Folien zur Vorlesung
Einführung in Datenbanken

Kapitel 5

Datenmodelle

Fachhochschule Wedel

Prof. Dr. Ulrich Hoffmann

basierend auf den Lehrmaterialien von
Prof. Dr. Hans–Detlef Gerhardt

Kapitel 5: Datenmodelle

fhwedel
UNIVERSITY OF APPLIED SCIENCES

5.1 Einführung

5.2 Das Entity-Relationship-Modell

- 5.2.1 Entities
- 5.2.2 Relationships
- 5.2.3 Entity-Relationship-Diagramm (ERD)

5.3 Grundlagen des relationalen Datenmodells

- 5.3.1 Relationen
- 5.3.2 Relationenschemas und Datenabhängigkeiten
- 5.3.3 Relationale Datenbanken
- 5.3.4 Entwurfs-Theorie relationaler DB-Schemas
  - 5.3.4.1 Update-Anomalien
  - 5.3.4.2 Normalformen
  - 5.3.4.3. Zweite Normalform
  - 5.3.4.4. Dritte Normalform

5.1 Einführung

Begriff *Datenmodell*

1. **konkrete Beschreibung** eines gegebenen Realitätsausschnitts — **umgangssprachlich**
2. **Methodik** zur Beschreibung von Datenstrukturen — **formal richtig**

Ein Datenmodell ist ein **Werkzeug** zur **Modellierung** und bietet

- eine Menge von Strukturprimitiven, mit deren Hilfe ein Datenbank-Schema definiert werden kann,
- eine Menge von Operationen, mit denen von DBS auszuwertende sprachliche Ausdrücke gebildet werden.

Verwandtschaft zu **abstrakten Datentypen:** (Datenstruktur, Operation)

Kurz: Datenmodell (DM):

- sprachliches Mittel zur Beschreibung von mehreren anwendungs-abhängigen Strukturen und Operationen.
- Verschiedene Datenmodelle unterscheiden sich entsprechend dervStrukturprimitiven und der Operationen.

Datenmodelle

basieren auf: Typen, Operatoren, Konstruktoren, Konsistenz

verschiedene Arten von Datenmodellen:

- Hierarchisches Modell
- Netzwerkmodell

➡ **Relationales Modell**
➡ **Entity-Relationship-Modell**

- Semantische Datenmodelle
- Non-Standard-Datenmodelle

- **Objektorientierte Datenmodelle**
- **XML-Datenmodelle**

Definition eines Datenmodells
(nach Brodie 1984)

Menge mathematisch wohldefinierter Konzepte, die alle statischen und dynamischen Eigenschaften der Anwendungswelt erfassen sollen, und zwar

- **statische Eigenschaften** wie Objekte, Eigenschaften von Objekten und Beziehungen zwischen Objekten,
- **dynamische Eigenschaften** wie Operationen auf Objekten, Eigenschaften dieser Operationen und Beziehungen zwischen Operationen,
- **Integritätsbedingungen**
  - über Objekten (statische Integritätsbedingungen) und
  - über Operationen (dynamische Integritätsbedingungen).

Zielstellung:
Realisierung einer Anwendung in einem Datenbanksystem

Anforderungsanalyse mit Ergebnis **Anforderungsspezifikation:**
- Beschreibung der bestehenden **Anwendungswelt** (Ist-Analyse)
- Beschreibung der zu realisierende Anforderungen **(Soll-Konzept)** ergänzt durch Graphiken, Ablaufdiagramme, Formulare, evtl. vorgelegt als Pflichtenheft.

Erforderlich ist, dass die **wesentlichen Aspekte formalisiert** werden, damit sie mit der zur Verfügung stehenden **Datenbanksprache implementiert** und benutzt werden können.

Die Formalisierung erfolgt schrittweise und wird immer detaillierter bezüglich des Zielsystems.

Wichtiger Schritte:

- **Konzeptueller Entwurf** unabhängig vom Ziel-DBMS,
  Ausschnitt der realen Welt soll möglichst **ohne Informationsverlust** modelliert werden.

- **Logischer Entwurf** bezogen auf eine Art von Datenbanksystem, z.B. Relationale Datenbanksysteme

Methoden unterscheiden sich in den Elementen, die zur Verfügung stehen, um Syntax und Semantik der Außenwelt zu modellieren.

Konzeptionelle Datenmodellierung für ein betriebliches Informationssystem (1)

Ein Betrieb möchte im Rahmen eines **Informationssystems** wichtige Informationen über alle mit dem Betrieb verbundenen **Personen** speichern.

Weiter sollen Informationen über

- die verfügbaren **Maschinen**,
- die benötigten **Rohstoffe** und
- deren **Lieferanten** zur Verfügung stehen.

Im einzelnen sind folgende Festlegungen getroffen:

PERSON:

- Nummer zur Identifikation (PNR)
- Name (NAME)
- Adresse (ADR)
- Telefonnummer (TEL)
- E-mail-Adresse (E-MAIL)

MASCHINE:

- Maschinennummer (MNR)
- Maschinenname (NAME)
- Anschaffungsdatum (ANSCH_DATUM)
- Neuwert (NEUWERT)
- Zeitwert (ZEITWERT)

ROHSTOFF:

- Rohstoffnummer (RNR)
- Rohstoffname (R_NAME)
- Menge (MENGE)
- Preis (PREIS)

LIEFERANT:

- Lieferantennummer (LNR)
- Firma (FIRMA)
- Adresse (ADR)
- Ansprechpartner (ANSPRECHP)
- Geschlecht (GESCHLECHT)

5.2 Das Entity-Relationship-Modell

5.2.1 Entities

Begriffe: Entity, Entität, Entity–Set, Entitätsmenge, Relationship Beziehung

Grundidee: Reale Welt besteht aus Objekten, welche in Beziehung zueinander stehen können.

Entity: individuelles und identifizierbares **Exemplar** von Dingen, Personen oder Begriffen der realen oder der Vorstellungswelt.

Ein **Entity** kann sein

- ein **Individuum**, z.B. Einwohner, Student, HSL
- ein **reales Objekt**, z.B. eine Maschine, Gebäude, Produkt
- ein **abstraktes Konzept**, z.B. ein Fachgebiet, eine Vorlesung
- ein **Ereignis**, z.B. ein Buchungsvorgang, die Immatrikulation eines Studenten

Entity ist immer eine Einheit,

- die eindeutig identifizierbar ist
- deren Existenz auf einem geeigneten Speichermedium aufgrund eines Identifikationsmerkmales darstellbar sein muss, und zwar unabhängig von der Existenz anderer Entities
- für die Informationen gesammelt und auf einem geeigneten Speichermedium festgehalten werden sollen.

Die **Sachlage** bestimmt, was als Einheit in Frage kommt.

Entity–Set: (vorläufige Definition)

Zusammenfassung von „ähnlichen“, „vergleichbaren“, „zusammengehörigen“ Entities zu einer Menge.

Entities besitzen Eigenschaften.

Eigenschaften werden:

- zur Charakterisierung von Entities genutzt
- haben Namen und Wert

Wertebereich (DOMAIN):

Zusammenfassung aller möglichen bzw.
zu gelassenen Werte für eine Eigenschaft.

Die interessierenden Eigenschaften werden im Modell durch **Attribute** dargestellt.

Ein **Attribut** ordnet jedem Entity eines Sets einen Wert aus einem bestimmten **Wertebereich** zu.

Attributklassen:

- einwertig A
- mehrwertig $\{A\}$
- zusammengesetzt $A(B_1, \ldots, B_k)$

Ordnet man jedem Attribut einer Entity-Set einen konkreten Wert zu,
so erhält man ein **konkretes Entity**.

Definition:
Ein **Entity e** ist **ein** Element des kartesischen Produkts aller Wertebereiche, d.h.

$$e \in dom(A_1) \times \ldots \times dom(A_m).$$

Ein **Entity-Set** E^t ist eine beliebige **Teilmenge** der Menge aller Entities.

Konzeptionelle Datenmodellierung für ein betriebliches Informationssystem (2)

Aus (1) können wir das Entity-Format ableiten:

PERSON_F = {PNR, NAME , ADR, TEL, E-MAIL}

MASCHINE_F = {MNR , NAME , ANSCH_DATUM , NEUWERT, ZEITWERT}

ROHSTOFF_F = {RNR , R_NAME , MENGE , PREIS}

LIEFERANT_F = {LNR, FIRMA , ADR, ANSPRECHP , GESCHLECHT}

Entity–Format: Menge aller Attributnamen eines Entity-Sets.
(zeitinvarianter Aspekt)

Inhalt eines Entity-Sets (d.h. das Set selbst) ist **zeitveränderlich**.

Beispiel:

Buchbestand besteht zu einem Zeitpunkt t_1 aus den zwei Entities:

e_1 = (1, Begg, Datenbanksysteme, Addison-Wesley)

e_2 = (2, Vetter, Aufbau betrieblicher Informationssysteme, Teubner-Verlag),

und zum Zeitpunkt t_2 ist außerdem das Entity

e_3 = (3, Date, Data Base Systems, Addison-Wesley)

enthalten.

Problem: Identifikation einer speziellen Entity.

Es interessiert die **Menge der Attribute**, deren Werte bereits jedes einzelne Entity **eindeutig identifizieren** können.

Wir suchen die Eigenschaften (**Attribute**), die ausreichend sind, ein Entity aus dem Entity-Set **eindeutig zu identifizieren.**

Wir betrachten die Tabelle:

MNR	NAME	ANSCH-DATUM	NEUWERT	ZEITWERT
1	Bohrmaschine	01-feb-99	30.000	15.000
2	Bohrmaschine	01-jul-02	30.000	18.000
3	Fraesmaschine	04-jan-98	40.000	10.000
11	Hobelmaschine	15-jan-02	29.000	19.000
12	Drehbank	01-aug-99	31.000	21.000
14	Hobelmaschine	01-nov-98	32.000	22.000
16	Drehbank	25-nov-01	32.000	23.000
17	Bohrmaschine	01-feb-03	31.000	25.000

MNR hat diese Eigenschaft.

Jede **Menge** mit dieser Eigenschaft heißt **Schlüsselkandidat**.

Hat ein Attribut die Eigenschaft der **eindeutigen Identifizierbarkeit**, und gibt man weitere Attribute hinzu, dann bleibt diese Eigenschaft erhalten.

Alle diese Mengen von Attributen heißen **Schlüsselkandidaten**.

Deshalb sucht man stets die **minimale Menge von Attributen** mit dieser Eigenschaft.

Schlüssel : Schlüsselkandidat mit **Minimalitätseigenschaft**.

Definition:

Ein **Schlüssel K** ist eine **minimale identifizierende** Attributkombination.

Schlüsselattribute dürfen keine Nullwerte annehmen.

Unter allen **Schlüsseln** zeichnet man einen besonders aus, dieser heißt **Primärschlüssel**.

Oft werden **künstliche Primärschlüssel** eingeführt.

Konzeptionelle Datenmodellierung für ein betriebliches Informationssystem (3)

Alle zur Verfügung stehenden Informationen sollen zusammengefasst und formal dargestellt werden:

Das Entity-Format **wird ergänzt durch den gewählten Primärschlüssel:**

Das Ergebnis heißt **Entity–Deklaration**.

Wir können daraus die folgenden Entity-Deklarationen ableiten:

PERSON	=	({PNR, NAME, ADR, TEL, E-MAIL}, {PNR})
MASCHINE	=	({MNR, NAME, ANSCH_DATUM, NEUWERT, ZEITWERT}, {MNR})
ROHSTOFF	=	({RNR , R_NAME , MENGE , PREIS}, {RNR})
LIEFERANT	=	({LNR, FIRMA, ADR, ANSPRECHP, GESCHLECHT}, {LNR})

Definition :

Ein Entity-Set E^t

$$E^t \subseteq dom(A_1) \times \ldots \times dom(A_m)$$

ist eine beliebige Teilmenge der Menge aller Entities, welche (zumindest) den Primärschlüssel erfüllt.

Definition:

Eine **Entity-Deklaration** (kurz: Entity-Typ) hat die Form

E = (attr(E), K);

sie besteht aus einem **Namen** (E),

einem **Format** (attr(E)),

einem **Primärschlüssel** (K),
welcher aus einwertigen Elementen von attr(E) zusammengesetzt ist.

attr(E) ist darstellbar als Menge oder Folge.

fhwedel
UNIVERSITY OF APPLIED SCIENCES

5.2.2 Relationships

- zur Darstellung von Beziehungen zwischen Entity–Sets.

Auch Beziehungen können eigene Attribute besitzen.

Man unterscheidet:

- zeitinvariante Beschreibung der Beziehung
- zeitveränderlicher Inhalt

Definition:

Eine **Relationship–Deklaration** hat die Form

R = (ent(R), attr(R)).

R **Name** der Deklaration
ent(R) Folge der Namen der Entity-Deklarationen, zwischen denen eine Beziehung definiert werden soll.

attr(R) **Menge** oder **Folge** von Attributen der Beziehung

Konzeptionelle Datenmodellierung für ein betriebliches Informationssystem (4)

Es gibt Personen, von denen man mehr Informationen abspeichern möchte. Das sind die Mitarbeiter des Betriebes.

Wir führen dazu formal eine neue Entity-Deklaration **MITARBEITER** ein.

Zwischen den festgelegten **Entity-Deklarationen** bestehen
Beziehungen, die für das Informationssystem wichtig sind.

Informal können wir feststellen, dass

- Mitarbeiter Maschinen bedienen,
- Maschinen Rohstoffe benötigen,
- Lieferanten Rohstoffe liefern**.**

Maschinen benötigen Rohstoffe in einer bestimmten Mindestmenge **M_MENGE**.

Lieferanten liefern Rohstoffe in einer bestimmten Menge **L_MENGE**.

Daraus ergeben sich **Relationship–Deklarationen:**

BEDIENT = ((MITARBEITER, MASCHINE), ∅)

BENÖTIGT = ((MASCHINE, ROHSTOFF), M_MENGE)

LIEFERT = ((LIEFERANT, ROHSTOFF), L_MENGE)

Sei **ent(R) = (E_1, . . . , E_k)** , und für beliebiges, aber festes t sei

E_i^t der Inhalt der Entity-Deklaration E_i , i = 1(1) k.

Sei **attr(R) = (B_1, . . . , B_n).**

Ein **Relationship r** ist ein Element des kartesischen Produktes aus allen E_i^t und den Wertebereichen der B_j, d.h.

$$r \in E_1^t \times \ldots \times E_k^t \times dom(B_1) \times \ldots \times dom(B_n),$$

$$r = (e_1, \ldots, e_k, b_1, \ldots, b_n)$$

mit $e_i \in E_i^t$ **für i = 1(1)k** und $b_j \in dom(B_j)$ **für j = 1(1)n.**

Ein **Relationship-Set R^t** ist eine **Menge von Relationships**, d.h.

$$R^t \subseteq E_1^t \times \ldots \times E_k^t \times dom(B_1) \times \ldots \times dom(B_n).$$

Definition:

Sei **R = (ent(R), attr(R))** eine Relationship-Deklaration mit
ent(R) = (E_1, . . . , E_k).

Dann heißt **k** die **Stelligkeit** oder der Grad von R, kurz **grad(R).**

Konzeptionelle Datenmodellierung für ein betriebliches Informationssystem (5)

Relationship-Deklarationen:

BEDIENT = ((MITARBEITER, MASCHINE), ∅)
BENÖTIGT = ((MASCHINE, ROHSTOFF), M_MENGE)
LIEFERT = ((LIEFERANT, ROHSTOFF), L_MENGE)

Diese Deklarationen haben die **Stelligkeit** k = 2.

BEDIENT ist eine **1:n-Beziehung**,
jeder Mitarbeiter darf eine oder mehrere Maschinen bedienen, aber jede Maschine darf nur von einem Mitarbeiter bedient werden.

BENÖTIGT ist eine **n:m-Beziehung**,
jede Maschine benötigt mehrere Rohstoffe, jeder Rohstoff wird von mehreren Maschinen benötigt.

LIEFERT ist eine **1:n-Beziehung**,
jeder Lieferant liefert ein oder mehrere Rohstoffe, aber für jeden Rohstoff ist genau ein Lieferant festgelegt.

Für **k = 2** wird definiert:

Komplexität:

Die **Komplexität** sagt aus, mit wie vielen Entities des zweiten Typs ein bestimmtes Entity des ersten Typs in einer konkreten Beziehung stehen kann, darf oder sogar muss.

Für zweistellige Beziehungen wird festgelegt:

Eine **Relationship R** kann von den folgenden Typen sein:

1 : 1 hierarchisch one to one
1 : 1 one to one
1 : n hierarchisch one to many
1 : n one to many
m : n many to many

Es bestehen folgende **Beziehungen**:

Mitarbeiter	**bedient**	**Maschine.**
Maschine	**benötigt**	**Rohstoffe.**
Lieferant	**liefert**	**Rohstoffe.**

Daraus ergeben sich **Relationship-Deklarationen:**

BEDIENT = ((MITARBEITER, MASCHINE), ∅)
BENÖTIGT = ((MASCHINE, ROHSTOFF), M_MENGE)
LIEFERT = ((LIEFERANT, ROHSTOFF), L_MENGE)

(M_MENGE Mindestmenge, L_MENGE Liefermenge)

Jeder kann eine oder auch mehrere Maschinen bedienen, jeder Maschine ist genau ein Mitarbeiter für die Bedienung zugeordnet.
Beziehung: **1:m Beziehung** zwischen MITARBEITER und MASCHINE.
Primärschlüssel: Primärschlüssel von MASCHINE (**MNR**)

Jede Maschine benötigt verschiedene Rohstoffe, jeder Rohstoff kann auf verschiedenen Maschinen verarbeitet werden.
Beziehung: m:n Beziehung zwischen ROHSTOFF und MASCHINE.
Primärschlüssel: setzt sich aus den Primärschlüsseln von ROHSTOFF und MASCHINE zusammen: **(MNR, RNR)**

Jeder Lieferant liefert einen oder auch mehrere Rohstoffe. Für jeden Rohstoff gibt es nur einen Lieferanten.
Beziehung: 1:n Beziehung zwischen LIEFERANT und ROHSTOFF.
Primärschlüssel: Primärschlüssel von ROHSTOFF (**RNR**).

Hierarchische Beziehungen: IS-A (Generalisierung/Spezialisierung)

Konzeptionelle Datenmodellierung für ein betriebliches Informationssystem (6)

Informationssystem mit allgemeinen Daten von Personen
(Mitarbeiter, Zeitarbeitskräfte, ehemalige Mitarbeiter, Bewerber, Manager, Kundenberater etc.)

PERSON = ((PNR, NAME, ADR(PLZ, STADT, STRASSE), TEL, E-MAIL), (PNR)).

MITARBEITER:
Sind PERSON +

- Gehaltsstufe (GEH_STUFE)
- Gehalt (BETRAG)
- Abteilung (ABT_NR, ABT_NAME)
- Krankenkasse (KRANKENKASSE)
- Prämienbetrag (P_BETRAG)
- Kinder (K_NAME, K_VORNAME, K_GEB)

ZEITAK:
Sind PERSON +

- Leihfirma (L_FIRMA)
- Stundenlohn (H_LOHN)
- Beginn und Ende Einsatz (VON_BIS)
- Gesamtstunden (H_SUMME)

Also:

MITARBEITER_V = ({PNR, NAME, ADR(PLZ, STADT, STRASSE), TEL, E-MAIL, GEH_STUFE, BETRAG, ABT_NR, ABT_NAME, KRANKENKASSE, {P_BETRAG}, {KINDER(K_NAME, K_VORNAME, K_GEB)}}, {PNR})

ZEITAK_V = ({PNR, NAME, ADR(PLZ, STADT, STRASSE), TEL, E-MAIL, L_FIRMA, H_LOHN, {VON_BIS}, H_SUMME}, {PNR}).

Es gilt:

Alle Attribute von **PERSON** kommen in **MITARBEITER** bzw. **ZEITAK** vor.

Zu jedem Mitarbeiter mit einer vorgegebenem PNR existiert eine Person mit der gleichen PNR.

Zu jeder Zeitarbeitskraft mit einer vorgegebenem PNR existiert eine Person mit der gleichen PNR.

Für gemeinsame Attribute werden stets **gleiche** Werte angenommen.

Beispiel:

Relation PERSON:

PNR	NAME	VORNAME	...		
167	Krause	Gustav			
168	Hahn	Egon			
123	Lehmann	Karl			
133	Schulz	Harry			
124	Meier	Richard			
125	Wutschke	Oskar			
126	Schroeder	Karl-Heinz			
227	Wagner	Walter			
234	Krohn	August			
135	Tietze	Lutz			
156	Hartmann	Juergen			
127	Ehlert	Siegfried			

Relation MITARBEITER:

PNR	...	GEH_STUFE	ABT_NR	ABT_NAME	...
123		it3	d13		
133		it1	d13		
124		it5	d13		
125		it3	d13		
126		it4	d13		
135		it2	d13		
127		it1	d15		

Relation ZEITAK:

PNR	L_FIRMA	H_Lohn	...	
227	Quick-Personal	20,00		
234	Zeit-Renner	18,00		

Definition:

Sind $E_1 = (attr(E_1), K_1)$ und $E_2 = (attr(E_2), K_2)$
zwei Entity-Deklarationen,
so besteht zwischen diesen eine
IS-A-Beziehung der Form

E_1 IS-A E_2 ,

falls gilt:

1. Alle Elemente von $attr(E_2)$ kommen in $attr(E_1)$ vor,

2. Zu jedem Zeitpunkt t gilt:

   Ist $e_1 \in E_1^t$, so existiert ein $e_2 \in E_2^t$ mit

   $e_1[A] = e_2[A]$ für jedes gemeinsame Attribut A.

Man schreibt kurz **$E_1 \subseteq E_2$** .

Vereinfachung durch Aufbau einer **hierarchischen IS-A-Beziehung**:

Gemeinsame Attribute schreibt man in die allgemeinen Entity-Deklaration:

Also:

PERSON = ({PNR, NAME , ADR(PLZ, STADT, STRASSE), TEL, E-MAIL}, {PNR})

MITARBEITER = ({PNR, GEH_STUFE, BETRAG, ABT_NR, ABT_NAME, KRANKENKASSE, {P_BETRAG}, {KINDER(K_NAME, K_VORNAME, K_GEB)}}, {PNR})

ZEITAK = ({PNR, L_FIRMA, H_LOHN, { VON_BIS }, H_SUMME}, {PNR}).

Die Verbindung erfolgt über den **gemeinsamen Primärschlüssel** PNR.

Es gilt:

MITARBEITER $\subseteq$ PERSON und ZEITAK $\subseteq$ PERSON.

MITARBEITER IS-A PERSON und
ZEITAK IS-A PERSON.

Konzeptionelle Datenmodellierung für ein betriebliches Informationssystem (7)

Informationssystem mit allgemeinen Daten von Personen
(Mitarbeiter, Zeitarbeitskräfte, ehemalige Mitarbeiter, Bewerber, Manager, Kundenberater etc.)

PERSON = ((PNR, NAME, ADR(PLZ, STADT, STRASSE), TEL, E-MAIL), (PNR)).

MITARBEITER:
Sind PERSON +

- Gehaltsstufe (GEH_STUFE)
- Gehalt (BETRAG)
- Abteilung (ABT_NR, ABT_NAME)
- Krankenkasse (KRANKENKASSE)
- Prämienbetrag (P_BETRAG)
- Kinder (K_NAME, K_VORNAME, K_GEB)

ZEITAK:
Sind PERSON +

- Leihfirma (L_FIRMA)
- Stundenlohn (H_LOHN)
- Beginn und Ende Einsatz (VON_BIS)
- Gesamtstunden (H_SUMME)

MANAGER:
Sind MITARBEITER +

- Akademischer Grad (A_GRAD)
- Studienrichtung (STUD_RI)

KB (Kundenberater):
Sind MITARBEITER +

- Kundenname (KD_NAME)

Falls die Attribute von Manager bzw. KB vollständig angegeben werden sollen, so wird man in die Attributfolge der entsprechenden Deklaration formal alle Attribute der Verallgemeinerung aufnehmen, also

MANAGER_V = ({PNR, GEH_STUFE, BETRAG, ABT_NR, ABT_NAME, KRANKENKASSE, {P_BETRAG }, {KINDER(K_NAME, K_VORNAME, K_GEB)}, A_GRAD, STUD_RI}, {PNR})

KB_V = ({PNR, GEH_STUFE, BETRAG, ABT_NR, ABT_NAME, KRANKENKASSE, {P_BETRAG}, {KINDER(K_NAME, K_VORNAME, K_GEB)}, KD_NAME}, {PNR})

MANAGER und KUNDENBERATER sind Spezialisierungen von **MITARBEITER**.

⇒ separate Entity-Deklarationen:

MANAGER = ({PNR, A_GRAD, STUD_RI}, {PNR})

KB = ({PNR, {KD_NAME}}, {PNR})

Eigenschaften von IS-A-Beziehungen:

total (t)	**-**	**partiell (p)**
disjunkt (d)	**-**	**nicht disjunkt (n)**

Hier: **p n :**
Es gibt Personen, die weder Mitarbeiter noch Zeitarbeitskräfte sind, es gibt Personen, die Zeitarbeitskräfte waren und jetzt Mitarbeiter sind.

Also

MITARBEITER ∪ ZEITAK ⊆ PERSON

total wäre: MITARBEITER ∪ ZEITAK = PERSON

MITARBEITER ∩ ZEITAK =/= ∅ → **nicht disjunkt**.

Es gilt: MANAGER ⊆ MITARBEITER und KB ⊆ MITARBEITER

Hier: **p d** :
Es gibt Mitarbeiter, die weder Manager noch Kundenberater sind, ein Mitarbeiter ist entweder Manager oder Kundenberater, aber nicht beides.

Also

MANAGER ∪ KB ⊆ MITARBEITER

MANAGER ∩ KB = ∅ → **disjunkt.**

Alle Beziehungen: **hierarchische 1:1 (IS-A) Beziehungen**.

Gewählt ist als Primärschlüssel: **PNR**

Mit dem **Datenmodell** steht ein erster Formalismus zur Verfügung, um eine konkrete Anwendungssituation beschreiben zu können.

Es sind dabei alle **zeitinvarianten Deklarationen** aufzustellen:

- **Entity-Deklarationen** jeweils mit
  - Namen
  - Attributen und deren Wertebereichen, möglicherweise mit Nullwerten
  - Primärschlüssel
- **Relationsship-Deklarationen** jeweils mit
  - Namen
  - beteiligten Entity-Deklarationen
  - gegebenenfalls eigenen Attributen und deren Wertebereiche
  - Primärschlüssel
  - Komplexität/Typ der Beziehung
  - gegebenenfalls IS-A-Beziehungen
  - gegebenenfalls Hierarchiefestlegungen.

Vereinbarung:
An Stelle von „Entwickeln Sie ein mathematisches Modell als ein Entity-Relationship-Modell“ wollen wir dafür kurz sagen:

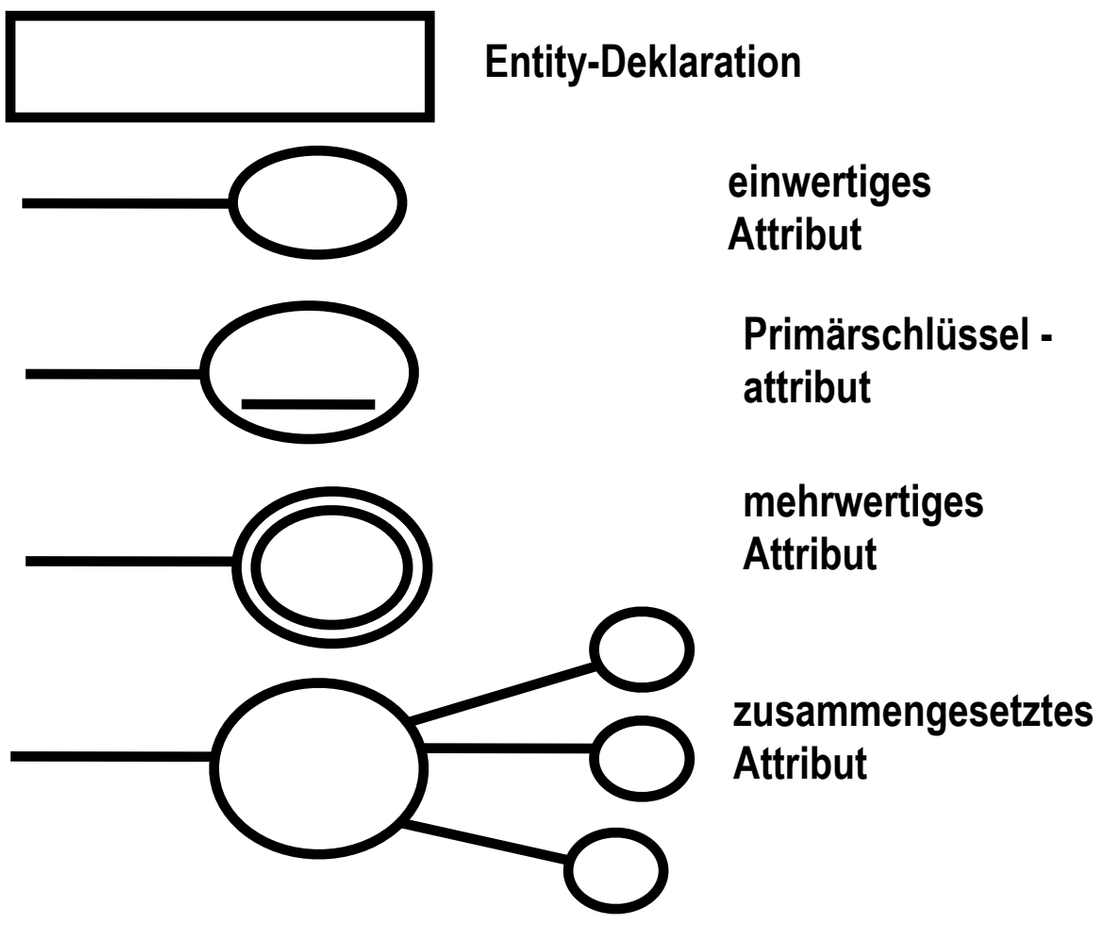
„Entwickeln Sie ein Entity-Relationship-Modell“

5.2.3 Entity-Relationship-Diagramm (ERD)

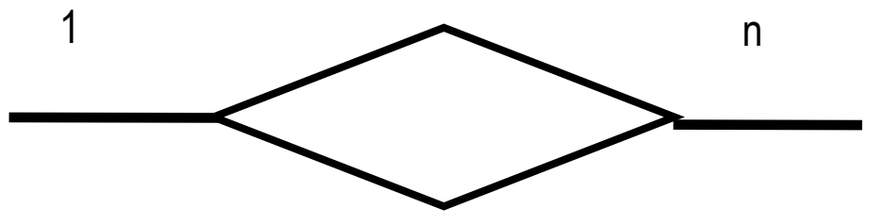
- Zur graphischen Veranschaulichung

Bausteine:

1. **Rechtecke** für Entity-Deklarationen, enthalten Namen der Deklaration.
2. **Kreise** für die Attribute, enthalten Attributnamen.
3. Primärschlüsselattribute werden unterstrichen.
4. Kreise durch **ungerichtete Kanten** mit Rechteck verbunden.



5. Rauten für Relationship-Deklarationen, enthalten Namen der Deklaration.



Relationship-Deklaration

6. Kreise für die Attribute, enthalten Attributnamen.

7. Kreise durch **ungerichtete Kanten** mit Raute verbunden.

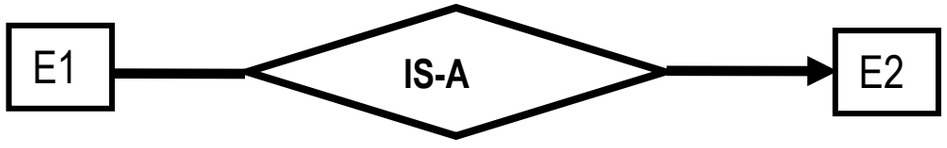
8. Raute durch **ungerichtete Kante** mit beteiligten Entity-Deklarationen verbunden, Typ der Beziehung wird an Kanten vermerkt.

9. Bei hierarchischen Beziehungen Raute durch gerichtete Kante mit der hierarchisch abhängigen Entity-Deklaration verbunden

Beispiel: Ist E_2 von E_1 hierarchisch abhängig, so wird Kante auf E_2 gerichtet

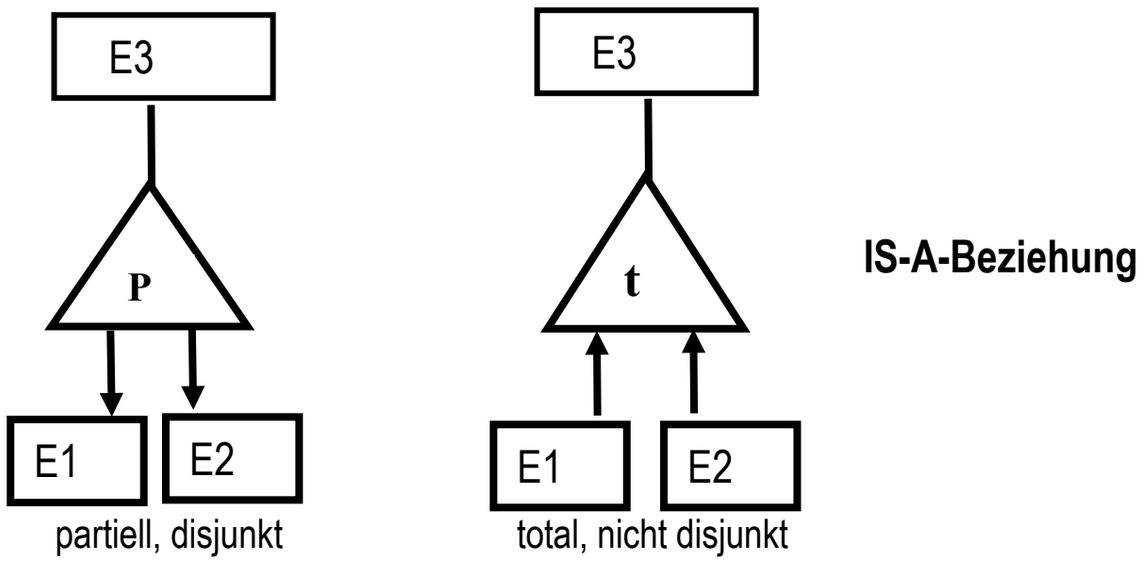
10. Falls die Anordnung der Entity-Deklaration zum Ausdruck gebracht werden soll, so können die Kanten entsprechend (in einem kleinen Kreis) nummeriert werden.

11. Eine IS-A-Beziehung der Form $E_1 \subseteq E_2$ wird **durch eine Raute mit der Beschriftung IS-A** dargestellt. Die Kante zu E_1 ist ungerichtet, die Kante zu E_2 ist **auf E_2** gerichtet.
Die Spezialisierung E_1 enthält als Attribute nur ihre speziellen Attribute sowie die ihres Schlüssels.



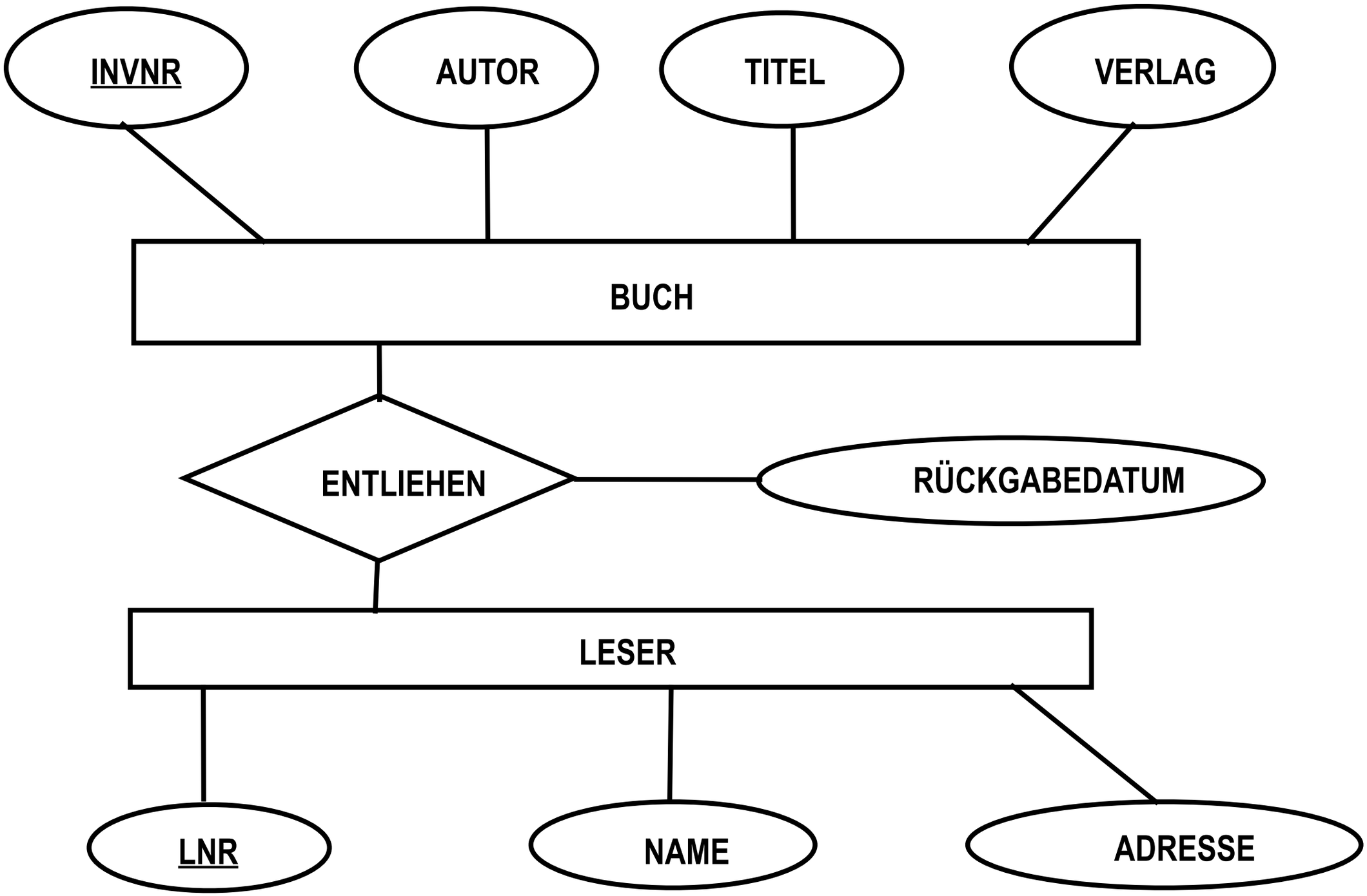
IS-A-Beziehung

12. Eine IS-A-Beziehung der Form $E_1 \subseteq E_3$ und $E_2 \subseteq E_3$ wird durch eine **Dreieck** mit der Beschriftung **p** für partiell und **t** für total dargestellt.
Ist die Beziehung disjunkt, werden die gerichteten Kanten auf E_1 und E_2 gerichtet.
Ist die Beziehung **nicht disjunkt,** werden die gerichteten Kanten auf das **Dreieck** gerichtet.

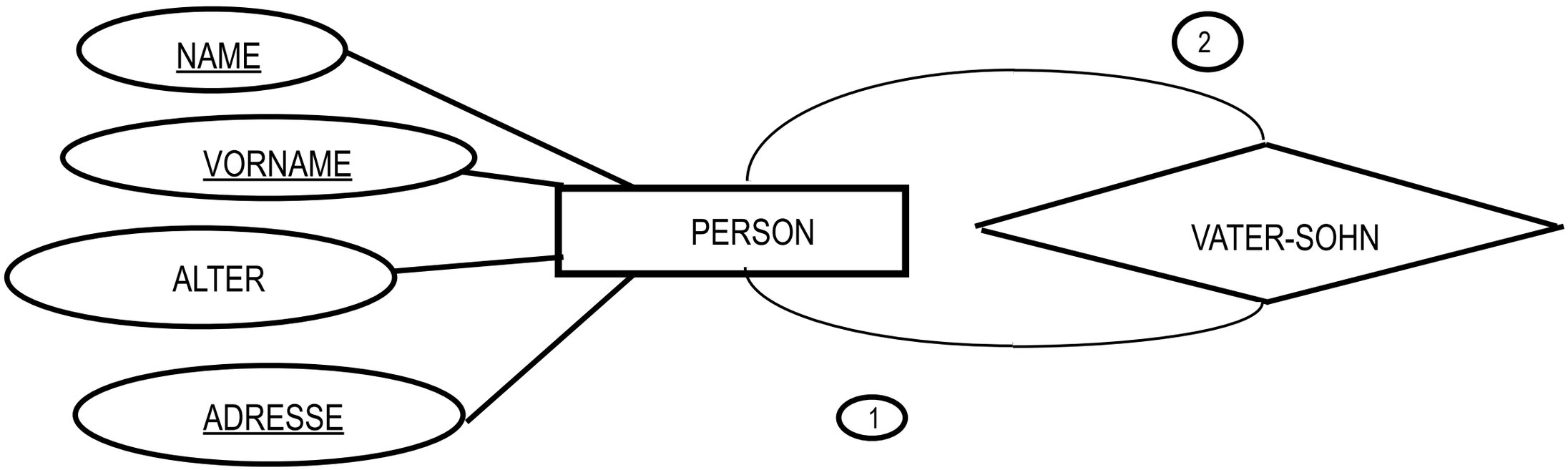


IS-A-Beziehung

Beispiel: ERD für Bibliothek



ERD für rekursive und hierarchische Beziehung: „VATER-SOHN“



E2 ist von E1 hierarchisch abhängig

Anordnung der Entity-Deklarationen in der Relationship–Deklaration

Konzeptionelle Datenmodellierung für ein betriebliches Informationssystem (8)

Ein Betrieb möchte im Rahmen eines Informationssystems wichtige Informationen über alle mit dem Betrieb verbundenen Personen speichern. Weiter sollen Informationen über die verfügbaren Maschinen, die benötigten Rohstoffe und deren Lieferanten zur Verfügung stehen.

Im einzelnen sind folgende Festlegungen getroffen:

PERSON:

- Nummer zur Identifikation (PNR)
- Name (NAME)
- Adresse (ADR)
- Telefonnummer (TEL)
- E-mail-Adresse (E-MAIL)

MITARBEITER:

Sind PERSON +

- Gehaltsstufe (GEH_STUFE)
- Gehalt (BETRAG)
- Abteilung (ABT_NR, ABT_NAME)
- Krankenkasse (KRANKENKASSE)
- Prämienbetrag (P_BETRAG)
- Kinder KINDER(K_NAME, K_VORNAME, K_GEB)

ZEITAK:
Sind PERSON +

- Leihfirma (L_FIRMA)
- Stundenlohn (H_LOHN)
- Beginn und Ende Einsatz (VON_BIS)
- Gesamtstunden (H_SUMME)

MANAGER:
Sind MITARBEITER +

- Akademischer Grad (A_GRAD)
- Studienrichtung (STUD_RI)

KB:
Sind MITARBEITER +

- Kundenname (KD_NAME)

MASCHINE:

- Maschinennummer (MNR)
- Maschinenname (NAME)
- Anschaffungsdatum (ANSCH_DATUM)
- Neuwert (NEUWERT)
- Zeitwert (ZEITWERT)

ROHSTOFF:

- Rohstoffnummer (RNR)
- Rohstoffname (R_NAME)
- Menge (MENGE)
- Preis (PREIS)

LIEFERANT:

- Lieferantennummer (LNR)
- Firma (FIRMA)
- Adresse (ADR)
- Ansprechpartner (ANSPRECHP)
- Geschlecht (GESCHLECHT)

PERSON = ({PNR, NAME , ADR (PLZ, STADT, STRASSE), TEL, E-MAIL}, {PNR})

MITARBEITER = ({PNR, GEH_STUFE, BETRAG, ABT_NR, ABT_NAME, KRANKENKASSE, {P_BETRAG}, KINDER (K_NAME, K_VORNAME, _GEB)}}, {PNR})

ZEITAK = ({PNR, L_FIRMA, H_LOHN, {VON_BIS}, H_SUMME}, {PNR})

MANAGER = ({PNR, A_GRAD, STUD_RI}, {PNR})

KB = ({PNR, KD_NAME}, {PNR})

MASCHINE = ({MNR, NAME, ANSCH_DATUM, NEUWERT, ZEITWERT}, {MNR})

ROHSTOFF = ({RNR, R_NAME, MENGE, PREIS}, {RNR})

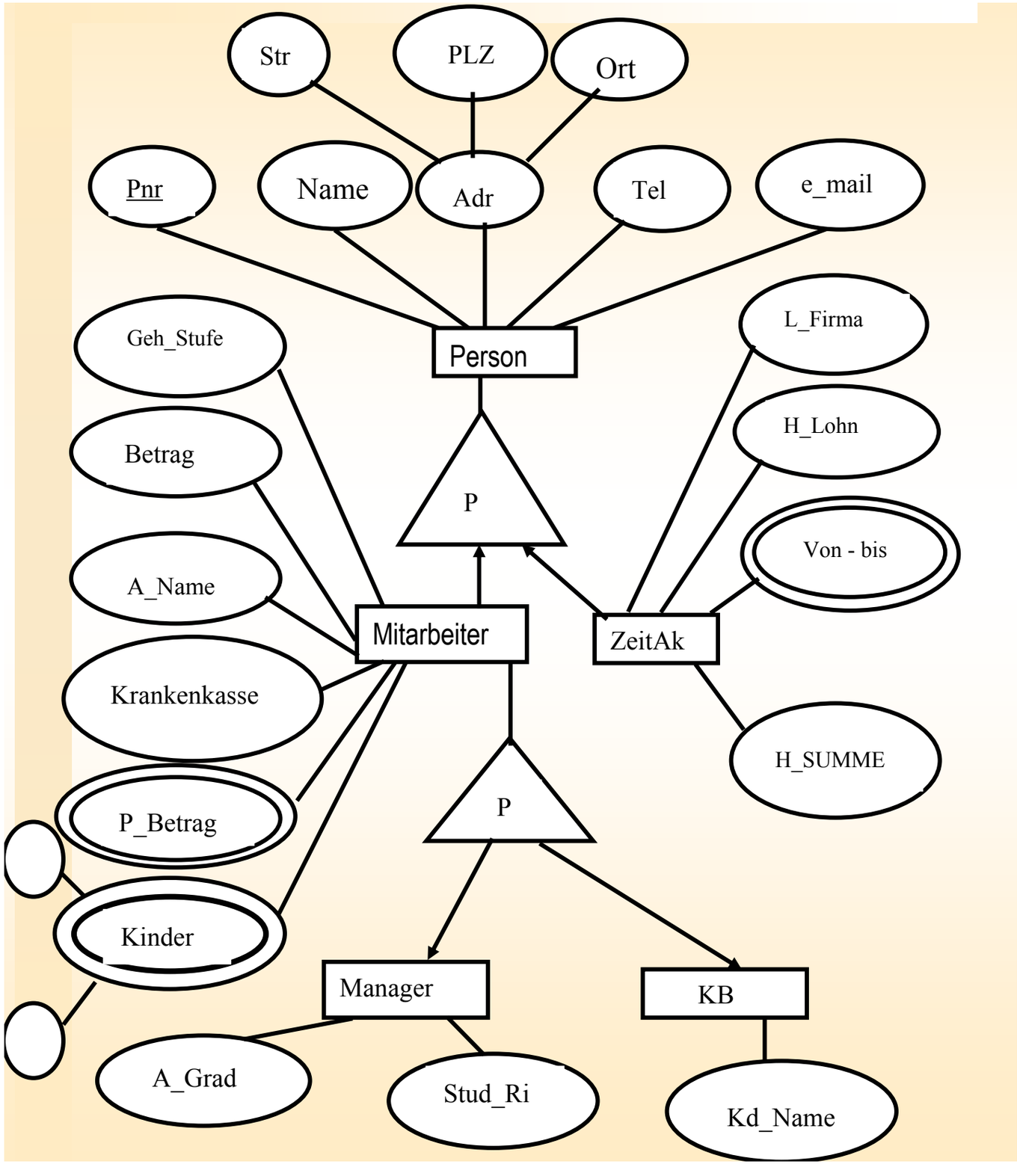
LIEFERANT = ({LNR, FIRMA , ADR(PLZ, STADT, STRASSE), ANSPRECHP ,GESCHLECHT }, {LNR})

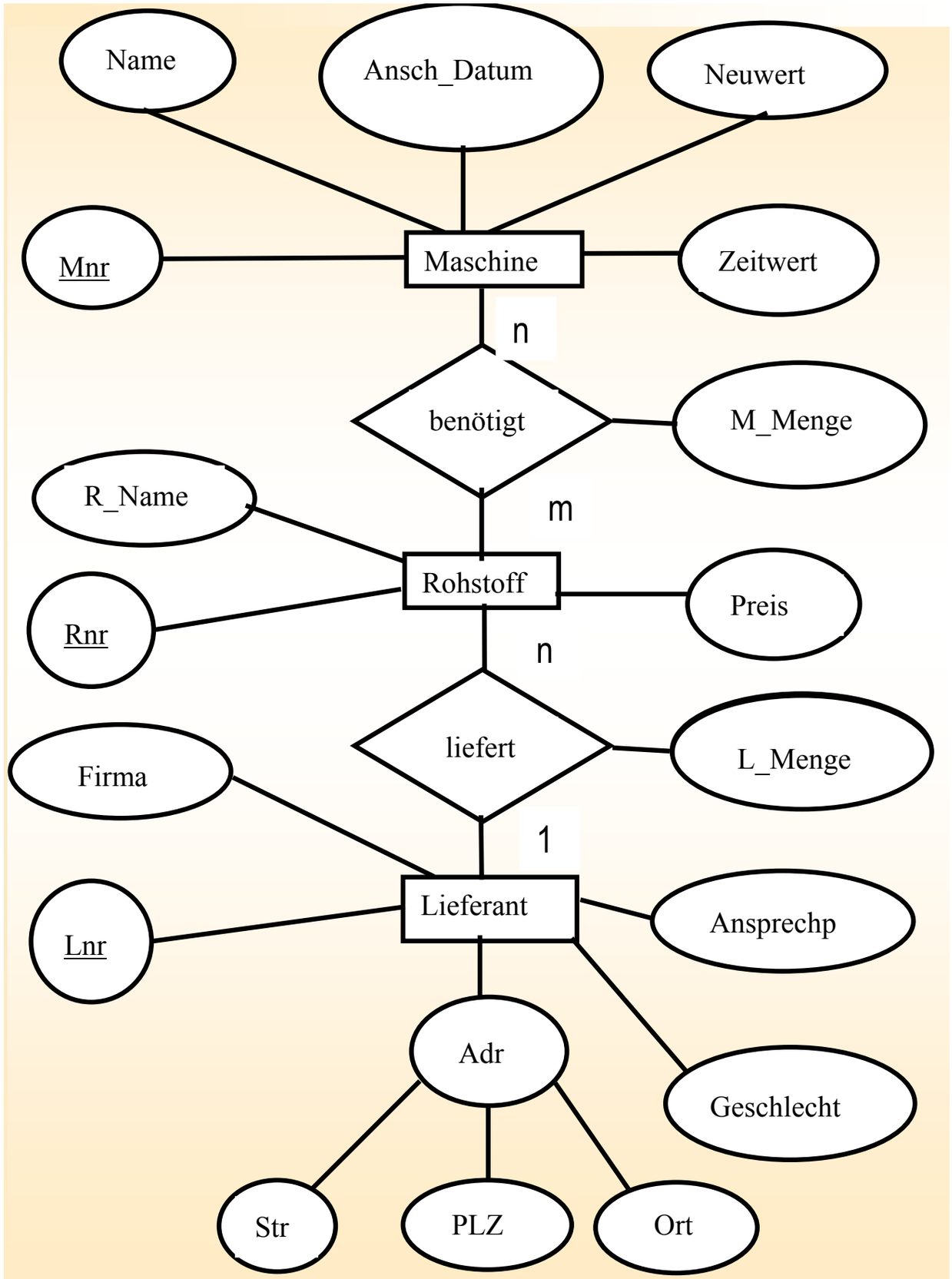
BEDIENT = ((MITARBEITER, MASCHINE), ∅)
PS: MNR Typ: 1:m

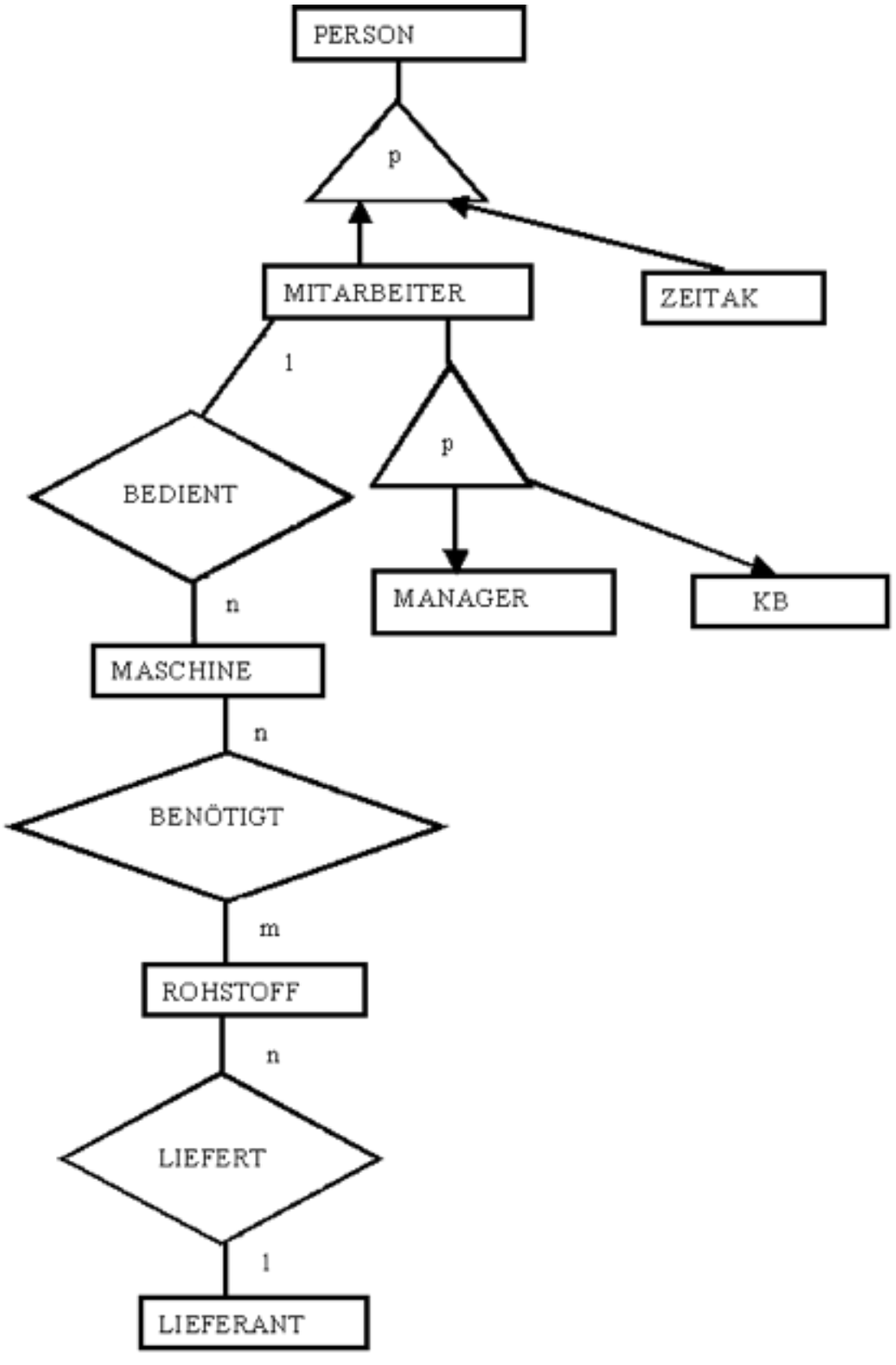
BENÖTIGT = ((MASCHINE, ROHSTOFF), M_MENGE)
PS: (MNR,RNR) Typ: m:n

LIEFERT = ((LIEFERANT, ROHSTOFF), L_MENGE)
PS: RNR Typ: 1:n

MITARBEITER	IS-A	PERSON	}	p, nd
ZEITAK	IS-A	PERSON	}	
MANAGER	IS-A	MITARBEITER	}	p, d
KB	IS-A	MITARBEITER		


Str
PLZ
Ort
Pnr
Name
Adr
Tel
e_mail
Person
Geh_Stufe
Betrag
L_Firma
H_Lohn
P
Von - bis
A_Name
Mitarbeiter
ZeitAk
Krankenkasse
H_SUMME
P_Betrag
P
Kinder
Manager
KB
A_Grad
Stud_Ri
Kd_Name


Name
Ansch_Datum
Neuwert
Mnr
Maschine
Zeitwert
n
benötigt
M_Menge
R_Name
m
Rohstoff
Preis
Rnr
n
liefert
L_Menge
Firma
1
Lieferant
Ansprechp
Lnr
Adr
Geschlecht
Str
PLZ
Ort


PERSON
p
MITARBEITER
ZEITAK
1
BEDIENT
p
n
MANAGER
KB
MASCHINE
n
BENÖTIGT
m
ROHSTOFF
n
LIEFERT
1
LIEFERANT

5.3 Grundlagen des relationalen Datenmodells

5.3.1 Relationen

Beispiel:
„Datenmodell“ Karteikarte (BUCHBESTAND)

INVNR	....	AUTOR	TITEL	VERLAG	...
2		Meier	Inf2	Springer	
3		Kron	DBS	IWT	
4		Liskin	ORACLE	Hill	

X = $\{A_1, ..., A_m\}$

dom(X) = $\bigcup_{A \in X} \mathrm{dom}(A)$

Tup (X) Menge aller Tupel über X

Relation r über X: endliche Menge von Tupeln über X, d.h. **r** $\subseteq$ **Tup(X)**

Rel(X) Menge aller Relationen über X

Relation über X: in Form einer Tabelle darstellbar, sofern eine Reihenfolge der Attribut von X festgelegt ist.

Definition: Ein R-Schema hat die Form

R = (X, K), es umfasst

R - Name des R-Schema
X - Relationenformat (eine Attributmenge)
K - Primärschlüssel.

5.3.3 Relationale Datenbanken

Gegeben: Entity-Relationship-Diagramm

Gesucht: Relationen-Modell

Gegeben: Entity-Deklaration **E = (attr(E), K)**

Daraus ist das R-Schema

R = (X, K) abzuleiten.

R Name der Entity-Deklaration E wird übernommen.

Sei attr(E) = $\{A_1, ..., A_m\}$, daraus wird abgeleitet

X = $(A_1, ..., A_m)$

K wird als **Primärschlüssel** übernommen.

Gegeben: Relationship-Deklaration

R = (ent(R), attr(R)),

wobei

ent(R) = (E_1 ..., E_k) und

attr(R) = (B_1 ..., B_n).

K_i sei **Primärschlüssel** von **E_i,** i = 1(1)k,

es gelte **$K_i = (A_{i\,1}, ..., A_{i\,n_i})$**.

Daraus wird das **R-Schema R** abgeleitet:

R = (X, K) mit

R Name der Relationship-Deklaration (wird übernommen)

X = $\{A_{11}, ..., A_{1n_1}, ..., A_{k1}, ..., A_{kn_k}, B_1 ..., B_n\}$

(die Namen aller in X vorkommenden Attribute seien verschieden)

Wie wird der Primärschlüssel für R ermittelt?

Beispiel:

Gegeben ERM der Bibliothek

⇒ für die Entity-Deklarationen BUCH und LESER

die **R-Schemas**

BUCH = ({INVNR, AUTOR, TITEL, VERLAG}, {INVNR})
LESER = ({LNR, NAME, ADRESSE}, {LNR})

⇒ für die Relationship-Deklaration ENTLIEHEN

das **R-Schema**

ENTLIEHEN = ({INVNR, LNR, RÜCKGABEDATUM}, {INVNR}).

Ist R eine **1:n-Beziehung** zwischen E_1 und E_2, dann ist K_2 ein Primärschlüssel für das R-Schema R.

Ist R eine **n:m-Beziehung** zwischen E_1 und E_2, dann kann $K_1 \cup K_2$ als Primärschlüssel gewählt werden.

Ist R eine **1:1 Beziehung** zwischen E_1 und E_2, dann kann K_1 oder K_2 als Primärschlüssel für R gewählt werden.

Ergebnis:

Menge $R = \{R_1, ..., R_k\}$ von Relationenschemas

Ergebnis:

Menge $R = \{R_1, ..., R_k\}$ von Relationenschemas

ER-Modell	**Relationen - Modell**
Entity - Deklaration Relationship - Deklaration	R - schema
Entity-, Relationship-Set	Relation (Tabelle)
Entity, Relationship	Tupel (Zeile)

Beispiel einer Datenbank d mit Datenbankformat

R = {BUCH, LESER , ENTLIEHEN}

buch:

INVNR	AUTOR	TITEL	VERLAG
1	Meier	Inf1	Springer
2	Meier	Inf2	Springer
3	Kron	DBS	IWT
4	Liskin	ORACLE	Hill

leser:

LNR	NAME	ADRESSE
500	Müller	Hamburg
550	Meier	Berlin
600	Schulz	Wedel

entliehen:

INVNR	LNR	RÜCKGABEDATUM
1	550	301110
2	550	301110
3	600	031210

fhwedel
UNIVERSITY OF APPLIED SCIENCES

Konzeptionelle Datenmodellierung für ein betriebliches Informationssystem (9)

Übertragung des ER-Modells in das relationale Datenmodell: Ableitung von R-Schemas:

PERSON = ({PNR, NAME , ADR(PLZ,STADT, STRASSE), TEL, E-MAIL}, {PNR})

MITARBEITER = ({PNR, GEH_STUFE, BETRAG, ABT_NR, ABT_NAME, KRANKEN-KASSE, {P_BETRAG}, {KINDER (K_NAME, K_VORNAME, _GEB)}}, {PNR})

ZEITAK = ({PNR, L_FIRMA, H_LOHN, {VON_BIS}, H_SUMME}, {PNR})

MANAGER = ({PNR, A_GRAD, STUD_RI}, {PNR})

KB = ({PNR, KD_NAME}, {PNR})

MASCHINE = ({MNR, NAME, ANSCH_DATUM, NEUWERT, ZEITWERT}, {MNR})

ROHSTOFF = ({RNR, R_NAME, MENGE, PREIS}, {RNR})

LIEFERANT = ({LNR, FIRMA, ADR(PLZ, STADT, STRASSE), ANSPRECHP, GESCHLECHT }, {LNR})

BEDIENT = ({MNR, PNR}, {MNR})

BENÖTIGT = ({MNR, RNR, M_MENGE}, {MNR, RNR})

LIEFERT = ({RNR, LNR, L_MENGE}, {RNR})

Modellierung von **Existenzabhängigkeiten** in RDM:

Inklusionsabhängigkeiten

Beispiel: Sei buch, leser wie eben und

entliehen':

INVNR	LNR	RDAT
1	550	301210
6	550	061210

$\Rightarrow$ Buch entliehen, was in Relation b nicht vorkommt

$\Rightarrow$ Widerspruch zur Semantik dieser Beziehung im ERM.

d. h. zu jedem $\mu \in$ elhn muss ein $v \in$ buch existieren mit **μ[INVNR] = v[INVNR],**

oder allgemein

zu jedem $\mu \in r$ muß ein $v \in r_i$ existieren mit $\mu[K_i] = v[K_i]$.

Definition: Sei R ein Datenbankformat,

$R_i, R_j \in R, R_i \neq R_j$, und
$R_i = (X_i, \sum_i)$
$R_j = (X_j, \sum_j)$
V sei Folge von n verschiedenen Attributen aus $\{X_i \cap X_j\}$, dabei sei $|\{X_i \cap X_j\}| \geq 1$.

Gilt $\{\mu[V]:\mu \in ri\} \subseteq \{\mu[V]:\mu \in rj\}$, so sagt man, zwischen Ri und Rj besteht bezüglich V bestehe eine ***Inklusionsabhängigkeit*** (Inclusion Dependency-ID)

Man schreibt dafür: $Ri\,[V] \subseteq Rj\,[V]$.

IS-A-Beziehung führt auf ID,
aus Beziehung KB **IS-A** MITARBEITER wird
im RDM: KB[PNR] $\subseteq$ MITARBEITER[PNR]

Existenzabhängigkeit im Beispiel Bibliothek:
ENTLIEHEN [INVNR] $\subseteq$ BUCH [INVNR]
ENTLIEHEN [LNR] $\subseteq$ LESER [LNR]

Die Attribute INVNR und LNR heißen **Fremdschlüssel-Attribute** in **ENTLIEHEN.**

Allgemein wird definiert:

$K \subseteq X$ heißt **Fremdschlüssel** in $R = (X, \sum_x)$,

falls K in $R' \neq R$ **Primärschlüssel** ist und zu jedem Wert von K in der

R-Relation ein Primärschlüsselwert in der R´-Relation existiert.

⇒ **Fremdschlüssel sind allgemein als Inklusionsabhängigkeit $R[K] \subseteq R'[K]$ zu beschreiben.**

Für die Transformation 2-stelliger Beziehungen vom ERM in das RDM gilt:

Sei R eine **m:n-Beziehung** zwischen E_1 und E_2 und
seien R bzw. $R_i = (X_j, \{K_j\})$, i=1, 2 die zugehörigen Relationenschemas,
so wird die Gültigkeit der **ID**

$R[K_1] \subseteq R_1[K_1]$ und
$R[K_2] \subseteq R_2[K_2]$ gefordert.

Konzeptionelle Datenmodellierung für ein betriebliches Informationssystem (10)

Noch nicht übertragen wurde die Information, dass **IS-A-Beziehungen** bestehen.
D.h. es bestehen **Existenzabhängigkeiten**, die als **interrelationale Datenabhängigkeiten** dargestellt werden können. Es muss also gelten:

MITARBEITER[PNR]	⊆	PERSON	[PNR]
ZEITAK[PNR]	⊆	PERSON	[PNR]
MANAGER[PNR]	⊆	MITARBEITER	[PNR]
KB [PNR]	⊆	MITARBEITER	[PNR]

Des weiteren gilt:

BEDIENT[MNR]	⊆	MASCHINE	[MNR]
BEDIENT[PNR]	⊆	MITARBEITER	[PNR]
BENÖTIGT [MNR]	⊆	MASCHINE	[MNR]
BENÖTIGT [RNR]	⊆	ROHSTOFF	[RNR]
LIEFERT [RNR]	⊆	ROHSTOFF	[RNR]
LIEFERT [LNR]	⊆	LIEFERANT	[LNR]

Man sieht:

MNR ist Fremdschlüssel in der Relation BEDIENT und BENÖTIGT,
PNR ist Fremdschlüssel in der Relation BEDIENT,
RNR ist Fremdschlüssel in der Relation BENÖTIGT und LIEFERT,
LNR ist Fremdschlüssel in der Relation LIEFERT.

Es sind folgende **Vereinfachungen** möglich:

Aufnahme von **PNR** in das **R-Schema** von MASCHINE:

→

MASCHINE = ({MNR, PNR, NAME, ANSCH_DATUM, NEUWERT, ZEITWERT}, {MNR})

Daraus folgt:

MASCHINE[PNR] $\subseteq$ **MITARBEITER [PNR]**

PNR ist **Fremdschlüssel** in der Relation MASCHINE.
Alle Informationen der Relation **BEDIENT** findet man jetzt in der Relation MASCHINE.

Eine solche Vereinfachung ist bezüglich der Relation **BENÖTIGT** nicht möglich, da dadurch eine **m:n-Beziehung** ausgedrückt wird und für jede **Kombination** der **Fremdschlüsselattribute** noch ein Wert des Attributes M_MENGE anzugeben ist.

Durch Aufnahme der Attribute LNR und L_MENGE in die Relation ROHSTOFF kann auf die Relation **LIEFERT** verzichtet werden:

ROHSTOFF = ({RNR, LNR, R_NAME, MENGE, PREIS, L_MENGE }, {RNR})

ROHSTOFF [LNR] $\subseteq$ **LIEFERANT[LNR].**

LNR ist **Fremdschlüssel** in der Relation ROHSTOFF.
Alle Informationen der Relation **LIEFERT** findet man jetzt in der Relation ROHSTOFF.

Es liegen die folgenden **R-Schemas** und **Abhängigkeiten** vor:

PERSON = ({PNR, NAME, ADR(PLZ, STADT, STRASSE), TEL, E-MAIL}, {PNR})

MITARBEITER = ({PNR, GEH_STUFE, BETRAG, ABT_NR, ABT_NAME, KRANKENKASSE, {P_BETRAG}, {KINDER(K_NAME, K_VORNAME, K_GEB)}}, {PNR})

ZEITAK = ({PNR, L_FIRMA, H_LOHN, {VON_BIS}, H_SUMME}, {PNR})

MANAGER = ({PNR, A_GRAD, STUD_RI}, {PNR})

KB = ({PNR, KD_NAME}, {PNR})

MASCHINE = ({MNR, PNR, NAME, ANSCH_DATUM, NEUWERT, ZEITWERT}, {MNR})

ROHSTOFF = ({RNR, LNR, R_NAME, MENGE, PREIS, L_MENGE}, {RNR})

LIEFERANT = ({LNR, FIRMA, ADR(PLZ, STADT, STRASSE), ANSPRECHP, GESCHLECHT}, {LNR})

BENÖTIGT = ({MNR, RNR, M_MENGE }, {MNR, RNR})

MITARBEITER[PNR] $\subseteq$ PERSON[PNR]
ZEITAK[PNR] $\subseteq$ PERSON[PNR]
MANAGER[PNR] $\subseteq$ MITARBEITER[PNR]
KB [PNR] $\subseteq$ MITARBEITER[PNR]

MASCHINE[PNR] $\subseteq$ MITARBEITER[PNR]
ROHSTOFF [LNR] $\subseteq$ LIEFERANT[LNR]
BENÖTIGT [MNR] $\subseteq$ MASCHINE[MNR]
BENÖTIGT [RNR] $\subseteq$ ROHSTOFF[RNR]

5.3.4 Entwurfs-Theorie relationaler DB-Schemas

5.3.5.1 Definitionen

Definition:
Sei V eine Attributmenge, X, Y ⊆ V.
Es besteht eine funktionale Abhängigkeit f

f: X → Y für eine Relation r ∈ Rel (V)

genau dann wenn für alle **μ, v** ∈ **r** gilt:

Aus **μ[X] = v[X]** folgt immer **μ[Y]=v[Y]**

f heißt Name der funktionalen Abhängigkeit X → Y.

Häufig wird an Stelle des Namens f einfach die Abhängigkeit X → Y geschrieben.

Eine Menge von funktionalen Abhängigkeiten wird mit F bezeichnet.

Ziel: logischer Entwurf eines relationalen Datenbankschemas

Konzeptionelle Datenmodellierung für ein betriebliches Informationssystem (11)

Untersuchung auf funktionelle Abhängigkeiten:

Relation PERSON:

PNR → NAME , ADR(PLZ, STADT, STRASSE), TEL, E-MAIL

PLZ → STADT

Relation MITARBEITER:

PNR → GEH_STUFE, BETRAG, ABT_NR, ABT_NAME, KRANKENKASSE {P_BETRAG}, {KINDER(K_NAME, K_VORNAME, K_GEB)}

GEH_STUFE → BETRAG

ABT_NR → ABT_NAME

Relation ZEITAK:

PNR → L_FIRMA, H_LOHN, {VON_BIS}, H_SUMME

Relation MANAGER:

PNR → A_GRAD , STUD_RI

Relation KB:

PNR → KD_NAME

Relation MASCHINE:

MNR → PNR, NAME, ANSCH_DATUM, NEUWERT, ZEITWERT

Relation ROHSTOFF:

RNR → LNR, R_NAME , MENGE , PREIS, L_MENGE

Relation LIEFERANT:

LNR → FIRMA , ADR(PLZ, STADT, STRASSE), ANSPRECHP, GESCHLECHT

PLZ → STADT

FIRMA → ADR(PLZ, STADT, STRASSE)

Relation BENÖTIGT:

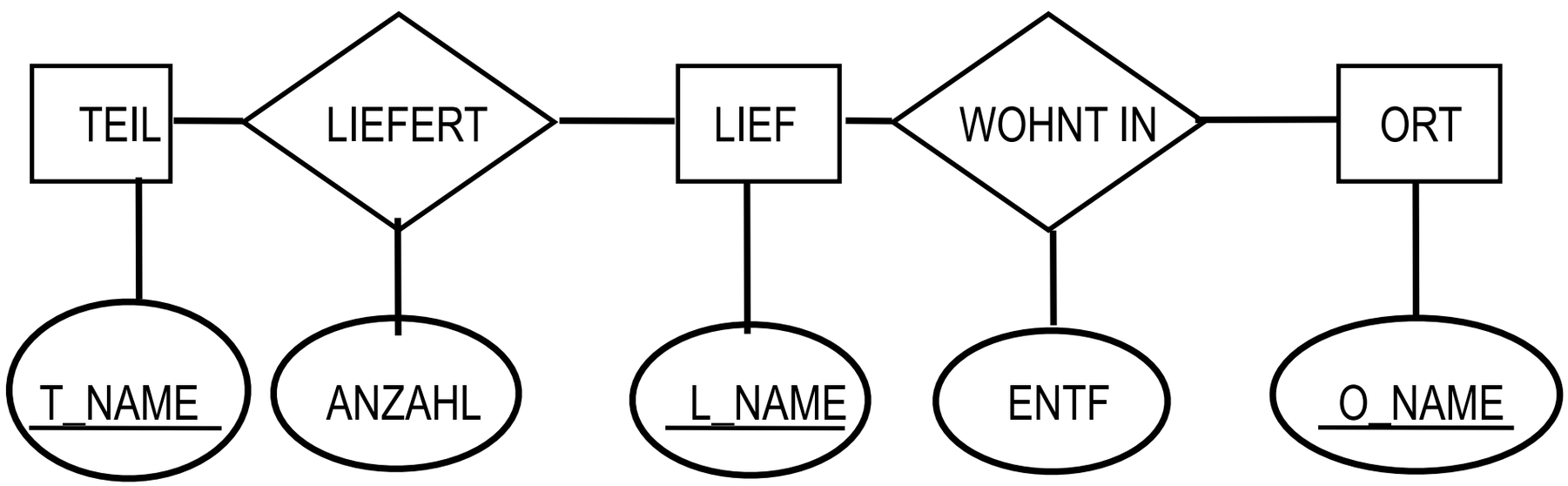
MNR,RNR → M_MENGE

Beispiel:
Firma möchte **Daten speichern**

über ihre **Lieferanten** (LIEF)
- Namen (des Lieferanten) (L_NAME)
- Ort des Firmensitzes (O_NAME)
- die Entfernung von der Firma zu diesem Ort des Lieferanten (ENTF)

für jedes gelieferte **Teil** (TEIL)
- dessen Bezeichnung (T_NAME)
- dessen Anzahl (ANZAHL) .

Darstellung als ERD:



Transformation in das RDM:

LIEF = (L_NAME, { L_NAME → L_NAME})
TEIL = (T_NAME, { T_NAME → T_NAME})
ORT = (O_NAME, { O_NAME → O_NAME})

LIEFERT = (L_NAME, T_NAME, ANZAHL, { L_NAME, T_NAME → ANZAHL })
WOHNT IN = (L_NAME, O_NAME, ENTF, {L_NAME → O_NAME, ENTF; O_NAME → ENTF })

Ergebnis unbefriedigend-mit Redundanz behaftet.

Alternatives Vorgehen:

- Relationenschema mit allen Attributen einer Anwendung
- funktionale Abhängigkeiten

Das heißt:

Aufstellen eines Universalrelationen-Schema auf Basis eines Universum U (entspricht Diskursbereich).

Für jede funktionale Abhängigkeit gilt **attr(f) ⊆ U.**
Ergebnis: Relationenschema der Form R = (U, F).

Ziel des logischen Entwurfs:
Schema zur Speicherung in einem Data Dictionary

5.3.4.1 Update-Anomalien

Universalschema für Beispiel:

R = **(U, F)** mit
U = L_NAME, T_NAME, ANZAHL, O_NAME, ENTF
F = **{** L_NAME, T_NAME → ANZAHL; L_NAME → O_NAME, O_NAME → ENTF **}**

r ∈ Rel(U)

L_NAME	T_NAME	ANZAHL	O_NAME	ENTF
L1	T1	300	Berlin	300
L1	T2	200	Berlin	300
L1	T3	400	Berlin	300
L1	T4	200	Berlin	300
L1	T5	100	Berlin	300
L1	T6	100	Berlin	300
L2	T1	100	München	800
L2	T2	400	München	800
L3	T2	200	München	800
L4	T2	200	Bremen	100
L4	T4	300	Bremen	100
L4	T5	400	Bremen	100

Relation weist **Anomalien auf:**

1. **Einfüge-Anomalie**
   Soll ein Tupel in die bestehende Relation eingefügt werden, dann setzt das eine Wertebelegung für alle Attribute voraus.
2. **Lösch-Anomalie**
   Werden die Informationen zu einem Teil nach dessen Lieferung gelöscht und liefert der Lieferant aktuell keine weiteren Teile, dann werden auch die zugehörigen Lieferanteninformationen gelöscht.
3. **Änderungs-Anomalie**
   Das Ändern eines Wertes kann dazu führen, dass an mehreren Stellen geändert werden muss.

Basis: **Normalformenlehre**, d.h. aus einer Relation werden durch Normalisierung Relationen abgeleitet.

5.3.4.2 Normalformen

Erste Normalform:

Sicht auf die Wertebereiche der Attribute.

Es gilt:

- Eine Relation ist in **erster Normalform**, wenn die

  Wertebereiche aller Attribute nur **elementare Werte** enthalten.

Nicht erlaubt sind damit

- Werte, die selbst eine **Struktur** besitzen
- **Wiederholungsgruppen**.

Abhilfe durch **Normalisierung**.

5.3.4.3. Zweite Normalform

Ist eine Relation nur in erster Normalform können bei Update–Operationen **Anomalien** auftreten.

Ursache für die Anomalien:

In der **Relation** R sind die Attribute Wohnort (O_NAME) und Entfernung (ENTF) nicht vom **vollständigen Primärschlüssel** (L_NAME, T_NAME) abhängig,
sondern **nur von einem Teil dieses Primärschlüssels** (L_NAME).

Funktionale Abhängigkeiten:

L_NAME, T_NAME → ANZAHL
L_NAME → O_NAME, ENTF
O_NAME → ENTF

Man bildet zwei **R-Schemata:**

R1 = ((L_NAME, T_NAME, ANZAHL) , { L_NAME, T_NAME }),

R2 = ((L_NAME , O_NAME , ENTF), { L_NAME }).

r1:

L_NAME	T_NAME	ANZAHL
L1	T1	300
L1	T2	200
L1	T3	400
L1	T4	200
L1	T5	100
L1	T6	100
L2	T1	100
L2	T2	400
L3	T2	200
L4	T2	200
L4	T4	300
L4	T5	400

r2:

L_NAME	O_NAME	ENTF
L1	Berlin	300
L2	München	800
L3	München	800
L4	Bremen	100

Definitionen:

Schlüssel: Jede **minimale identifizierende** Attributkombination einer Relation.

Schlüsselattribut: Jedes Attribut A, welches Bestandteil eines Schlüssels ist.

Nichtschlüsselattribut: Jedes Attribut A, welches nicht zu irgendeinem Schlüssel gehört.

Zweite Normalform:
Eine Relation R ist in **2. NF**, wenn

- die **1. NF** gilt und
- für jeden Schlüssel K gilt, dass jedes Nichtschlüsselattribut vom gesamten Schlüssel K abhängig ist (und nicht nur von einem Teil von K).

Die abgeleiteten Beispielrelationen sind in 2.NF.

Wir setzen i.f. voraus:
In jeder Relation haben wir **genau** einen Schlüssel K-den **Primärschlüssel**.

Allgemein gilt:

Besteht der **Primärschlüssel** einer Relation R nur aus **einem Attribut** und ist die Relation in 1.NF, so ist sie auch in 2.NF.

Man sieht:

- Relation ist nunmehr in 2. NF
- man hat noch Update-Anomalien.

Man sieht:

Man benötigt eine weitere Normalform.

5.3.4.4. Dritte Normalform

Die Ursache für die noch bestehenden Anomalien liegt in der Existenz von sog. **transitiven Abhängigkeiten**.

Es gilt:

Transitive Abhängigkeiten liegen dann vor, wenn es funktionale Abhängigkeiten im Bereich der **Nichtschlüsselattribute** gibt.

Wir betrachten R2:

R2 = ({L_NAME , O_NAME **,** ENTF }, { L_NAME }).

Nichtschlüsselattribute: Y = { O_NAME **,** ENTF}

Funktionale Abhängigkeiten im Bereich der **Nichtschlüsselattribute Y**:
O_NAME → ENTF

(Attribut ENTF ist transitiv vom Primärschlüssel L_NAME abhängig,
weil gilt:
L_NAME → O_NAME und
O_NAME → ENTF **)**

Definition:

Eine Relation ist in **3. NF**, wenn sie in 1. NF vorliegt und es keine funktionalen Abhängigkeiten zwischen Nichtschlüsselattributen gibt.

Nichtschlüsselattribute müssen also immer von (vollständigen) Schlüsseln funktional abhängig sein und nur von diesen.

Nichtschlüsselattribute dürfen also nicht transitiv von Schlüsseln abhängen.

Es gilt (ohne Beweis):

Ist eine Relation in 3. NF, dann ist sie auch in 2. NF.

Die 3. NF löst noch nicht alle **Anomalieprobleme**, deshalb wurden weitere Normalformen eingeführt.

Das R-Schema **R2** wird unter Benutzung der festgestellten Abhängigkeiten in die 3.NF überführt.
Aus der festgestellten Abhängigkeit leitet man eine eigene Relation ab und erhält somit die **zwei R-Schemas in 3. NF:**

R2 = ((L_NAME, O_NAME), {L_NAME })
R3 = ((O_NAME, ENTF) , { O_NAME }).

Ergebnis:

Aus

R = ((L_NAME, T_NAME, ANZAHL, O_NAME, ENTF),
{ L_NAME, T_NAME })

mit den funktionale Abhängigkeiten

L_NAME, T_NAME → ANZAHL,
L_NAME → O_NAME,
O_NAME → ENTF

wurde ein Datenbankschema R' abgeleitet, welches die gleichen Anwendungen wie R modelliert, aber dafür besser geeignet ist.

R' = {R1, R2, R2} mit
R1 = ((L_NAME, T_NAME, ANZAHL), { L_NAME, T_NAME})
R2 = ((L_NAME, O_NAME), {L_NAME })
R3 = ((O_NAME, ENTF), { O_NAME }),

→

- R und R' „eng" verwandt
  (gleiche Attribute, gleiche Abhängigkeiten)
- jedes Schema von R' ist in 3.NF

→ **keine Anomalie und Redundanzen mehr**

r1:

L_NAME	T_NAME	ANZAHL
L1	T1	300
L1	T2	200
L1	T3	400
L1	T4	200
L1	T5	100
L1	T6	100
L2	T1	100
L2	T2	400
L3	T2	200
L4	T2	200
L4	T4	300
L4	T5	400

r2:

L_NAME	O_NAME
L1	Berlin
L2	München
L3	München
L4	Bremen

r3:

O_NAME	ENTF
Berlin	300
München	800
Bremen	100

Konzeptionelle Datenmodellierung für ein betriebliches Informationssystem (12)

Normalisierung:

1. Normalform:

Relation PERSON:

ADR ist **zusammengesetztes** Attribut, anstelle dieses Attributes nehmen wir die drei Attribute **PLZ**,**STADT**, **STRASSE** als **eigenständige** Attribute auf.

Relation **PERSON** ist damit in **erster NF**:

PERSON = ({PNR, NAME, PLZ, STADT, STRASSE, TEL, E-MAIL}, {PNR})

Alternativ:

Eigenständige Relation ADRESSE:

ADRESSE = ({ADR_ID, PLZ, STADT, STRASSE}, {ADR_ID})

Daraus ergibt sich als **Relation PERSON**:

PERSON = ({PNR, NAME ADR_ID, TEL, E-MAIL}, {PNR})

Relation MITARBEITER:

P_BETRAG ist **Wiederholungsgruppe**, wir schaffen **eigenständige Relation** Prämie mit den Attributen PNR und P_BETRAG.

Aus Diskussionen folgt, dass auch das **Datum** der Prämienzahlung von Interesse ist.

Daraus folgt **Relation**

PRÄMIE = ({PNR, P_DATUM, P_BETRAG},{PNR, P_DATUM })
und

MITARBEITER = ({PNR, GEH_STUFE, BETRAG, ABT_NR, ABT_NAME, KRANKENKASSE,
{KINDER(K_NAME, K_VORNAME, K_GEB)}}, {PNR})

KINDER ist **zusammengesetztes Attribut** und **Wiederholungsgruppe**:

Wir schaffen **eigenständige Relation:**

KINDER = ({PNR, K_NAME, K_VORNAME, K_GEB}, {PNR, K_VORNAME})
und
MITARBEITER = ({PNR, GEH_STUFE, BETRAG, ABT_NR, ABT_NAME, KRANKENKASSE}, {PNR})

Relation ZEITAK :

VON_BIS ist Wiederholungsgruppe,
wir schaffen eine eigenständige Relation **EINSATZ**, wobei wir die Attribute trennen, also Attribute VON und BIS einführen.

EINSATZ = ({PNR, ENR, VON, BIS}, {PNR, ENR})

Zusätzlich wird das Attribut Einsatznummer ENR
eingefügt. (Zählung erfolgt pro Zeitarbeitskraft)

ZEITAK = ({PNR, L_FIRMA, H_LOHN, H_SUMME}, {PNR})

Relation **LIEFERANT:**

LIEFERANT = ({LNR, FIRMA, ADR(PLZ, STADT, STRASSE), ANSPRECHP, GESCHLECHT}, {LNR})

ADR ist ein zusammengesetztes Attribut, wir nehmen die drei einzelnen
Attribute auf:

LIEFERANT = ({LNR, FIRMA, PLZ, STADT, STRASSE, ANSPRECHP, GESCHLECHT}, {LNR})

Ergebnis: Alle Relationen sind jetzt in 1.NF.

fhwedel
UNIVERSITY OF APPLIED SCIENCES

2. Normalform:

PERSON = ({PNR, NAME, PLZ, STADT, STRASSE, TEL, E-MAIL}, {PNR})
MITARBEITER = ({PNR, GEH_STUFE, BETRAG, ABT_NR, ABT_NAME, KRANKENKASSE}, {PNR})

PRÄMIE = ({PNR, P_DATUM, P_BETRAG}, {PNR, P_DATUM })
KINDER = ({PNR, K_NAME, K_VORNAME, K_GEB}, {PNR, K_VORNAME})
ZEITAK = ({PNR, L_FIRMA, H_LOHN, H_SUMME}, {PNR})
EINSATZ = ({PNR, ENR, VON, BIS}, {PNR, ENR})

MANAGER = ({PNR, A_GRAD, STUD_RI}, {PNR})
KB = ({PNR, KD_NAME}, {PNR})

MASCHINE = ({MNR, PNR, NAME, ANSCH_DATUM, NEUWERT, ZEITWERT}, {MNR})
ROHSTOFF = ({RNR, LNR, R_NAME, MENGE, PREIS, L_MENGE}, {RNR})
LIEFERANT = ({LNR, FIRMA, PLZ, STADT, STRASSE, ANSPRECHP, GESCHLECHT }, {LNR})

BENÖTIGT = ({MNR, RNR, M_MENGE}, {MNR, RNR})

Es sind nur die Relationen zu prüfen, deren Primärschlüssel aus mindestens 2 Attributen zusammengesetzt sind.

Es gilt:

PNR, P_DATUM ⇒ P_BETRAG **(voll funktional abhängig)**

PNR , K_VORNAME ⇒ K_NAME, K_GEB

PNR, ENR ⇒ VON, BIS

MNR,RNR ⇒ M_MENGE

Alle Relationen somit in 2. NF.

3. Normalform :

Wir suchen **funktionale Abhängigkeiten** im Bereich der **Nichtschlüsselattribute**:

Dazu betrachten wir die funktionalen Abhängigkeiten.

Relation PERSON:

PERSON = ({PNR, NAME, PLZ, STADT, STRASSE,TEL, E-MAIL}, {PNR})
PLZ → STADT

Daraus eigene Relation **STADT** mit den Attributen PLZ, STADT:

STADT = ({PLZ, STADT}, { PLZ})
PERSON = ({PNR, NAME, PLZ, STRASSE, TEL, E-MAIL}, {PNR})

Relation MITARBEITER:

MITARBEITER = ({PNR, GEH_STUFE, BETRAG, ABT_NR, ABT_NAME, KRANKENKASSE}, {PNR})
GEH_STUFE → BETRAG
ABT_NR → ABT_NAME

Eigene Relationen:

GEHALT = ({GEH_STUFE, BETRAG}, {GEH_STUFE})
ABTEILUNG = ({ABT_NR, ABT_NAME}, {ABT_NR})
MITARBEITER = **({PNR, GEH_STUFE, ABT_NR, KRANKENKASSE}, {PNR})**

Relation Prämie:

PRÄMIE = ({PNR, P_DATUM, P_BETRAG},{PNR, P_DATUM})

Automatisch 3.NF (nur ein Nichtschlüsselattribut)

Relation KINDER:
KINDER = ({PNR, K_NAME, K_VORNAME, K_GEB }, {PNR, K_VORNAME })

Relation ZEITAK:
ZEITAK = ({PNR, L_FIRMA, H_LOHN, H_SUMME}, {PNR})

Frage: Hängt H_LOHN von der PERSON oder der L_FIRMA ab?
Wenn von PERSON abhängig → 3.NF,
wenn von L_FIRMA abhängig →Normalisierung
Z1=({PNR, L_FIRMA, H_SUMME},{PNR}) und
Z2= ({L_FIRMA, H_LOHN},{L_FIRMA})

Relation Einsatz:
EINSATZ = ({PNR, ENR, VON, BIS}, {PNR, ENR})

Relation MANAGER:
MANAGER = ({PNR, A_GRAD, STUD_RI}, {PNR})

Relation KB:
KB = ({PNR, KD_NAME}, {PNR})

Relation MASCHINE:
MASCHINE = ({MNR, PNR, NAME, ANSCH_DATUM, NEUWERT, ZEITWERT}, {MNR})

Relation ROHSTOFF:
ROHSTOFF = ({RNR, LNR, R_NAME, MENGE, PREIS, L_MENGE}, {RNR})

Relation LIEFERANT:

LIEFERANT = ({LNR, FIRMA, PLZ, STADT, STRASSE, ANSPRECHP, GESCHLECHT}, {LNR})

PLZ→STADT

LIEFERANT = ({LNR, FIRMA, PLZ, STRASSE, ANSPRECHP, GESCHLECHT }, {LNR})

Relation BENÖTIGT:

BENÖTIGT = ({MNR, RNR, M_MENGE}, {MNR, RNR})