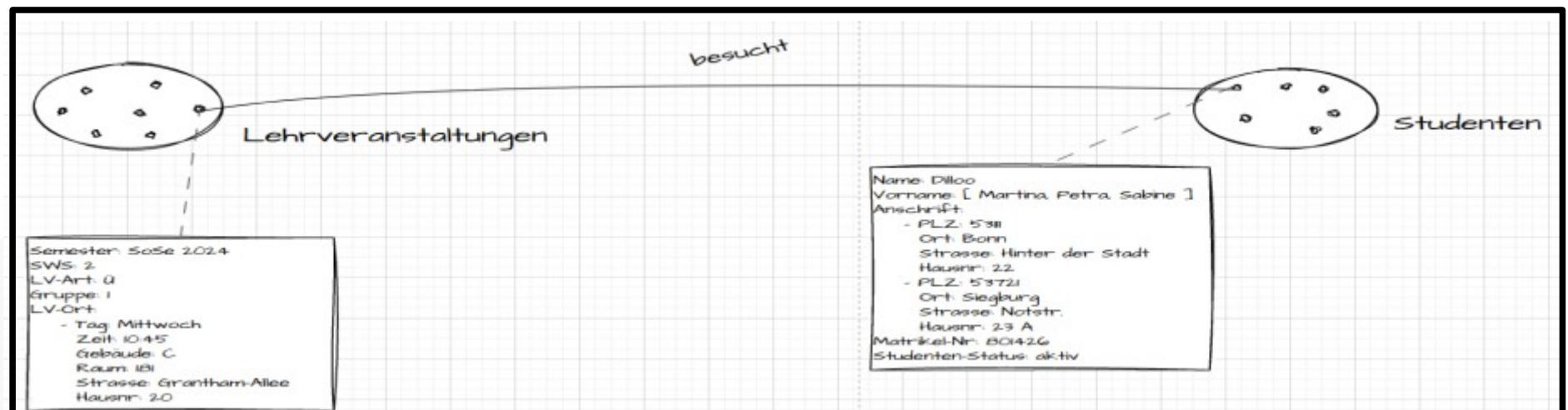
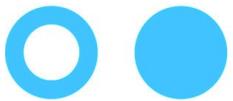
Darstellung in DB-MAIN

- Werden einige oder alle untergeordneten Entitäten gelöscht, hat dies im Allgemeinen keinen Einfluss auf die Existenz der übergeordneten Entität.
- Je nach Diskursbereich und Anwendungsfall kann in dieser Situation, dennoch die Löschung oder das Entfernen der übergeordneten Entität notwendig bzw. sinnvoll sein.



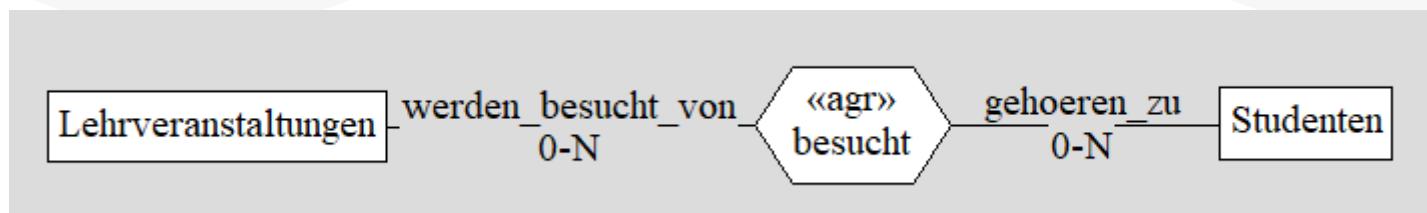
Komposition (Teil-Ganzes-Beziehung) : klassisches Beispiel „Gebäude haben Räume“

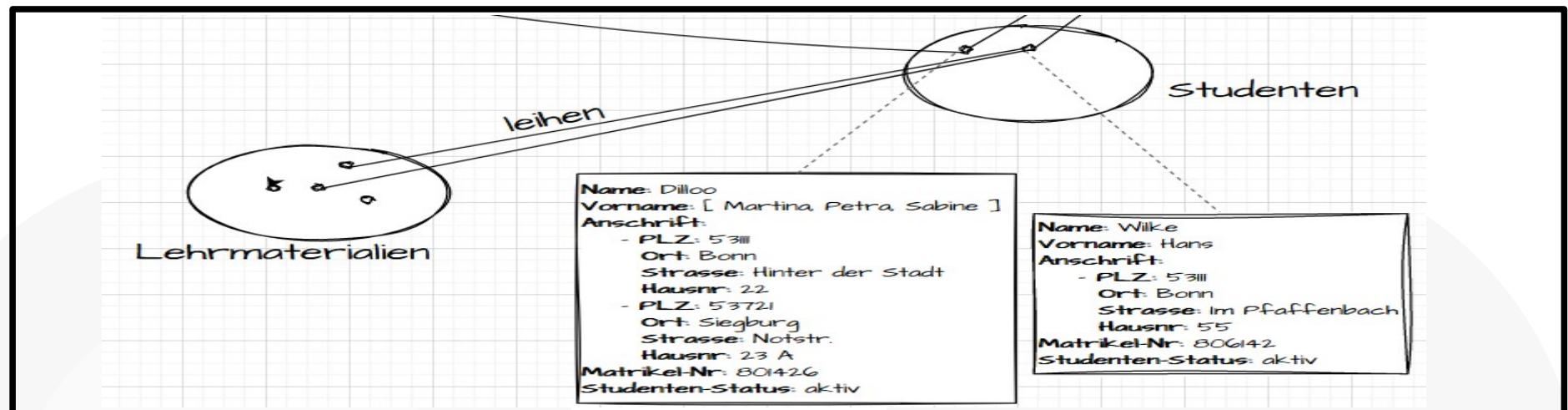
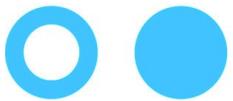




## Aggregation (Eigentums-Beziehung , Gehört-zu-Beziehung)

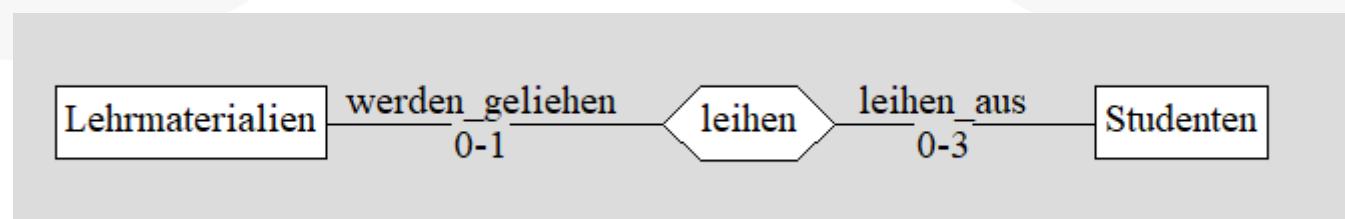
- Relationstyp als funktionale Einheit bestehend aus (gleich starken) Entitäten.
- Entitäten, die in einer Aggregation verbunden sind, haben weiterhin einen unabhängigen Lebenszyklus.
- Beteiligte Entitäten haben von einander unabhängige Identifizierungsmerkmale, d.h. unabhängige Identitäten
- „Unidirektionales Eigentum“
- „Besitzbares“ benötigt keinen „Besitzer“



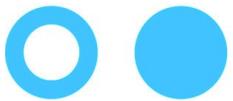


## Assoziation (Verwendungsbeziehung, Kommunikationsbeziehung)

- Relationstyp beschreibt eine lose Bindung zwischen (gleich starken) Entitäten.
- Allgemeinster Beziehungstyp
- Beteiligte Entitäten haben von einander unabhängige Identifizierungsmerkmale
- Jede Beziehung ist mindestens eine Assoziation.

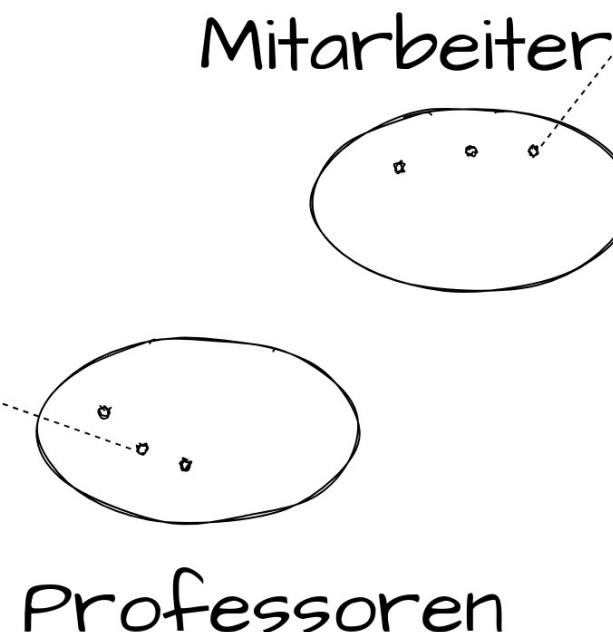


**Offene Frage:** Wie kann dasselbe Material über die Zeit mehrfach entliehen werden?



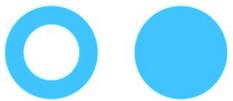
# Ist ein Professor ein Mitarbeiter?

Name: Knolle
Vorname: Harm
Anschrift:
- PLZ: 53859
Ort: Niederkassel
Strasse: Datenschutzstr.
Hausnr: 42
Personal-Nummer: 507252
Institution: Forschung und Lehre
Beruf: Professor
Gehalt: 2000
Mitarbeiter-Status: aktiv
Titel: Prof Dr. rer. nat.
Fachgebiet: Datenbanken



Name: Hartmann
Vorname: Robert
Anschrift:
- PLZ: 53859
Ort: Niederkassel
Strasse: Datenschutzstr.
Hausnr: 41
Personal-Nummer: 507899
Institution: Forschung und Lehre
Beruf: Wissenschaftlicher Mitarbeiter
Gehalt: 1000
Mitarbeiter-Status: aktiv

Wie kann man Professor Knolle auch als Mitarbeiter behandeln?



Name: Knolle  
Vorname: Harm  
Anschrift:  
- PLZ: 53859  
Ort: Niederkassel  
Strasse: Datenschutzstr.  
Hausnr: 42  
Personal-Nummer: 507252  
Institution: Forschung und Lehre  
Beruf: Professor  
Gehalt: 2000  
Mitarbeiter-Status: aktiv  
Titel: Prof Dr. rer. nat.  
Fachgebiet: Datenbanken

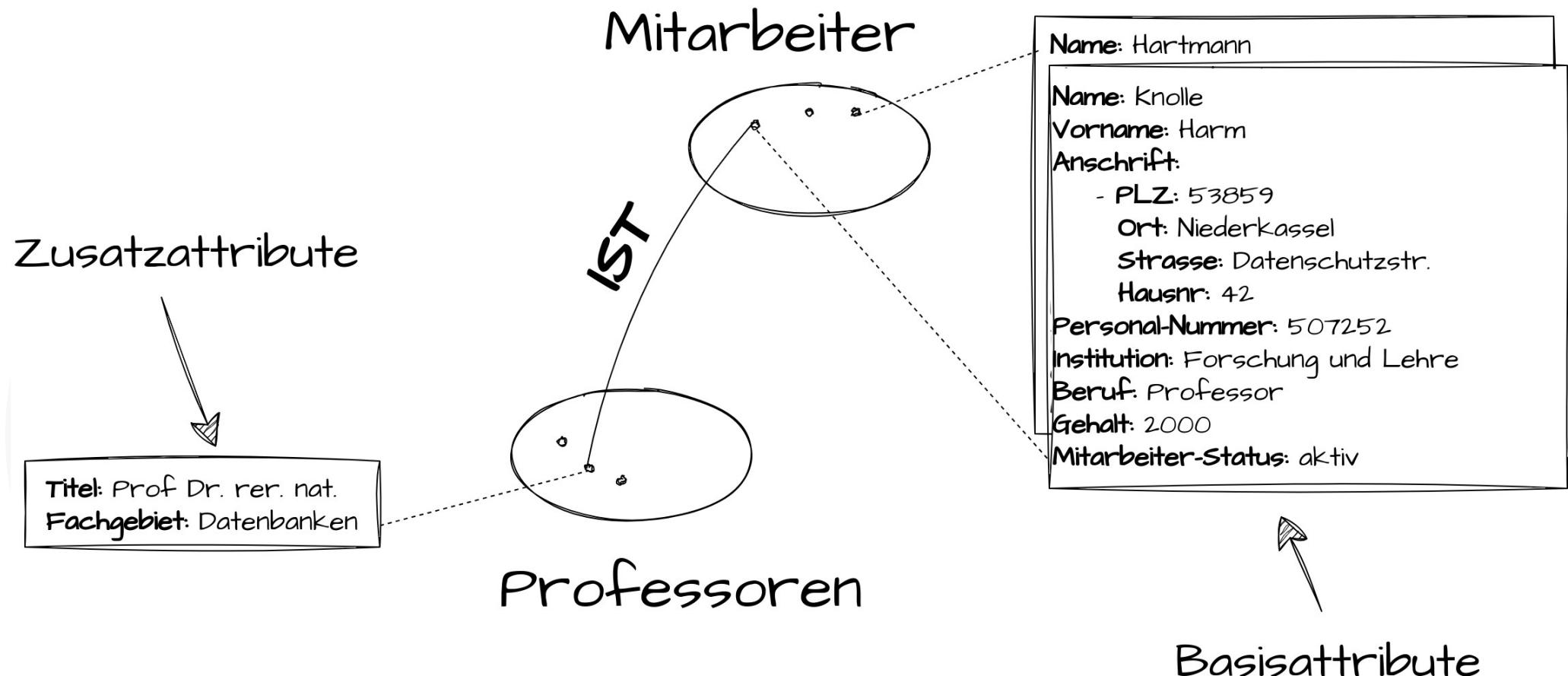
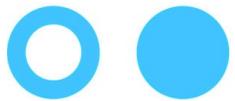
Mitarbeiter



Professoren

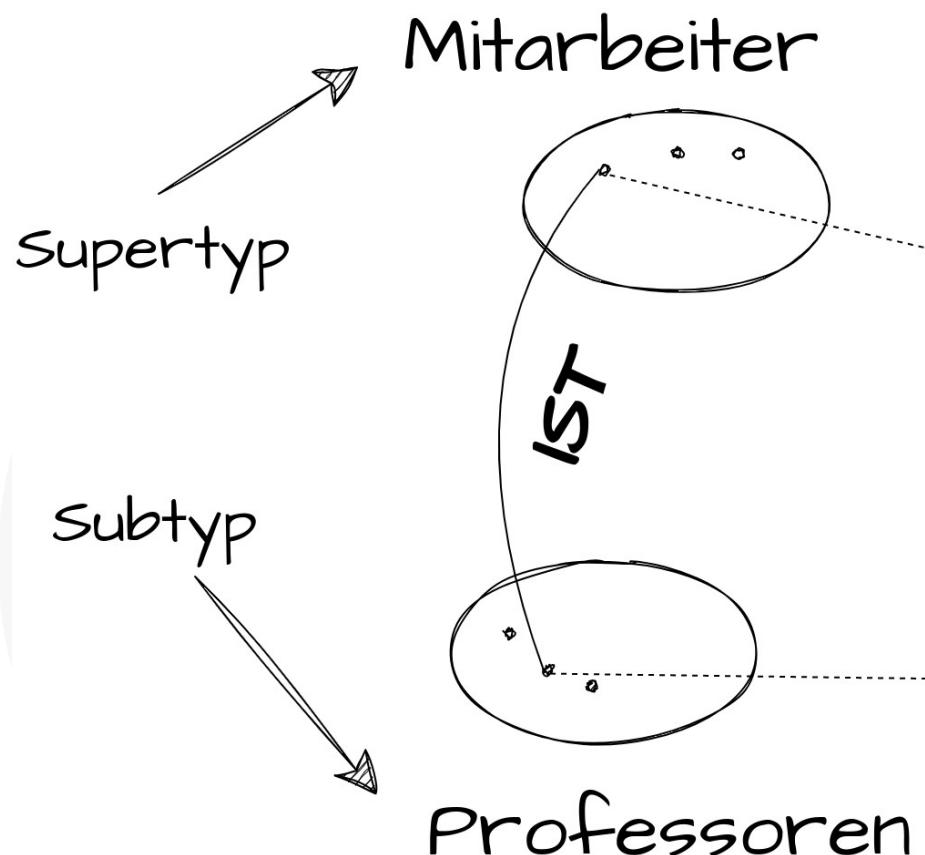
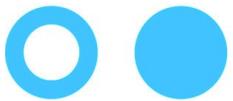
Name: Hartmann  
Name: Knolle  
Vorname: Harm  
Anschrift:  
- PLZ: 53859  
Ort: Niederkassel  
Strasse: Datenschutzstr.  
Hausnr: 42  
Personal-Nummer: 507252  
Institution: Forschung und Lehre  
Beruf: Professor  
Gehalt: 2000  
Mitarbeiter-Status: aktiv

Wie kann man die entstandene Redundanz vermeiden?



1. Aufteilung in Basis- und Zusatzattribute

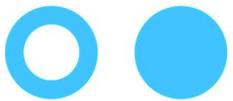
2. Beziehung "Professor IST Mitarbeiter" erfassen



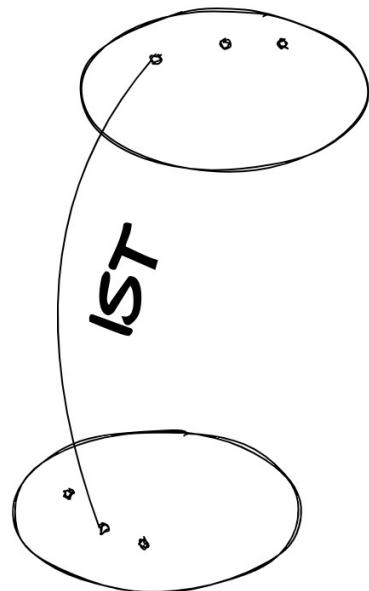
Name: Knolle  
Vorname: Harm  
Anschrift:  
- PLZ: 53859  
Ort: Niederkassel  
Strasse: Datenschutzstr.  
Hausnr: 42  
Personal-Nummer: 507252  
Institution: Forschung und Lehre  
Beruf: Professor  
Gehalt: 2000  
Mitarbeiter-Status: aktiv

Titel: Prof Dr. rer. nat.  
Fachgebiet: Datenbanken

... wird die gleiche Entität in zwei (oder mehr) Entitätsmengen aufgeteilt  
... eine Entität des Subtypen MUSS auch eine Entität des Supertypen sein



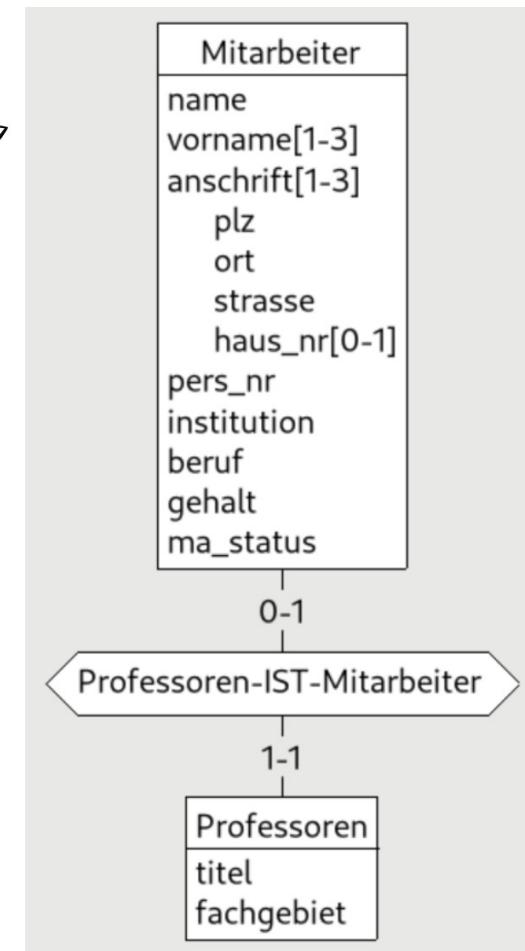
Mitarbeiter



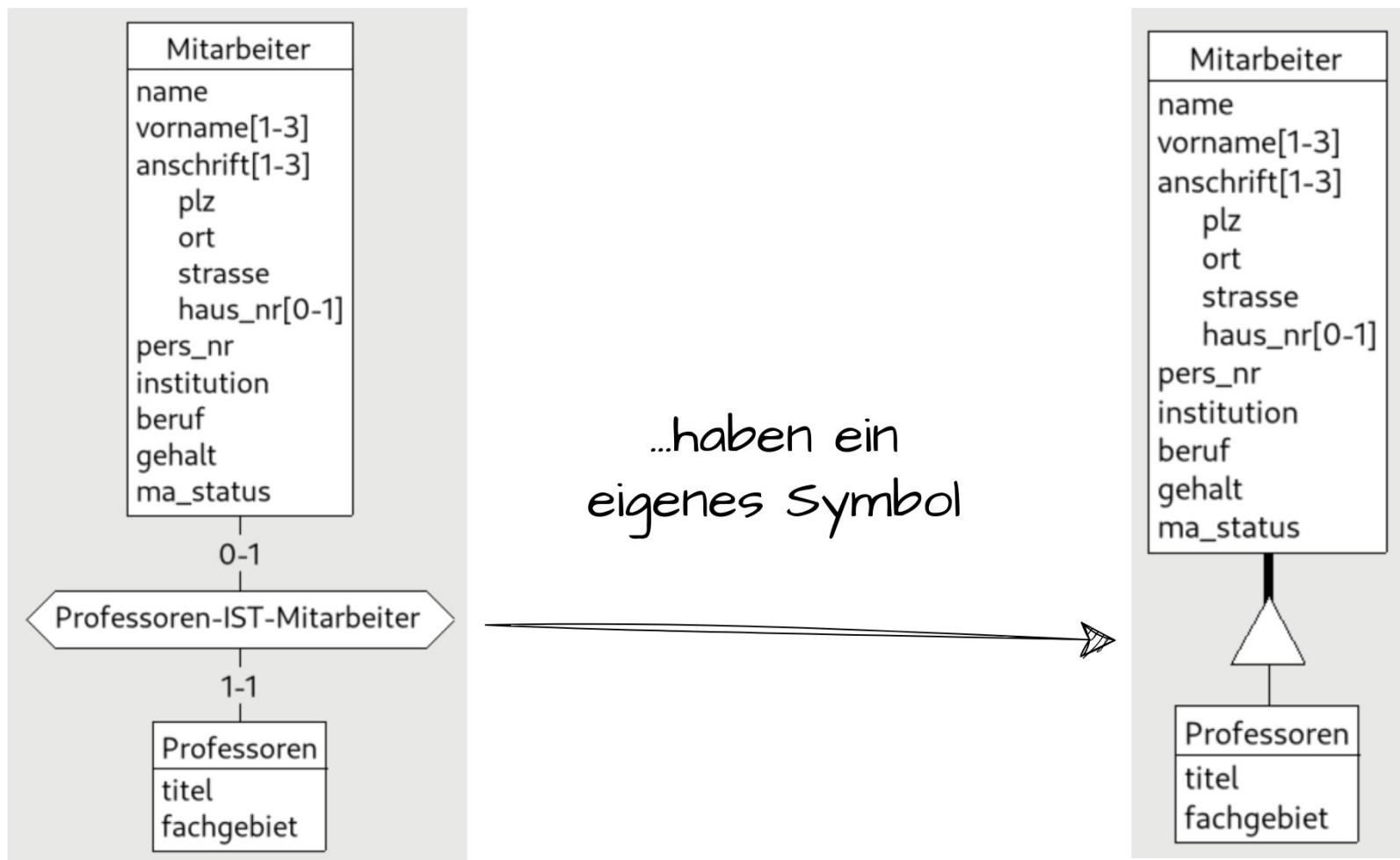
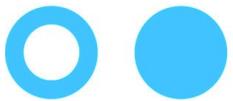
Professoren

Supertyp

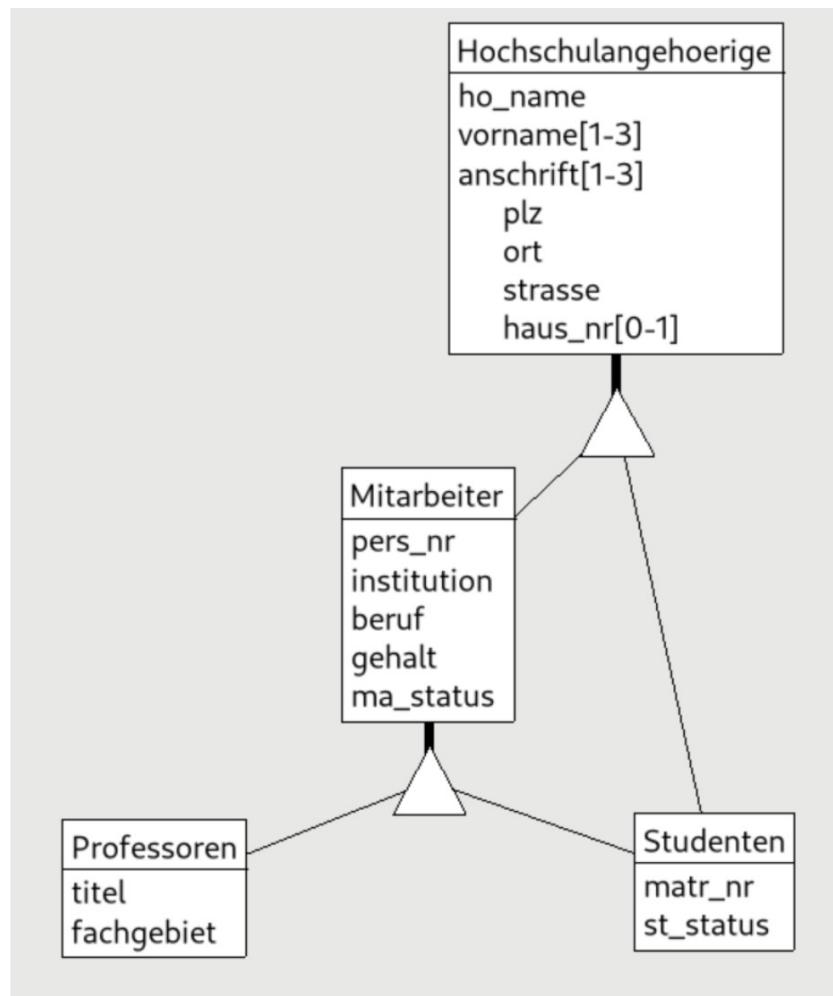
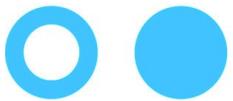
Subtyp



... wird die gleiche Entität in zwei (oder mehr) Entitätsmengen aufgeteilt  
... eine Entität des Subtypen MUSS auch eine Entität des Supertypen sein



... werden auch als Generalisierung/Spezialisierung bezeichnet



... ermöglichen Vererbungshierarchien wie in OOP



## - Generalisierung / Spezialisierung -

### Generalisierung

- Auslagerung redundanter Attribute ähnlicher Entity-Typen in einen übergeordneten Super-Typ

### Spezialisierung

- Hinzunahme spezieller Attribute eines Entity-Typen zur Erzeugung einer Menge ähnlicher untergeordneter Entity-Typen (Sub-Typen)

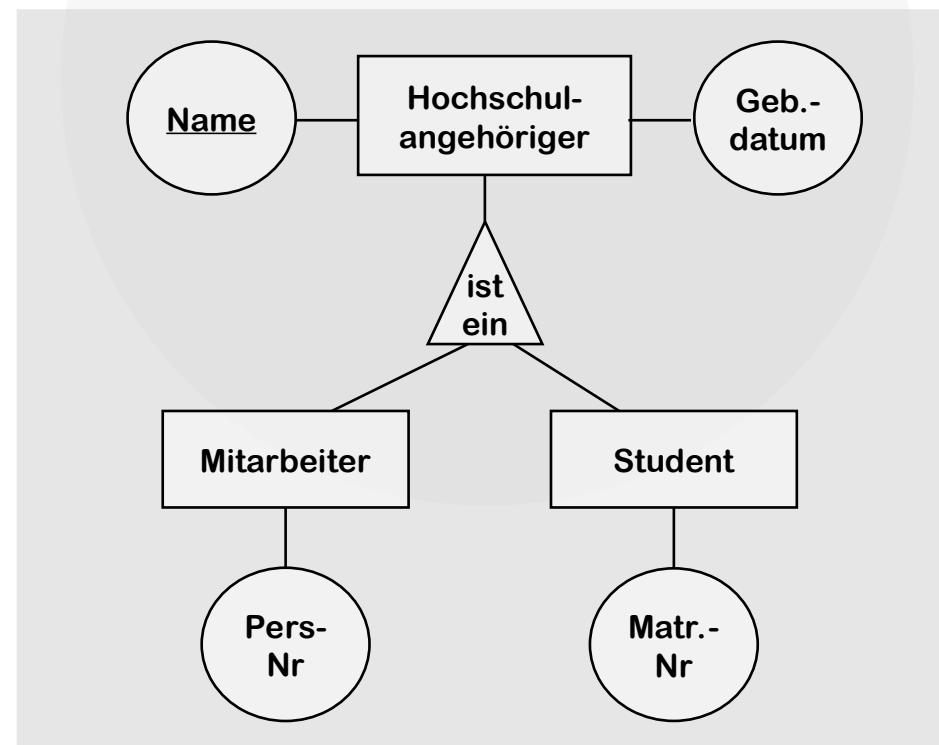
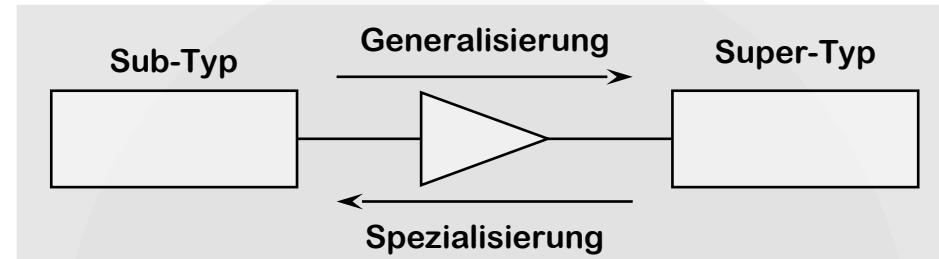
### Semantik

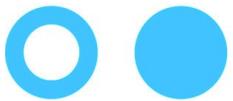
- Der Sub-Typ beschreibt keine anderen Entities als sein Super-Typ

### "is-a"-Beziehung

- Beziehung zwischen Sub-Typ und Super-Typ

### Graphische Darstellung





## - Klassifizierung der Generalisierung / Spezialisierung -

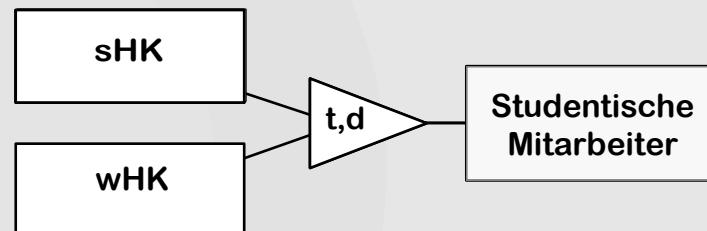
### Vier Generalisierungs- / Spezialisierungsarten

DBmain nutzt:  
 <Kein Zeichen> = partiell  
 t = total,  
 d = disjunkt,  
 p = t+d = partitioniert  
 partitioniert != partiell

d: disjunkt  
 (alternativ)

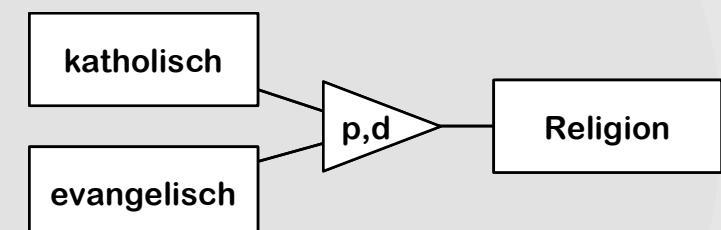
n: nicht-  
 disjunkt  
 (additiv)

t: total (alle)

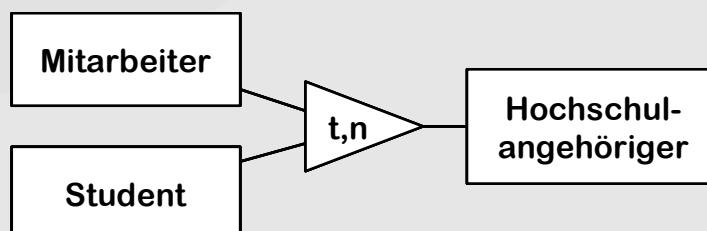


entweder studentische oder wiss. Hilfskraft  
 sHK : noch kein Bachelor-Abschluss  
 wHK: mit Bachelor-Abschluss

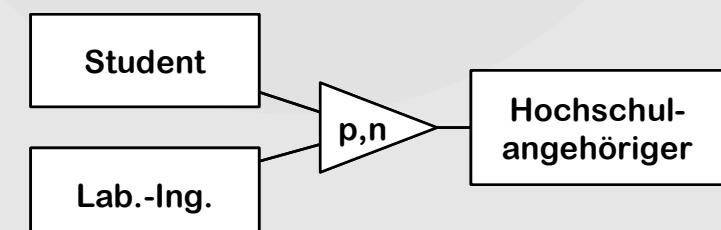
p: partiell (einige)



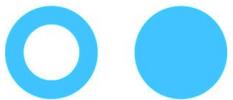
es gibt noch andere  
 Religionszugehörigkeiten



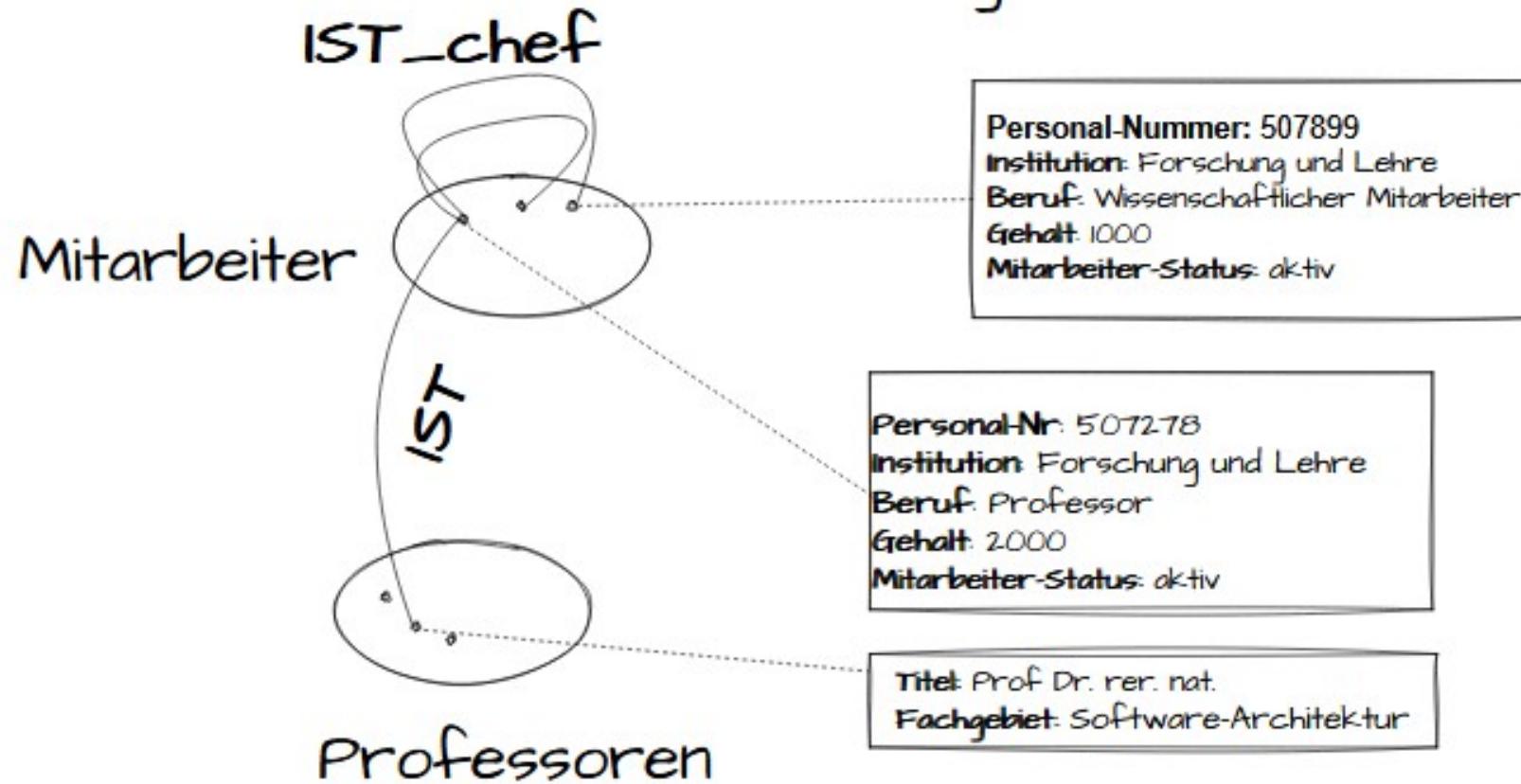
Mitarbeiter kann auch Student sein



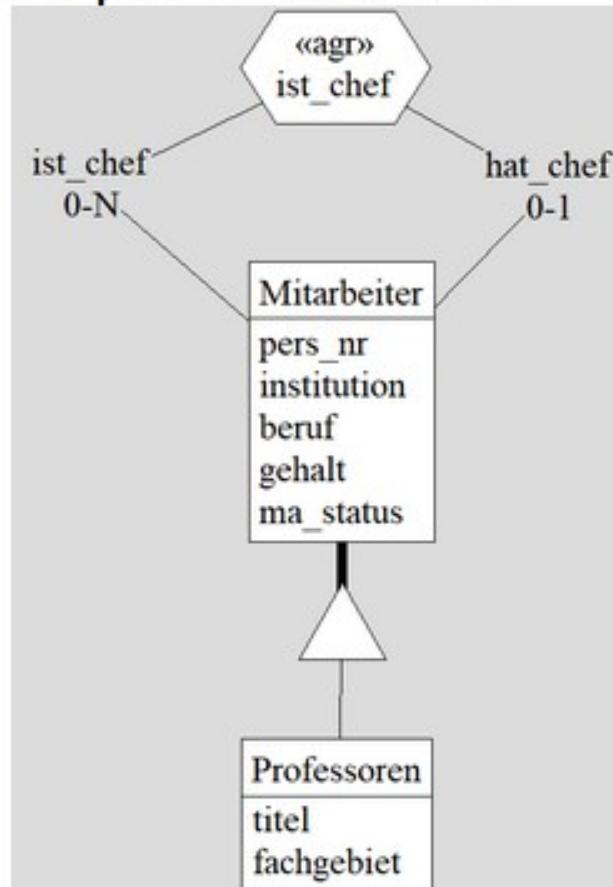
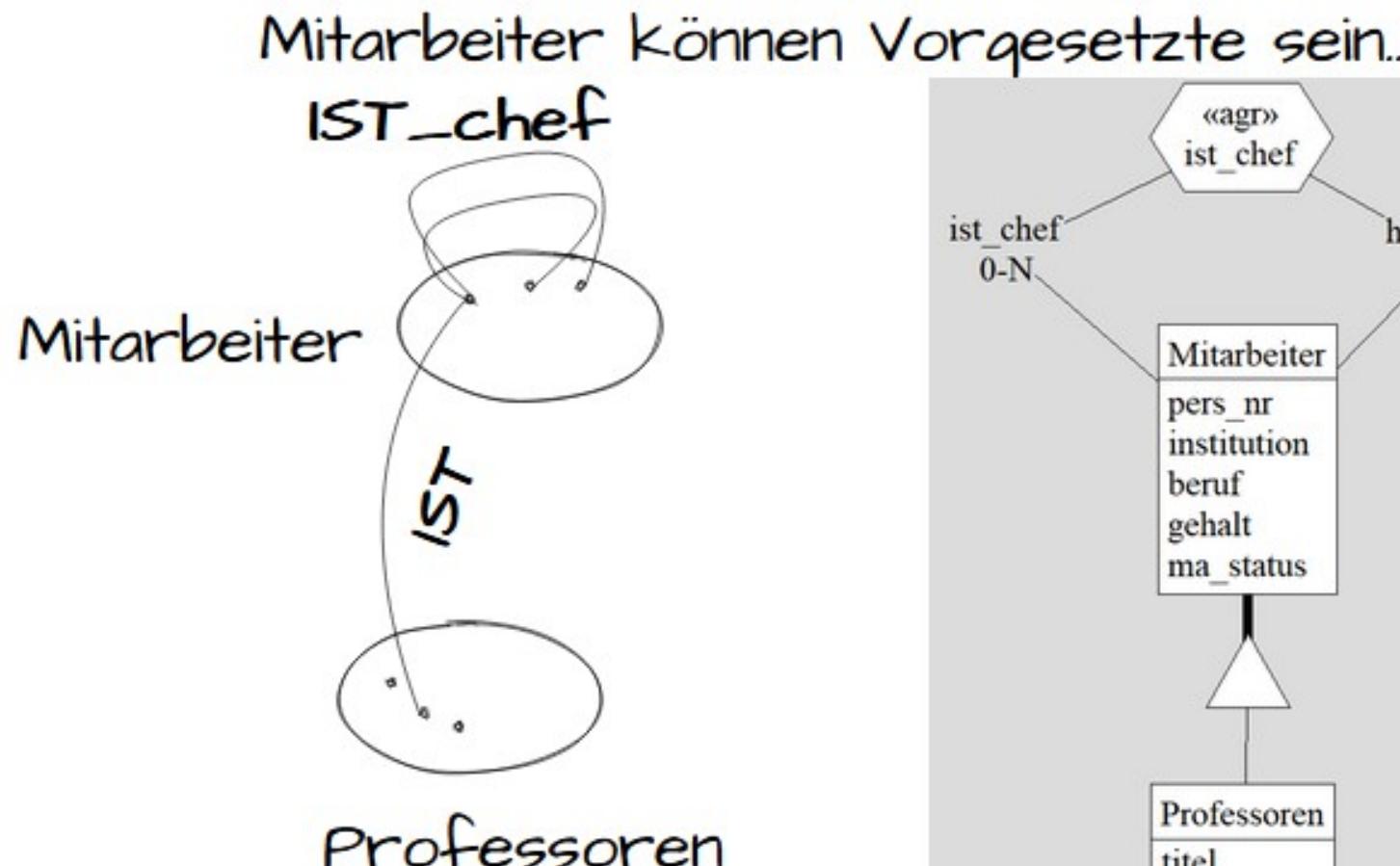
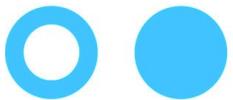
Lab.-Ing. kann auch Student sein,  
 Professoren sind auch Hochschulangehörige



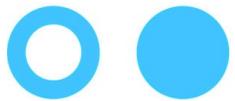
Mitarbeiter können Vorgesetzte sein.



- ... Beziehungen zwischen Entitäten innerhalb eines Entitätstyps sind möglich.
- ... Diese Art von Beziehungen werden "reflexiv" oder auch "rekursiv" genannt

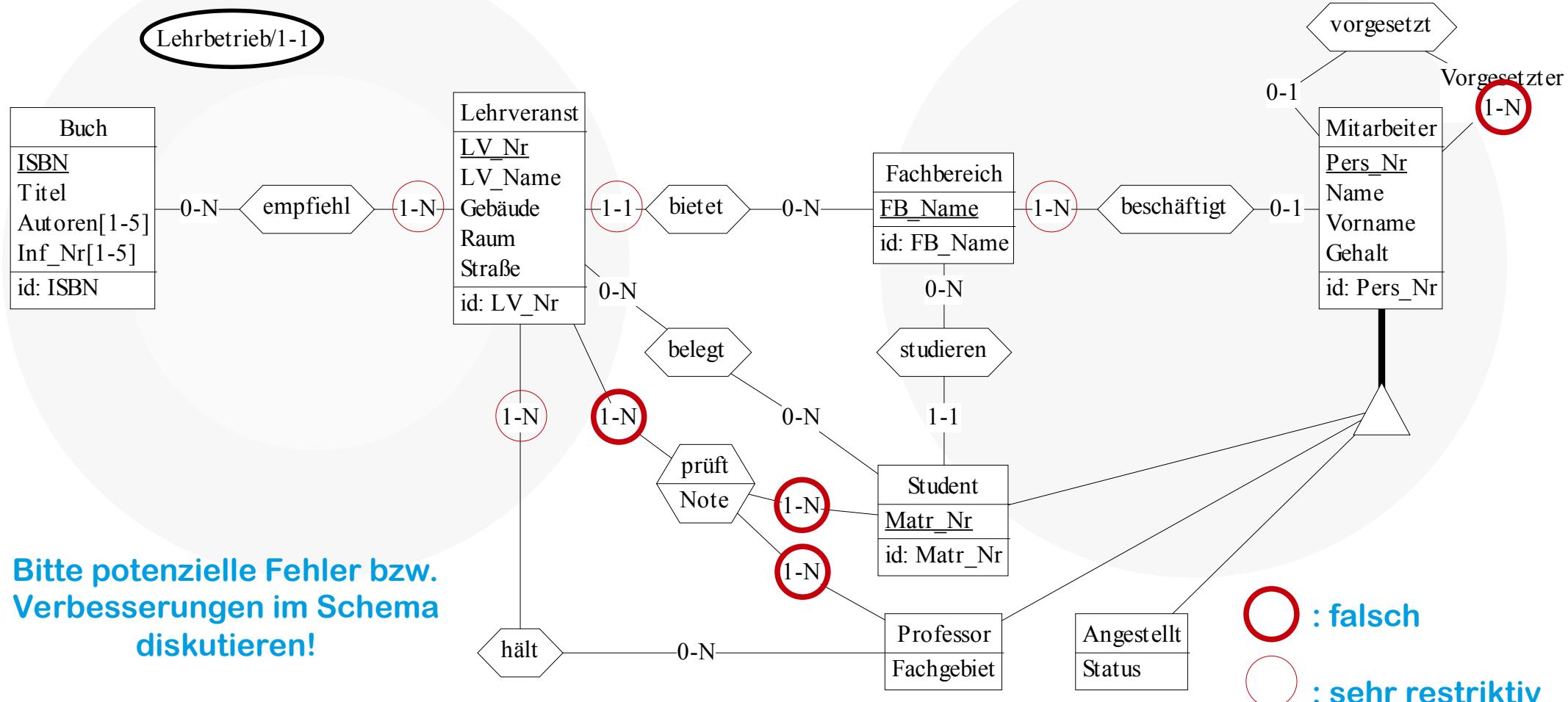


... Beschriftung der Kanten zwischen Entitätstyp und Beziehungstyp  
geben Klarheit über die Rollen innerhalb der Beziehung

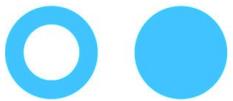


## Beispiel mit Reflexivität , Generalisierung / Spezialisierung und Min-Max-Kardinalitäten

Erstellt mit dem CASE-Werkzeug DB-MAIN



Bitte potenzielle Fehler bzw.  
Verbesserungen im Schema  
diskutieren!



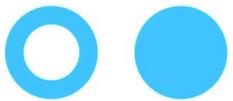
## - Zeit (Historisierung) und Versionen -

### Inhalt

- ◆ Einführung
- ◆ Entity Relationship Modell (ERM)
- ◆ Erweiterungen des ERM
- ◆ Zeit (Historisierung) und Versionen
- ◆ Objekt-orientierte Modelle

### Überblick

- ◆ Folgen der Merkmale von Beziehungen
- ◆ Abhängige Entitäten als Schlüssellieferanten



## - Folgen der Merkmale von Beziehungen -

### Folge bzw. Problem

#### Entitäten

- Schlüssellieferanten
- hat eine eigene (Objekt-)Identität
- kann ohne Beziehung existieren
- Beispiel
  - die Lehrveranstaltung „Datenbanksysteme“
  - der Student „Peter Müller“

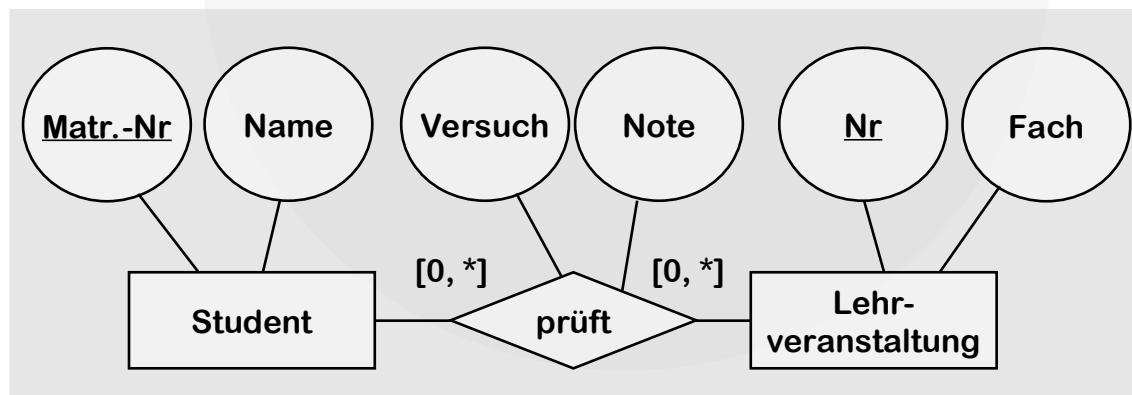
#### Beziehung

- kein Schlüssellieferant
- hat keine eigene Identität
- kann ohne zugeordnete Entitäten nicht existieren
- Identität wird über die zugeordneten Entitäten beschrieben
- Beispiel
  - die Prüfung „Datenbanksysteme“ des Studenten „Peter Müller“

- eine Beziehung vom selben Typ kann zwischen den selben Entitäten nur einmal ausgeprägt werden
- ansonsten wäre eine konkrete Beziehung nicht identifizierbar

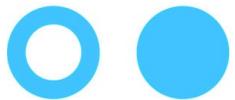
### Potenzielle Abhilfe

- Einführung von identifizierenden Beziehungsattributen wie z.B. Datum oder Version



### Weiterhin bestehendes Problem

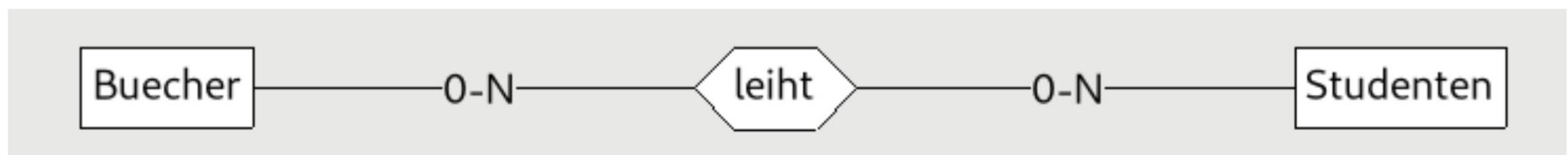
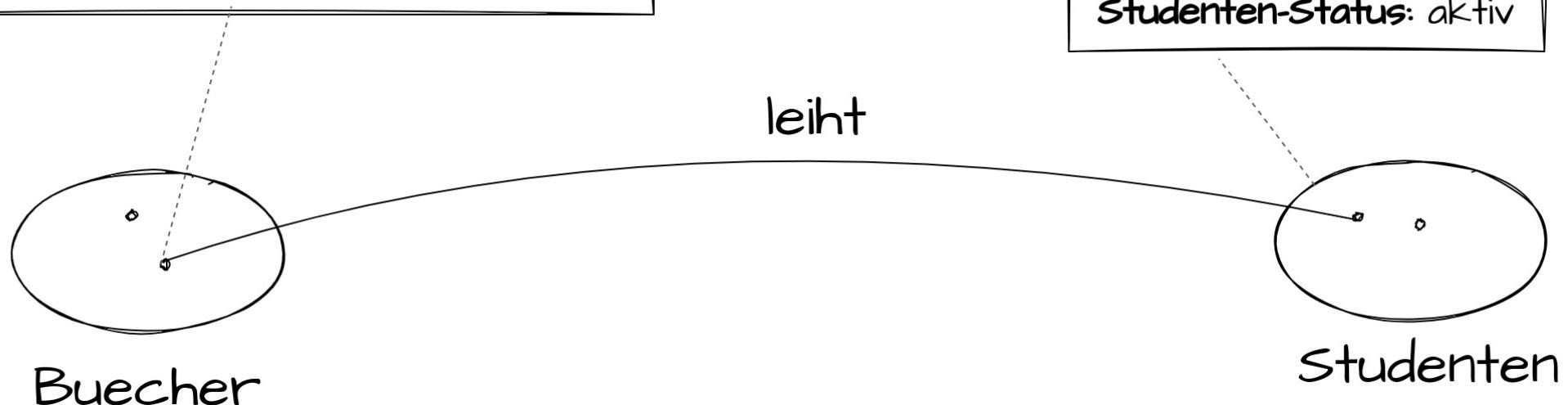
- Beziehungsattribute sind nicht Teil des Schlüssels
- eine neue Beziehung würde die alte ersetzen

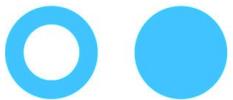


# Kann ein Student ein Buch 2x ausleihen (nacheinander) ?

**Titel:** Einführung in die Datenmodellierung  
**Autor:** Andreas Gaddatsch  
**ISBN:** 978-3-658-25730-9

**Name:** Wilke  
**Vorname:** Hans  
...  
**Matrikel-Nr.:** 806142  
**Studenten-Status:** aktiv

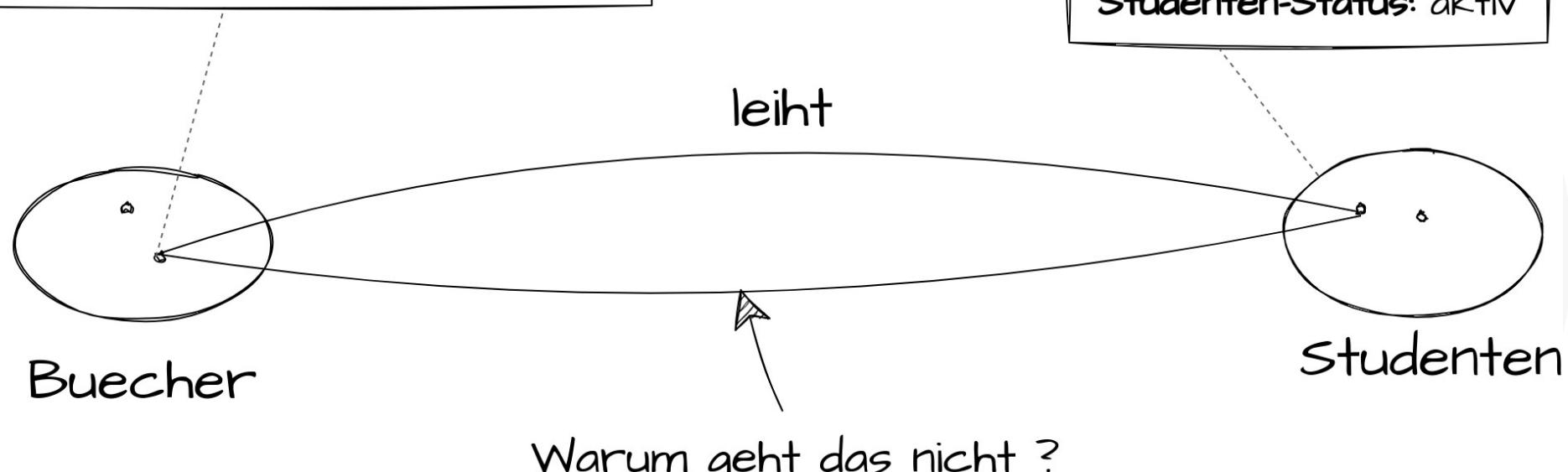




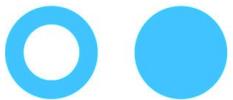
## Idee: Buch und Student 2x in Beziehung setzen

**Titel:** Einführung in die Datenmodellierung  
**Autor:** Andreas Gaddatsch  
**ISBN:** 978-3-658-25730-9

**Name:** Wilke  
**Vorname:** Hans  
...  
**Matrikel-Nr.:** 806142  
**Studenten-Status:** aktiv



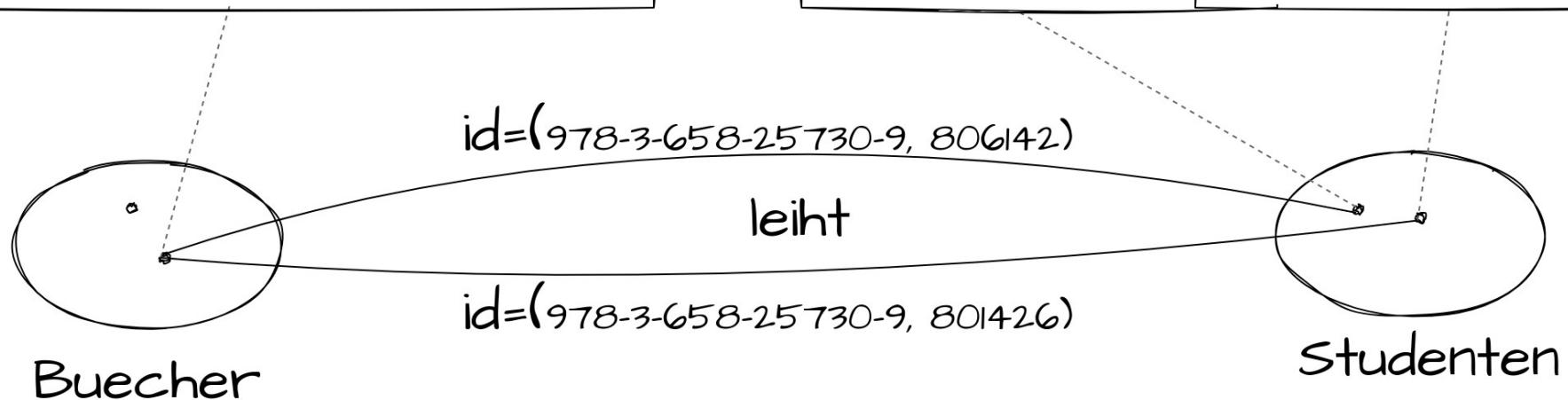
Wie werden eigentlich (konkrete) Beziehungen identifiziert?  
Haben Beziehungen auch Schlüssel?



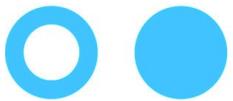
**Titel:** Einführung in die Datenmodellierung  
**Autor:** Andreas Gaddatsch  
**ISBN:** 978-3-658-25730-9

Name: Wilke  
Vorname: Hans  
...  
Matrikel-Nr: 806142  
Studenten-Status: ak

Name: Dilloo  
Vorname: ...  
...  
Matrikel-Nr: 801426  
Studenten-Status: aktiv

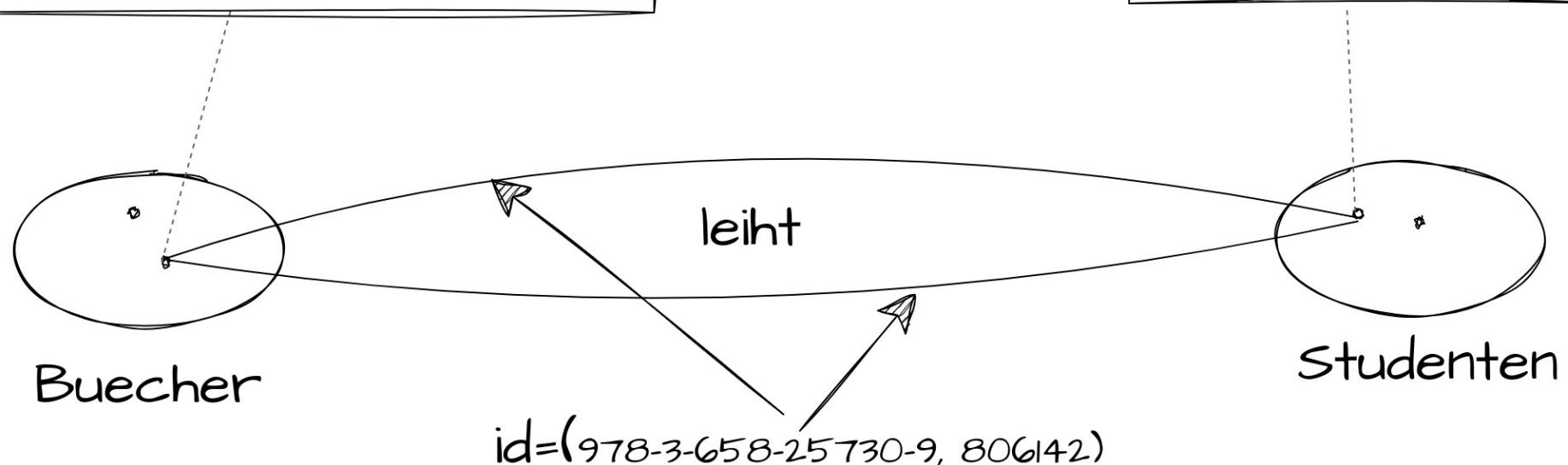


Die Identität einer Beziehung bestimmt sich aus den Identitäten der beteiligten Entitäten.

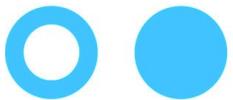


**Titel:** Einführung in die Datenmodellierung  
**Autor:** Andreas Gaddatsch  
**ISBN:** 978-3-658-25730-9

Name: Wilke  
Vorname: Hans  
...  
Matrikel-Nr: 806142  
Studenten-Status: aktiv



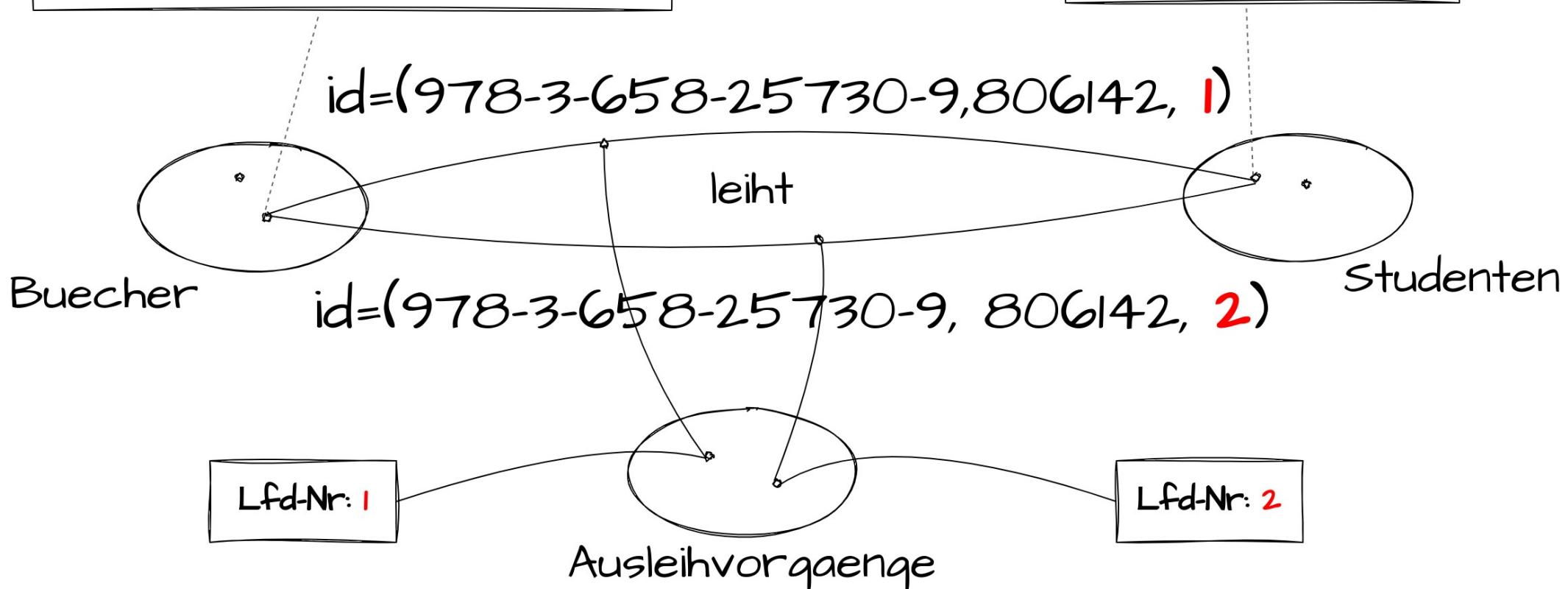
Wie können zwei Ausleihvorgänge mit unterschiedlicher Identität erfasst werden?

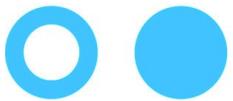


# Lösung: Versionierungsentität

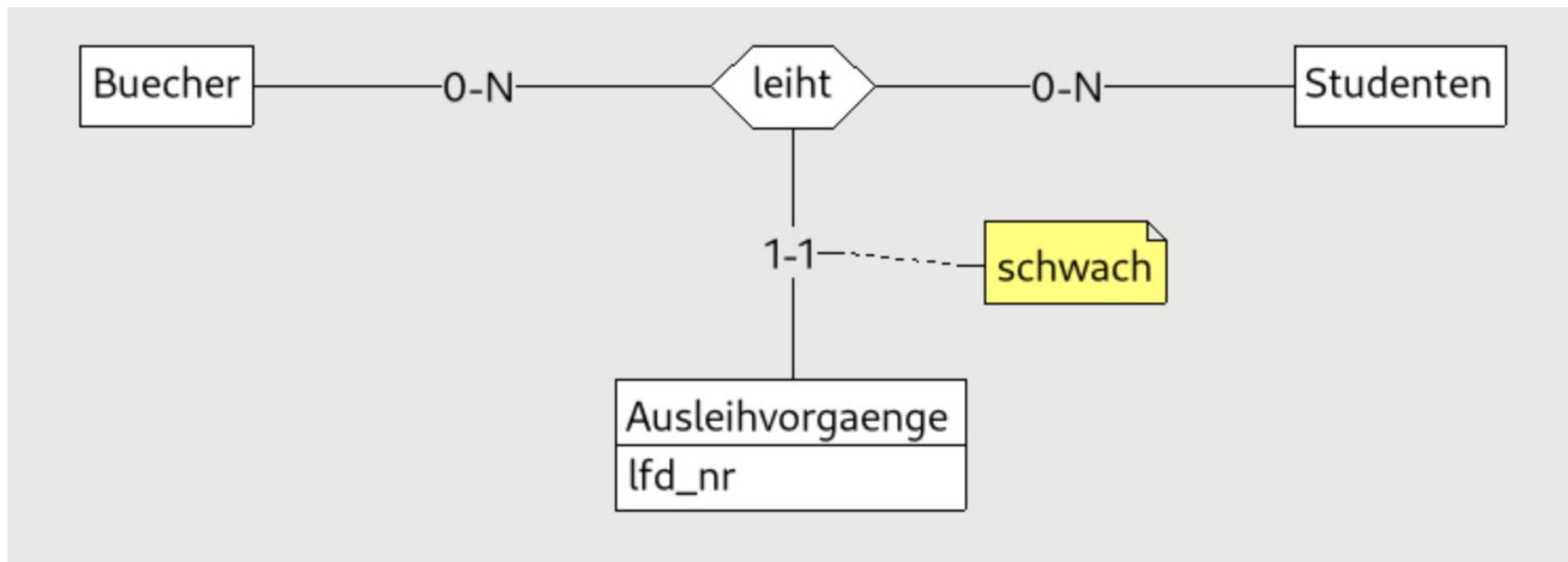
**Titel:** Einführung in die Datenmodellierung  
**Autor:** Andreas Gaddatsch  
**ISBN:** 978-3-658-25730-9

Name: Wilke  
Vorname: Hans  
...  
Matrikel-Nr: 806142  
Studenten-Status: aktiv



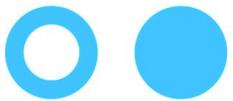


## Versionierte Beziehung im ERD



Eine Versionierungsentität...

... ist selbst eine schwache Entität  
... liefert einen weiteren Identitätsteil für die Beziehung



## - Abhängige Entitäten als Schlüssellieferanten -

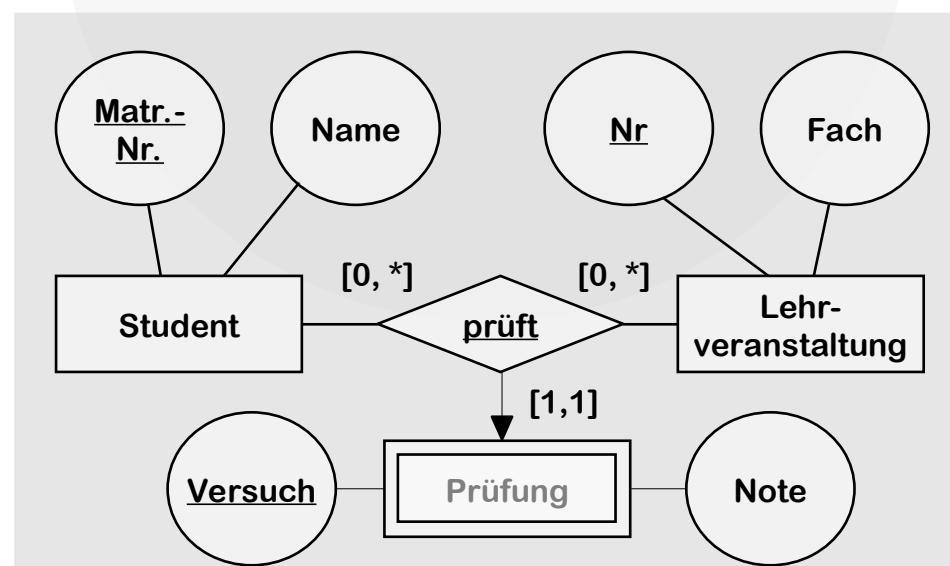
### Zu lösendes Problem

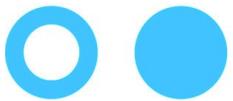
- sollen mehrere Ausprägungen eines Beziehungstyps zwischen den selben Entitäten gleichzeitig möglich und eindeutig identifizierbar sein, wird ein Schlüssellieferant benötigt
  - Versionen
  - Historie
- Merkmale wie „laufende Nummer“, „Version“, „Versuch“, „Datum“ müssen Teil des Beziehungsschlüssels werden
  - „Müller“, „Datenbanksysteme“, „1. Versuch“, 5
  - „Müller“, „Datenbanksysteme“, „2. Versuch“, 2
- Schlüssellieferant muss eine Entität sein
- der Teilschlüssel ist kein Merkmal der beteiligten Entitäten

**Achtung:** Beachte die Pfeilrichtung,  
vgl. Folie 12 („Abhängigkeit“).  
In beiden Fällen zeigt die Pfeilspitze in die  
Richtung der Schlüsselweitergabe.

### Lösung

- an der Beziehung nimmt ein weiterer abhängiger (schwacher) Entitätstyp mit der einfachen Muss-Kardinalität teil
- dieser schwache Entitätstyp übernimmt u.a. die Aufgabe des Schlüssellieferanten und somit die Rolle eines Versionszählers
- die Identifizierung erfolgt unter Ausnutzung der in Beziehung stehenden anderen Entities und deren Schlüsselattribute





## - Objekt-orientierte Modelle -

### Philosophie

- ◆ Darstellbarkeit komplex strukturierter Objekte
- ◆ Objekte erhalten lebenslange (typunabhängige) Identität
- ◆ das Schema besteht aus Klassen, deren Attribute jeweils von einem bestimmten Typ sind (Vergleichbar mit Entity-Typen)
- ◆ Klassen können hierarchisch angeordnet sein und sich hierbei Attribute vererben (Generalisierung / Spezialisierung)
- ◆ Klassen können sich gegenseitig (auch zyklisch) referenzieren (Bildung von Assoziationen, Aggregationen und Kompositionen)
- ◆ neben statischen Struktureigenschaften erhalten Klassen auch dynamische Verhaltensregeln (Methoden, Funktionen)
- ◆ Realisierung von Datenstrukturen und Methoden werden gekapselt und sind nach außen hin nicht sichtbar

- ◆ innerhalb einer Vererbungshierarchie können Methoden umdefiniert werden, um so bei den Objekten speziellerer Klassen dem Kontext angepasst agieren zu können
- ◆ fertig vordefinierte Klassen und anwendungsspezifisch definierte Klassen sind gleichgestellt

### Entwicklung

- ◆ Erweiterung der ERM um objekt-orientierte Konzepte
- ◆ Starke Verschmelzung der Konzepte objekt-orientierter Programmiersprachen mit denen semantischer Datenmodelle
- ◆ UML (Unified Modelling Language), wichtiger Schritt in Richtung der Vereinheitlichung unterschiedlicher objekt-orientierter Modellierungsansätze