Das Entity-Relationship-Modell

Datenmodelle

- Datenmodell
  - System von Konzepten zur abstrakten Darstellung eines Ausschnitts der realen Welt mittels Daten
  - Verschiedene Abstraktionsebenen
  - Bestehen aus:
    - Strukturen (statische Eigenschaften)
    - Operatoren (dynamische Eigenschaften)
    - Constraints (Korrektheitsbedingungen)
  - Werte ohne Struktur sind sinnlos

Beispiel:

301	91	19	5
302	91	18	2
303	91	22	9
304	91	12	-3

Tag	Jahr	Tmax	Tmin
301	91	19	5
302	91	18	2
303	91	22	9
304	91	12	-3

Datenmodelle (Forts.)

Operatoren erlauben Zugriff, Speicherung, Änderung von Werten, z.B.
Insert 305 91 -6 11

Constraints garantieren die syntaktische und semantische Korrektheit einer Operation (und dadurch die Konsistenz eines Datenbankzustandes)

Beispiel:

Tmax > Tmin macht obige Operation ungültig

Oft sind Konsistenzregeln in den Strukturen inhärent (für den Benutzer intuitiv), müssen aber für das DBMS explizit dargestellt werden.

Konzeptueller DB-Entwurf

- Konzeptueller Entwurf
  - Entity-Relationship-Modell ist traditioneller Ansatz
  - Was sind die Entitäten und die Beziehungen im gewählten Weltausschnitt?
  - Welche Information über diese Entitäten und Beziehungen sollen in der DB gespeichert werden (Informationsbedarfsanalyse)?
  - Was sind die Integritätsbedingungen (oder Business Rules), die gelten müssen?
  - Ein DB-Schema kann graphisch im ER-Modell repräsentiert werden (ER-Diagramm)
  - Ein ER-Diagramm läßt sich in ein relationales Schema übersetzen (logischer DB-Entwurf)

Phasen des DB-Entwurfs

- Requirements-Analyse
  - Welche Daten?
  - Welche (häufigen) Operationen?
  - Welche Anwendungen?
  - Nicht-funktionale Anforderungen, z.B. Performance
- Konzeptueller DB-Entwurf
  - Spezifikation der gesammelten Anforderungen in einer high-level-Darstellung (z.B. ER-Modell)
- Logischer DB-Entwurf
  - Übersetzung des konzeptuellen DB-Entwurfs in ein Schema im Datenmodell des Ziel-DBMS (zumeist relationales DBMS)
- Schema-Verfeinerung
  - Normalisierung des relationalen Schemas soweit erforderlich (Nutzung Normalformen-Theorie)
- Physischer DB-Entwurf
  - Phys. Entwurfsentscheidungen (Index, Clusters) entsprechend Last-Profilen und Performance-Anforderungen
- Security-Entwurf
  - Definition von Benutzergruppen, Rollen, Zugriffsrechten

Konzept 1: Entity-Menge

- Entity: “A thing that has a real or individual existence in reality or in mind“ (Webster)
- Entity ist von anderen Objekten unterscheidbar, wird beschrieben durch eine Menge von Attributen (in DB)
- Entity-Menge: Zusammenfassung aller Entities mit gemeinsamen Eigenschaften

  Elemente einer Menge $e \in E$

  z.B. Personen, Bücher, Projekte, Kunden, Wein
- Zugehörigkeit über Prädikat entscheidbar

  $e_i \in E_j \Leftrightarrow$ is $E_j(e_i)$
- DB enthält endlich viele Entity-Mengen

  E_1, E_2, E_3

  z.B. E_1 ... Personen

  E_2 ... Kunden $\quad E_2 \subseteq E_1$

Angestellter

Konzept 2: Relationship-Menge

- Relationship: Beziehung zwischen zwei oder mehreren Entities, z.B. “John arbeitet in der Vertriebsabteilung“
- Zusammenfassung von gleichartigen Beziehungen (Relationships) zwischen Entities, die jeweils gleichen Entity-Mengen angehören, z.B. *ist Hörer von* zwischen *Student* und *Vorlesung*
- R ... Relationship-Menge = math. Relation zwischen n E_i

  $R \subseteq E_1 \times E_2 \ldots \times E_n$

  d.h. $R = \{r = [e_1, e_2 \ldots e_n] \mid e_1 \in E_1 \ldots e_n \in E_n\}$

  gewöhnlich n=2 oder n=3

  Keine Disjunktheit der Entity-Mengen, die an einer R_i beteiligt sind, gefordert (d.h. dieselbe Entity-Menge nimmt in verschiedenen Rollen teil)

  z.B. HEIRAT: PERSON (MANN), PERSON (FRAU)

Konzept 3: Wertemengen

- Information über e_i oder r_i wird ausgedrückt durch Attribut-Wert-Paare
- Wertemengen W_i (Domains) bestimmen Zulässigkeit konkreter Werte für e_i und r_i
- Definition durch Aufzählung, Prädikate
  - Beispiele:
    - NUMMER = neunstellige natürliche Zahl
    - QUALITÄT = {1,2,3,4,5}
    - NACHNAME = Menge der max. 35 langen Zeichenfolgen über Alphabet

Konzept 4: Attribute

- Attribut A zu einer Entity-Menge E oder Relationship-Menge R
- Mathematische Funktion

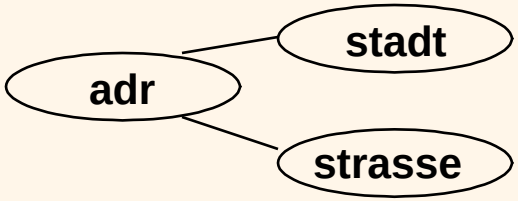
  A: $E \rightarrow W$ bzw. $W_1 \times W_2 \ldots \times W_k$

  oder

  A: $R \rightarrow W$ bzw. $W_1 \times W_2 \ldots \times W_m$

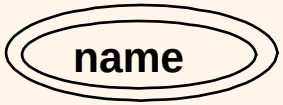
Darstellung als Oval

- Einfache vs. Zusammengesetzte Attribute
  - KDNR
  - NAME
  - ANSCHRIFT



→

- Einwertige (single-valued) vs. Mehrwertige (multi-valued) Attribute
  - FARBE
  - KINDER



- Nullwerte: Attributwert nicht möglich bzw. unbekannt A(e) ={}, z.B. private Tel.-Nr.
- Wertemengen W_i nicht notwendig verschieden
- Relationship-Mengen können auch Attribute besitzen, z.B. DATUM einer HEIRAT

Konzept 5: Schlüssel

- Information über ein Entity ausschließlich über Attribut-Werte
- Identifikation eines Entities durch Attribut (oder Kombination von Attributen)
- $\{ A_1, A_2 \ldots A_m \} = A$ sei Menge der Attribute zur Entity-Menge E

  $R = \{r = [e_1, e_2 \ldots e_n] \mid e_1 \in E_1 \ldots e_n \in E_n \}$

  $K \subseteq A$ ist Schlüsselkandidat von E

  $\Leftrightarrow$ K $\neq$minimal; $e_i, e_k \in E$;

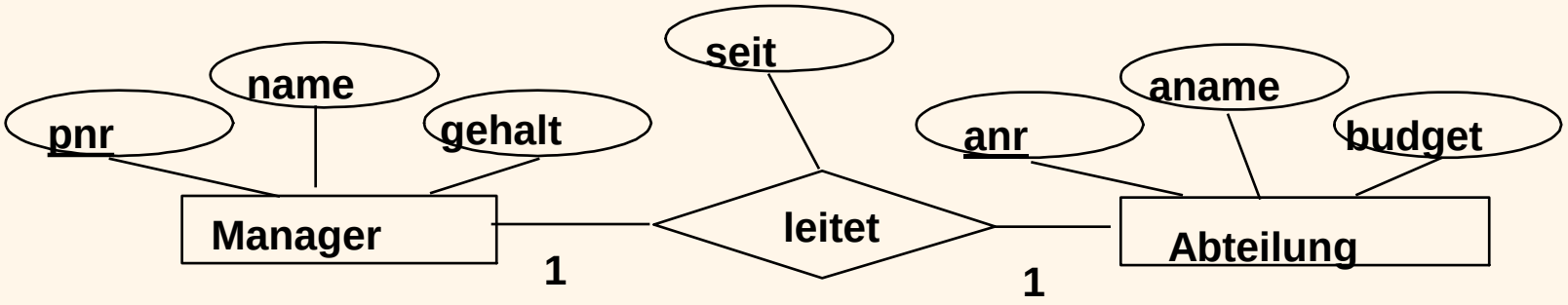
  $e_i \neq e_k \rightarrow K(e_i) \neq K(e_k)$,
- Mehrere Schlüsselkandidaten möglich

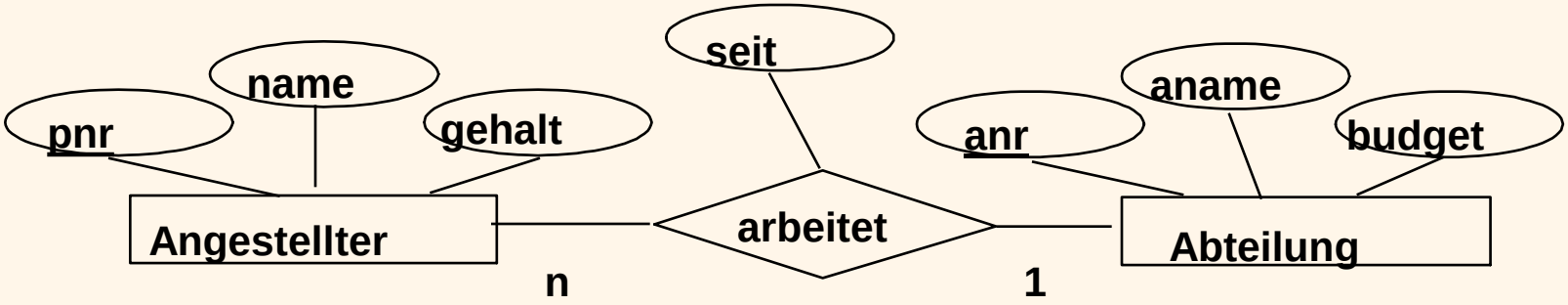
  z.B. Personalausweis-Nr. oder Sozialversicherungs Nr. für Angestellte

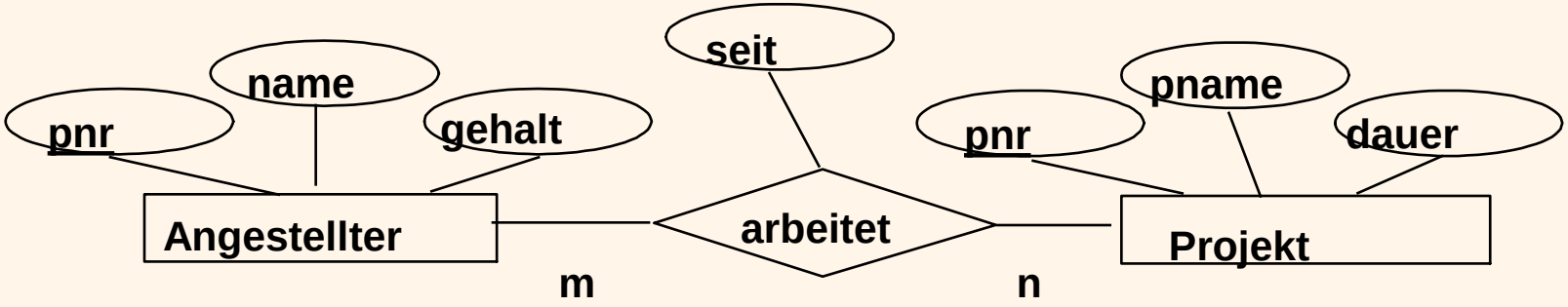
  $\Rightarrow$ Auswahl eines Primärschlüssels

Attributname unterstrichen

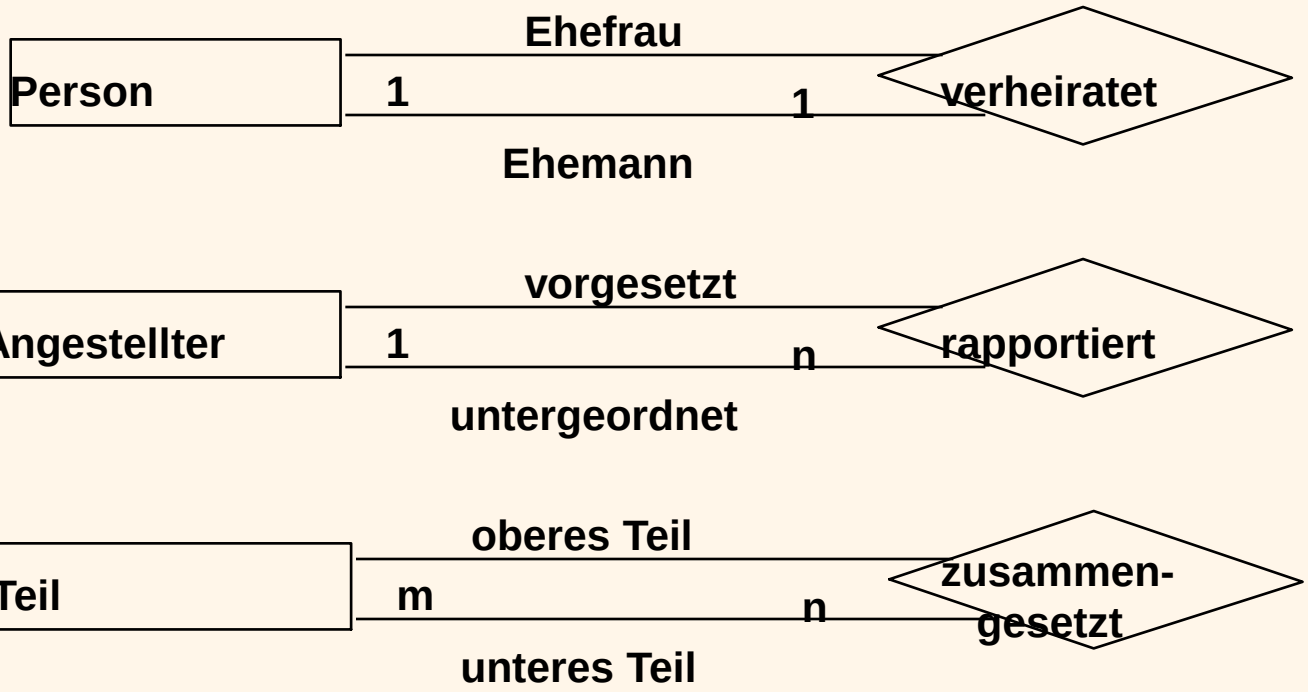
Beispiele (1)


seit
name
pnr
gehalt
aname
anr
budget
Manager
leitet
Abteilung
1
1

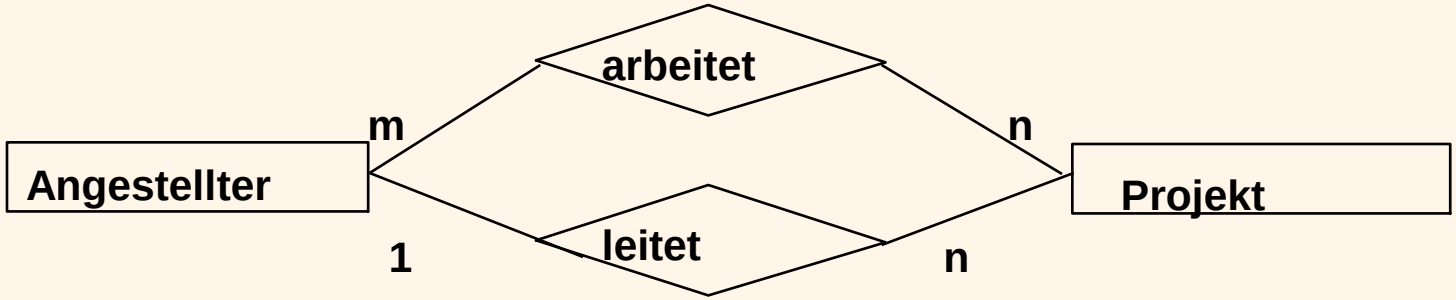

seit
name
pnr
gehalt
aname
anr
budget
Angestellter
arbeitet
Abteilung
n
1


seit
name
pnr
gehalt
pname
pnr
dauer
Angestellter
arbeitet
Projekt
m
n

Beispiele (2)



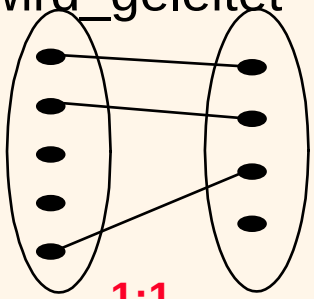
Zwischen den gleichen Entity-Mengen können jeweils unterschiedliche Relationship-Mengen definiert werden.



Kardinalität von Beziehungen

leitet / wird_geleitet

Manager

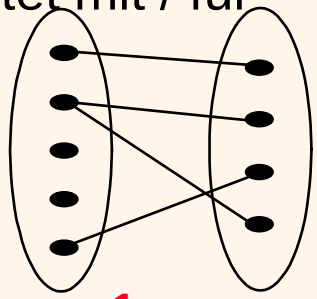


Abteilung

1:1

arbeitet mit / für

Manager

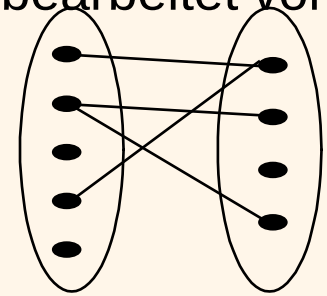


Mitarbeiter

1:n

arbeitet an / bearbeitet von

Mitarbeiter



Projekt

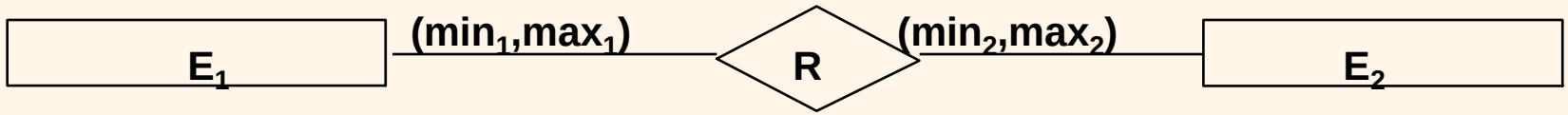
m:n

Integritätsbedingungen in Relationships

- Verfeinerung der Semantik einer Beziehung

  Sei $R \subseteq E_1 \times E_2 \ldots \times E_n$

  card(R, E_i) = (min,max) bedeutet, daß jedes Element aus E_i in wenigstens min und höchstens max Ausprägungen von R enthalten sein muß (mit $0 \leq min \leq max$, $max \geq 1$)
- Graphische Darstellung

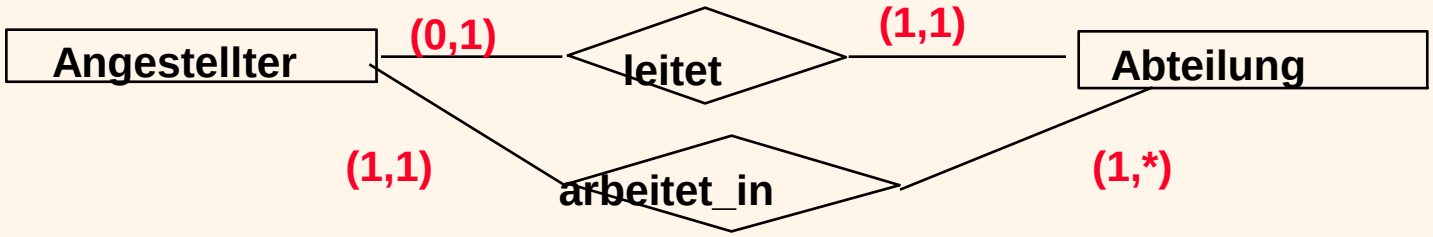


Verfeinerung der Semantik einer Beziehung

e_1 nimmt an (**min_1,max_1**) Beziehungen von Typ R teil

e_2 nimmt an (**min_2,max_2**) Beziehungen von Typ R teil

Beispiel

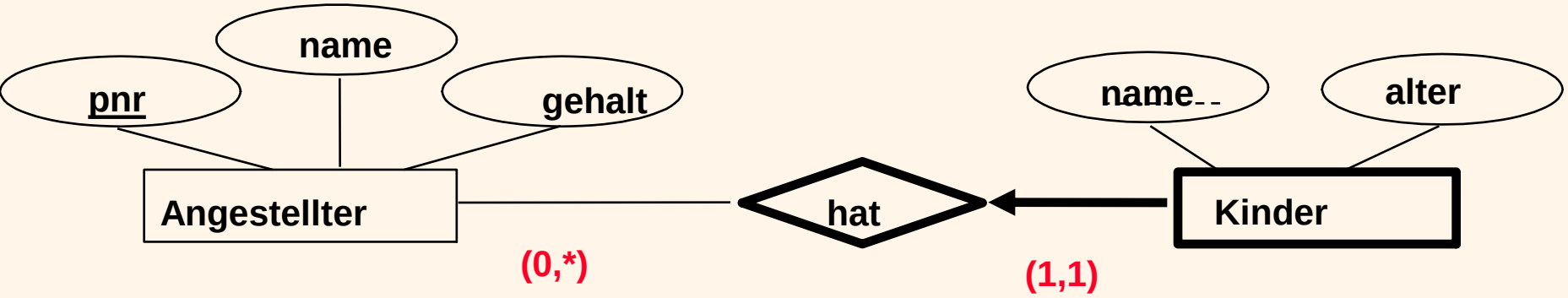


Komplexität binärer Relationships

Kompl.	Bemerkung	Beispiel
(1,1)(1,1)	Strenge 1:1-Beziehung, umkehrbar eindeutige Funktion	Ehe zwischen Ehemännern und Ehefrauen
(1,1)(0,1) (0,1)(1,1)	Partielle 1:1	Ehe zwischen Ehemännern und Frauen
(0,1)(0,1)	Allgemeine 1:1	Ehe zwischen Männern und Frauen
(1,1)(1,*) (1,*)(1,1)	Strenge hierarchische Beziehung	Angestellte in einer Abteilung
(1,1)(0,*) (0,*)(1,1)	Funktionale Abhängigkeit ohne Existenzbedingung	Beziehung zwischen Männern (potentiellen Vätern) und Kindern
(0,1)(1,*) (0,1)(0,*) (1,*)(0,1) (0,*)(0,1)	Allgemeine hierarch. Beziehung (1:n)	
(k,l)(r,s)	Âllgemeine m:n-Beziehung (l,s $\geq$1)	Angestellte arbeiten für Projekte

Schwache Entities

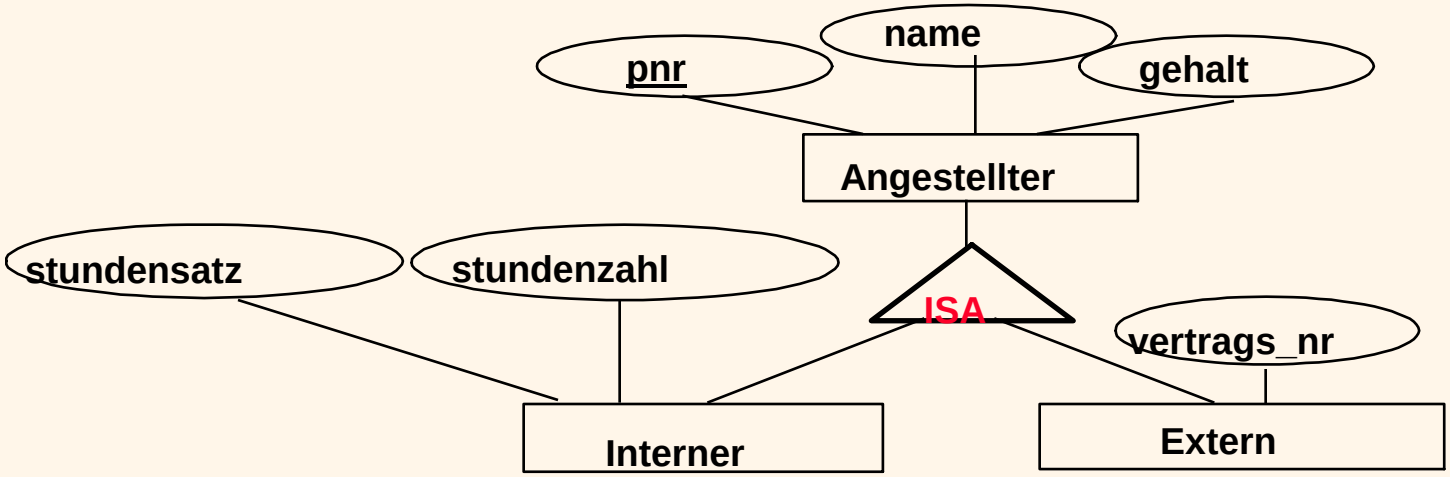
- Schwaches Entity (*weak entity*) kann eindeutig identifiziert werden nur über den Primärschlüssel einer anderen (Owner) Entity.
- Owner Entity und Weak Entity müssen in einer 1:n-Beziehung stehen (ein Owner, mehrere Weak Entities)



Jedes Entity aus Kinder **muß** an der Beziehung teilnehmen (*total Participation Constraint*)

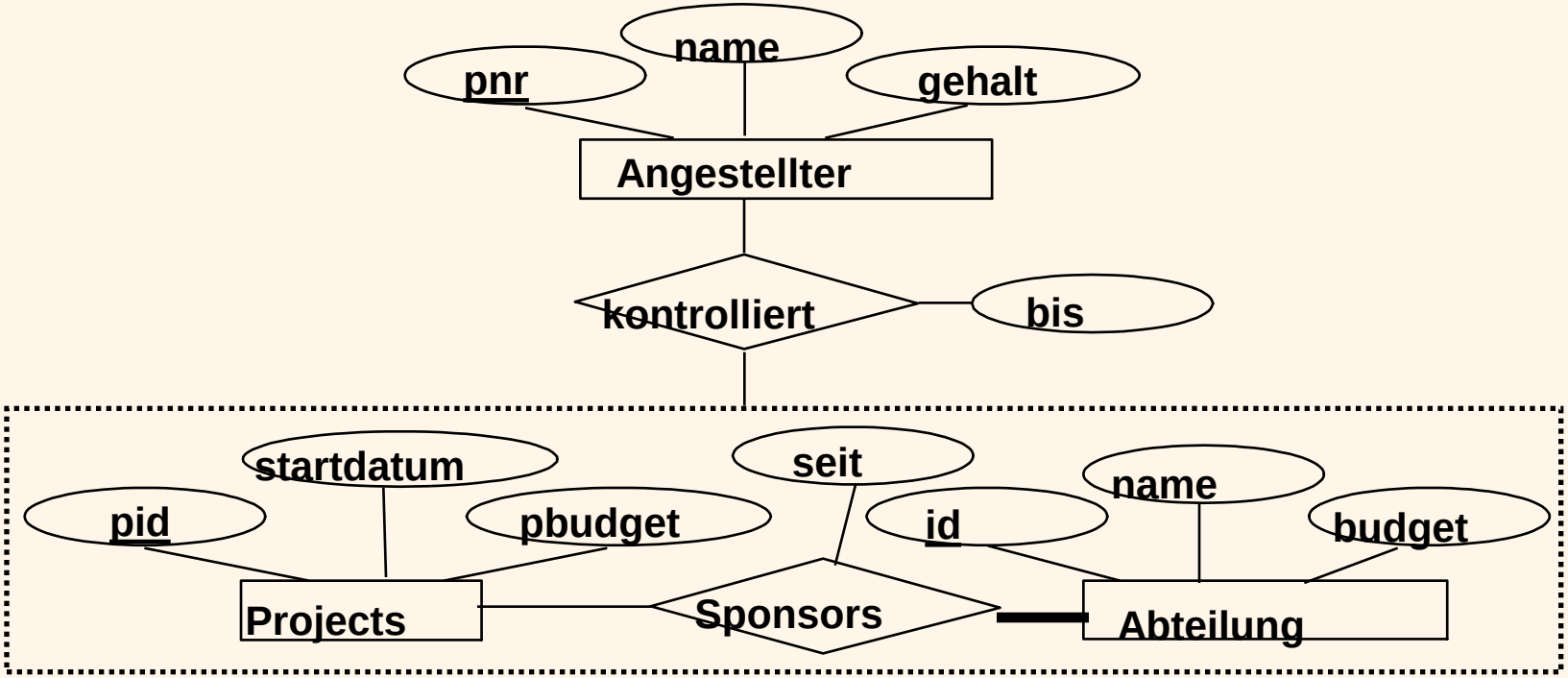
ISA-Beziehung

- Vererbung von Attributen (vgl. OO Sprachen)
- A “is a“ B heißt: Jedes A Entity ist zugleich auch ein B Entity
- *Overlap Constraints*: Kann Joe zugleich ein Fest-Angestellter und ein Contractor sein (erlaubt / nicht erlaubt)?
- *Covering Constraints*: Kann jeder Angestellte in Intern oder Extern klassifiziert werden (ja/nein)
- Nutzen von ISA:
  - Hinzufügen von Attributen spezifisch für einen Subtyp
  - Identifizieren von Entities, die an Beziehungen teilnehmen (Generalisierung)



Aggregation*

- Zur Modellierung von Relationships zwischen Entity-Mengen und Relationship-Mengen
- Erlaubt es, eine Relationship-Menge (z.B. “Sponsors“) als eine Entity-Menge zu betrachten, um an einer anderen Relationship teilzunehmen

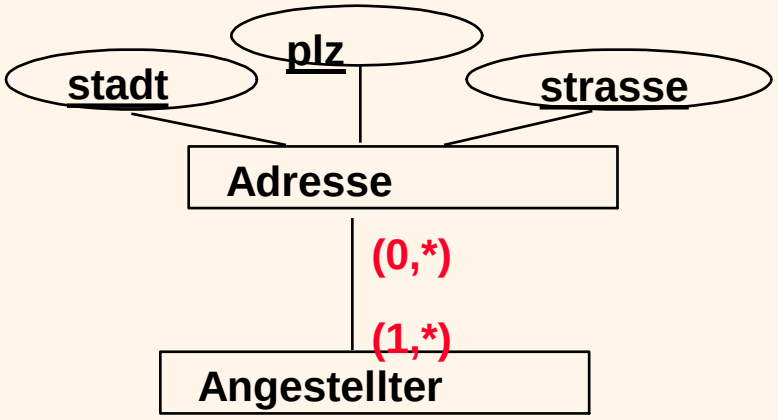


Konzeptueller Entwurf im ER-Modell

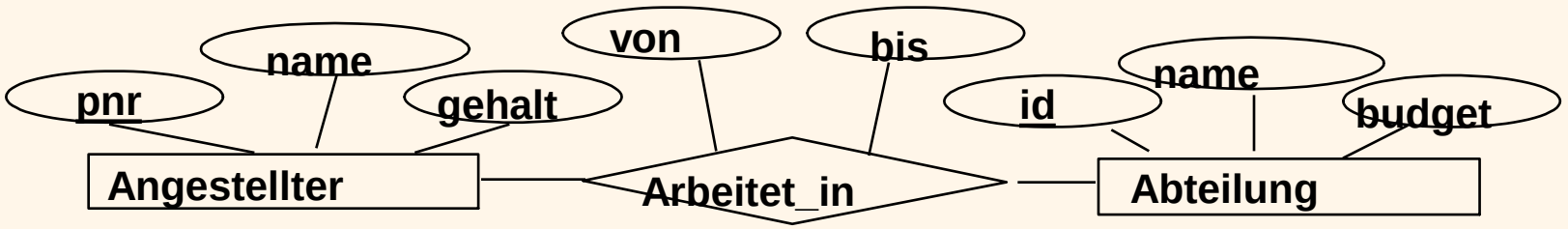
- Entwurfsentscheidungen
  - Sollte ein Konzept als Entity oder Attribut modelliert werden?
  - Sollte ein Konzept als Entity oder Relationship modelliert werden?
  - Bestimme die Beziehungen: Binär oder ternär? Aggregation?
- Constraints im ER-Modell
  - Möglichst viel Datensemantik sollte erfaßt werden
  - Einige Constraints können mit den Mitteln des ER nicht ausgedrückt werden (z.B. Wertabhängigkeiten zwischen Attributen)

Entity vs. Attribut

- Sollte *Adresse* ein Attribut von *Angestellter* sein oder ein Entity (in Beziehung zu Angestellter)
- Abhängig von der Benutzung der Adress-Information und der Semantik der Daten
  - Wenn mehrere Adressen pro Angestellter vorhanden, dann sollte Adresse ein eigenständiges Entity sein (wenn mengenwertige Attribute ausgeschlossen sind)
  - Wenn die Struktur der Adresse wichtig ist, d.h. Zugriff auf Bestandteile der Adresse (wie *Stadt*), dann muß Adresse als separates Entity modelliert werden (mit atomaren Attributwerten)

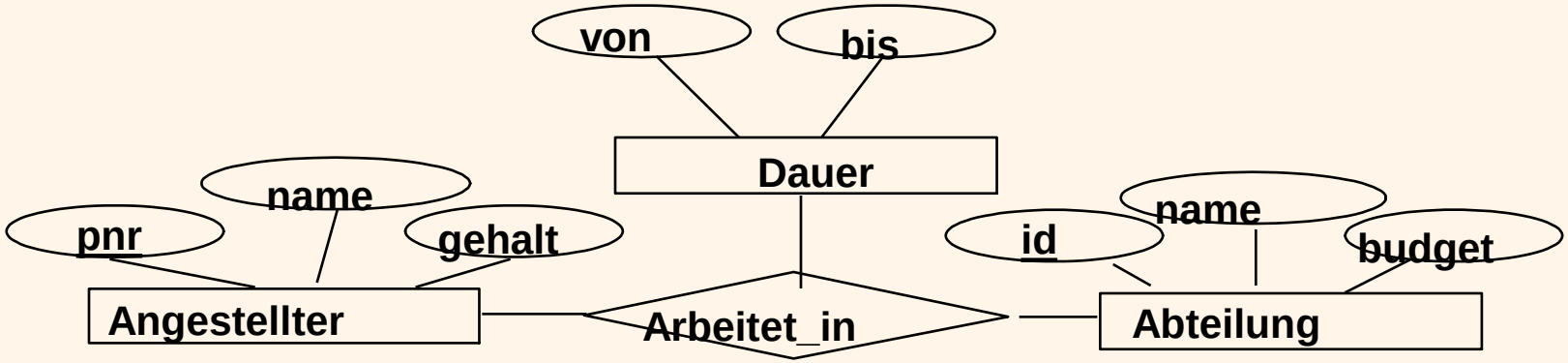


Entity vs. Attribut (Forts.)



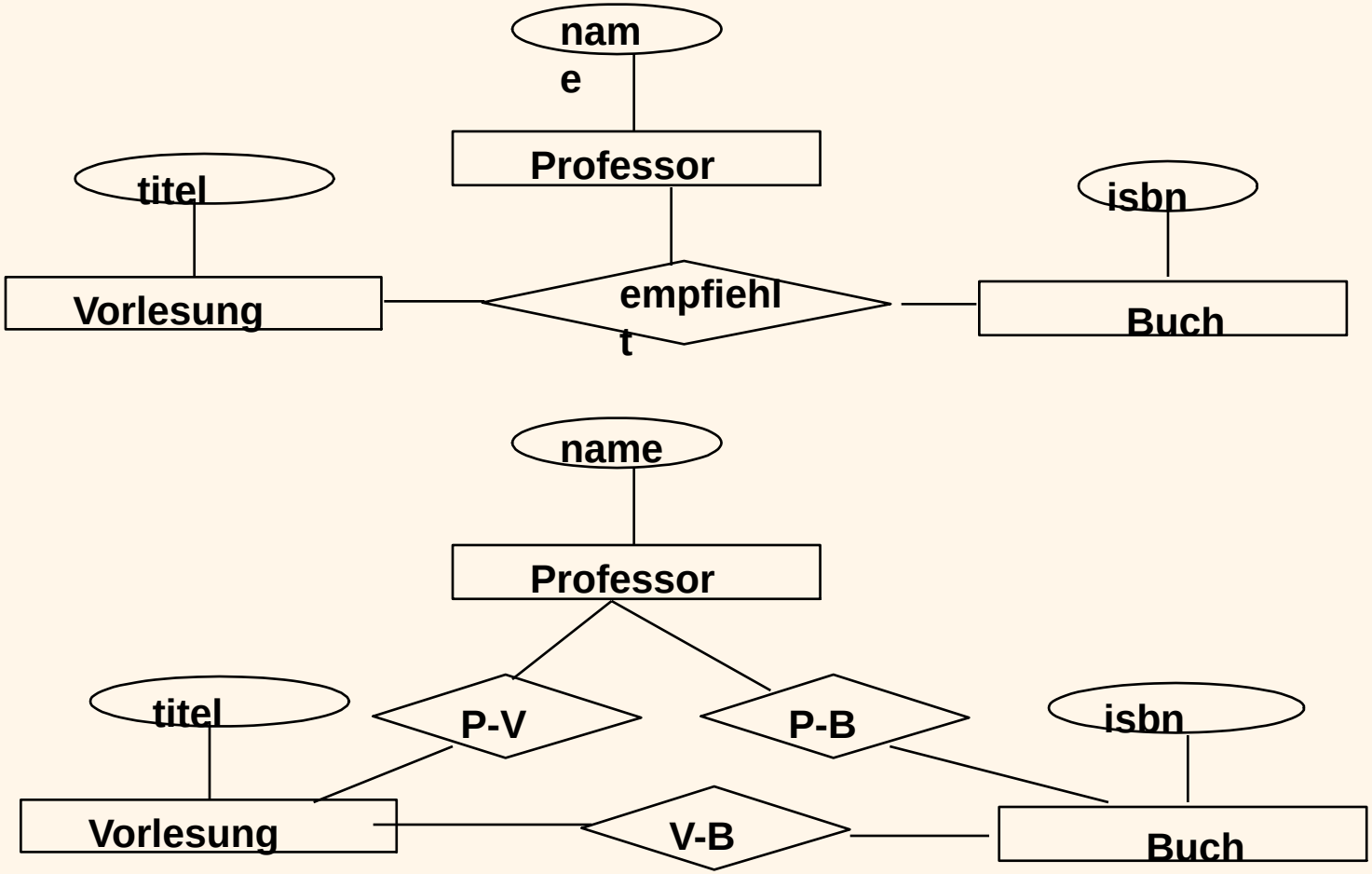
- So ist es nicht möglich, die Mitarbeit eines Angestellten über mehrere Zeiträume zu modellieren (ähnlich wie mehrere Adressen eines Mitarbeiters) → Ersetzen der zeitbezogenen Attribute (von,bis) durch das neue Entity Dauer

  Es entsteht eine ternäre Beziehung



Ternäre Beziehungen

- Ternäre Beziehungen können nicht automatisch in binäre Beziehungen aufgebrochen werden



Ternäre Beziehungen (Forts.)

empfiehlt	Professor	Vorlesung	Buch
	Heuer	DB1	1-234
	Heuer	DB2	9-876
	Saake	DB1	9-876
	Saake	DB2	9-876

P-V	Prof	Vorl
	Heuer	DB1
	Heuer	DB2
	Saake	DB1
	Saake	DB2

V-B	Vorl	Buch
	DB1	1-234
	DB2	9-876
	DB1	9-876

P-B	Prof	Buch
	Heuer	1-234
	Heuer	9-876
	Saake	9-876

Hier verlieren wir die Information, daß Heuer das Buch 9-876 nur für die Vorlesung DB2 aber nicht für DB1 empfiehlt.

Könnten auch zusätzliche (fragwürdige) Information abspeichern, die sonst nicht darstellbar wäre, z.B. das Buch 7-000 für DB3, ohne daß ein Professor diese Vorlesung hält oder das Buch empfiehlt

Konzeptueller Entwurf (Zusammenfassung)

- Konzeptueller Entwurf folgt der Anforderungsanalyse
- Ergebnis: eine “high-level“ Beschreibung der zu speichernden Daten
- ER-Modell populär für konzeptuellen Entwurf
- Basis-Konstrukte: Entities, Relationships, Attribute (von Entities und Relationships)
- Zusätzliche Konstrukte: Weak Entities, ISA-Beziehungen, Aggregation
- Constraints (Integritätsbedingungen) im ER:
  - Kardinalitätsrestriktionen (1:1, 1:n, m:n)
  - Komplexität von Beziehungen in min,max-Notation (Participation Constraints)
  - Overlap/Covering-Constraints in ISA-Hierarchien
  - Bestimmen von Constraints wichtig für guten DB-Entwurf
  - Einige Constraints (z.B. funktionale Abhängigkeiten) lassen sich im ER-Modell nicht ausdrücken
- Entwurf ist subjektiv
  - Entity vs. Attribut, Entity vs. Relationship, Binär oder Ternär, mit oder ohne ISA, mit oder ohne Aggregation