



Skript zur Vorlesung
Datenbanksysteme I
Wintersemester 2011/2012

Kapitel 6: Das E/R-Modell

Vorlesung: Prof. Dr. Christian Böhm
Übungen: Andreas Züfle, Sebastian Goebel

Skript © 2005 Christian Böhm

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



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:



- Beschreibung im relationalen Modell:
`create table student (...);`
`create table vorlesung (...);`

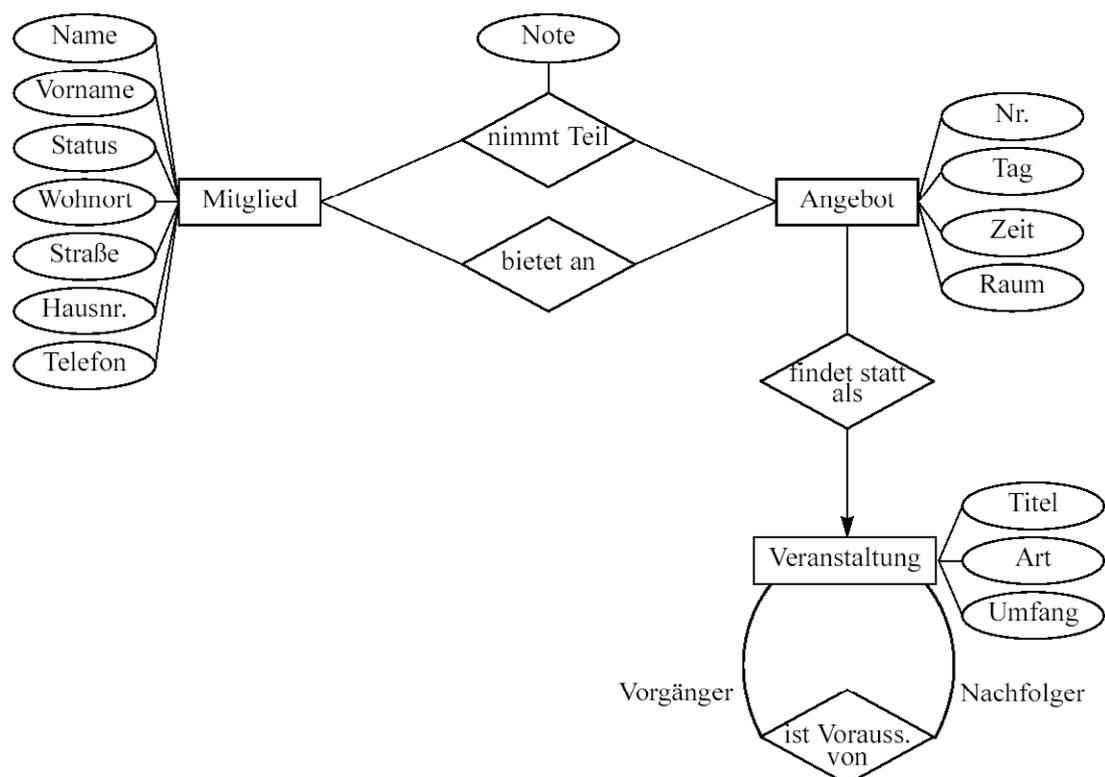


Das Entity/Relationship Modell

- 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 relationales Schema überführt werden
 - Einfache Grundregeln zur Transformation
 - Gewinnung eines *effizienten* Schemas erfordert tiefes Verständnis vom Zielmodell



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, ...

Attribute

- Entities durch charakterisierende Eigenschaften beschrieben
- Einfache Datentypen wie INT, STRING usw.
- Bsp: Farbe, Gewicht, Name, Titel, ...
- Häufig beschränkt man sich auf die wichtigsten Attribute

Schlüssel

- ähnlich definiert wie im relationalen Modell
- Primärschlüssel-Attribute werden unterstrichen

Matrikel-Nr.

Student

Name

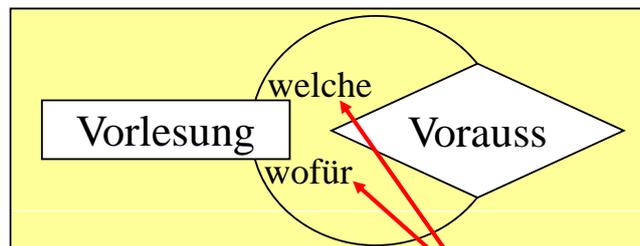


Relationships (Beziehungen)

- Stellen Zusammenhänge zwischen Entities dar
- Beispiele:



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



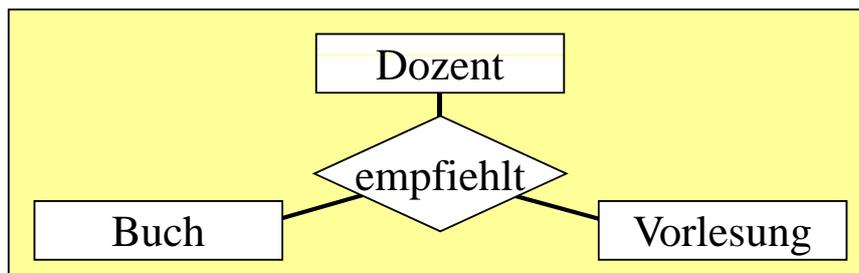
Ausprägung:
Voraus (I 1, DBS 1)
Voraus (I 1, Software-Eng.)
Voraus (DBS 1, DBS 2)
Voraus (I 1, Wissensrepr.)

unterschiedliche Rollen



Relationships (Beziehungen)

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

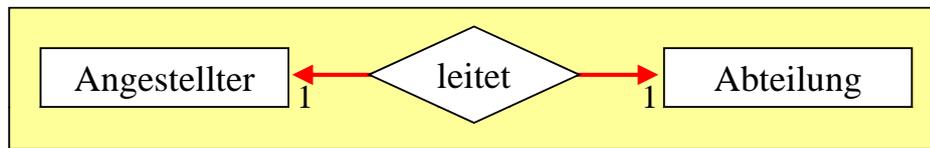


Ausprägung: empfiehl (Böhm, Heuer&Saake, DBS 1)
empfiehl (Böhm, Kemper&Eickler, DBS 1)
empfiehl (Böhm, Bishop, Informatik 1)
empfiehl (Böhm, Bishop, Programmierkurs)



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)

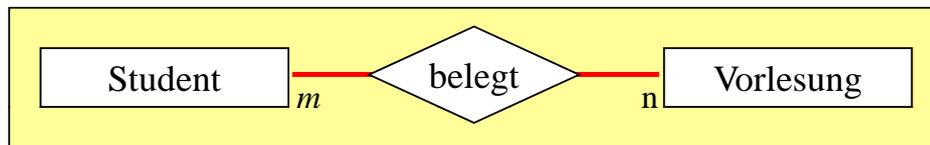


- Charakteristik:
Jedes Objekt auf der "many"-Seite steht in Beziehung zu höchstens einem Objekt auf der "one"-Seite (im Allgemeinen nicht umgekehrt)
- Grafische Notation:
Pfeil in Richtung zur "one"-Seite



Funktionalität von Relationships

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

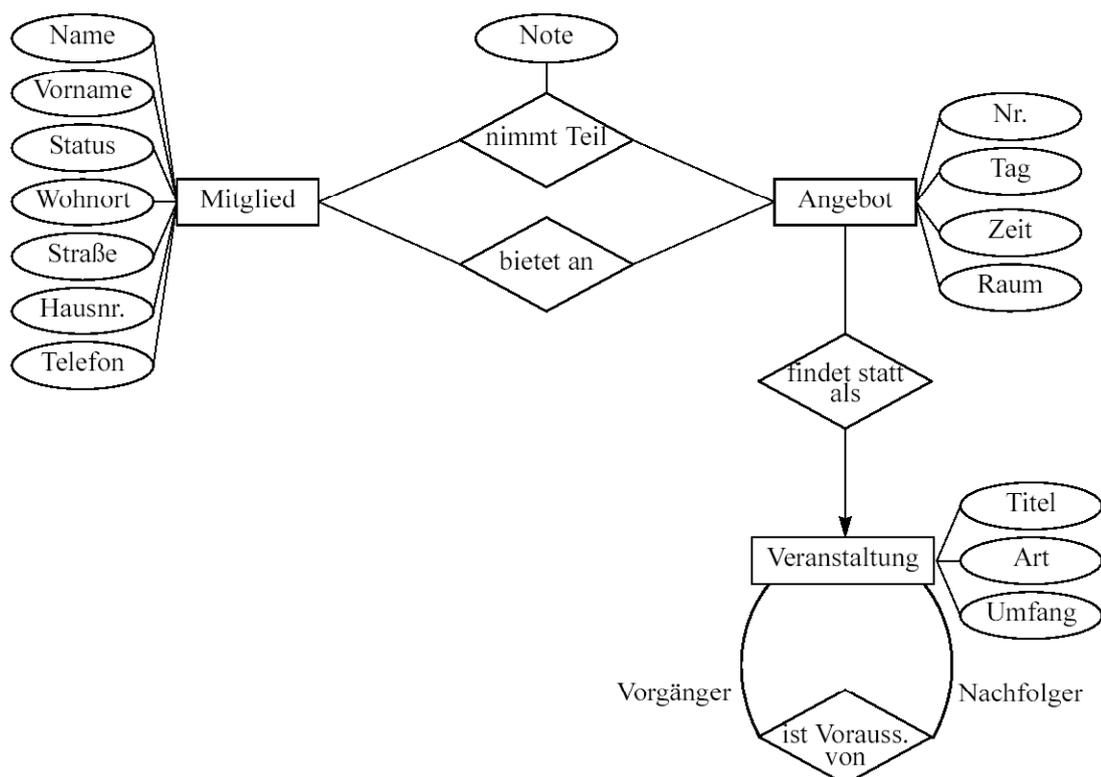


- Charakteristik:
Jedes Objekt auf der linken Seite kann zu mehreren Objekten auf der rechten-Seite in Beziehung stehen (d.h. keine Einschränkung)
- Grafische Notation:
Kein Pfeil

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



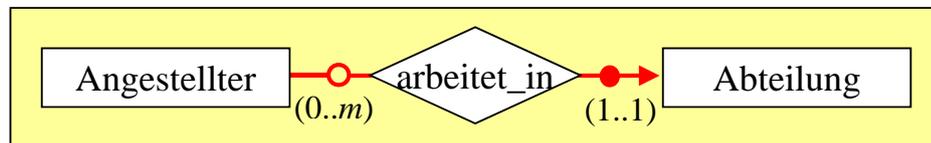
Beispiel: Lehrveranstaltungen





Optionalität

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

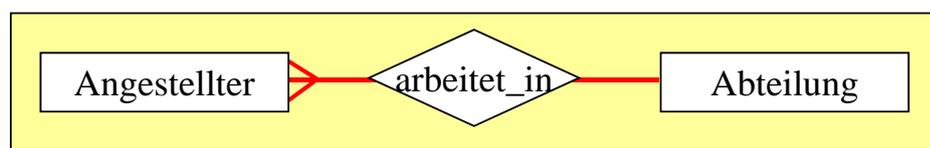


Verschiedene Notationen

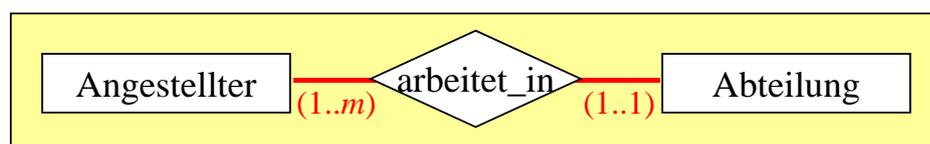
- Pfeil-Notation der Funktionalität:



- Krähenfuß-Notation:



- Kardinalitäts-Notation:

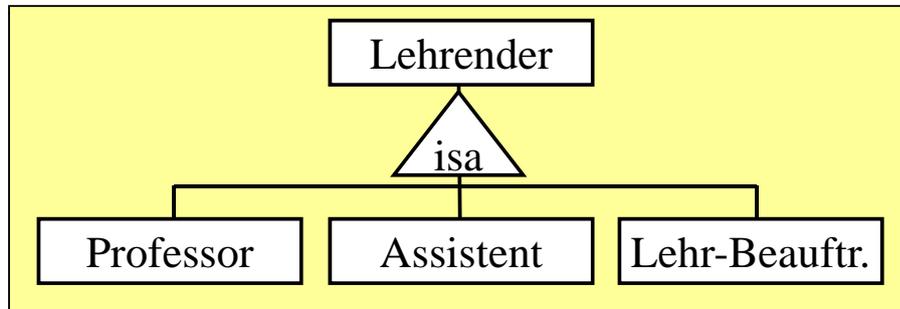


auch m , $(1..∞)$, $(1..*)$, verschiedene Kombinationsformen



ISA-Beziehung/Vererbung

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



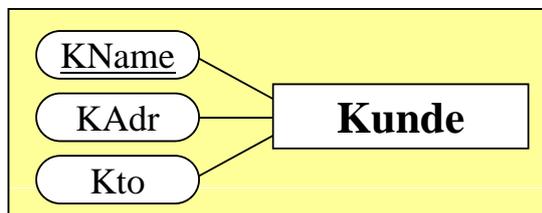
- Charakteristik und Bedeutung:
 - Assistent *ist ein* Lehrender (engl.: *is a*)
 - Vererbung aller Attribute und Beziehungen



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 } Zusätzliche Attribute in bestehende Relationen
 - 1:n }
 - n:m 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:



- Keine zusätzliche Tabelle wird 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, ...)

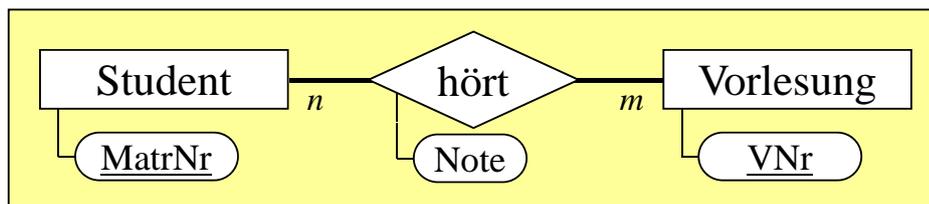
Mitarbeiter (PNr, Name, Vorname, ..., ANr)

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



Vom E/R-Modell zur Relation

- Many-To-Many-Relationships:

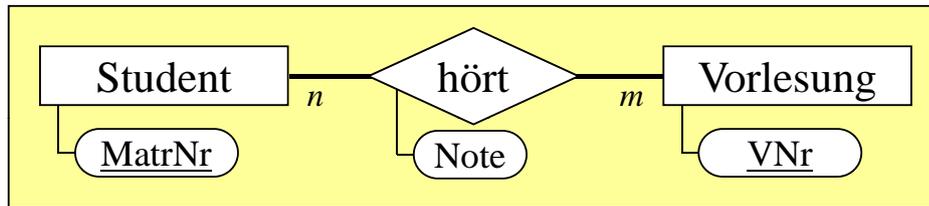


- Einführung einer zusätzlichen Relation mit dem Namen der Relationship
- Attribute: Die Primärschlüssel-Attribute der Relationen, den Entities beider Seiten zugeordnet sind
- Diese Attribute sind jeweils Fremdschlüssel
- Zusammen sind diese Attribute Primärschlüssel der neuen Relation
- Attribute der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

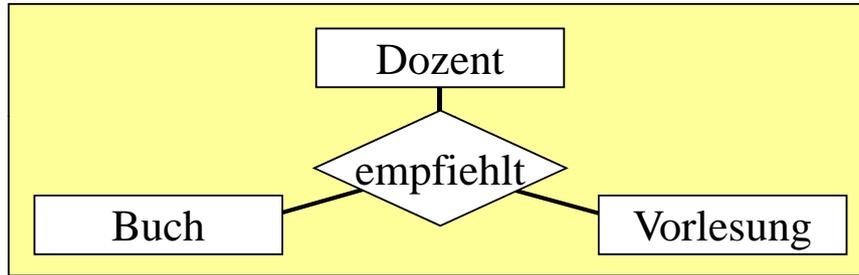
Geraet (GerID, ..., AktenNr, ...)

- 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 (PNr, ...)

Buch (ISBN, ...)

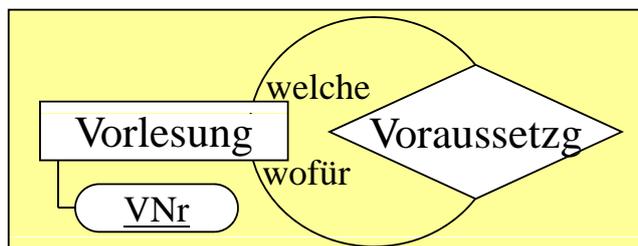
Vorlesung (VNr, ...)

empfiehl (PNr, ISBN, VNr)



Vom E/R-Modell zur Relation

- Selbstbezug



Keine Besonderheiten:

Vorgehen je nach Funktionalität.

Vorlesung (VNr, ...)

Voraussetzg (Welche, Wofuer)

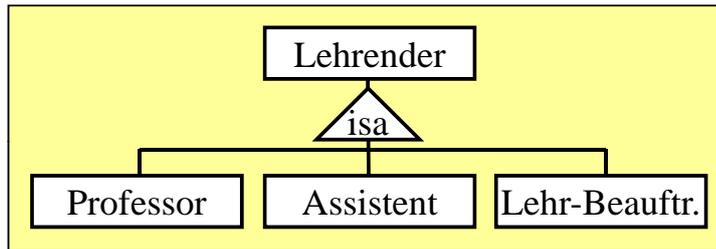
foreign key Welche **references** Vorlesung (VNr),

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

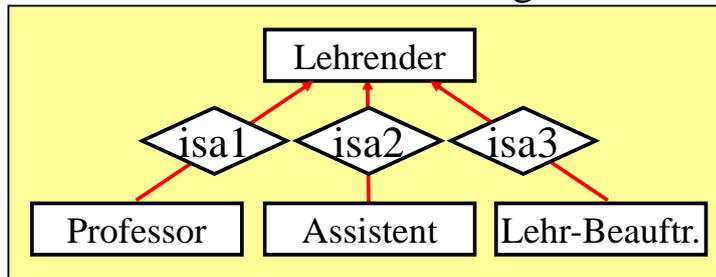


Vom E/R-Modell zur Relation

- 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)

