



Vorlesung
Datenbanksysteme I
Wintersemester 2014/2015

Kapitel 5:
Das E/R-Modell

Vorlesung: PD Dr. Arthur Zimek
Übungen: Sebastian Goebel, Jan Stutzki

Skript © Christian Böhm, LMU
ergänzt von Arthur Zimek (2014)

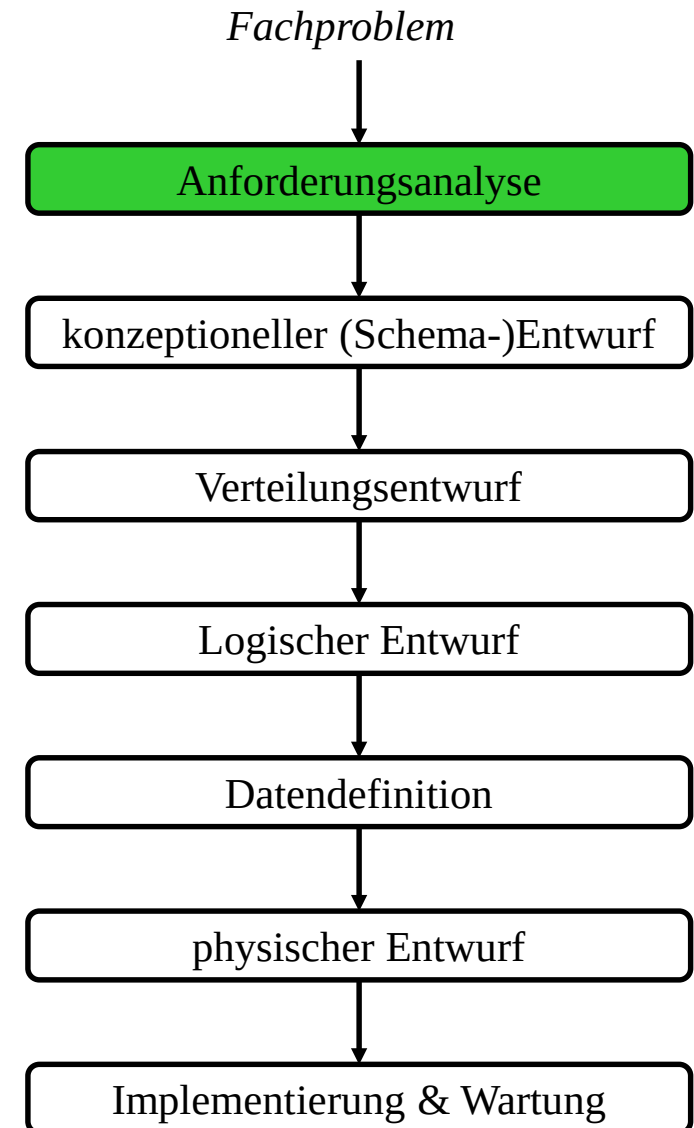
http://www.dbs.ifi.lmu.de/cms/Datenbanksysteme_I



Datenbank-Entwurf

Entwurfsphasen:

- Sammeln und Analysieren der Anforderungen für die zu realisierende Datenbank
- Welche Informationen (und Funktionen) werden (von welchen späteren Benutzern) benötigt?

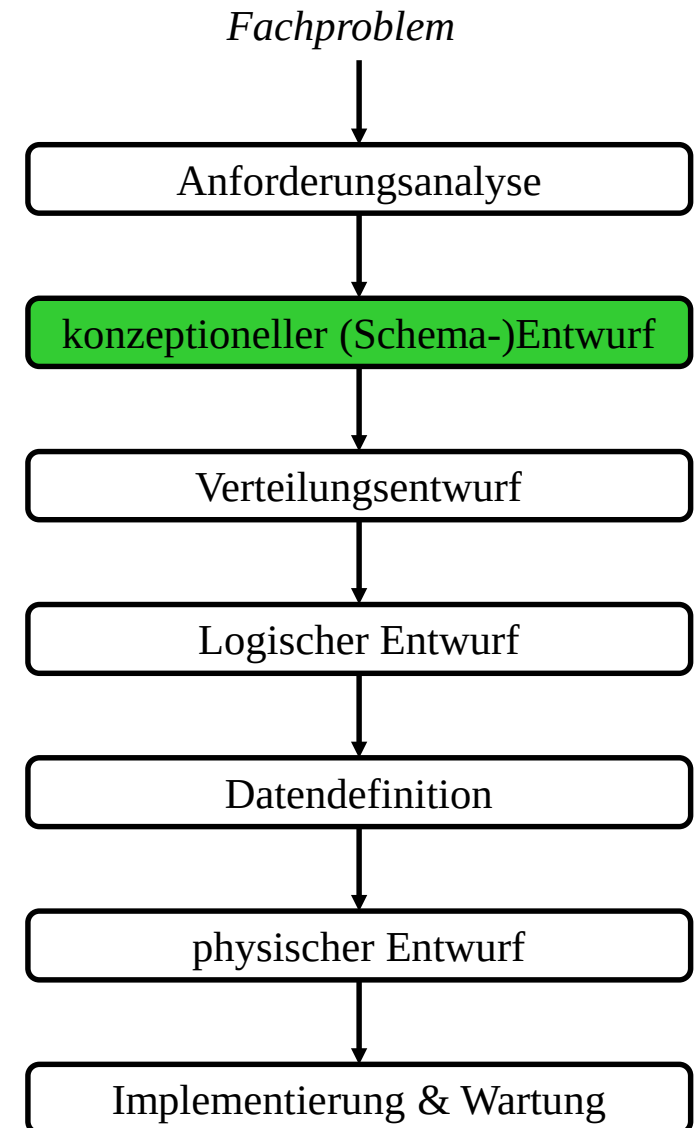




Datenbank-Entwurf

Entwurfsphasen:

- Entwurf eines Gesamtschemas für die Datenbank, das unabhängig von dem später zur Implementierung verwendeten Datenbanksystem ist
- in der Regel werden zunächst *Sichten* für einzelne Benutzer(-gruppen) entworfen (*Sichtenentwurf*), die dann auf Übereinstimmungen und Diskrepanzen untersucht werden (*Sichtenanalyse*)
- schließlich werden die Sichten zu einem Gesamtschema integriert (*Sichtenintegration*)

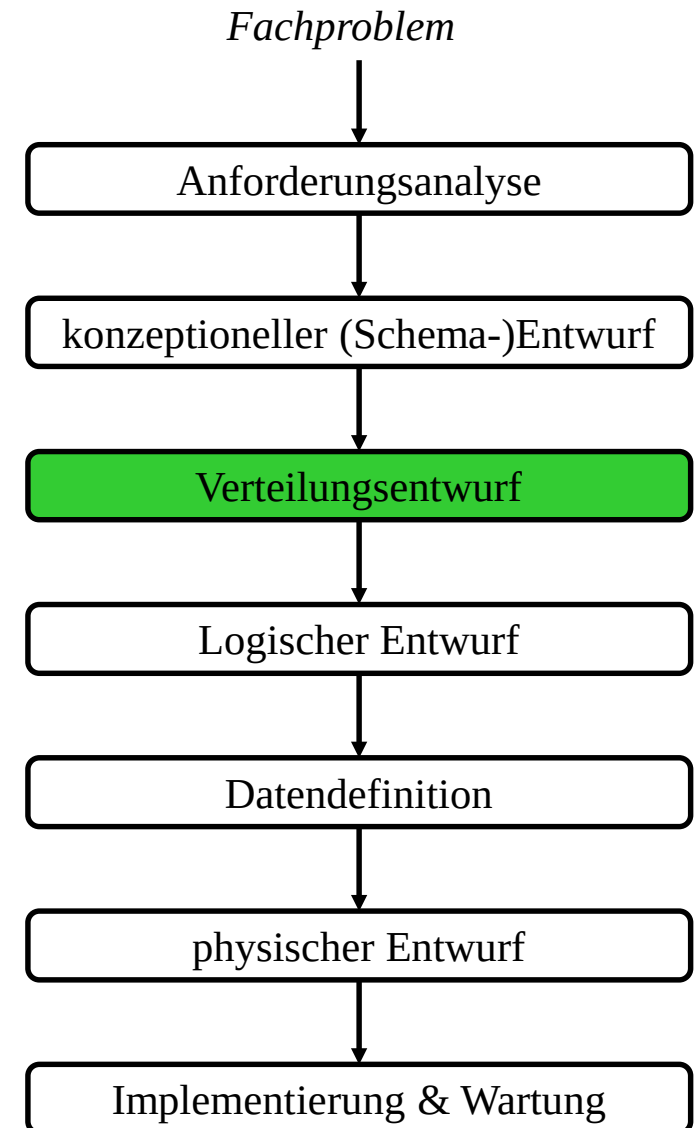




Datenbank-Entwurf

Entwurfsphasen:

- Bestimmung, welche Teile des konzeptionellen Schemas auf welchen Knoten realisiert werden sollen
- nur bei verteilten Datenbanksystemen
- im Prinzip unabhängig von konkreten Systemen für die einzelnen Knoten

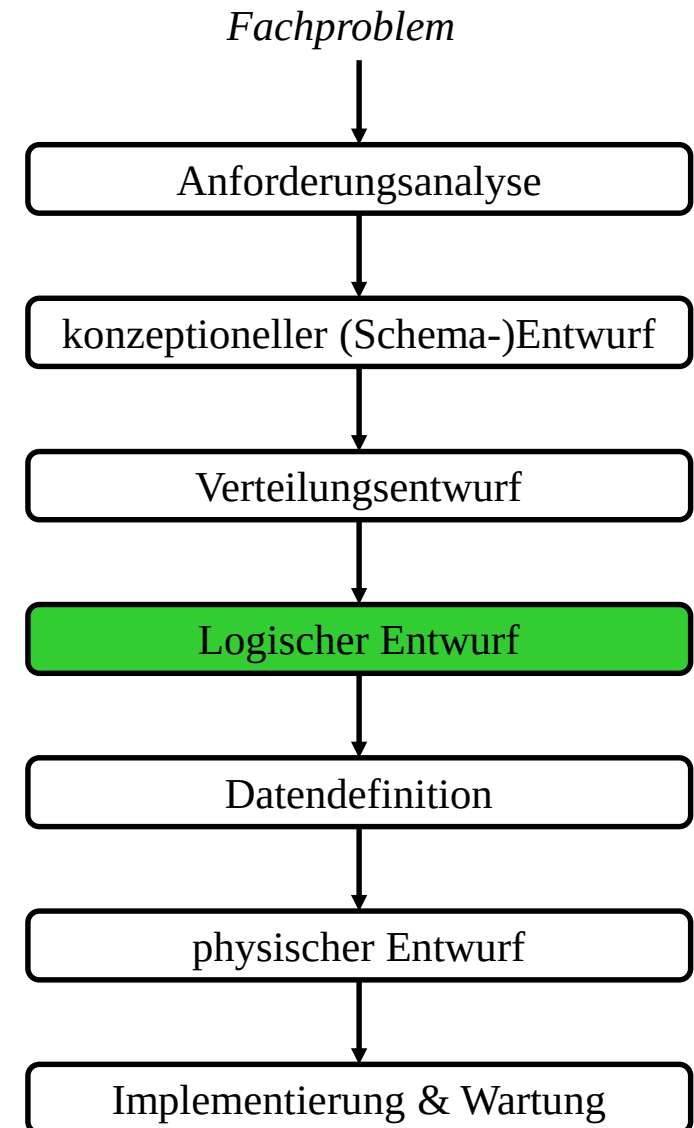




Datenbank-Entwurf

Entwurfsphasen:

- Auswahl eines konkreten Datenmodells, das vom Zielsystem unterstützt wird
- Transformation des konzeptionellen Schemas in dieses Datenmodell
- anschließend Optimierung anhand unterschiedlicher Qualitätskriterien
- unabhängig von Besonderheiten einzelner Datenbanksysteme

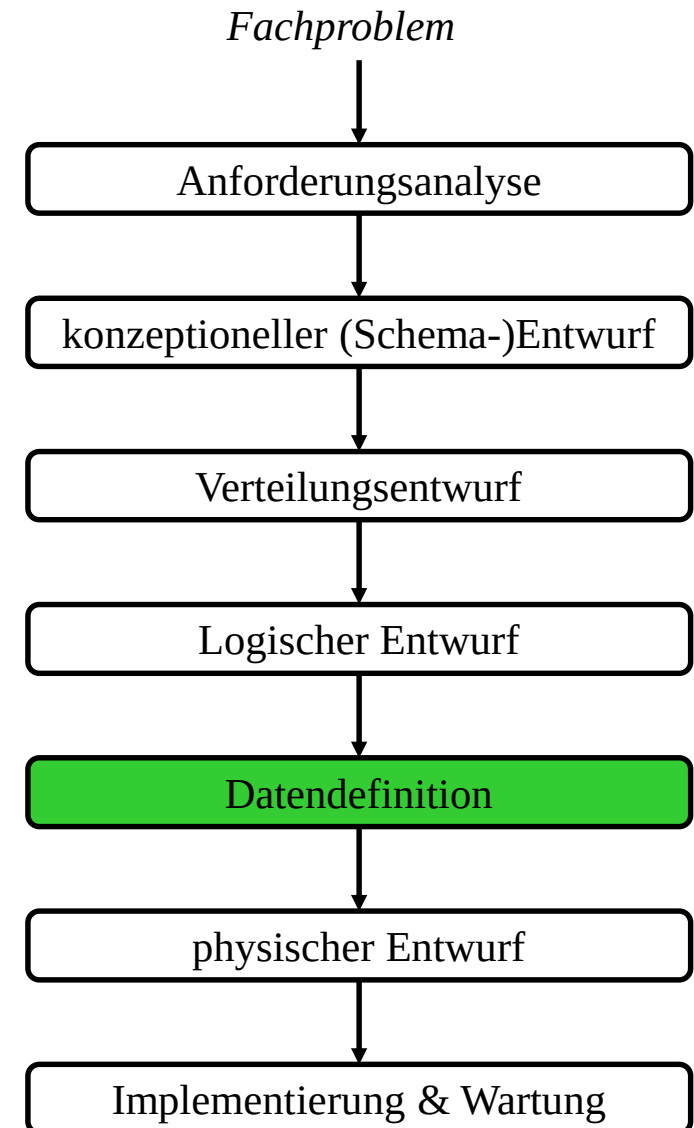




Datenbank-Entwurf

Entwurfsphasen:

- Umsetzung des Ergebnis des logischen Entwurfs in eine konkrete Deklaration für ein konkretes Datenbanksystem
- Definition von Benutzersichten

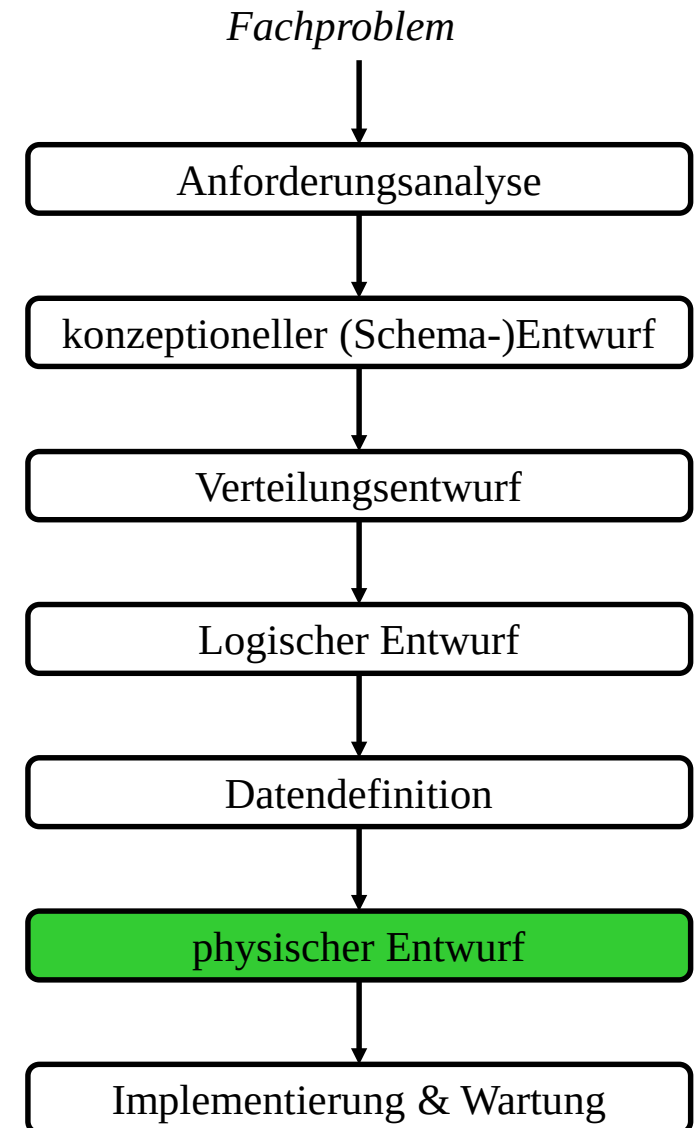




Datenbank-Entwurf

Entwurfsphasen:

- Definition der Zugriffs- und Speicherstrukturen auf der internen Ebene
- Ausnutzung der von dem verwendeten Datenbanksystem angebotenen Möglichkeiten (Indexstrukturen...)

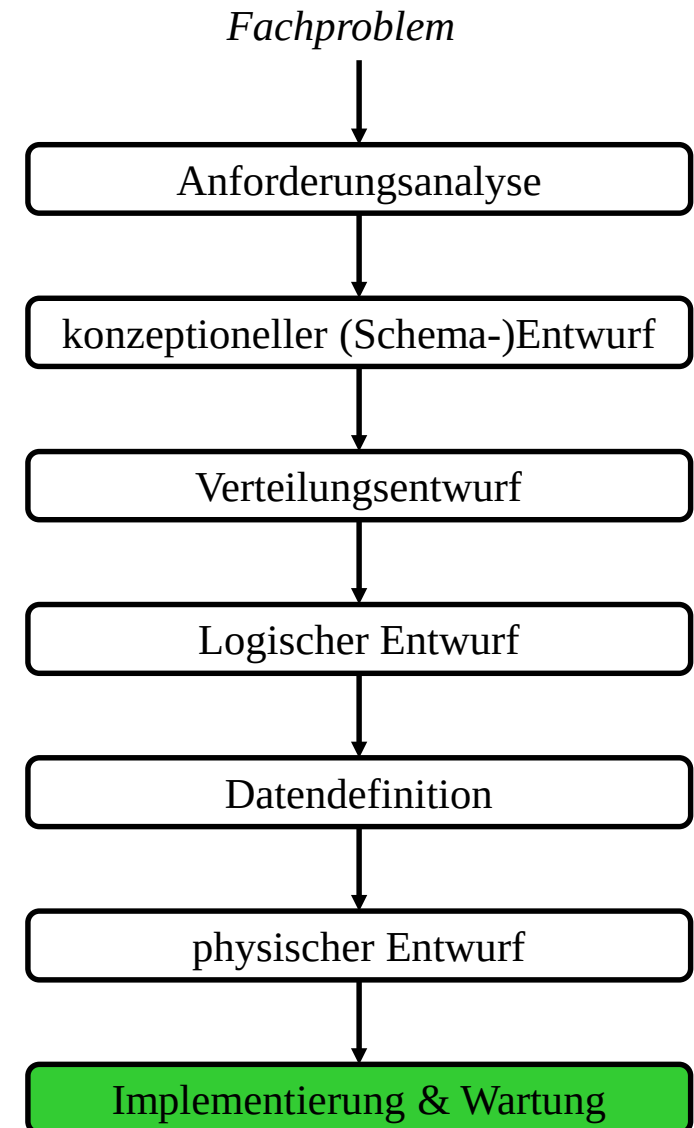




Datenbank-Entwurf

Entwurfsphasen:

- Phase der tatsächlichen Installation der Datenbank-Anwendungen
- fortlaufende Anpassung an neue Anforderungen bzw. bei Wechsel der Systemplattformen





Schema-Entwurf

- Generelle Aufgabe:
Finde eine formale Beschreibung (Modell) für einen zu modellierenden Teil der realen Welt
- Zwischenstufen:
 - Beschreibung durch natürliche Sprache (Pflichtenheft):
Beispiel: *In der Datenbank sollen alle Studierenden mit den durch sie belegten Lehrveranstaltungen gespeichert sein*
 - Beschreibung durch abstrakte grafische Darstellungen:


```
graph LR; Student[Student] --- belegt{belegt} --- Vorlesung[Vorlesung];
```
 - Beschreibung im relationalen Modell:
create table student (...);
create table vorlesung (...);



Das Entity/Relationship Modell

- Dient dazu, für einen Ausschnitt der realen Welt ein konzeptionelles Schema zu erstellen
- Grafische Darstellung: E/R-Diagramm
- Maschinenfernes Datenmodell
- Hohes Abstraktionsniveau
- Überlegungen zur Effizienz spielen keine Rolle
- Das E/R-Modell muss in ein konkretes Datenschema (hierarchisches Modell, Netzwerkmodell, relationales Modell,...) überführt werden
 - Einfache Grundregeln zur Transformation
 - Gewinnung eines *effizienten* Schemas erfordert tiefes Verständnis vom Zielmodell



Beispiel: Lehrveranstaltungen

Verschiedene Objekte spielen eine Rolle im Lehrbetrieb:

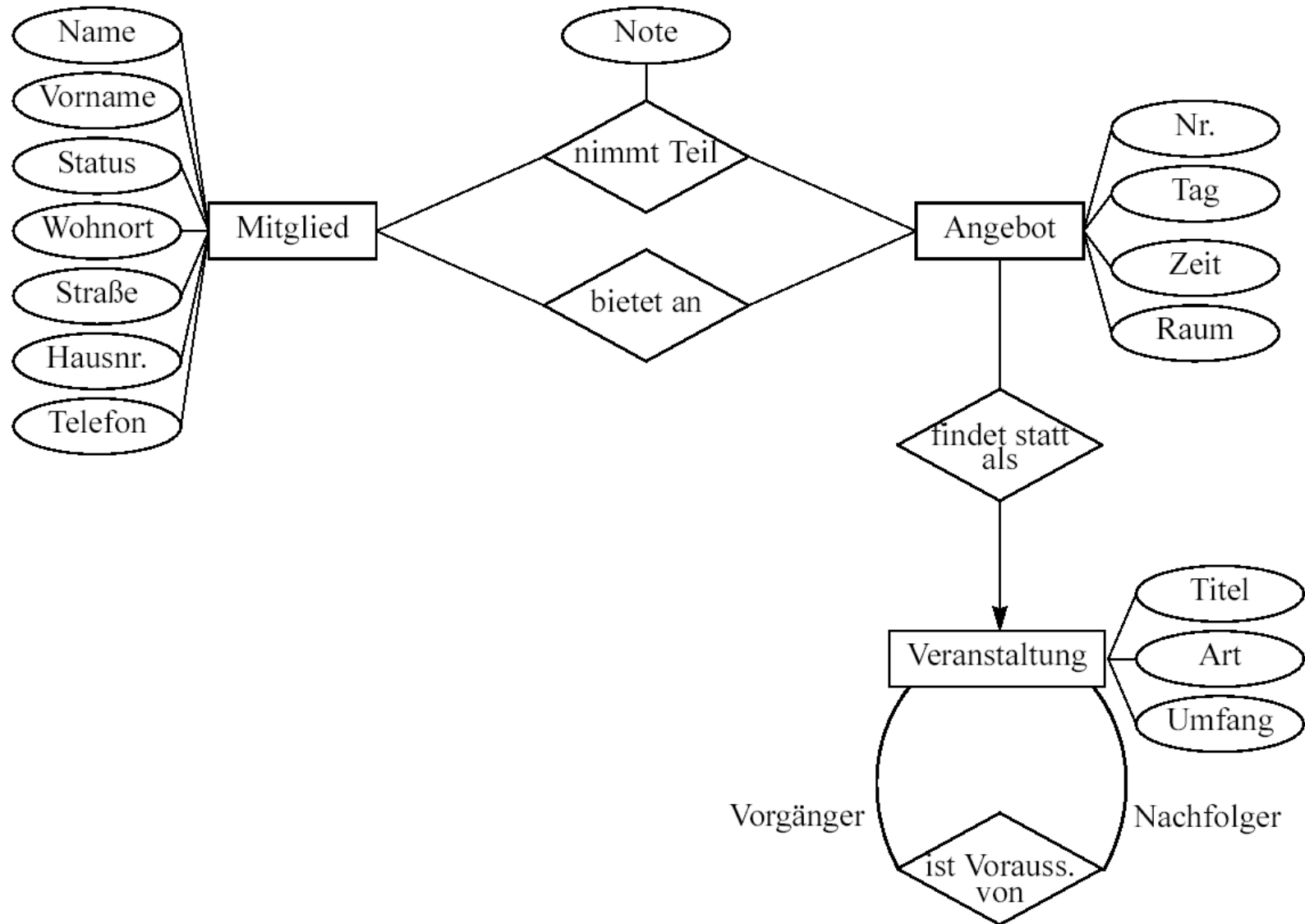
- Personen (genauer: Studenten, Professoren, Mitarbeiter, Hausmeister,...)
- Räume
- Zeiten
- Lehrveranstaltungen
- Bücher
- ...

Diese Objekte sind nicht voneinander isoliert, sondern stehen in verschiedenen Beziehungen zueinander, z.B.:

- ...veranstaltet...
- ...nimmt teil an...
- ...findet statt in...
- ...wohnt in...



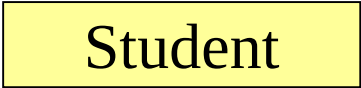
Beispiel: Lehrveranstaltungen



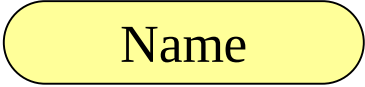


Elemente des E/R-Modells

- Entities:
(eigentlich: Entity Sets)
Objekttypen
- Attribute:
Eigenschaften
- Relationships:
Beziehungen zw. Entities



Student



Name



belegt

Entscheidende Aufgabe des Schema-Entwurfs:

- Finde *geeignete* Entities, Attribute und Relationships



Entities und Attribute

Entities

- Objekte, Typen, "Seiendes"
- Objekte der realen Welt, unterscheidbar
- Bsp: Mensch, Haus, Vorlesung, Bestellung, ...
 - Menge gleichartiger Entities: Entity-Set (Objektmenge, Objekttyp)
 - Einzelne Entities des gleichen Typs sind unterscheidbar (haben eine Objektidentität)

Student

Vorlesung

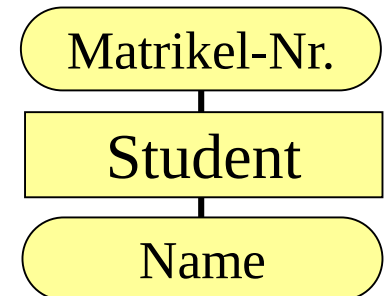
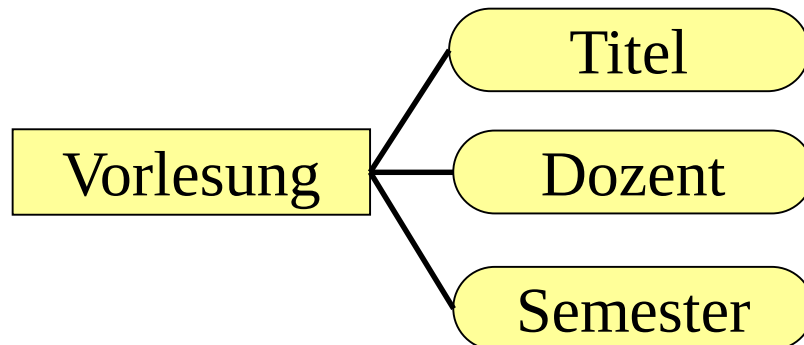
Leistungsnachweis



Entities und Attribute

Attribute

- Entities durch charakterisierende Eigenschaften beschrieben
- normalerweise einfache Datentypen wie INT, STRING usw.
- aber auch komplexe Attribute sind denkbar
- Bsp: Farbe, Gewicht, Name, Titel, ...
- Häufig beschränkt man sich auf die wichtigsten Attribute

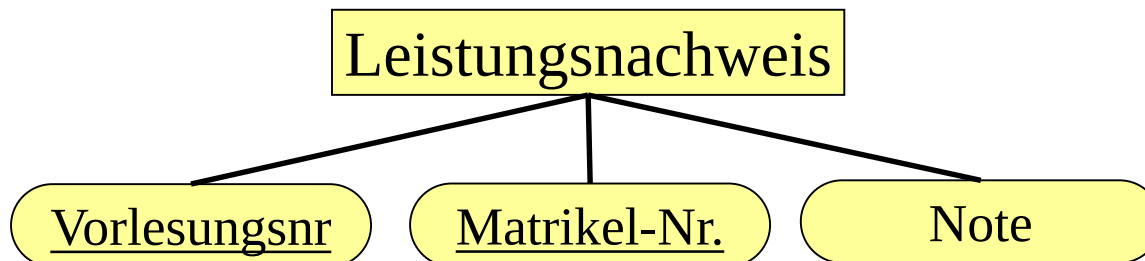
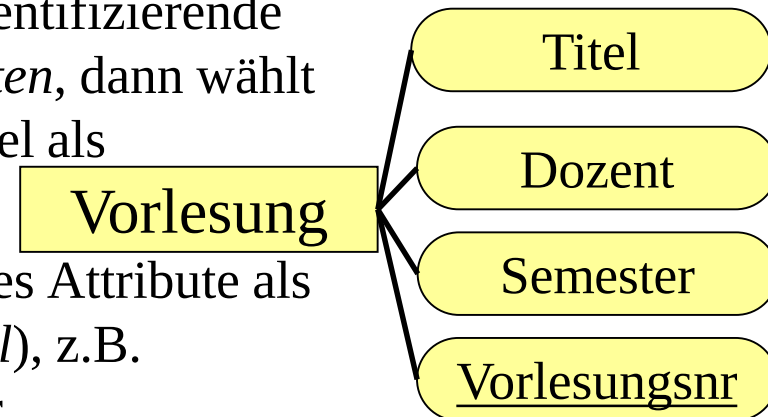
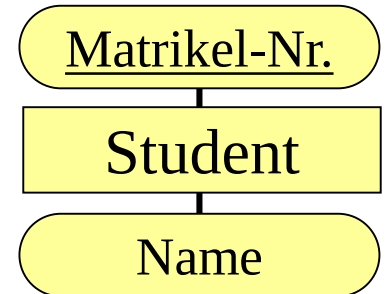




Entities und Attribute

Schlüssel

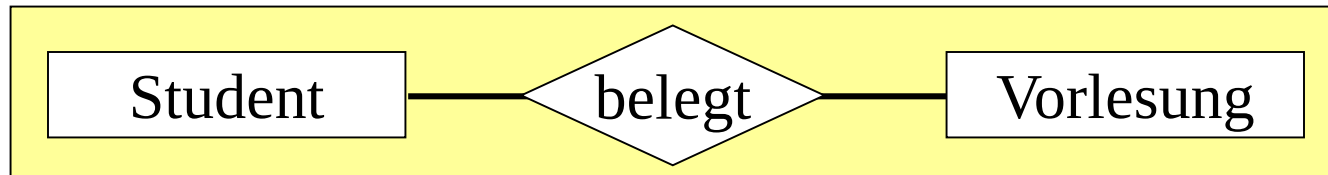
- ähnlich definiert wie im relationalen Modell: eine minimale Menge von Attributen, deren Werte das zugeordnete Entity eindeutig innerhalb aller Entities seines Typs identifizieren
- gibt es verschiedene minimale und identifizierende Attributmengen als *Schlüsselkandidaten*, dann wählt man einen dieser Kandidaten-Schlüssel als *Primärschlüssel* aus
- häufig wird einem Typ ein zusätzliches Attribute als Schlüssel hinzugefügt (*Kunstschlüssel*), z.B. Personalnummer, Vorlesungsnummer
- Primärschlüssel-Attribute werden unterstrichen



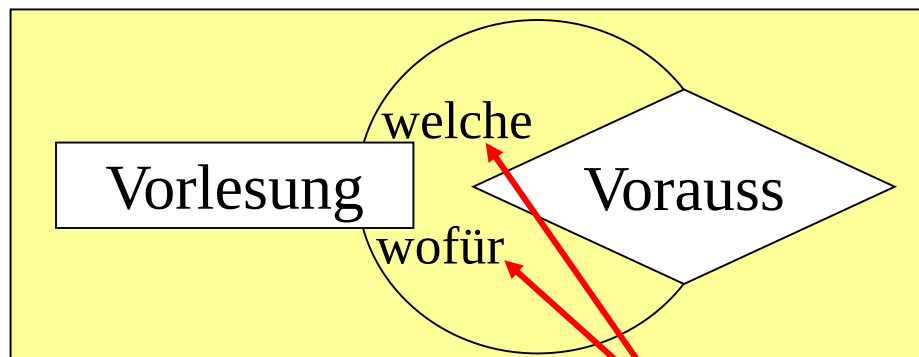


Relationships (Beziehungen)

- Stellen Zusammenhänge zwischen Entities dar
- Beispiele:



Ausprägung: belegt (Anton, Informatik 1)
belegt (Berta, Informatik 1)
belegt (Caesar, Wissensrepräsentation)
belegt (Anton, Datenbanksysteme 1)



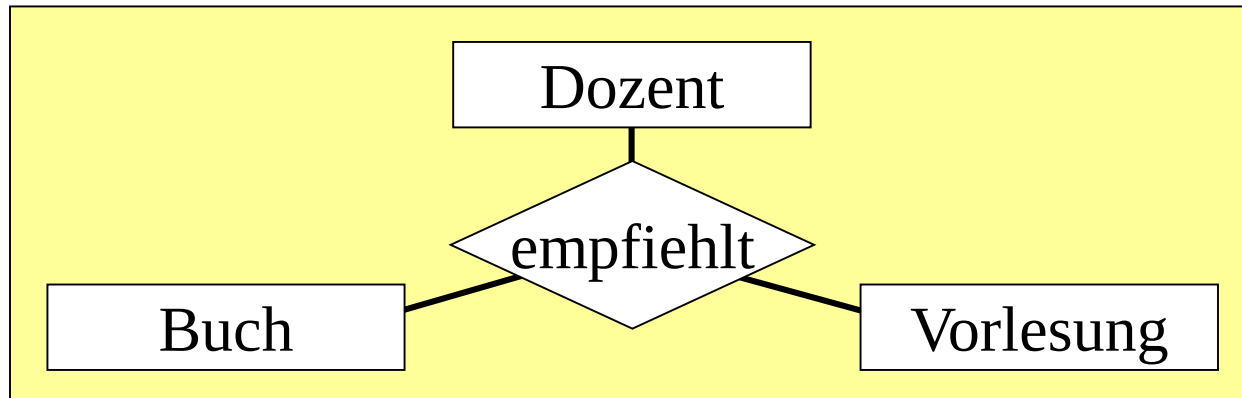
Ausprägung:
Vorauss (I 1, DBS 1)
Vorauss (I 1, Software-Eng.)
Vorauss (DBS 1, DBS 2)
Vorauss (I 1, Wissensrepr.)

unterschiedliche Rollen



Relationships (Beziehungen)

- Relationships können eigene Attribute haben.
Beispiel: Student *belegt* Vorlesung
belegt hat Attribut *Note*
- Mehrstellige (ternäre usw.) Relationships:

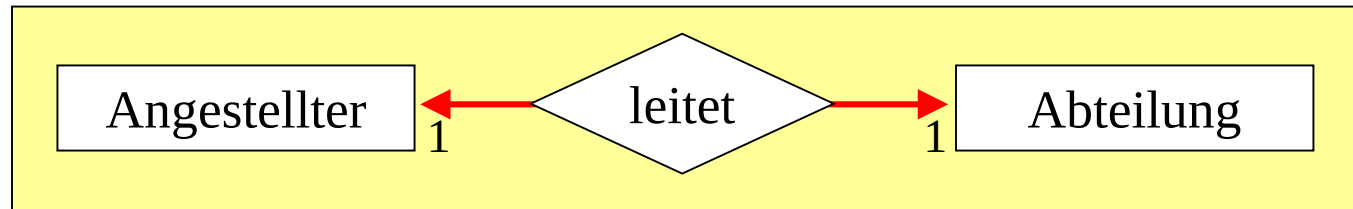


Ausprägung: empfiehlt (Böhm, Heuer&Saake, DBS 1)
empfiehlt (Böhm, Kemper&Eickler, DBS 1)
empfiehlt (Böhm, Bishop, Informatik 1)
empfiehlt (Böhm, Bishop, Programmierkurs)



Funktionalität von Relationships

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

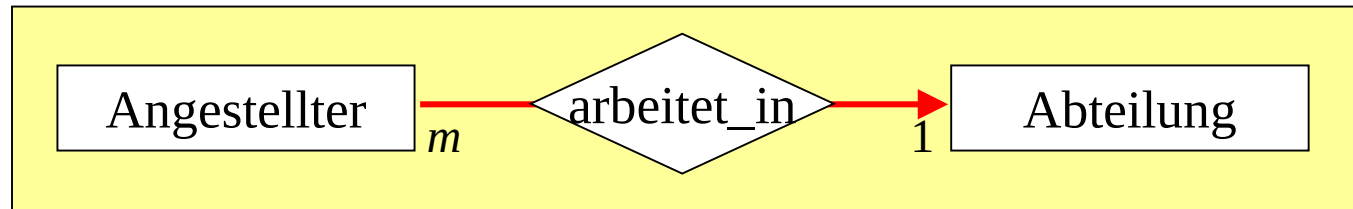


- Charakteristik:
Jedes Objekt aus dem linken Entity steht in Beziehung zu **höchstens** einem Objekt aus dem rechten Entity und umgekehrt.
- Grafische Notation: Pfeile auf jeder Seite
- Beispiel gilt unter der Voraussetzung, dass jede Abteilung max. einen Leiter hat und kein Angestellter mehrere Abteilungen leitet



Funktionalität von Relationships

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

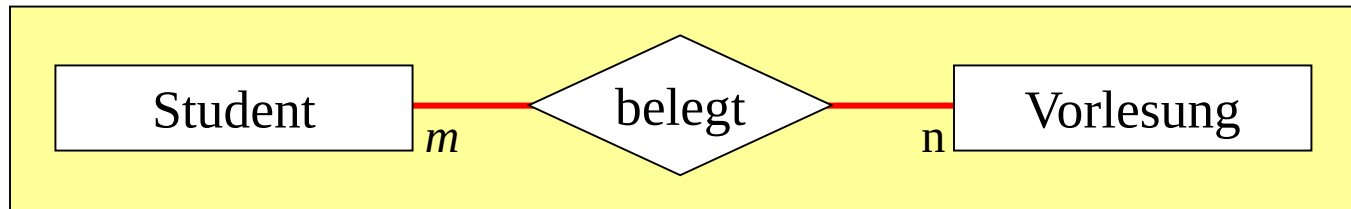


- Charakteristik:
Jedes Objekt auf der "*many*"-Seite steht in Beziehung zu höchstens einem Objekt auf der "*one*"-Seite (im Allgemeinen nicht umgekehrt)
- Grafische Notation:
Pfeil in Richtung zur "*one*"-Seite



Funktionalität von Relationships

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



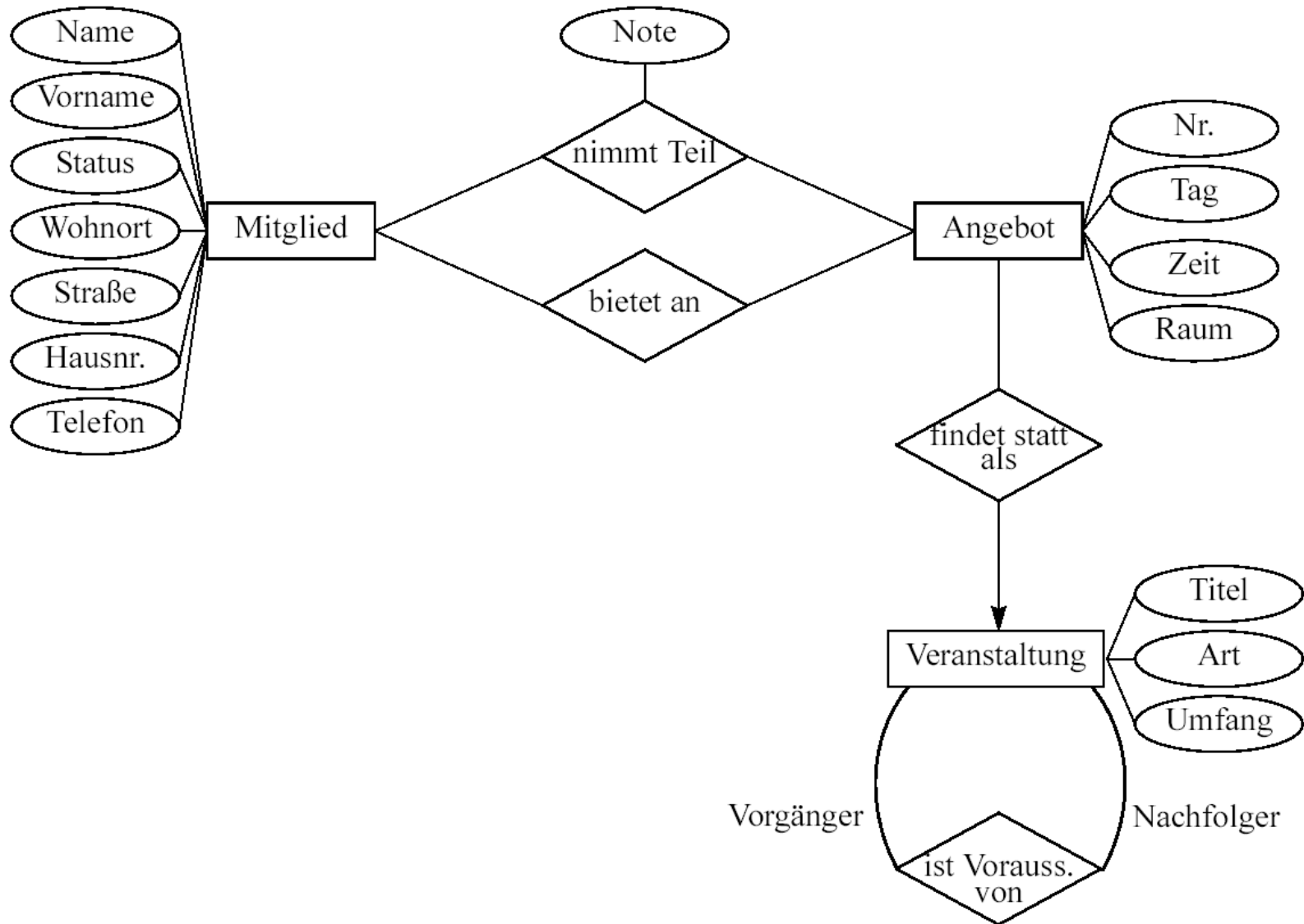
- Charakteristik:
Jedes Objekt auf der linken Seite kann zu mehreren Objekten auf der rechten-Seite in Beziehung stehen (d.h. keine Einschränkung)
- Grafische Notation:
Kein Pfeil

Beispiel-Ausprägung:

belegt (Anton, Informatik 1)
belegt (Berta, Informatik 1)
belegt (Caesar, Wissensrepräsentation)
belegt (Anton, Datenbanksysteme 1)



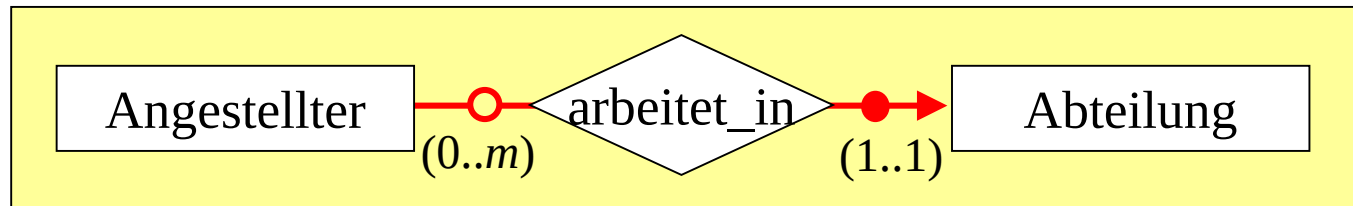
Beispiel: Lehrveranstaltungen





Optionalität

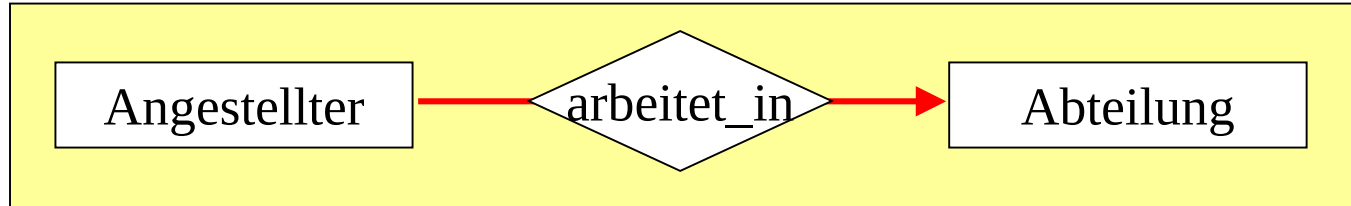
- Das E/R-Modell trifft lediglich Aussagen darüber, ob ein Objekt zu mehreren Objekten in Beziehung stehen darf oder zu max. einem
- keine Aussage, ob es zu mindestens einem Objekt in Beziehung stehen muss
- Erweiterung der Notation mit:
 - Geschlossener Kreis: Verpflichtend mindestens eins.
 - Offener Kreis: Optional
 - Gilt auch für Attribute (vgl. NOT NULL)



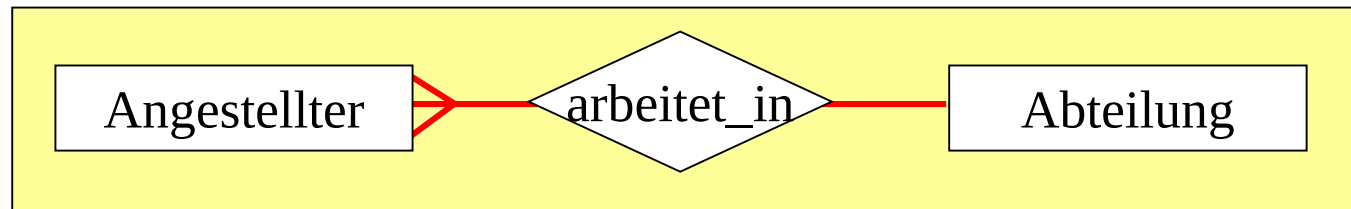


Verschiedene Notationen

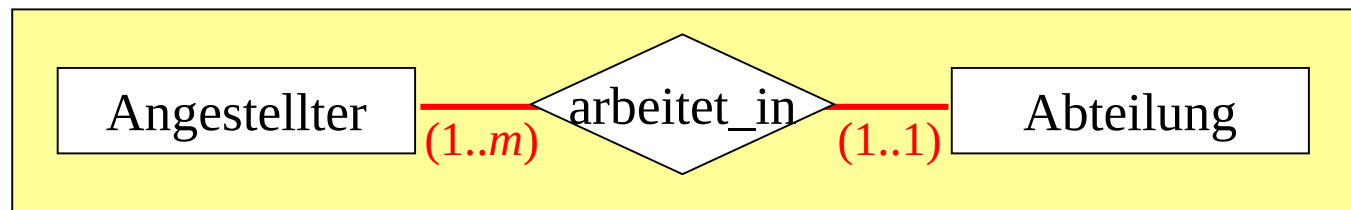
- Pfeil-Notation der Funktionalität:



- Krähenfuß-Notation:



- Kardinalitäts-Notation:

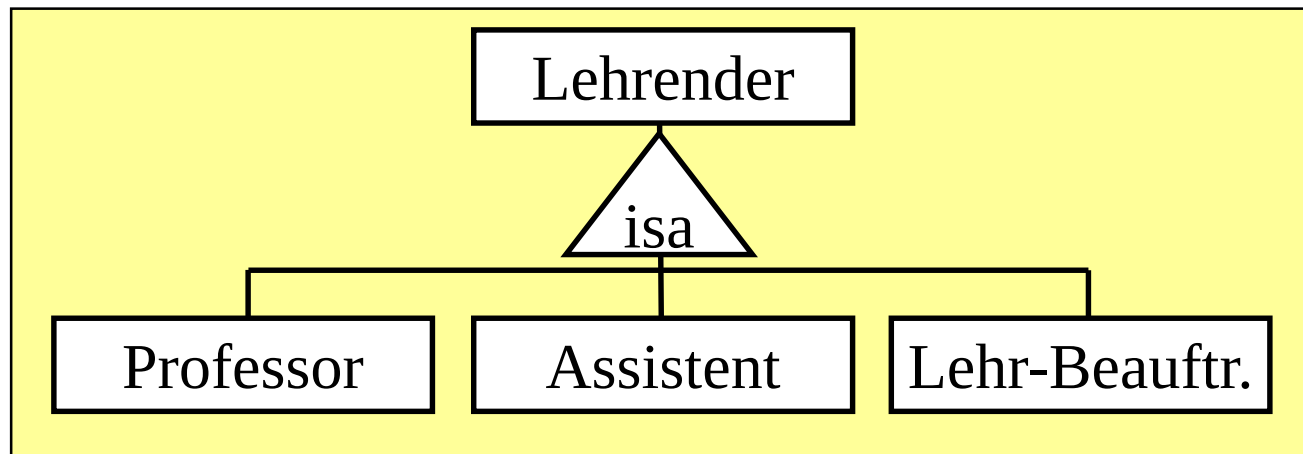


auch m , $(1..\infty)$, $(1..*)$, verschiedene Kombinationsformen



ISA-Beziehung/Vererbung

- Das Erweiterte E/R-Modell kennt Vererbungs-Beziehungen für Entities
- Beispiel:



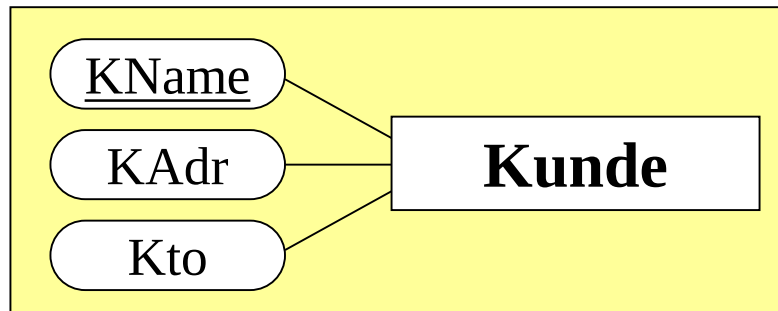
- Charakteristik und Bedeutung:
 - Assistent ist ein Lehrender (engl.: *is a*)
 - Vererbung aller Attribute und Beziehungen



Vom E/R-Modell zur Relation

Einfache Umsetzungsregeln:

- Entities und Attribute:



- Jedem Entity-Typ wird eine Relation zugeordnet
- Jedes Attribut des Entity wird ein Attribut der Relation
- Der Primärschlüssel des Entity wird Primärschlüssel der Relation
- Attribute können im weiteren Verlauf dazukommen

Kunde (KName, KAdr, Kto)



Vom E/R-Modell zur Relation

- Bei Relationships:
Umsetzung abh. von Funktionalität/Kardinalität:
 - 1:1
 - 1:n
 - n:m

Zusätzliche Attribute in bestehende Relationen

Erzeugung einer zusätzlichen Relation

Die ersten beiden Funktionalitäten sind Spezialfälle der dritten.

Deshalb ist es immer *auch* möglich, zusätzliche Relationen einzuführen, jedoch nicht erforderlich



Vom E/R-Modell zur Relation

- One-To-Many Relationships:



- Eine zusätzliche Tabelle wird nicht angelegt
- Der Primärschlüssel der Relation auf der **one**-Seite der Relationship kommt in die Relation der **many**-Seite (Umbenennung bei Namenskonflikten)
- Die neu eingeführten Attribute werden Fremdschlüssel
- Die Primärschlüssel der Relationen ändern sich nicht
- Attribute der Relationship werden ebenfalls in die Relation der **many**-Seite genommen (kein Fremdschl.)



Vom E/R-Modell zur Relation

- Beispiel One-To-Many-Relationship:



Abteilung (ANr, Bezeichnung, ...)

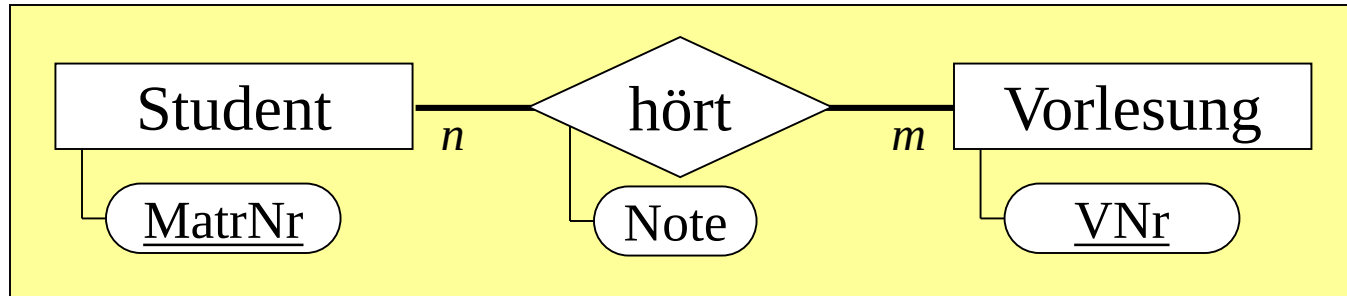
Mitarbeiter (PNr, Name, Vorname, ..., **ANr**)

```
create table Mitarbeiter (  
    PNr char(3) primary key,  
    ...  
    ANr char(3) references Abteilung (ANr) );
```



Vom E/R-Modell zur Relation

- Many-To-Many-Relationships:

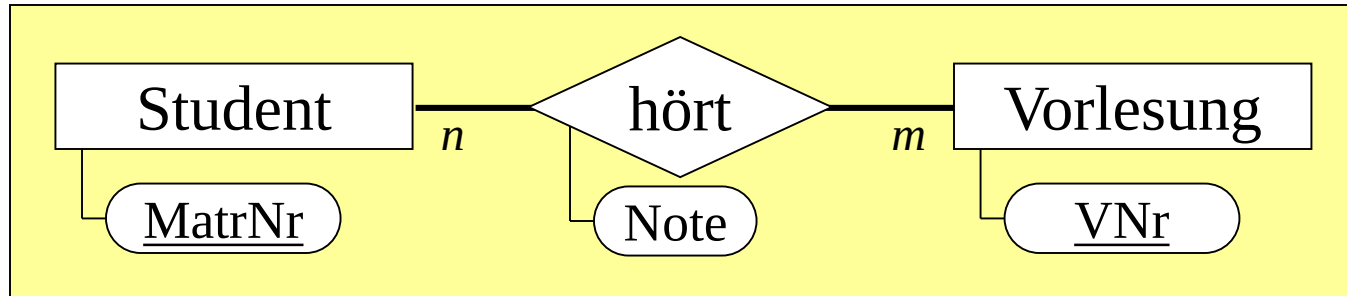


- Einführung einer zusätzlichen Relation mit dem Namen der Relationship
- Attribute: Die Primärschlüssel-Attribute der Relationen, den Entities beider Seiten zugeordnet sind
- Diese Attribute sind jeweils Fremdschlüssel
- Zusammen sind diese Attribute Primärschlüssel der neuen Relation
- Attribute der Relationship ebenfalls in die neue Rel.



Vom E/R-Modell zur Relation

- Beispiel: Many-To-Many-Relationships



Student (MatrNr, ...)

Vorlesung (VNr, ...)

Hoert (MatrNr, VNr, Note)

...

primary key (MatrNr, VNr),

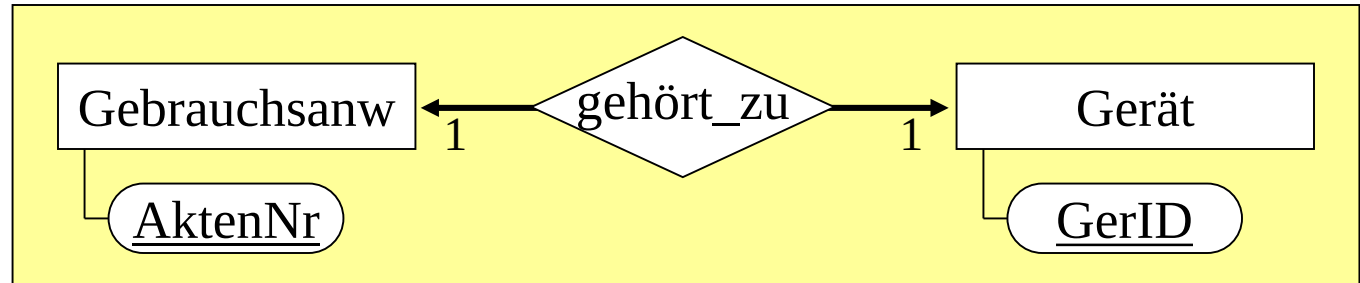
foreign key MatrNr **references** Student,

foreign key VNr **references** Vorlesung...



Vom E/R-Modell zur Relation

- One-To-One-Relationships:



- Die beiden Entities werden zu einer Relation zusammengefasst
- Einer der Primärschlüssel der Entities wird Primärschlüssel der Relation

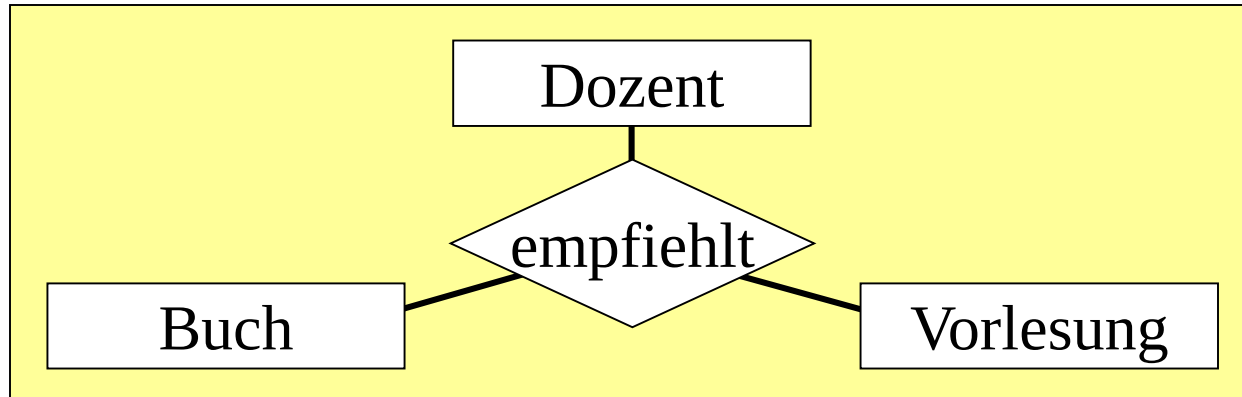
Geraet (GerID, ..., AktenNr, ...)

- Häufig auch Umsetzung wie bei 1:n-Beziehung (insbes. wenn eine Seite optional ist), wobei die Rollen der beteiligten Relationen austauschbar sind



Vom E/R-Modell zur Relation

- Mehrstellige Relationen



- Eigene Relation für *empfiehl*, falls mehr als eine Funktionalität **many** ist:

Dozent (PN_r, ...)

Buch (ISBN, ...)

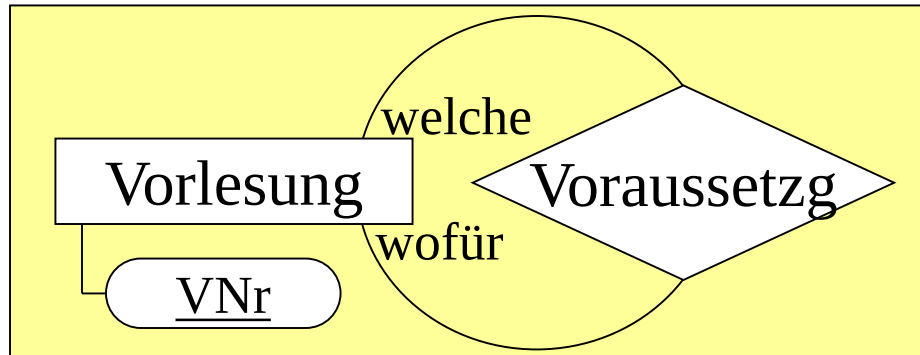
Vorlesung (VN_r, ...)

empfiehl (PN_r, ISBN, VN_r)



Vom E/R-Modell zur Relation

- Selbstbezug



Keine Besonderheiten:

Vorgehen je nach Funktionalität.

Vorlesung (VNr, ...)

Voraussetzg (Welche, Wofuer)

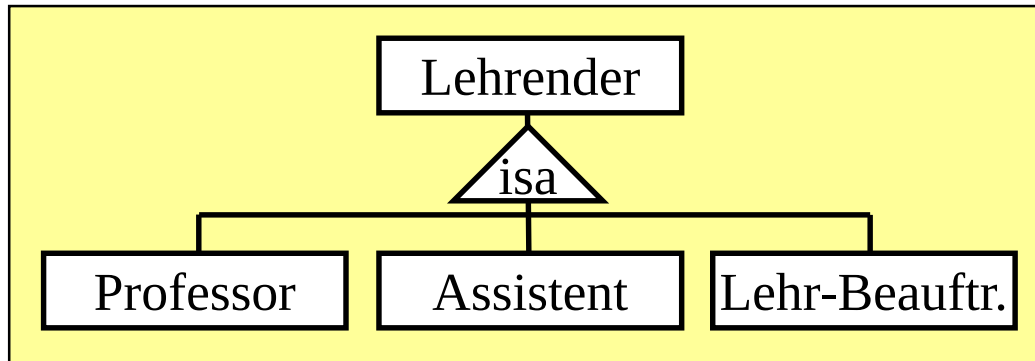
foreign key Welche **references** Vorlesung (VNr),

foreign key Wofuer **references** Vorlesung (VNr)

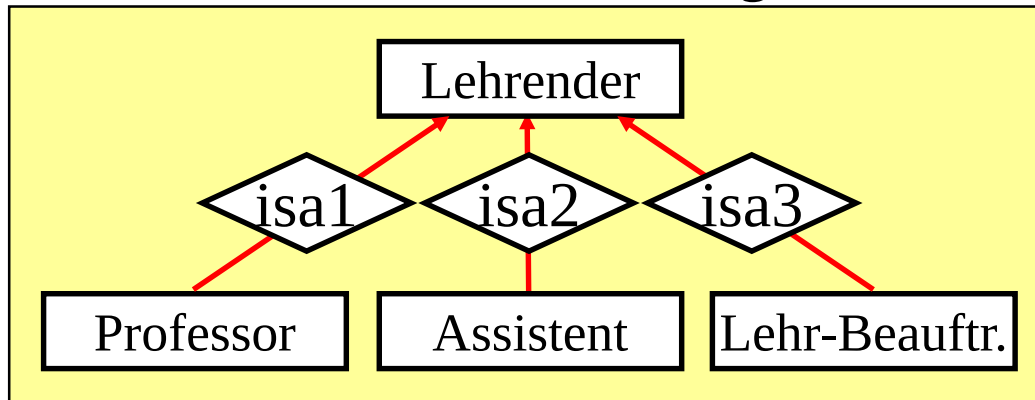


Vom E/R-Modell zur Relation

- Umsetzung der ISA-Beziehung:



- Meist wie bei 1:m-Beziehungen:



- Alternative: Attribute und Relationships von *Lehrender* explizit in *Professor* (...) übernehmen



UML

- Wegen Unübersichtlichkeit und Einschränkungen Verdrängung der E/R-Diagramme durch UML
- Unterschiede:
 - Attribute werden direkt im Entity-Kasten notiert
 - Relationships ohne eigenes Symbol (nur Verbindung)
Ausnahme: Ternäre Relationships mit Raute
 - Verschiedene Vererbungsbeziehungen, Part-of-Bez.
 - (Methoden: Nicht gebraucht bei DB-Modellierung)

