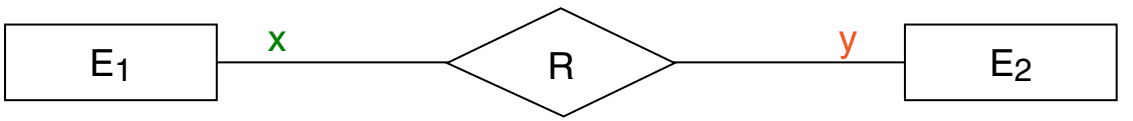
Kapitel DB:II (Fortsetzung)

II. Konzeptueller Datenbankentwurf

- Entwurfsprozess
- Datenbankmodelle
- Einführung in das Entity-Relationship-Modell
- ER-Konzepte und ihre Semantik
- Charakterisierung von Beziehungstypen
- Existenzabhängige Entity-Typen
- Abstraktionskonzepte
- Konsolidierung, Sichtenintegration
- Konzeptuelle Modellierung mit UML

Charakterisierung von Beziehungstypen

Funktionalitäten bei zweistelligen Beziehungen (Formalismus I für Kardinalitäten)

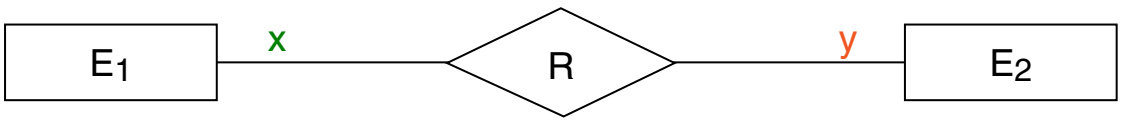


Zweistellige Beziehungstypen und ihre Semantik:

- 1:1-Beziehung
- 1:n-Beziehung
- n:m-Beziehung

Charakterisierung von Beziehungstypen

Funktionalitäten bei zweistelligen Beziehungen (Formalismus I für Kardinalitäten)



Zweistellige Beziehungstypen und ihre Semantik:

- 1:1-Beziehung
  Jedem Entity $e_1 \in \mathit{state}(E_1)$ ist höchstens ein Entity $e_2 \in \mathit{state}(E_2)$ zugeordnet und umgekehrt.

- 1:n-Beziehung
  Jedem Entity $e_1 \in \mathit{state}(E_1)$ sind beliebig viele (auch gar keine) Entities aus $\mathit{state}(E_2)$ zugeordnet, und jedem Entity $e_2 \in \mathit{state}(E_2)$ ist höchstens ein Entity $e_1 \in \mathit{state}(E_1)$ zugeordnet.

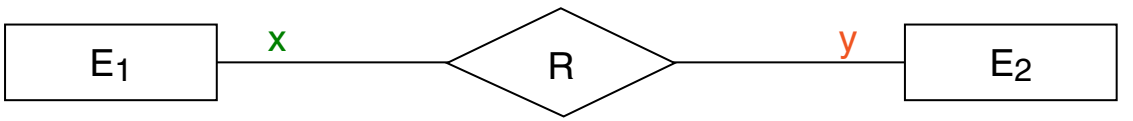
- n:m-Beziehung
  Keine Restriktionen gelten: jedes Entity $e_1 \in \mathit{state}(E_1)$ kann mit beliebig vielen Entities $e_2 \in \mathit{state}(E_2)$ in Beziehung stehen und umgekehrt.

Bemerkungen:

- Mit dem Wort „Funktionalität“ wird hier eine Abhängigkeitsbeziehung zwischen Entity-Typen bzw. zwischen Entity-Typen und einem Beziehungstyp bezeichnet. Eine andere Bezeichnung für Funktionalität ist *Constraint*.
- Die Semantik von n:1-Beziehungen ist analog zu der von 1:n-Beziehungen.
- Bei jeder Art von zweistelligen Beziehungen kann es Entities aus *state*(E_1) (bzw. *state*(E_2)) geben, denen kein Element aus *state*(E_2) (bzw. *state*(E_1)) zugeordnet ist.

Charakterisierung von Beziehungstypen

Funktionalitäten bei zweistelligen Beziehungen (Formalismus I für Kardinalitäten)



Beispiele:

- 1:1-Beziehung. „verheiratet“-Relation zwischen Männern und Frauen nach europäischem Recht.
- 1:n-Beziehung. „beschäftigen“-Relation zwischen Firmen und Personen.
- n:m-Beziehung. „unterrichtet“-Relation zwischen Studierenden und Professoren.

Bemerkungen:

- Funktionalitäten stellen *Integritätsbedingungen* dar, die in der zu modellierenden Welt immer gelten müssen.

- Die binären 1:1, 1:n und n:1-Beziehungen stellen partielle Funktionen dar. Insbesondere ist jede 1:1-Beziehung umkehrbar, und es existieren folgende Funktionen:
  $R : E_1 \rightarrow E_2$
  $R^{-1} : E_2 \rightarrow E_1$

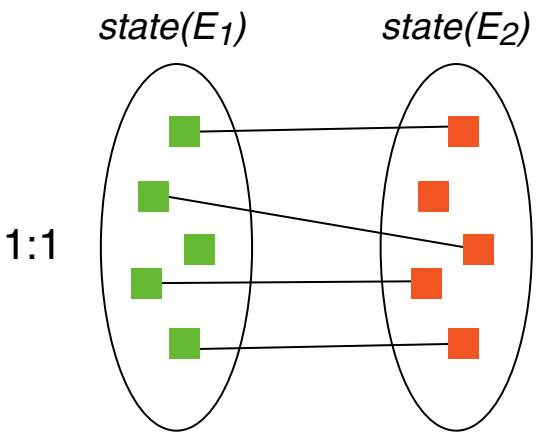
  Beispiel:
  Ehemann: Frauen $\rightarrow$ Männer
  Ehefrau: Männer $\rightarrow$ Frauen

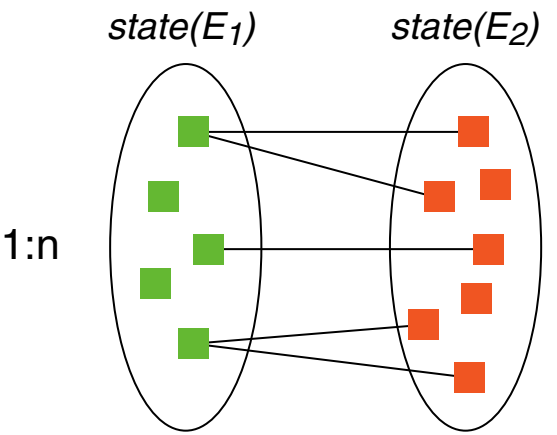
- Bei einer 1:n-Beziehung ist die Richtung der Funktion zwingend, vom „n-Entity-Typ“ zum „1-Entity-Typ“.

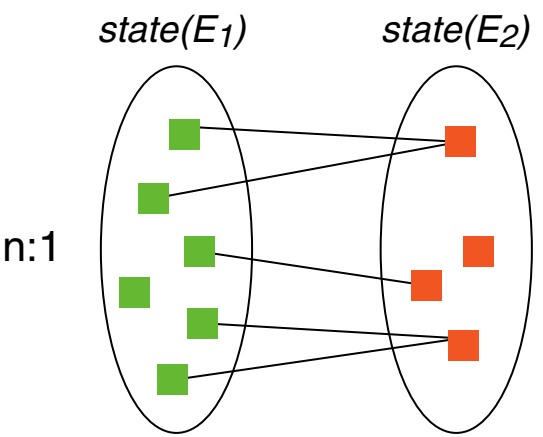
  Beispiel:
  beschäftigen: Personen $\rightarrow$ Firmen

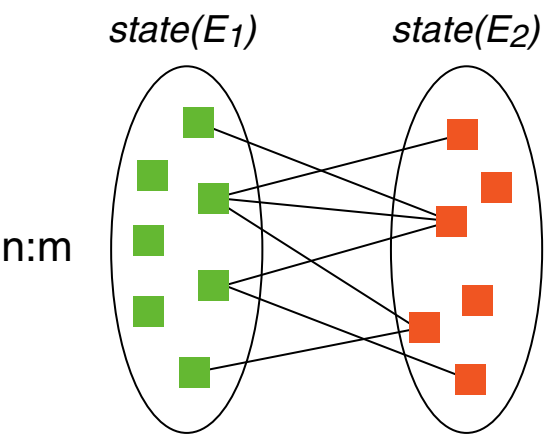
Charakterisierung von Beziehungstypen

Funktionalitäten bei zweistelligen Beziehungen (Formalismus I für Kardinalitäten)





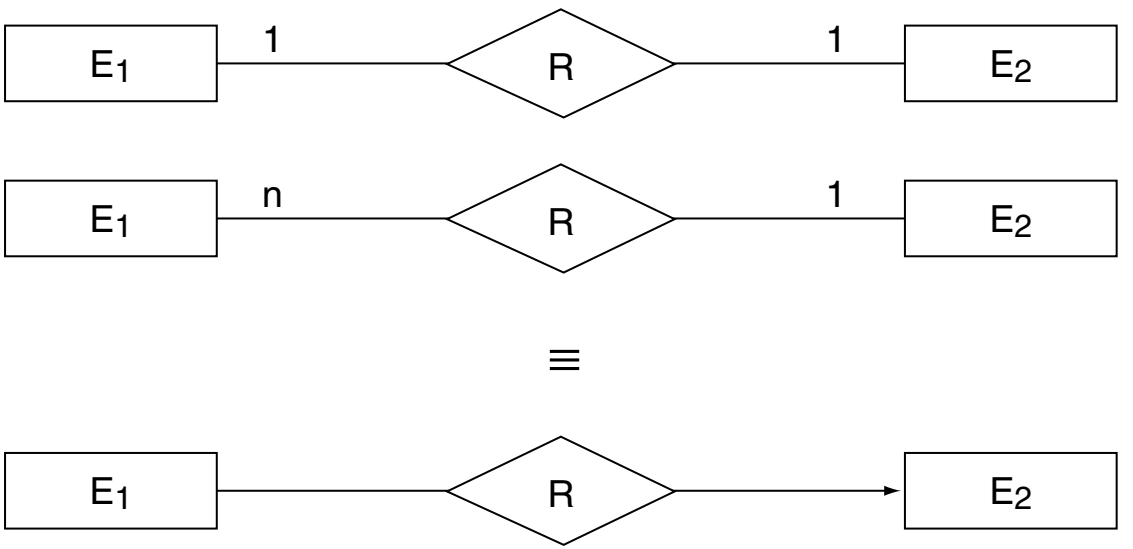




Charakterisierung von Beziehungstypen

Funktionalitäten bei zweistelligen Beziehungen (Formalismus I für Kardinalitäten)

Alternative graphische Darstellungen für zweistellige funktionale Beziehungen:



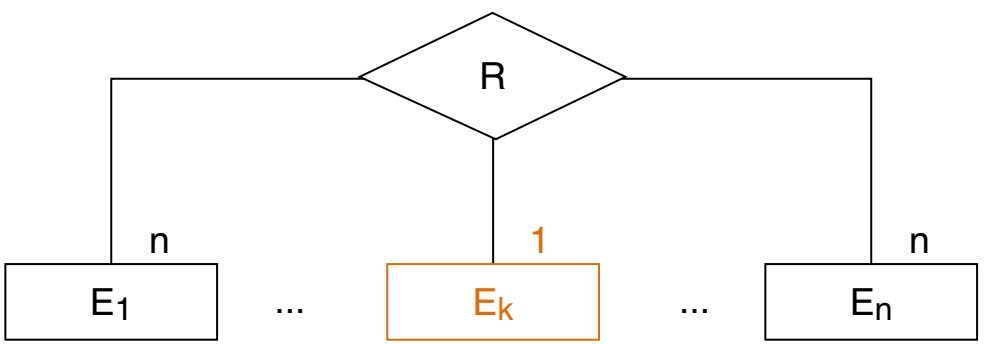
Charakterisierung von Beziehungstypen

Funktionalitäten bei n-stelligen Beziehungen (Formalismus I für Kardinalitäten)

Sei R eine Beziehung zwischen den Entity-Typen $E_1, \ldots, E_n$, wobei die Funktionalität bei Entity-Typ E_k, $1 \leq k \leq n$, mit 1 spezifiziert ist. Dann wird durch R folgende partielle Funktion vorgegeben:

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

[Kemper/Eickler 2011]



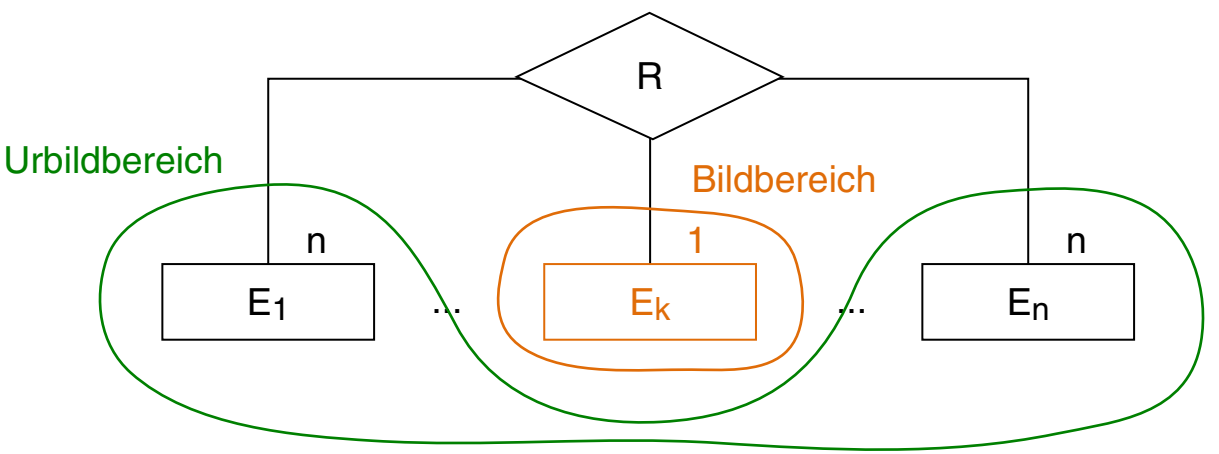
Charakterisierung von Beziehungstypen

Funktionalitäten bei n-stelligen Beziehungen (Formalismus I für Kardinalitäten)

Sei R eine Beziehung zwischen den Entity-Typen $E_1, \ldots, E_n$, wobei die Funktionalität bei Entity-Typ E_k, $1 \leq k \leq n$, mit 1 spezifiziert ist. Dann wird durch R folgende partielle Funktion vorgegeben:

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

[Kemper/Eickler 2011]



Bemerkungen:

- n-stellige (n-äre) Beziehungstypen spezifizieren genau *die* Menge der Funktionen, für deren Signatur gilt: die Bildmenge besteht aus einem Entity-Typ mit der Kardinalität 1; die Urbildmenge ist das kartesische Produkt der restlichen $n-1$ Entity-Typen.
- Diese Definition von n-stelligen Beziehungstypen erweitert die binären n:1-Beziehungen in kanonischer Weise. Dabei ist der 1-Entity-Typ der rechten Seite in der Rolle des rechten 1-Entity-Typs der binären Funktionalität; die restlichen $n-1$ Entity-Typen sind zusammen in der Rolle des linken n-Entity-Typs der binären Funktionalität.
- n-stellige Beziehungstypen sind in der Regel *keine* Kurzschreibweise von $\binom{n}{2}$ binären Beziehungstypen. Selbst eine ternäre 1:1:1-Relation R lässt sich nicht zwangsläufig verlustlos bzw. verbundtreu in binäre Beziehungstypen zerlegen.

Charakterisierung von Beziehungstypen

Funktionalitäten bei n-stelligen Beziehungen (Formalismus I für Kardinalitäten)

Beispiel Prüfungsordnung (= Ausschnitt der realen Welt) :

(a) Ein Student darf bei demselben Professor nur ein Seminarthema machen.

(b) Ein Student darf dasselbe Thema nur einmal bearbeiten und nicht zu einem anderen Professor damit gehen.

(c) Ein Professor kann dasselbe Thema an verschiedene Studenten vergeben.

(d) Dasselbe Thema kann von verschiedenen Professoren vergeben werden.

Konzeptuelles Modell (= ER-Diagramm) :

?

Charakterisierung von Beziehungstypen

Funktionalitäten bei n-stelligen Beziehungen (Formalismus I für Kardinalitäten)

Beispiel Prüfungsordnung (= Ausschnitt der realen Welt) :

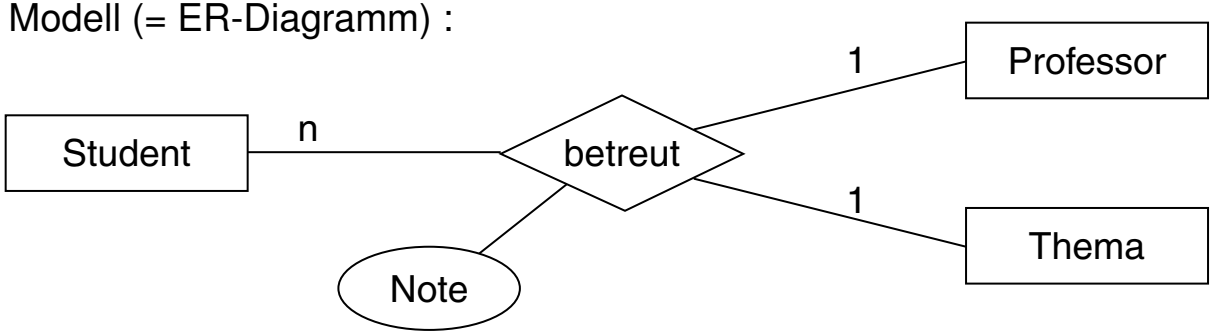
(a) Ein Student darf bei demselben Professor nur ein Seminarthema machen.

(b) Ein Student darf dasselbe Thema nur einmal bearbeiten und nicht zu einem anderen Professor damit gehen.

(c) Ein Professor kann dasselbe Thema an verschiedene Studenten vergeben.

(d) Dasselbe Thema kann von verschiedenen Professoren vergeben werden.

Konzeptuelles Modell (= ER-Diagramm) :



Charakterisierung von Beziehungstypen

Funktionalitäten bei n-stelligen Beziehungen (Formalismus I für Kardinalitäten)

Beispiel Prüfungsordnung (= Ausschnitt der realen Welt) :

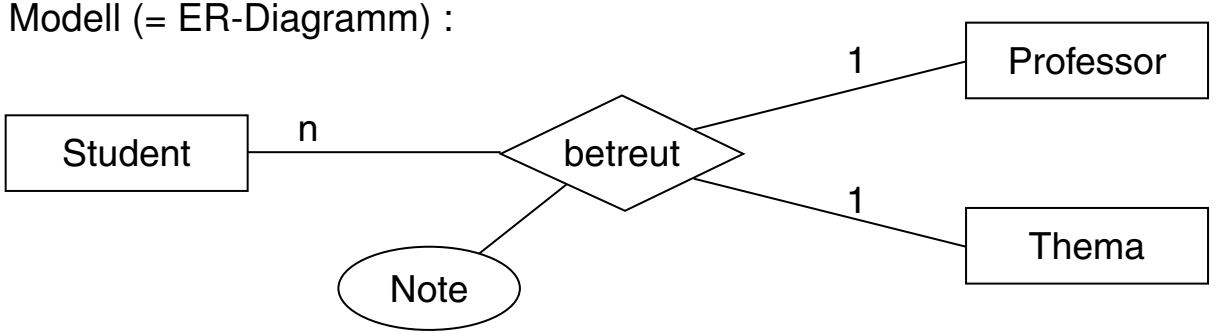
(a) Ein Student darf bei demselben Professor nur ein Seminarthema machen.

(b) Ein Student darf dasselbe Thema nur einmal bearbeiten und nicht zu einem anderen Professor damit gehen.

(c) Ein Professor kann dasselbe Thema an verschiedene Studenten vergeben.

(d) Dasselbe Thema kann von verschiedenen Professoren vergeben werden.

Konzeptuelles Modell (= ER-Diagramm) :



Analyse des konzeptuellen Modells:

zu (a)

zu (b)

zu (c)

zu (d)

Charakterisierung von Beziehungstypen

Funktionalitäten bei n-stelligen Beziehungen (Formalismus I für Kardinalitäten)

Beispiel Prüfungsordnung (= Ausschnitt der realen Welt) :

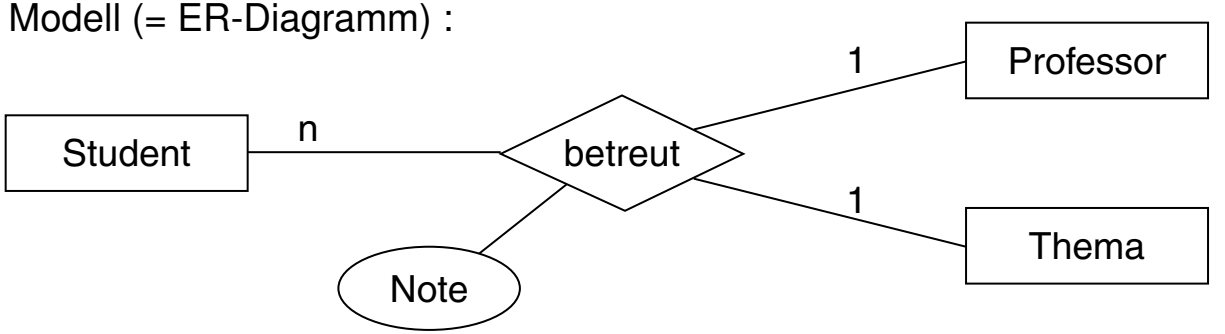
(a) Ein Student darf bei demselben Professor nur ein Seminarthema machen.

(b) Ein Student darf dasselbe Thema nur einmal bearbeiten und nicht zu einem anderen Professor damit gehen.

(c) Ein Professor kann dasselbe Thema an verschiedene Studenten vergeben.

(d) Dasselbe Thema kann von verschiedenen Professoren vergeben werden.

Konzeptuelles Modell (= ER-Diagramm) :



Analyse des konzeptuellen Modells:

zu (a) Abgebildet durch $betreut_{f_1}$: Professor $\times$ Student $\rightarrow$ Thema

zu (b)

zu (c)

zu (d)

Charakterisierung von Beziehungstypen

Funktionalitäten bei n-stelligen Beziehungen (Formalismus I für Kardinalitäten)

Beispiel Prüfungsordnung (= Ausschnitt der realen Welt) :

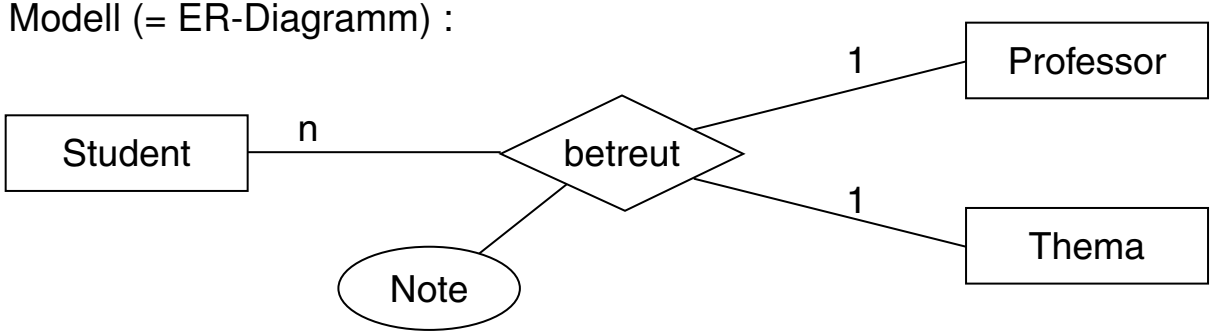
(a) Ein Student darf bei demselben Professor nur ein Seminarthema machen.

(b) Ein Student darf dasselbe Thema nur einmal bearbeiten und nicht zu einem anderen Professor damit gehen.

(c) Ein Professor kann dasselbe Thema an verschiedene Studenten vergeben.

(d) Dasselbe Thema kann von verschiedenen Professoren vergeben werden.

Konzeptuelles Modell (= ER-Diagramm) :



Analyse des konzeptuellen Modells:

zu (a) Abgebildet durch $\mathit{betreut}_{f_1}$: Professor $\times$ Student $\rightarrow$ Thema

zu (b) Abgebildet durch $\mathit{betreut}_{f_2}$: Thema $\times$ Student $\rightarrow$ Professor

zu (c)

zu (d)

Charakterisierung von Beziehungstypen

Funktionalitäten bei n-stelligen Beziehungen (Formalismus I für Kardinalitäten)

Beispiel Prüfungsordnung (= Ausschnitt der realen Welt) :

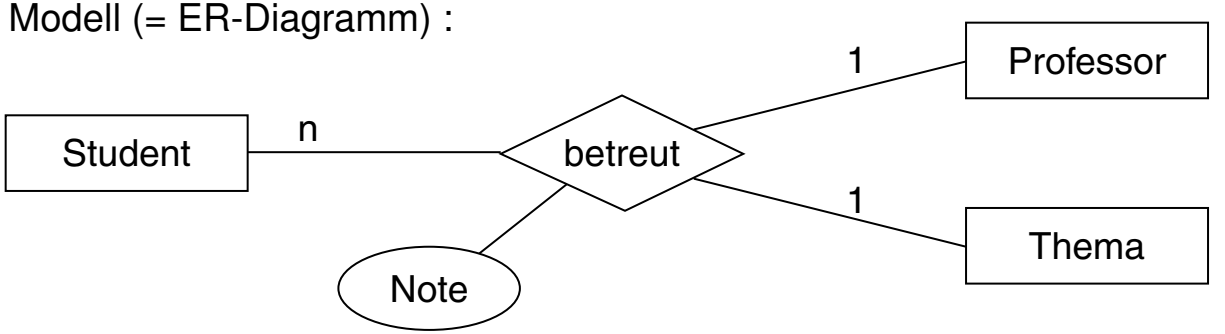
(a) Ein Student darf bei demselben Professor nur ein Seminarthema machen.

(b) Ein Student darf dasselbe Thema nur einmal bearbeiten und nicht zu einem anderen Professor damit gehen.

(c) Ein Professor kann dasselbe Thema an verschiedene Studenten vergeben.

(d) Dasselbe Thema kann von verschiedenen Professoren vergeben werden.

Konzeptuelles Modell (= ER-Diagramm) :



Analyse des konzeptuellen Modells:

zu (a) Abgebildet durch $\mathit{betreut}_{f_1} : \text{Professor} \times \text{Student} \rightarrow \text{Thema}$

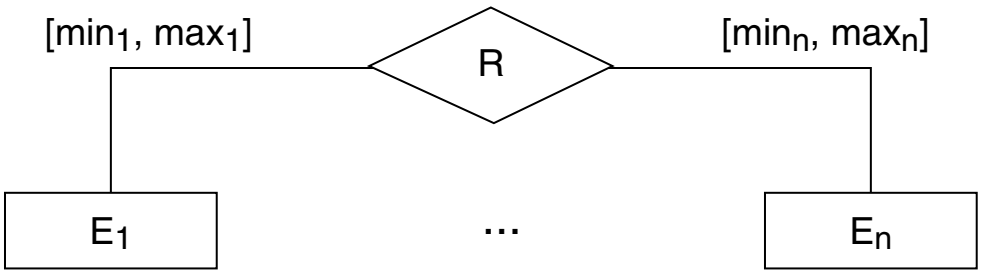
zu (b) Abgebildet durch $\mathit{betreut}_{f_2} : \text{Thema} \times \text{Student} \rightarrow \text{Professor}$

zu (c) Zugelassen durch $\mathit{betreut}_{f_1}$.

zu (d) Zugelassen durch $\mathit{betreut}_{f_2}$.

Charakterisierung von Beziehungstypen

[*min*, *max*]-Notation (Formalismus II für Kardinalitäten)

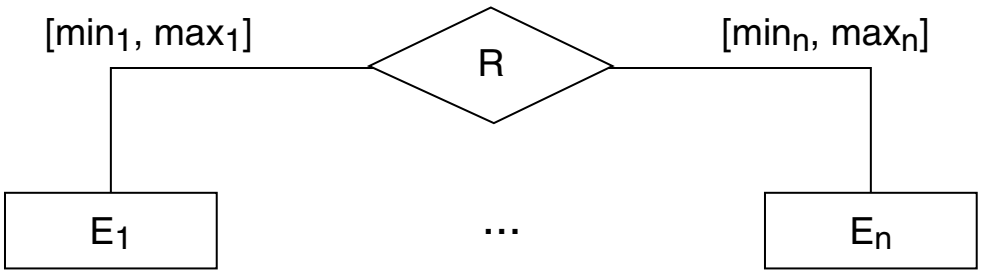


Schreibweise:

$$R(E_1[min_1, max_1], \ldots, E_i[min_i, max_i], \ldots, E_n[min_n, max_n])$$

Charakterisierung von Beziehungstypen

$[min, max]$-Notation (Formalismus II für Kardinalitäten)



Schreibweise:

$$R(E_1[min_1, max_1], \ldots, E_i[min_i, max_i], \ldots, E_n[min_n, max_n])$$

Semantik der $[min, max]$-Notation:

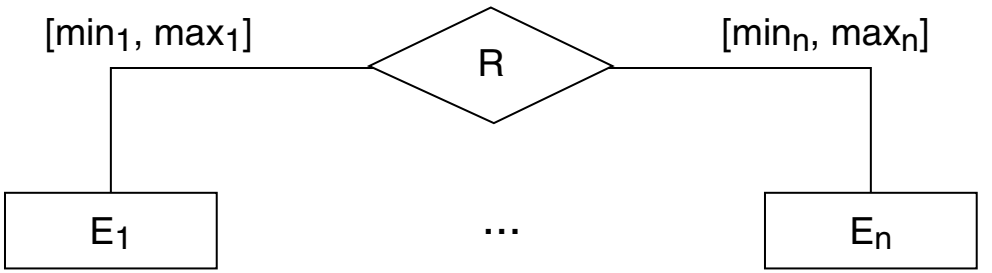
Die Anzahl der Beziehungsinstanzen vom Typ R mit Beteiligung der Instanz e_i vom Typ E_i ist zwischen min_i und max_i:

$$\forall i \in \{1, \ldots, n\} \;\; \forall e_i \in state(E_i) : \;\; min_i \leq |\{t \in state(R) \mid t.E_i = e_i\}| \leq max_i$$

$t.E_i$ bezeichnet die Projektion der Beziehungsinstanz t auf den Entity-Typ E_i.

Charakterisierung von Beziehungstypen

$[min, max]$-Notation (Formalismus II für Kardinalitäten)



Schreibweise:

$$R(E_1[min_1, max_1], \ldots, E_i[min_i, max_i], \ldots, E_n[min_n, max_n])$$

Semantik der $[min, max]$-Notation:

Die Anzahl der Beziehungsinstanzen vom Typ R mit Beteiligung der Instanz e_i vom Typ E_i ist zwischen min_i und max_i:

$$\forall i \in \{1, \ldots, n\} \;\; \forall e_i \in state(E_i) : \;\; min_i \leq |\{t \in state(R) \mid t.E_i = e_i\}| \leq max_i$$

$t.E_i$ bezeichnet die Projektion der Beziehungsinstanz t auf den Entity-Typ E_i.

Charakterisierung von Beziehungstypen

[*min*, *max*]-Notation (Formalismus II für Kardinalitäten)

Beispiele:

- *arbeitet_in*(Mitarbeiter[0,1], Raum[0,3])

  „Mitarbeitern ist ein oder kein Raum zugeordnet. Pro Raum gibt es höchstens drei Mitarbeiter.“

Charakterisierung von Beziehungstypen

[*min*, *max*]-Notation (Formalismus II für Kardinalitäten)

Beispiele:

- *arbeitet_in*(Mitarbeiter[0,1], Raum[0,3])

  „Mitarbeitern ist ein oder kein Raum zugeordnet. Pro Raum gibt es höchstens drei Mitarbeiter.“

  $$\mathit{state}(\mathit{Mitarbeiter}) = \{m_1, m_2, m_3, m_4, m_5, m_6\}$$

  $$\mathit{state}(\mathit{Raum}) = \{r_1, r_2, r_3\}$$

  $$\mathit{state}(\mathit{arbeitet\_in}) = \{(m_1, r_1), (m_2, r_1), (m_3, r_1), (m_5, r_3)\}$$

Charakterisierung von Beziehungstypen

[*min*, *max*]-Notation (Formalismus II für Kardinalitäten)

Beispiele:

- *arbeitet_in*(Mitarbeiter[0,1], Raum[0,3])

  „Mitarbeitern ist ein oder kein Raum zugeordnet. Pro Raum gibt es höchstens drei Mitarbeiter.“

  $$state(Mitarbeiter) = \{\overset{1}{m_1}, \overset{1}{m_2}, \overset{1}{m_3}, \overset{0}{m_4}, \overset{1}{m_5}, \overset{0}{m_6}\}$$

  $$state(Raum) = \{\overset{3}{r_1}, \overset{0}{r_2}, \overset{1}{r_3}\}$$

  $$state(arbeitet\_in) = \{(m_1, r_1), (m_2, r_1), (m_3, r_1), (m_5, r_3)\}$$

Charakterisierung von Beziehungstypen

[*min*, *max*]-Notation (Formalismus II für Kardinalitäten)

Beispiele:

- *arbeitet_in*(Mitarbeiter[0,1], Raum[0,3])

  „Mitarbeitern ist ein oder kein Raum zugeordnet. Pro Raum gibt es höchstens drei Mitarbeiter.“

  $$state(Mitarbeiter) = \{\overset{1}{m_1}, \overset{1}{m_2}, \overset{1}{m_3}, \overset{0}{m_4}, \overset{1}{m_5}, \overset{0}{m_6}\}$$

  $$state(Raum) = \{\overset{3}{r_1}, \overset{0}{r_2}, \overset{1}{r_3}\}$$

  $$state(arbeitet\_in) = \{(m_1, r_1), (m_2, r_1), (m_3, r_1), (m_5, r_3)\}$$

- *verantwortlich_für*(Mitarbeiter[0,*], Computer[1,1])

  „Es gibt Mitarbeiter, die für keinen Computer verantwortlich sind – aber auch Mitarbeiter, die für mehrere Computer verantwortlich sind. Jedem Computer ist genau ein Mitarbeiter zugeordnet, der verantwortlich für diesen Computer ist.“

Bemerkungen:

- Eine spezielle Wertangabe für max_i ist $*$. Sie dient zum Anzeigen einer unbegrenzten Anzahl.
- $[0, *]$ bedeutet keine Einschränkung und ist die Standardannahme, wenn keine Kardinalitäten angegeben sind.
- $R(E_1[0, 1], E_2)$ entspricht einer *partiellen funktionalen* Beziehung $R : E_1 \to E_2$.
- $R(E_1[1, 1], E_2)$ entspricht einer *totalen funktionalen* Beziehung $R : E_1 \to E_2$.
- Die Kardinalitätsangabe auf der rechten Seite verrät auch etwas über die Surjektivität der Abbildung: $[0, *]$ bedeutet, dass nicht jedes Element in einer Beziehung stehen muss; $[1, *]$ bedeutet, dass jedes Element erreicht wird, die Funktion also surjektiv ist.

Charakterisierung von Beziehungstypen

Formalismus I versus Formalismus II

Unterschied in der Semantik:

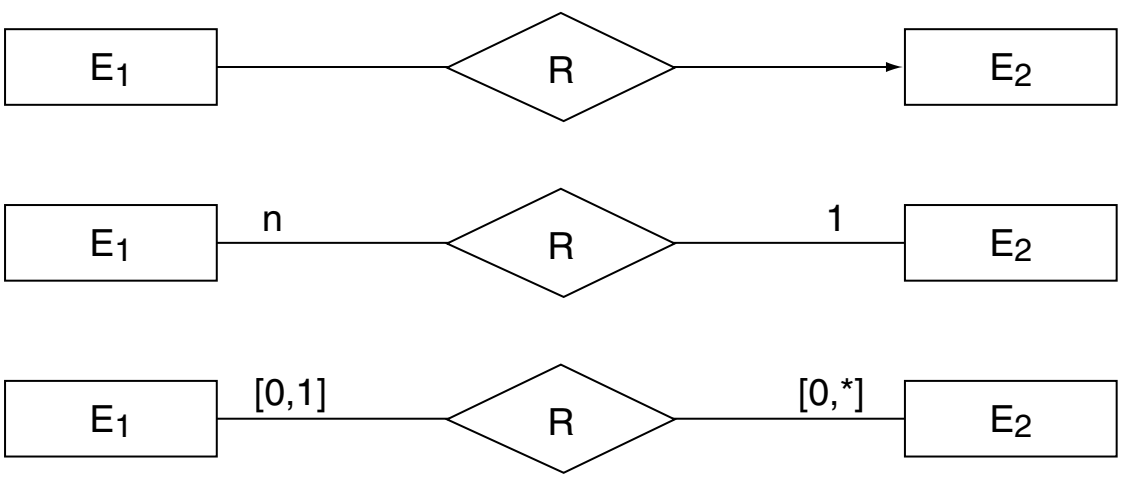
- Die Semantik der Kardinalitäten bei Formalismus I bezieht sich auf Instanzen von Entity-Typen.
- Die Semantik der Kardinalitäten bei Formalismus II ($[min, max]$-Notation) bezieht sich auf Instanzen von Beziehungstypen.

Charakterisierung von Beziehungstypen

Formalismus I versus Formalismus II

Unterschied in der Semantik:

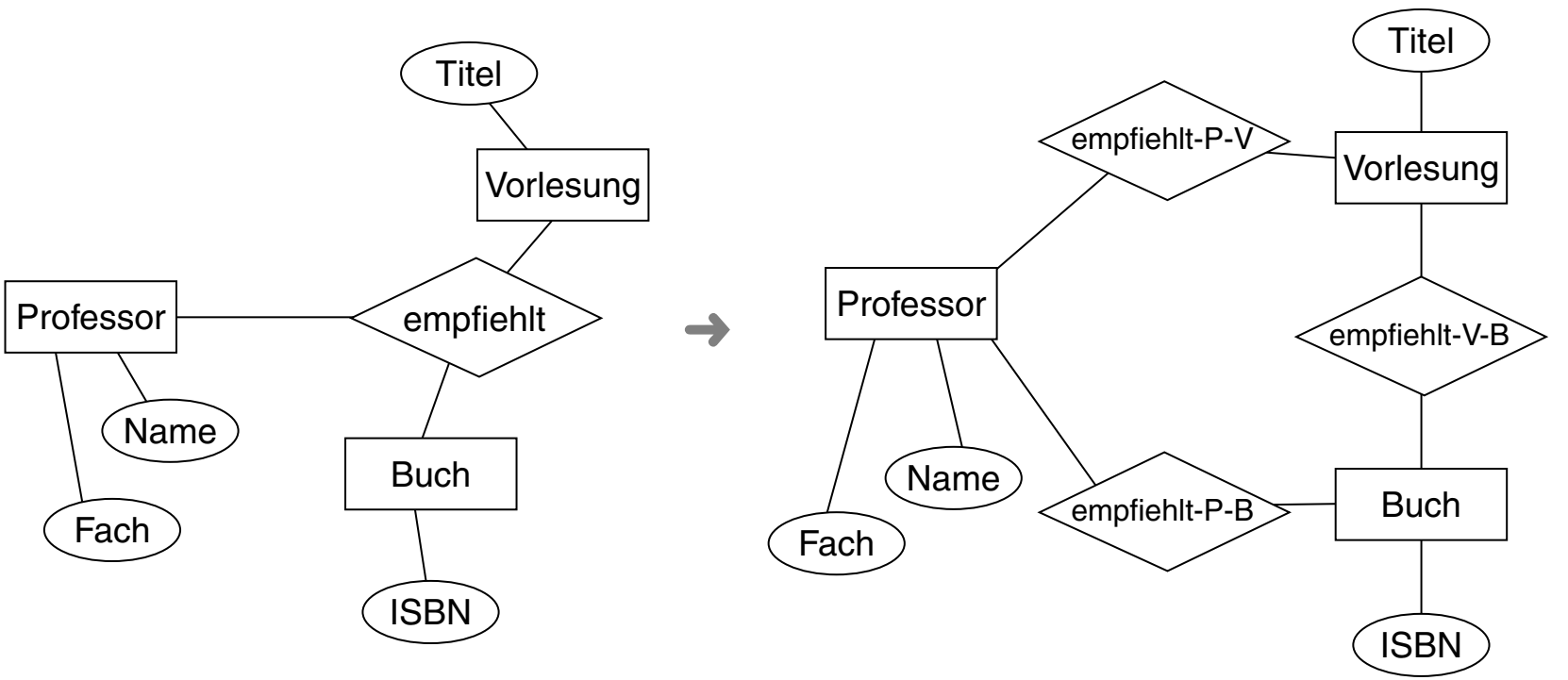
- Die Semantik der Kardinalitäten bei Formalismus I bezieht sich auf Instanzen von Entity-Typen.
- Die Semantik der Kardinalitäten bei Formalismus II ($[min, max]$-Notation) bezieht sich auf Instanzen von Beziehungstypen.
- Somit kann eine partielle Funktion $R : E_1 \rightarrow E_2$ graphisch u.a. auf folgende Arten dargestellt werden:



Charakterisierung von Beziehungstypen

Umwandlung n-stelliger Beziehungen

Beispiel für die Umwandlung einer dreistelligen Beziehung in drei zweistellige Beziehungen:



Charakterisierung von Beziehungstypen

Umwandlung n-stelliger Beziehungen (Fortsetzung)

ursprünglicher Zustand:

empfiehlt		
Professor	Vorlesung	Buch (ISBN)
Pearl	Datenbanken	0-341...
Pearl	IT-Systeme	2-305...
Graham	Datenbanken	2-305...
Graham	IT-Systeme	2-305...

Zustände der drei zweistelligen Beziehungen:

empfiehlt-P-V	
Professor	Vorlesung
Pearl	Datenbanken
Pearl	IT-Systeme
Graham	Datenbanken
Graham	IT-Systeme

empfiehlt-P-B	
Professor	Buch (ISBN)
Pearl	0-341...
Pearl	2-305...
Graham	2-305...

empfiehlt-V-B	
Vorlesung	Buch (ISBN)
Datenbanken	0-341...
IT-Systeme	2-305...
Datenbanken	2-305...

Charakterisierung von Beziehungstypen

Umwandlung n-stelliger Beziehungen (Fortsetzung)

ursprünglicher Zustand:

empfiehlt		
Professor	Vorlesung	Buch (ISBN)
Pearl	Datenbanken	0-341...
Pearl	IT-Systeme	2-305...
Graham	Datenbanken	2-305...
Graham	IT-Systeme	2-305...

Zustände der drei zweistelligen Beziehungen:

empfiehlt-P-V	
Professor	Vorlesung
Pearl	Datenbanken
Pearl	IT-Systeme
Graham	Datenbanken
Graham	IT-Systeme

empfiehlt-P-B	
Professor	Buch (ISBN)
Pearl	0-341...
Pearl	2-305...
Graham	2-305...

empfiehlt-V-B	
Vorlesung	Buch (ISBN)
Datenbanken	0-341...
IT-Systeme	2-305...
Datenbanken	2-305...

Charakterisierung von Beziehungstypen

Umwandlung n-stelliger Beziehungen (Fortsetzung)

ursprünglicher Zustand:

empfiehlt		
Professor	Vorlesung	Buch (ISBN)
Pearl	Datenbanken	0-341...
Pearl	IT-Systeme	2-305...
Graham	Datenbanken	2-305...
Graham	IT-Systeme	2-305...
Pearl	**Datenbanken**	**2-305...**

Problem 1

Zustände der drei zweistelligen Beziehungen:

empfiehlt-P-V	
Professor	Vorlesung
Pearl	Datenbanken
Pearl	IT-Systeme
Graham	Datenbanken
Graham	IT-Systeme

empfiehlt-P-B	
Professor	Buch (ISBN)
Pearl	0-341...
Pearl	2-305...
Graham	2-305...

empfiehlt-V-B	
Vorlesung	Buch (ISBN)
Datenbanken	0-341...
IT-Systeme	2-305...
Datenbanken	2-305...

Charakterisierung von Beziehungstypen

Umwandlung n-stelliger Beziehungen (Fortsetzung)

ursprünglicher Zustand:

empfiehlt		
Professor	Vorlesung	Buch (ISBN)
	?	

Problem 2

Zustände der drei zweistelligen Beziehungen:

empfiehlt-P-V	
Professor	Vorlesung
Pearl	Datenbanken
Pearl	IT-Systeme
Graham	Datenbanken
Graham	IT-Systeme

empfiehlt-P-B	
Professor	Buch (ISBN)
Pearl	0-341...
Pearl	2-305...
Graham	2-305...

empfiehlt-V-B	
Vorlesung	Buch (ISBN)
Datenbanken	0-341...
IT-Systeme	2-305...
Datenbanken	2-305...
Web-Services	**1-100**

Bemerkungen:

- Verschiedene Zustände der dreistelligen Relation (oben) bilden auf dieselben zweistelligen Relationen (unten) ab. Bei der Umkehrung dieser Zerlegung (von unten nach oben) können zusätzliche Tupel enstehen.
- Die illustrierte Problematik wird als die Eigenschaft der *Verlustlosigkeit* bzw. *Verbundtreue* in der Entwurfstheorie relationaler Datenbanken behandelt.
- Die im Beispiel durchgeführte Zerlegung ist nicht verlustlos: verloren gegangen ist die Information, dass Pearl das Buch 2-305 für IT-Systeme, aber nicht für Datenbanken empfiehlt. D.h., Anfragen auf der Datenbank mit dem zerlegten Schema können (ungewollter Weise) zu anderen Ergebnissen als auf der ursprünglichen Datenbank führen.
- Mit den drei zweistelligen Beziehungstypen können (ungewollter Weise) auch Zusammenhänge abgebildet werden, die vorher in der dreistelligen Relation nicht möglich waren: ein Buch (1-100) wird für eine Vorlesung (Web-Services) empfohlen, für die keine Professorin angegeben ist.
- Manche Datenbanksysteme bieten für die Modellierung ausschließlich binäre Beziehungstypen an.

Existenzabhängige Entity-Typen

Bisher vorausgesetzt: Entities existieren autonom und sind innerhalb ihres Typs über die Schlüsselattribute eindeutig identifizierbar.

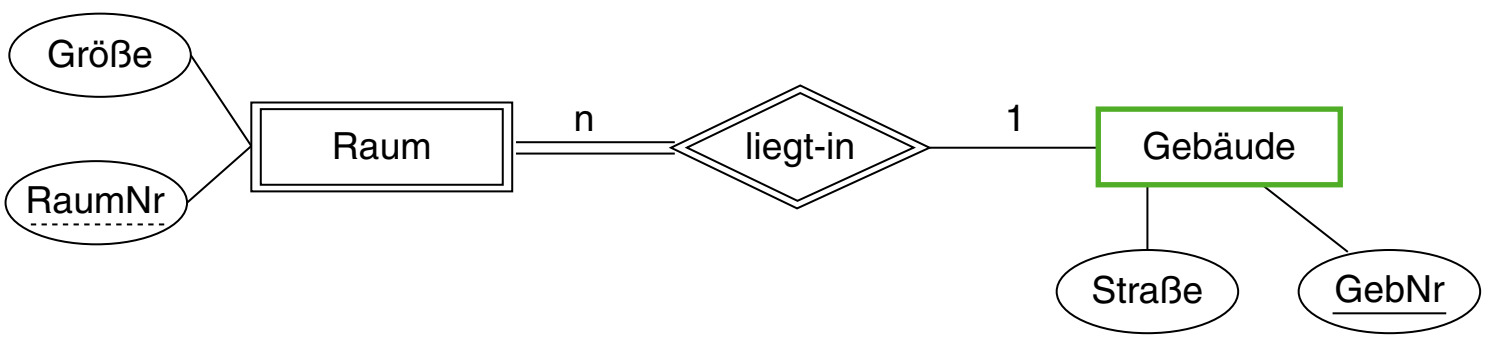
Existenzabhängige Entity-Typen

Bisher vorausgesetzt: Entities existieren autonom und sind innerhalb ihres Typs über die Schlüsselattribute eindeutig identifizierbar.

Aus Modellierungssicht sind auch Entity-Typen sinnvoll, die

1. von einem anderen, übergeordneten Entity-Typ abhängig sind, und
2. *in Kombination* mit dem Schlüssel des übergeordneten Entity-Typs eindeutig identifizierbar sind.

Beispiel:



Bemerkungen:

- Existenzabhängige Entity-Typen werden auch als *schwache* Entity-Typen bezeichnet.
- Existenzabhängige Entity-Typen werden durch doppelt umrandete Rechtecke, die Beziehung zu dem übergeordneten Entity-Typ durch eine doppelt umrandete Raute und die zugehörige Kante durch eine Doppellinie repräsentiert.
- Die Beziehung eines existenzabhängigen Entity-Typs zu dem übergeordneten Entity-Typ ist meist n:1, manchmal 1:1, aber nie n:m.
- Existenzabhängige Entity-Typen haben im Allgemeinen keinen Schlüssel, der alle Entities eindeutig identifiziert. Statt dessen gibt es ein Attribut (bzw. eine Menge von Attributen), das alle existenzabhängigen Entities, die *einem* übergeordneten Entity zugeordnet sind, voneinander unterscheidet. Diese Attribute werden im ER-Diagramm gestrichelt unterstrichen.
  Im Beispiel dient das Attribut „RaumNR“ zur Unterscheidung aller Räume *eines* Gebäudes.

Abstraktionskonzepte

Generalisierung / Spezialisierung

Die Generalisierung wird im konzeptuellen Entwurf eingesetzt, um eine bessere – im Sinne von „natürlichere“ – Strukturierung der Entity-Typen zu erzielen.

Abstraktionskonzepte

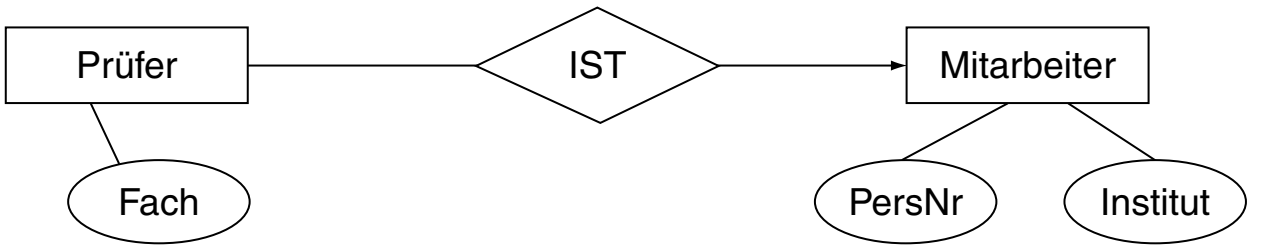
Generalisierung / Spezialisierung

Die Generalisierung wird im konzeptuellen Entwurf eingesetzt, um eine bessere – im Sinne von „natürlichere“ – Strukturierung der Entity-Typen zu erzielen.

Vorgehensweise:

- Die Eigenschaften ähnlicher Entity-Typen werden „faktorisiert“ und einem gemeinsamen *Obertyp* zugeordnet.
- Eigenschaften, die nicht faktorisierbar sind, verbleiben beim jeweiligen *Untertyp*, der somit eine Spezialisierung des Obertyps darstellt.

Beispiel:



Prüfer(*Institut*, *PersNr* , *Fach*)

geerbt von Mitarbeiter

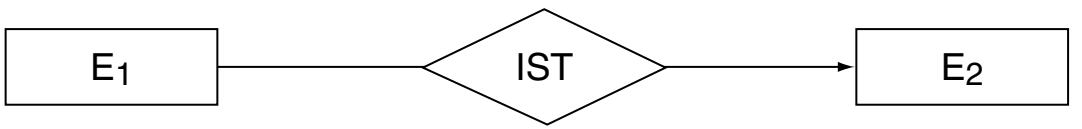
Bemerkungen:

- Im Beispiel ist „Mitarbeiter“ der generellere Obertyp und „Prüfer“ der speziellere Untertyp.
- Ein wichtiges Prinzip der Generalisierung ist die Vererbung: ein Untertyp erbt alle Eigenschaften des Obertyps.
- Bei der Übersetzung in das Relationenmodell werden Generalisierungsbeziehungen besonders behandelt.

Abstraktionskonzepte

Generalisierung / Spezialisierung (Fortsetzung)

Die Entities eines Untertyps sind gleichzeitig Entities des Obertyps; es handelt sich also um eine Teilmengenbeziehung: $\mathit{state}(E_1) \subseteq \mathit{state}(E_2)$



Schreibweise: E_1 IST E_2

Hinsichtlich der Teilmengensicht sind zwei Fälle von besonderem Interesse:

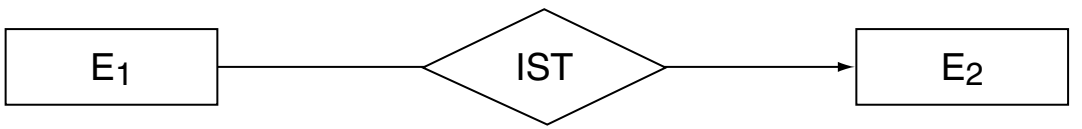
1. Disjunkte Spezialisierung

2. Vollständige Spezialisierung

Abstraktionskonzepte

Generalisierung / Spezialisierung (Fortsetzung)

Die Entities eines Untertyps sind gleichzeitig Entities des Obertyps; es handelt sich also um eine Teilmengenbeziehung: $state(E_1) \subseteq state(E_2)$



Schreibweise: E_1 IST E_2

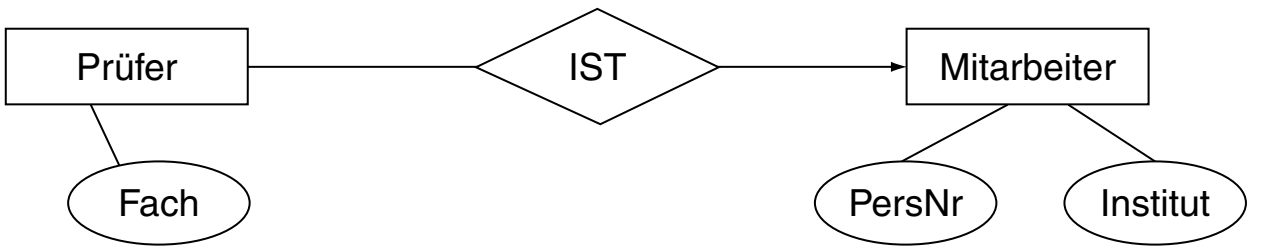
Hinsichtlich der Teilmengensicht sind zwei Fälle von besonderem Interesse:

1. Disjunkte Spezialisierung
   Die Mengen der Entity-Untertypen eines Entity-Obertyps sind paarweise disjunkt.
2. Vollständige Spezialisierung
   Die Menge des Entity-Obertyps enthält keine unspezialisierten Elemente sondern ergibt sich vollständig durch die Vereinigung der Mengen der Entity-Untertypen.

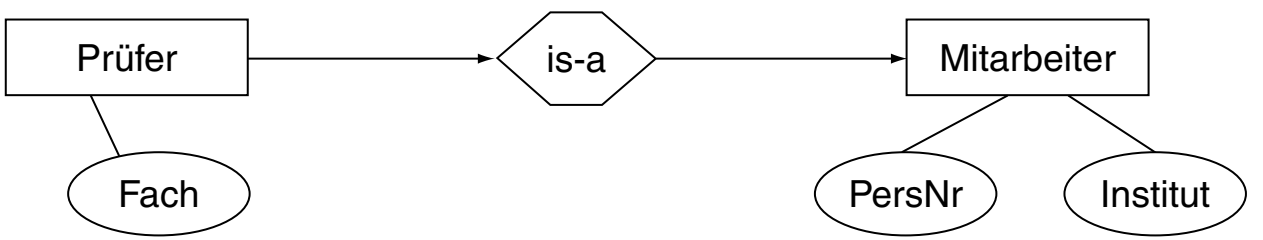
Abstraktionskonzepte

Generalisierung / Spezialisierung (Fortsetzung)

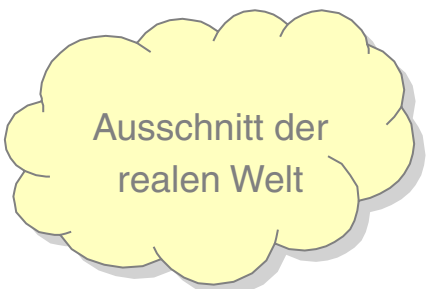
Graphische Notation als Standard-Beziehungstyp:



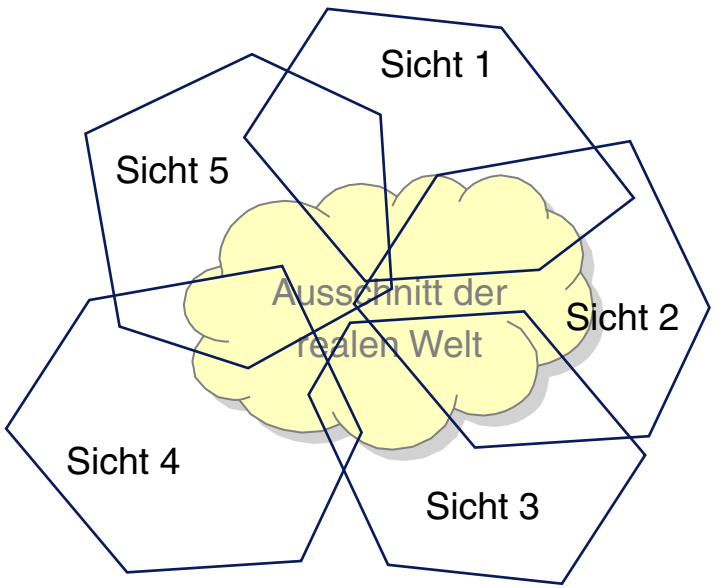
Alternative graphische Notation:



Konsolidierung, Sichtenintegration



Konsolidierung, Sichtenintegration



Konsolidierung

globales Schema

- redundanzfrei
- widerspruchsfrei
- Synonyme bereinigt
- Homonyme bereinigt
- ...

Bemerkungen: [DB:II Entwurfsprozess > Konzeptueller Entwurf]

- Bei großen Anwendungen ist es nicht praktikabel, den konzeptuellen Entwurf in einem Schritt durchzuführen. Sinnvoll ist die Aufteilung des Modellierungsproblems nach Anwendergruppen, die jeweils ihre Sicht auf das Gesamtmodell beschreiben.
- Konsolidierung bedeutet die Zusammenfassung einzelner Sichten zu einem globalen Schema, das u.a. redundanz- und widerspruchsfrei ist.
- Bei der Konsolidierung einer großen Anwendung sollte man schrittweise vorgehen und jeweils nur zwei Teilschemata gleichzeitig betrachten.
- Arten von Widersprüchen, die bei einer Konsolidierung aufzulösen sind:
  - unterschiedliche Benennung gleicher Sachverhalte (Synonyme)
  - gleiche Benennung unterschiedlicher Sachverhalte (Homonyme)
  - Sachverhalt einmal als Entity-Typ und ein andermal als Beziehungstyp modelliert (struktureller Widerspruch)
  - widersprüchliche Funktionalitätsangaben
  - widersprüchliche Datentypen, widersprüchliche Schlüsselattribute

Konsolidierung, Sichtenintegration

Beispiel: drei Sichten einer Universitätsdatenbank

Erstellung von
Dokumenten

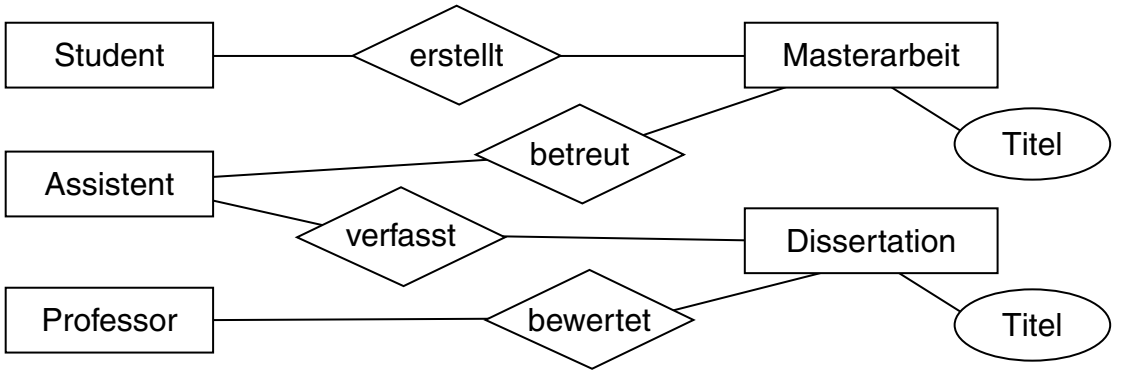
Bibliotheks
-verwaltung

Buchempfehlungen

Konsolidierung, Sichtenintegration

Beispiel: drei Sichten einer Universitätsdatenbank

Erstellung von Dokumenten



Bibliotheks -verwaltung

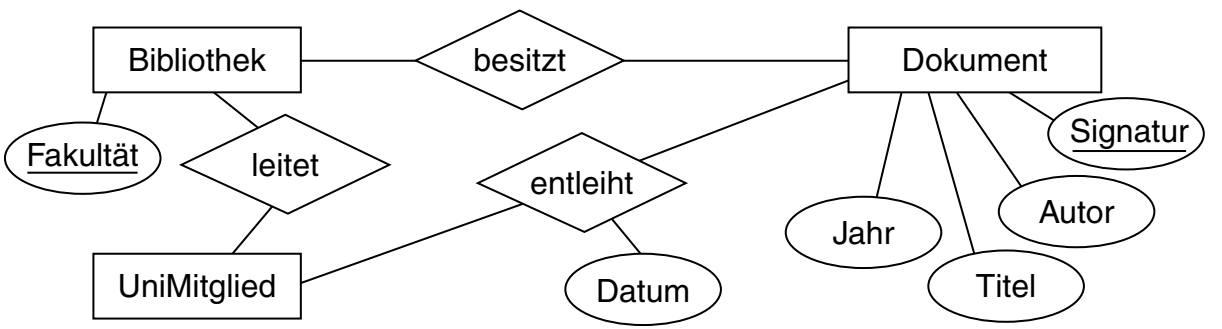
Buchempfehlungen

Konsolidierung, Sichtenintegration

Beispiel: drei Sichten einer Universitätsdatenbank

Erstellung von Dokumenten

Bibliotheks -verwaltung



Buchempfehlungen

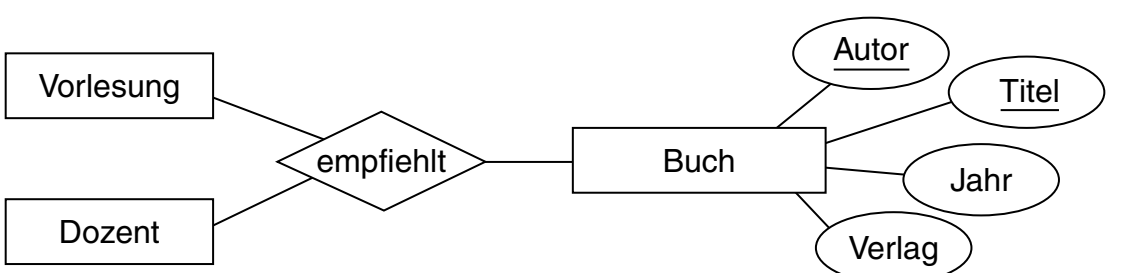
Konsolidierung, Sichtenintegration

Beispiel: drei Sichten einer Universitätsdatenbank

Erstellung von
Dokumenten

Bibliotheks
-verwaltung

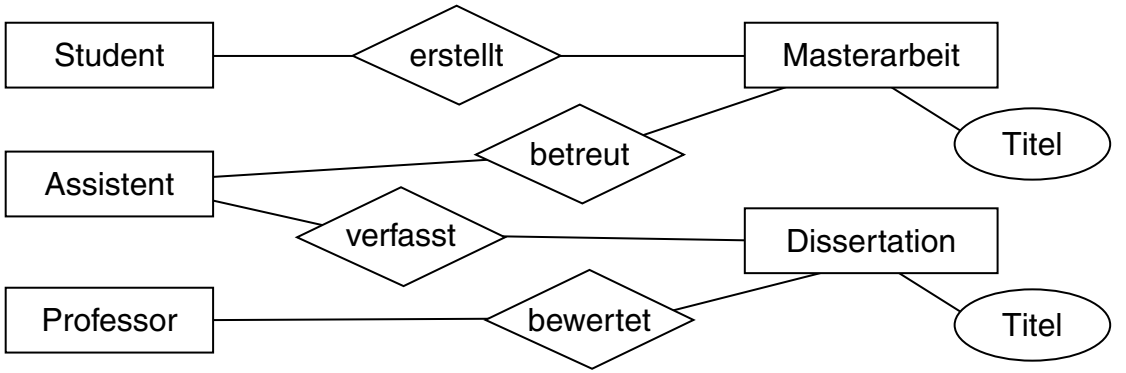
Buchempfehlungen



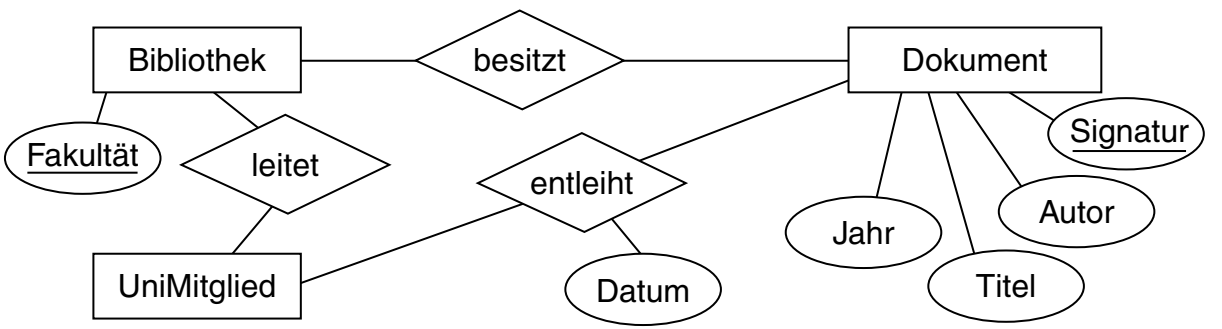
Konsolidierung, Sichtenintegration

Beispiel: drei Sichten einer Universitätsdatenbank

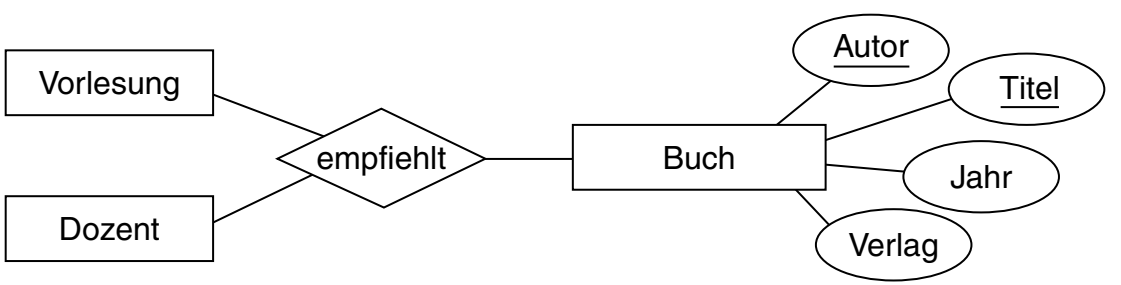
Erstellung von Dokumenten



Bibliotheks -verwaltung



Buchempfehlungen



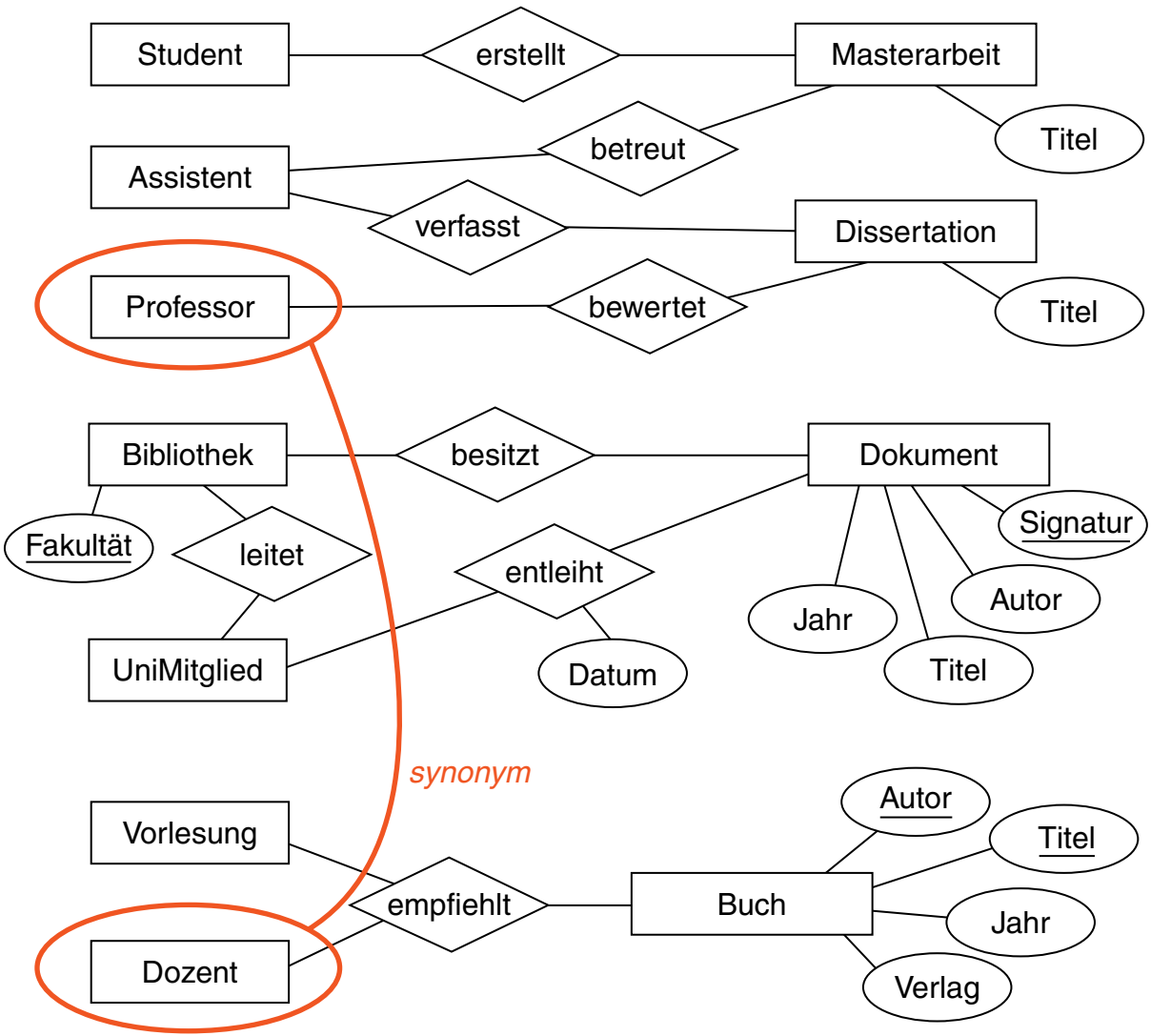
Konsolidierung, Sichtenintegration

Beispiel: drei Sichten einer Universitätsdatenbank

Erstellung von Dokumenten

Bibliotheks -verwaltung

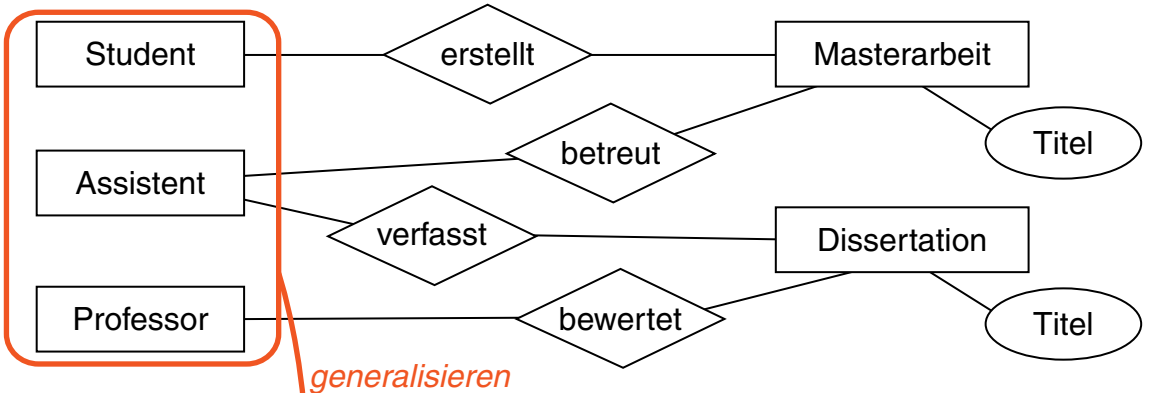
Buchempfehlungen



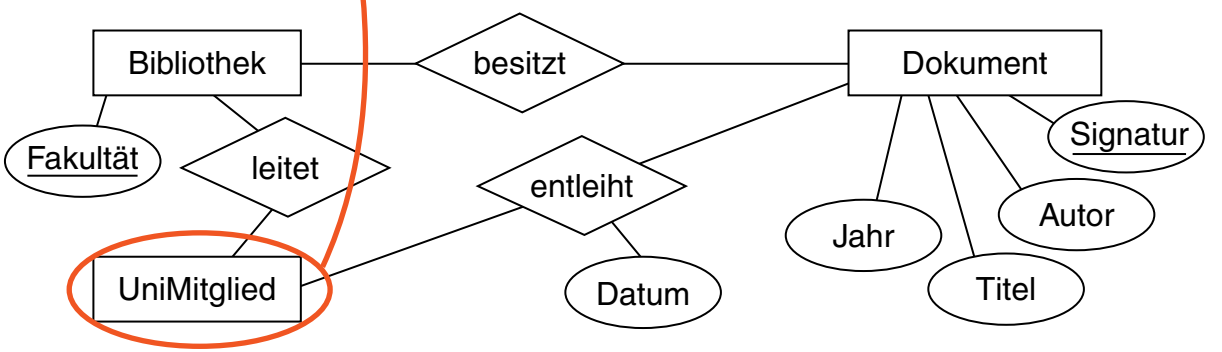
Konsolidierung, Sichtenintegration

Beispiel: drei Sichten einer Universitätsdatenbank

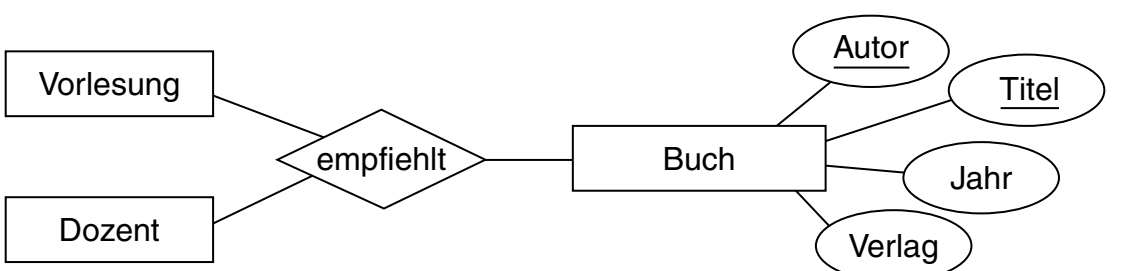
Erstellung von Dokumenten



Bibliotheks -verwaltung



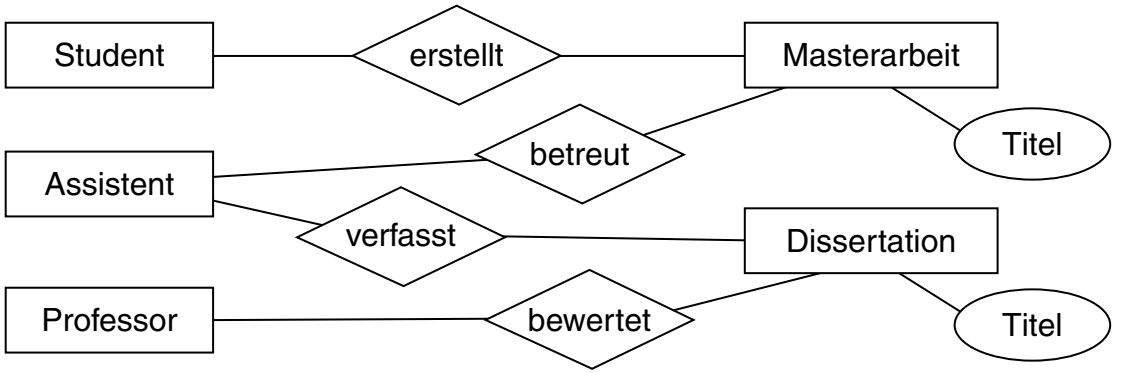
Buchempfehlungen



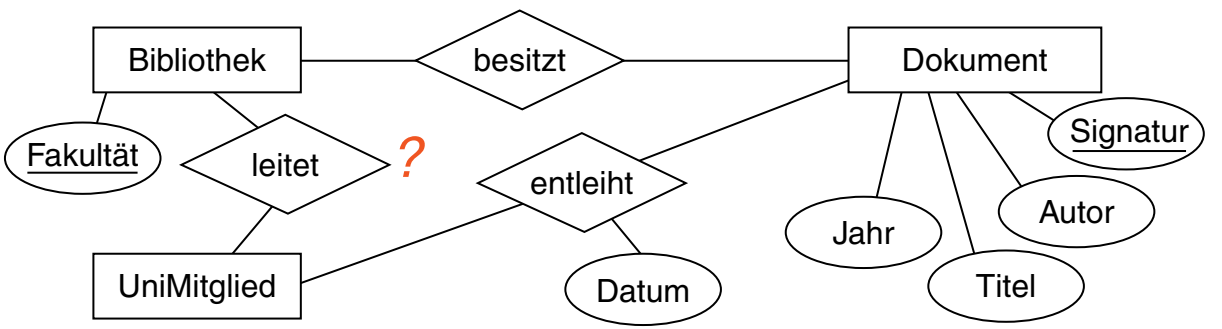
Konsolidierung, Sichtenintegration

Beispiel: drei Sichten einer Universitätsdatenbank

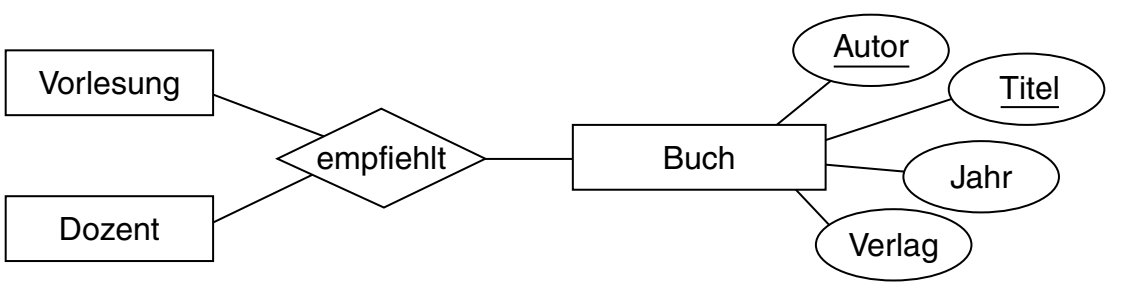
Erstellung von Dokumenten



Bibliotheks -verwaltung



Buchempfehlungen



Bemerkungen:

- Aufgabe ist die Konsolidierung dieser Sichten in *ein* Schema.
- Die Entity-Typen „Dozent“ und „Professor“ sind synonym verwendet worden.
- Der Entity-Typ „UniMitglied“ ist eine Generalisierung der Entity-Typen „Student“, „Professor“ und „Assistent“.
- Fakultätsbibliotheken werden von Angestellten und nicht von Studierenden geleitet: der Beziehungstyp „leitet“ in Sicht 2 sollte revidiert werden.
- Die Entity-Typen „Dissertation“, „Masterarbeit“ und „Buch“ sind Spezialisierungen des Entity-Typs „Dokument“.
- Alle an der Universität erstellten Diplomarbeiten und Dissertationen werden in Bibliotheken verwaltet.
- Die Beziehungtypen „erstellt“ und „verfasst“ in Sicht 1 modellieren denselben Sachverhalt wie das Attribut „Autor“ vom Entity-Typ „Dokument“ in Sicht 2.
- Alle in einer Bibliothek verwalteten Dokumente werden durch die Signatur identifiziert.