

Datenbanksysteme

Semantische Modellierung mit dem Entity/Relationship-Modell

Burkhardt Renz

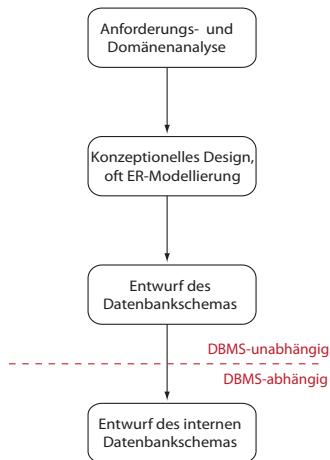
Fachbereich MNI
Technische Hochschule Mittelhessen

Sommersemester 2020

Inhalt

- Vorgehensweise und ein Beispiel
 - Phasen im Datenbank-Entwurf
 - Ein Beispiel
- Entity-Relationship-Modell
 - Entitäten, Attribute, Assoziationen
 - Erweitertes Entity-Relationship-Modell
 - Design-Prinzipien
- Schema-Entwurf
 - Regeln für das ER-Modell
 - Transformation des erweiterten ER-Modells
 - Rezept für den Datenbank-Entwurf

Vorgehen im Datenbank-Entwurf



Inhalt

- Vorgehensweise und ein Beispiel
 - Phasen im Datenbank-Entwurf
 - Ein Beispiel
- Entity-Relationship-Modell
 - Entitäten, Attribute, Assoziationen
 - Erweitertes Entity-Relationship-Modell
 - Design-Prinzipien
- Schema-Entwurf
 - Regeln für das ER-Modell
 - Transformation des erweiterten ER-Modells
 - Rezept für den Datenbank-Entwurf

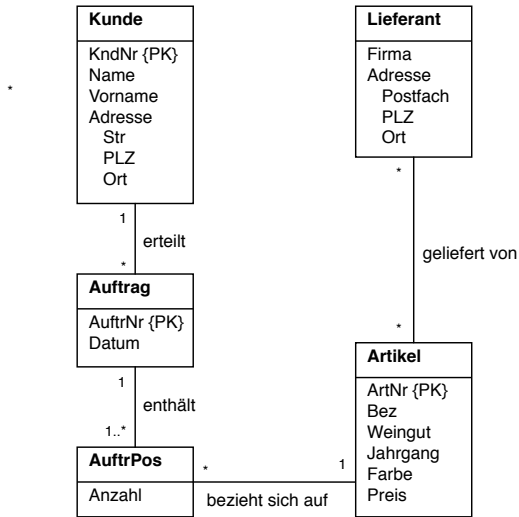
Domänen- und Anforderungsanalyse I

- Der Weinhändler verkauft Weine an Kunden. Die Artikel werden mit einer eindeutigen Artikelnummer gekennzeichnet. Im Katalog der Artikel findet der interessierte Kunde die folgenden Angaben: Bezeichnung des Weins, Weingut oder Winzer, d.h. den Hersteller des Weins, Jahrgang, Farbe und Preis. Manche Weine haben keine Jahrgangsangabe.
- Die Kunden werden unter einer Kundennummer verzeichnet. Der Weinhändler hat die Namen, Vornamen und Adressen seiner Kunden, die übrigens auch aus dem europäischen Ausland sein können.

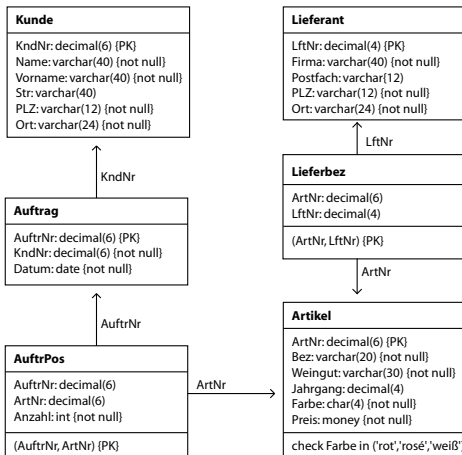
Domänen- und Anforderungsanalyse II

- Das zu konstruierende System soll die Aufträge der Kunden aufzeichnen können. Aufträge werden an einem bestimmten Datum erteilt und erhalten eine Auftragsnummer. Sie enthalten Auftragspositionen, nämlich jeweils die Anzahl der bestellten Artikel.
- Der Weinhandler bezieht seine Weine nicht direkt bei den Weingütern, sondern über Lieferanten. Die Lieferanten sind Firmen mit einer Adresse mit folgenden Angaben: Postfach, PLZ und Ort. Ein Lieferant kann die ausgewählten Weine mehrerer Weingüter anbieten. Ein Wein kann auch von mehreren Lieferanten bezogen werden.

ER-Diagramm (in UML-Notation)



Datenbankschema (als Diagramm)



Inhalt

- Vorgehensweise und ein Beispiel
 - Phasen im Datenbank-Entwurf
 - Ein Beispiel
- Entity-Relationship-Modell
 - Entitäten, Attribute, Assoziationen
 - Erweitertes Entity-Relationship-Modell
 - Design-Prinzipien
- Schema-Entwurf
 - Regeln für das ER-Modell
 - Transformation des erweiterten ER-Modells
 - Rezept für den Datenbank-Entwurf

Entitäten

Definition (Entität)

Eine **Entität** ist ein identifizierbares „Ding“, Objekt aus dem Anwendungsgebiet

Beispiel

Der Kunde namens „Thomas Lehr“
ein bestimmter Artikel
der Auftrag, den Kundin Riesling heute erteilt hat. . .

Entitätsmenge

Definition (Entitätsmenge)

Eine **Entitätsmenge** ist eine Menge gleichartig strukturierter Entitäten, die zu einem bestimmten Zeitpunkt betrachtet werden.

Beispiel

Alle Kunden, die bei uns kaufen

Alle Artikel, die wir im Angebot haben

Alle Aufträge, die noch nicht bezahlt sind ...

Entitätstyp

Definition (Entitätstyp)

Ein **Entitätstyp** ist die Menge aller möglichen gleichartig strukturierten Entitäten.

Wir identifizieren Entitätstypen durch einen eindeutigen Namen. Jeder Entitätstyp hat eine Menge von *Attributen*, das sind die Merkmale, die diesen Entitätstyp charakterisieren.

Beispiel

Kunde

Artikel

Auftrag ...

Attribut

Definition (Attribut)

Ein **Attribut** ist eine Eigenschaft eines Entitätstyps oder einer Assoziation.

Ein **Attributwert** ist der Wert eines Attributs einer bestimmten Entität.

Ein **Datentyp** ist eine Menge von Werten. Jedes Attribut hat einen Datentyp und der Attributwert hat diesen Typ.

Zum Merken

Entitätstypen sind die Arten von „Dingen“, **über** die wir sprechen
Attribute sind die Eigenschaften solcher Dinge, also **was** wir über sie sagen

Beziehung

Definition (Beziehung)

Eine **Beziehung** ist eine Verbindung zwischen Entitäten

Beispiel

Kunde „Thomas Lehr“ hat Auftrag 1003 erteilt
Lieferant „Louis Max“ liefert Artikel 100101...

Beziehungsmenge

Definition (Beziehungsmenge)

Eine **Beziehungsmenge** ist eine Menge gleichartiger Verbindungen zwischen Entitäten zu einem bestimmten Zeitpunkt

Beispiel

Alle Kombinationen von Kunden mit den Aufträgen, die sie erteilt haben

Alle Aufträge mit ihren Auftragspositionen, so wie sie heute vorliegen. . .

Assoziation

Definition (Assoziation)

Eine **Assoziation** (auch: **Beziehungstyp**) ist die Menge aller möglichen gleichartigen Beziehungen zwischen Entitäten von je bestimmtem Typ.

Man sagt: Die Assoziation verbindet die Entitätstypen

Beispiel

Kunde *erteilt* Auftrag

Lieferant *liefert* Artikel...

Grad einer Assoziation

Definition (Grad einer Assoziation)

Der **Grad** einer Assoziation ist die Zahl der beteiligten Entitätstypen.

An einer **binären** Assoziation sind zwei Entitätstypen beteiligt.

An einer **ternären** Assoziation sind drei Entitätstypen beteiligt.

Beispiel

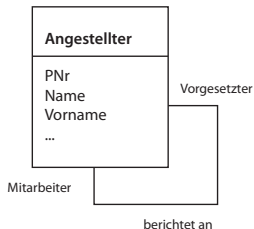
Die Assoziation „Eine $\langle \text{Person} \rangle$ verwendet in einem $\langle \text{Projekt} \rangle$ eine $\langle \text{Programmiersprache} \rangle$ “ ist eine ternäre Assoziation

Reflexive Assoziation

Definition (Reflexive Assoziation)

Eine **reflexive Assoziation** ist eine Assoziation, an der derselbe Entitätstyp in verschiedenen **Rollen** beteiligt ist.

Beispiel



Eigenschaften von Attributen

Definition

Einfache Attribute haben einen atomaren Wert.

Zusammengesetzte Attribute haben mehrere Wert-Komponenten.

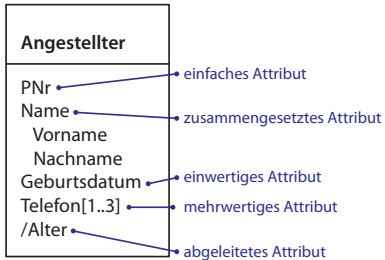
Ein Attribut ist **einwertig**, wenn jede Entität nur einen Wert zu diesem Attribut hat.

Ein Attribut ist **mehrwertig**, wenn jede Entität mehrere Werte zu diesem Attribut haben kann.

Ein Attribut ist **abgeleitet**, wenn sein Wert aus dem anderer Attribute berechnet werden kann.

Eigenschaften von Attributen

Beispiel



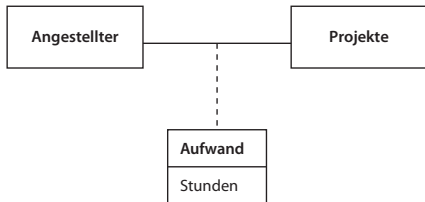
Attribute von Assoziationen

Assoziationen können auch Attribute haben:

Merkmale, die weder dem einen noch dem anderen Entitätstyp zuzuordnen sind, sondern zu ihrer Beziehung gehören.

In der UML nennt man dies eine **Assoziationsklasse**

Beispiel



Schlüssel

Definition

Ein **Superschlüssel** ist eine Menge von Attributen, deren Werte eine Entität eindeutig bestimmen.

Ein **Schlüssel** ist ein Superschlüssel mit minimaler Anzahl von Attributen

Ein **Primärschlüssel** ist ein ausgewählter, dedizierter Schlüssel

Starke und schwache Entitätstypen

Definition

Ein **starker Entitätstyp** ist ein Entitätstyp, dessen Primärschlüssel nicht von einem anderen Entitätstyp abhängt.

Ein **schwacher Entitätstyp** ist ein Entitätstyp, dessen Primärschlüssel von der Existenz anderer Entitäten abhängig ist.

Beispiel

Artikel ist ein *starker* Entitätstyp

AuftrPos ist ein *schwacher* Entitätstyp

Integritätsbedingungen für Assoziationen

Eine wichtige Integritätsbedingung in der Entity-Relationship-Modellierung ist die Frage nach der **Multiplizität** einer Assoziation:

Wieviele Entitäten eines Typs dürfen eine Beziehung zu Entitäten eines anderen Typs haben?

Multiplizität bezeichnet zwei Konzepte bezüglich von Assoziationen:

- ➊ **Kardinalität**, die Zahl möglicher Beziehungen der Entitäten
- ➋ **Beteiligung**, die Frage ob die Beziehung für eine Entität optional oder obligatorisch ist

1-1-Assoziation

Definition

In einer **1-1-Assoziation** zwischen E_1 und E_2 hat eine Entität vom Typ E_1 höchstens eine Beziehung zu einer Entität vom Typ E_2 und umgekehrt.

Beispiel

Ein Angestellter leitet eine Abteilung und jede Abteilung wird von einem Angestellten, dem Abteilungsleiter geleitet.



1-n-Assoziation

Definition

In einer **1-n-Assoziation** zwischen E_1 und E_2 hat eine Entität vom Typ E_1 mehrere Beziehungen zu Entitäten vom Typ E_2 , umgekehrt hat jedoch eine Entität vom Typ E_2 höchstens eine Beziehung zu einer Entität vom Typ E_1 .

Beispiel

Eine Abteilung hat mehrere Angestellte, ein Angestellter gehört jedoch zu genau einer Abteilung



n-m-Assoziation

Definition

In einer **n-m-Assoziation** zwischen E_1 und E_2 hat eine Entität vom Typ E_1 mehrere Beziehungen zu Entitäten vom Typ E_2 und umgekehrt

Beispiel

Ein Angestellter arbeitet an mehreren Projekten und ein Projekt hat mehrere Angestellte, die in ihm arbeiten.



Multiplizitäten

Wichtige Multiplizitäten

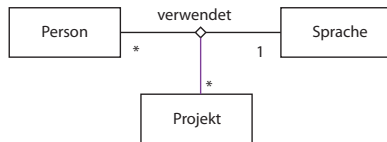
0..1	eins oder keins, eins optional
1	genau eins
0..*, *	beliebig viele, evtl. auch keins
1..*	mindestens eins

Multiplizität bei ternären Assoziationen

Definition

Die Multiplizität eines Assoziationsendes einer **n-ären** Assoziation ist die mögliche Zahl von Entitäten, wenn man je eine Entität der anderen $n-1$ Entitätstypen wählt.

Beispiel



- Eine Person verwendet in einem Projekt wieviele Sprachen? 1
- Eine Person verwendet eine Sprache in wievielen Projekten? *
- Eine Sprache wird in einem Projekt von wievielen Personen verwendet? *

Inhalt

- Vorgehensweise und ein Beispiel
 - Phasen im Datenbank-Entwurf
 - Ein Beispiel
- **Entity-Relationship-Modell**
 - Entitäten, Attribute, Assoziationen
 - **Erweitertes Entity-Relationship-Modell**
 - Design-Prinzipien
- Schema-Entwurf
 - Regeln für das ER-Modell
 - Transformation des erweiterten ER-Modells
 - Rezept für den Datenbank-Entwurf

Super- und Subklassen

Entitätstypen sind Mengen von „Dingen“ gleicher Art. Oft kann man diese Mengen unterteilen:

Es gibt verschiedene Typen von Angestellten: z.B. Techniker, Ingenieure, Manager etc.

Definition

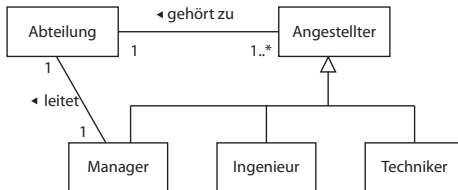
Eine **Subklasse** ist eine Teilmenge eines Entitätstyps, die eine eigene Repräsentation im Entity-Relationship-Modell hat.

Eine **Superklasse** ist ein Entitätstyp, der die Vereinigung von mehreren Entitätstypen ist.

Man spricht auch von einer „**ist-ein**“ („**is-a**“)-**Beziehung**:
„Ein Techniker *ist ein* Angestellter“

Beispiel für Subklassen-Bildung

Beispiel



Generalisierung und Spezialisierung

Definition

Generalisierung ist das Bilden von Superklassen, indem man Gemeinsamkeiten von Entitätstypen identifiziert und zu einer eigenen Superklasse macht.

Spezialisierung besteht darin, bedeutungsvolle Teilmengen von Entitätstypen zu bilden.

Eine Subklasse *erbt* alle Eigenschaften, d.h. Attribute und Assoziationen der Superklasse – weil ja eine Entität der Subklasse auch ein Element der Superklasse ist.

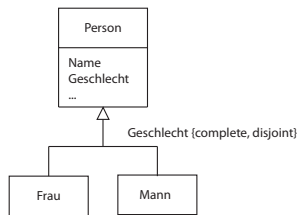
Spezialisierung durch Merkmale von Entitäten

Definition

Wenn eine Bedingung über die Ausprägung von Attributen festlegt, ob eine Entität zu einer Subklasse gehört, spricht man von **bedingungsdefinierter** Spezialisierung.

Bestimmt der Wert eines Attributs die Zugehörigkeit zu einer Subklasse, ist es eine **attributdefinierte** Spezialisierung

Beispiel

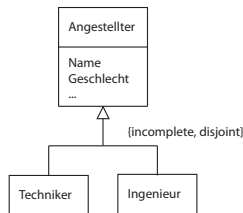


Vollständigkeitseinschränkung

Die **Vollständigkeitseinschränkung** bestimmt, ob jede Entität einer Superklasse auch in einer ihrer Subklassen vorkommen muss.

- complete \cong Jede Entität der Superklasse kommt in mindestens einer Subklasse vor
- incomplete \cong Nicht jede Entität der Superklasse muss in einer Subklasse vorkommen

Beispiel

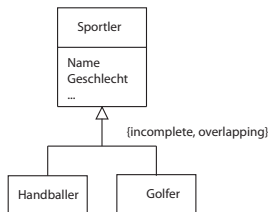


Disjunktheitseinschränkung

Die **Disjunktheitseinschränkung** bestimmt, ob eine Entität Element mehrerer Subklassen sein darf oder nicht

- disjoint $\hat{=}$ Eine Entität kann höchstens in *einer* Subklasse vorkommen
- overlapping $\hat{=}$ Eine Entität kann in *mehreren* Subklassen vorkommen

Beispiel



Inhalt

- Vorgehensweise und ein Beispiel
 - Phasen im Datenbank-Entwurf
 - Ein Beispiel
- Entity-Relationship-Modell
 - Entitäten, Attribute, Assoziationen
 - Erweitertes Entity-Relationship-Modell
 - Design-Prinzipien
- Schema-Entwurf
 - Regeln für das ER-Modell
 - Transformation des erweiterten ER-Modells
 - Rezept für den Datenbank-Entwurf

Design-Prinzipien

- 1 Treue Abbildung des Anwendungsgebiets:
Entitätstypen, Attribute sowie Assoziationen und Multiplizitäten sollten die Gegebenheiten des Anwendungsgebiets korrekt wiedergeben.
- 2 Redundanz vermeiden:
Merkmale und Beziehungen, die sich aus bereits modellierten Elementen herleiten lassen, sollten nicht explizit modelliert werden, allenfalls als abgeleitete Attribute
- 3 Einfache und angemessene Modelle konstruieren:
Modelle sollten relevante Fakten über das Anwendungsgebiet widerspiegeln und keine technischen Konstrukte enthalten, die später für die Implementierung benötigt werden.

Das angemessene Element wählen

- Entitätstyp vs. Attribut:

Entitätstyp \triangleq die interessierenden „Dinge“

Attribut \triangleq ihre Eigenschaften

Beispiel: Person, Adresse als Attribut oder eigenen Entitätstyp?

- Entitätstyp vs. Assoziation:

Entitätstyp \triangleq eigenständige unabhängige Identifikation
(starker Entitätstyp)

Assoziation \triangleq definiert durch beteiligte Assoziationstypen

Beispiel: Angestellter – Firma, Anstellung als
Assoziation(sklasse) oder eigenen Entitätstyp?

Inhalt

- Vorgehensweise und ein Beispiel
 - Phasen im Datenbank-Entwurf
 - Ein Beispiel
- Entity-Relationship-Modell
 - Entitäten, Attribute, Assoziationen
 - Erweitertes Entity-Relationship-Modell
 - Design-Prinzipien
- Schema-Entwurf
 - Regeln für das ER-Modell
 - Transformation des erweiterten ER-Modells
 - Rezept für den Datenbank-Entwurf

Vorgehen beim Schema-Entwurf

Transformation des Entity-Relationship-Modells in ein Datenbankschema im relationalen Modell

Systematisches Vorgehen nach bestimmten Regeln
plus

Ergänzung des ER-Modells um weitere Angaben

Regel für Entitätstypen

- Entitätstyp ergibt Tabelle mit allen einfachen und einwertigen Attributen
- Von zusammengesetzten Attributen werden die einfachen Komponenten als Attribute übernommen
- Datentypen werden übernommen bzw. ergänzt

1. Regel für Entitätstypen

Entitätstyp \Rightarrow Tabelle

Regel für Entitätstypen

Beispiel

Angestellter
PNr
Name
Vorname
Nachname
Geburtsdatum
Telefon[1..3]
/Alter



Angestellter
PNr: dec(6)
Vorname: varchar(40)
Nachname: varchar(40)
Geburtsdatum: date

Regel über Primärschlüssel

- Hat der Entitätstyp einen Primärschlüssel, wird er übernommen
- Bei schwachen Entitätstypen wird der Primärschlüssel des bestimmenden Entitätstyps übernommen und zu einem Teil des Primärschlüssels
- Ansonsten: Primärschlüssel aus Attributen oder „künstlichen“ Primärschlüssel einführen

2. Regel für Primärschlüssel

Tabelle bekommt Primärschlüssel

Regel für Primärschlüssel

Beispiel

Angestellter
PNr
Name
Vorname
Nachname
Geburtsdatum
Telefon[1..3]
/Alter



Angestellter
PNr: dec(6) {PK}
Vorname: varchar(40)
Nachname: varchar(40)
Geburtsdatum: date

Regel für mehrwertige Attribute

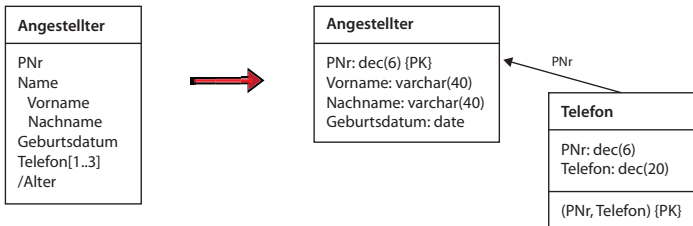
- Mehrwertige Attribute werden zu einer eigenen Tabelle
- Die Zuordnung geschieht über einen Fremdschlüssel
- Oft werden mehrwertige Attribute wie ein schwacher Entitätstyp behandelt

3. Regel für mehrwertige Attribute

Mehrwertiges Attribut \Rightarrow Tabelle samt Fremdschlüsselbeziehung

Regel für mehrwertige Attribute

Beispiel



Regel für 1-1-Assoziationen

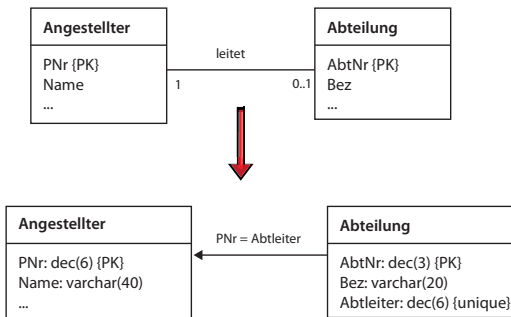
- Bei Multiplizität 1 - 1:
Verschmelzen der beiden Entitätstypen
- Bei Multiplizität 1 - 0..1:
Fremdschlüsselbeziehung zwischen den beiden zugehörigen Tabellen
- Bei Multiplizität 0..1 - 0..1:
Fremdschlüsselbeziehung zwischen den beiden zugehörigen Tabellen

4. Regel für 1-1-Assoziationen

1-1-Assoziation \Rightarrow Fremdschlüsselbeziehung

Regel für 1-1-Assoziationen

Beispiel



Regel für 1-n-Assoziationen

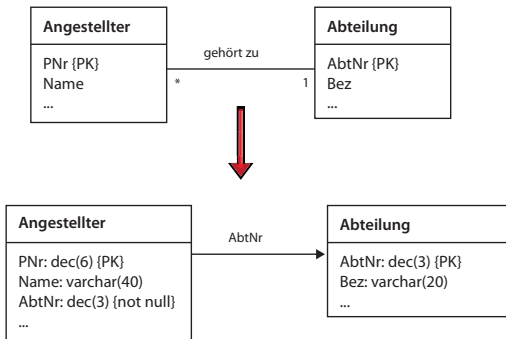
- Bei Multiplizität 1 - 1..*:
Fremdschlüsselbeziehung, Fremdschlüssel not null
- Bei Multiplizität 1 - *:
Fremdschlüsselbeziehung, Fremdschlüssel not null
- Bei Multiplizität 0..1 - 1..*:
Fremdschlüsselbeziehung, Fremdschlüssel darf auch null sein
- Bei Multiplizität 0..1 - *:
Fremdschlüsselbeziehung, Fremdschlüssel darf auch null sein

5. Regel für 1-n-Assoziationen

1-n-Assoziation \Rightarrow Fremdschlüsselbeziehung

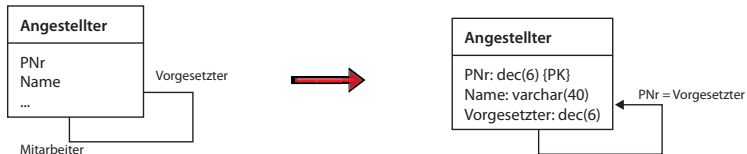
Regel für 1-n-Assoziationen

Beispiel



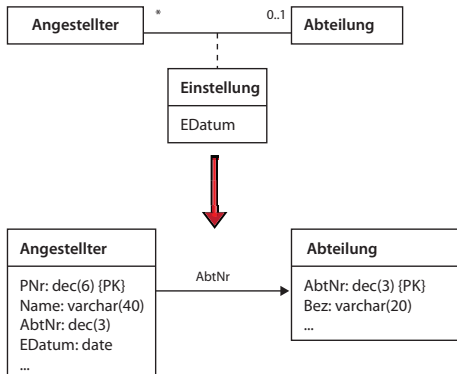
Sonderfall reflexive Assoziation

Beispiel



Sonderfall Assoziationsklasse bei 1 - n

Beispiel



Regel für n-m-Assoziationen

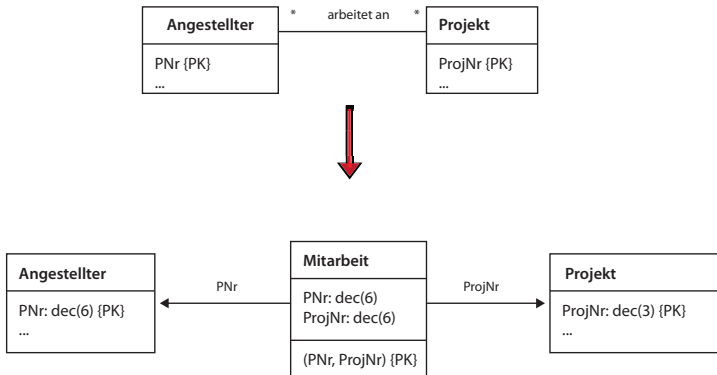
- Zwischentabelle mit 2 Fremdschlüsselbeziehungen zu den beiden beteiligten Tabellen
- Fremdschlüssel werden zusammengesetzter Primärschlüssel der Zwischentabelle

6. Regel für n-m-Assoziationen

n-m-Assoziation \Rightarrow Zwischentabelle mit 2 Fremdschlüsselbeziehungen

Regel für n-m-Assoziationen

Beispiel



Regel für n-stellige Assoziationen

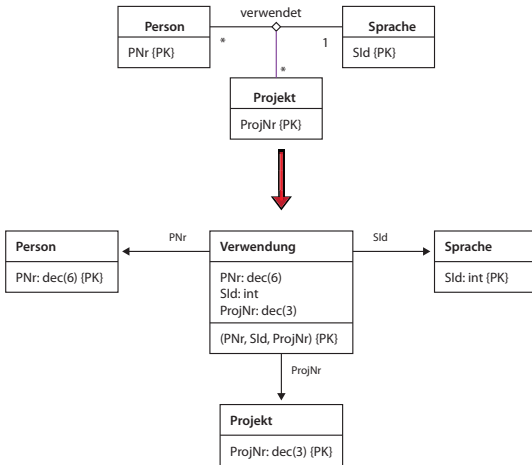
Eine n-stellige Assoziation ergibt eine Zwischentabelle mit n Fremdschlüsselbeziehungen

7. Regel für n-m-Assoziationen

n-stellige Assoziation \Rightarrow Zwischentabelle mit n Fremdschlüsselbeziehungen

Regel für n-stellige Assoziationen

Beispiel



Regel für Indexe

Zusätzlich zum Entwurf der Tabellen muss man sich überlegen, welche typischen und häufigen Zugriffspfade es für die Daten gibt.

Bei Primärschlüsseln und Eindeutigkeitsbedingungen richten die DBMS in der Regel automatisch einen Index ein.

Beispiel

```
alter table Angestellter  
  add constraint idx_name unique (Nachname, Vorname)  
  
create index idx_name on Angestellter(Nachname, Vorname)
```

8. Regel für Eindeutigkeit und Indexe

Typischer Suchpfad \Rightarrow Eindeutigkeitsbedingung oder Index

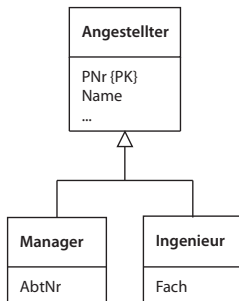
Inhalt

- Vorgehensweise und ein Beispiel
 - Phasen im Datenbank-Entwurf
 - Ein Beispiel
- Entity-Relationship-Modell
 - Entitäten, Attribute, Assoziationen
 - Erweitertes Entity-Relationship-Modell
 - Design-Prinzipien
- Schema-Entwurf
 - Regeln für das ER-Modell
 - Transformation des erweiterten ER-Modells
 - Rezept für den Datenbank-Entwurf

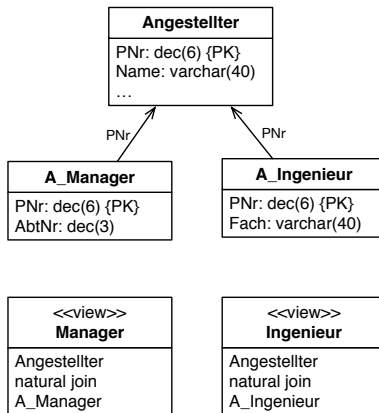
Transformation der Super-Subklassen-Beziehung

Für die Super-Subklassen-Beziehung gibt es mehrere Varianten, die wir an einem Beispiel betrachten:

Beispiel



Variante 1: Eine Tabelle pro Entitätstyp sowie geeignete Views



Variante 1: Eine Tabelle pro Entitätstyp sowie geeignete Views

```
create table Angestellter(  
    PNr dec(6) primary key,  
    Name varchar(40) not null,  
    ...  
);
```

```
create table A_Manager(  
    PNr dec(6) primary key references Angestellter(PNr),  
    AbtNr dec(3)  
);
```

```
create table A_Ingenieur(  
    PNr dec(6) primary key references Angestellter(PNr),  
    Fach varchar(40)  
);
```

Views für Variante 1

```
create view Manager as  
    select * from Angestellter natural join A_Manager;  
  
create view Ingenieur as  
    select * from Angestellter natural join A_Ingenieur;
```

Variante 2: Eine Tabelle pro Subklasse sowie geeignete View

Manager
PNr: dec(6) {PK}
Name: varchar(40)
...
AbtNr: dec(3)

Ingenieur
PNr: dec(6) {PK}
Name: varchar(40)
...
Fach: varchar(40)

<<view>> Angestellter
PNr, Name, ... from Manager union PNr, Name, ... from Ingenieur

Variante 2: Eine Tabelle pro Subklasse sowie geeignete View

Voraussetzung: Die Superklasse ist abstrakt, d.h. die Subklassenbildung ist „complete“

```
create table Manager(  
    PNr dec(6) primary key,  
    Name varchar(40) not null,  
    AbtNr dec(3)...  
);
```

```
create table Ingenieur(  
    PNr dec(6) primary key,  
    Name varchar(40) not null,  
    Fach varchar(40)  
);
```

View für Variante 2

```
create view Angestellter as  
  select PNr, Name from Manager  
union  
  select PNr, Name from Ingenieur;
```

Variante 3: Eine Tabelle für alle Entitätstypen sowie geeignete Views

Ang
PNr: dec(6) {PK} Name: varchar(40) ... AbtNr: dec(3) {null} Fach: varchar(40) {null} Typ: char(3) {in ('Mit', 'Mgr', 'Ing')}

<<view>> Angestellter
PNr, Name, ... from Ang

<<view>> Manager
PNr, Name, ..., AbtNr from Ang where Typ = 'Mgr'

<<view>> Manager
PNr, Name, ..., Fach from Ang where Typ = 'Ing'

Variante 3a: Eine Tabelle für alle Entitätstypen sowie geeignete Views

Voraussetzung: Die Subklassenbildung ist „disjoint“

```
create table Ang(  
  PNr dec(6) primary key,  
  Name varchar(40) not null,  
  AbtNr dec(3),  
  Fach varchar(40),  
  Typ char(3) check (Typ in ('Mit', 'Mgr', 'Ing'))  
    -- gibt den Typ an  
);
```

Views für Variante 3a

```
create view Angestellter as  
  select PNr, Name from Ang;
```

```
create view Manager as  
  select PNr, Name, AbtNr from Ang where Typ = 'Mgr';
```

```
create view Ingenieur as  
  select PNr, Name, Fach from Ang where Typ = 'Ing';
```

Variante 3b: Eine Tabelle für alle Entitätstypen mit Flags sowie geeignete Views

```
create table Ang(  
  PNr dec(6) primary key,  
  Name varchar(40) not null,  
  AbtNr dec(3),  
  Fach varchar(40),  
  AngFlag boolean not null,  
  MgrFlag boolean not null,  
  IngFlag boolean not null  
);
```

Views für Variante 3b

```
create view Angestellter as
  select PNr, Name from Ang where AngFlag = true;

create view Manager as
  select PNr, Name, AbtNr from Ang where MgrFlag = true;

create view Ingenieur as
  select PNr, Name, Fach from Ang where IngFlag = true;
```

Inhalt

- Vorgehensweise und ein Beispiel
 - Phasen im Datenbank-Entwurf
 - Ein Beispiel
- Entity-Relationship-Modell
 - Entitäten, Attribute, Assoziationen
 - Erweitertes Entity-Relationship-Modell
 - Design-Prinzipien
- Schema-Entwurf
 - Regeln für das ER-Modell
 - Transformation des erweiterten ER-Modells
 - Rezept für den Datenbank-Entwurf

Rezept Teil A: ER-Modell entwickeln

A1 Finde Entitätstypen

= „die Dinge, über die wir reden“

A2 Finde Attribute

= „was wir über diese Dinge sagen“

A3 Finde Beziehungen/Assoziationen

= „was die Dinge miteinander zu tun haben“

A4 Finde Multiplizitäten der Assoziationen

= „wieviele Dinge stehen zu einander in Beziehung“

⇒ Entity-Relationship-Diagramm + Glossar + Beschreibungen

Rezept Teil B: Datenbankschema entwickeln

B1 Finde Tabellen

Entitätstyp \rightarrow Tabelle

n-m-Assoziation \rightarrow Zwischentabelle

B2 Finde Primär- und Fremdschlüssel

Entitätstyp \rightarrow Primärschlüssel

1-n-Assoziation \rightarrow Fremdschlüssel

B3 Vervollständige Tabellen

Datentypen, Integritätsbedingungen

B4 Bilde Indexe

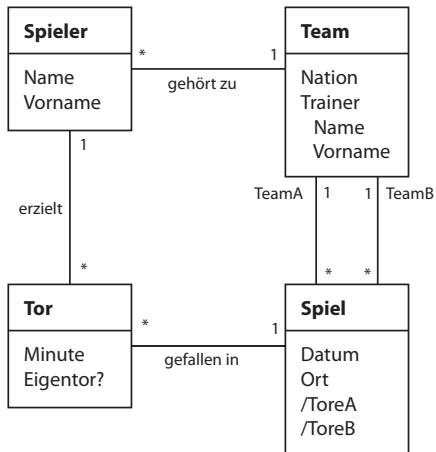
durch Analyse der typischen Zugriffspfade

\Rightarrow Datenbankschema in Diagrammform, daraus sind die DDL-Anweisungen unmittelbar ablesbar

Noch ein Beispiel: Fußball-WM

- Teams, gestellt von Nationen, haben einen Trainer
- Teams haben Spieler
- Teams tragen Spiele gegeneinander aus
- Spieler schießen in diesen Spielen (manchmal) Tore
- Ab und an geht der Ball auch ins eigene Tor
- Spiele enden mit einem Ergebnis

Fußball-WM ER-Diagramm



Fußball-WM Datenbank-Schema

