Inhaltsübersicht

Diese Seite bleibt aus technischen Gründen frei.

Verzeichnis der Abbildungen

Lernziele

Nach dem Durcharbeiten der vorliegenden Kurseinheit sollen Sie sich mit dem Gegenstand der Datenmodellierung sowie mit den Grundelementen von Datenmodellen soweit vertraut gemacht haben, dass sie den Zweck und den besonderen Stellenwert von Datenmodellen in der betrieblichen Datenverarbeitung charakterisieren können. Außerdem sollen Sie mit

- dem hierarchischen Datenmodell,
- dem netzwerkartigen Datenmodell und
- dem klassischen Relationenmodell

Datenmodelle

drei grundlegende Ansätze der Datenmodellierung einschließlich ihrer spezifischen Vorteile und Nachteile nicht nur kennengelernt haben, sondern im Einzelnen auch begründen können, warum der relationale Ansatz den anderen Modellierungsansätzen vorzuziehen ist.

Darüber hinaus sollen Sie sich mit einem auf

- der Unterscheidung von globalen und lokalen Attributen,
- der Verwendung von Relationen in dritter Normalform,
- der Vorgabe von statischen oder dynamischen Wertebereichen für bestimmte Arten von Attributen,
- der ausschließlichen Verwendung von hierarchischen Beziehungen zwischen Relationen und
- der präzisen Darstellung von Ober- und Untermengenbeziehungen zwischen Relationen

Eigenschaften eines erweiterten Relationenmodells

beruhenden erweiterten Relationenmodell intensiver auseinandergesetzt haben. Einerseits sollen Sie die wesentlichen Eigenschaften dieses erweiterten Relationenmodells benennen und erläutern können, und andererseits sollen Sie auch ein weitergehendes, verhaltensorientiertes Lernziel erreicht haben: Sie sollen in der Lage sein, einen gegebenen Ausschnitt aus der Datenwelt eines konkreten Unternehmens in ein erweitertes Relationenmodell abzubilden und damit eine konzeptionelle Datenbasis für die Entwicklung von Datenbankanwendungen zu schaffen.

Diese Seite bleibt aus technischen Gründen frei.

3 Logische Datenorganisation

logische Gesamtsicht

Unter logischer Datenorganisation versteht man die Modellierung der Datenwelt eines gegebenen Realitätsausschnitts unter dem Blickwinkel der logischen Gesamtsicht. Das Ziel der logischen Datenorganisation besteht letztlich in der Formulierung eines konzeptionellen Modells, welches die verarbeitungsrelevanten Phänomene des Realitätsausschnitts durch Datenobjekte und Beziehungen zwischen Datenobjekten abbildet. Ein konzeptionelles Modell ist in der Regel auf ein konkretes DBVS ausgerichtet. Bei der Modellierung sind daher Restriktionen zu berücksichtigen, die sich aus dem verwendeten DBVS ergeben. Solche Restriktionen betreffen beispielsweise die Art der verwendbaren Beziehungen zwischen Datenobjekten.

semantisches Datenmodell und DBVS-gestütztes konzeptionelles Modell

Allgemeinere Datenmodelle erhält man, falls bei der Modellierung die durch das DBVS gesetzten Restriktionen außer Acht bleiben. Mit allgemeinen Datenmodellen verbinden sich zwei Zwecke: Einerseits sollen sie die Datenwelt eines Realitätsausschnitts semantisch vollständig - also mit vollständigem Bedeutungsgehalt - wiedergeben; man bezeichnet solche Modelle daher auch als semantische Datenmodelle. Andererseits bilden sie hilfreiche Zwischenstufen auf dem Weg zu einem DBVS-gestützten konzeptionellen Modell. Wie die Abb. 3.1 zeigt, umschließt die logische Datenorganisation beide Modellformen, allgemeine bzw. semantische und DBVS-gestützte konzeptionelle Datenmodelle.

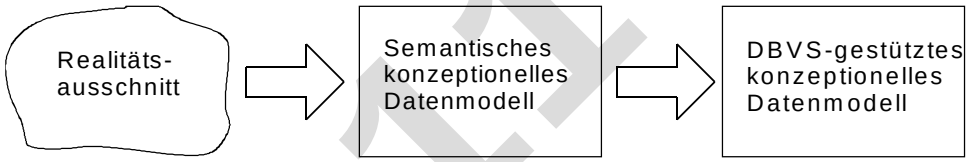


Abb. 3.1. Datenmodelle in der logischen Datenorganisation.

Semantische Datenmodelle zeichnen sich durch eine semantische Reichhaltigkeit aus, die nicht durch DBVS-Restriktionen eingeschränkt ist. Dagegen sind DBVS-gestützte, d.h. durch DBVS verarbeitbare Datenmodelle, meist semantisch ärmer. Sprachen und Werkzeuge zur Entwicklung und Beschreibung von semantischen Datenmodellen eignen sich somit gleichermaßen zur Darstellung DBVS-gestützter Datenmodelle. Im Folgenden wird daher nur dann zwischen semantischen und DBVS-gestützten Datenmodellen unterschieden, wenn dies erforderlich ist. Ansonsten wird allgemein von Datenmodellen die Rede sein. Ausdrücklich sei darauf hingewiesen, dass es dabei stets um die logische Gesamtsicht und somit um konzeptionelle Modelle geht.

Entwicklung von Datenmodellen

In der Vergangenheit unterlagen die in Datenbanksystemen verwendeten Datenmodelle einem erheblichen Wandel. Auf hierarchische Datenmodelle folgten rasch netzwerkartige Modelle. Und die heute verwendeten DBVS basieren fast durchweg auf Relationenmodellen. Die vorliegende Kurseinheit befasst sich schwerpunktmäßig mit Relationenmodellen, geht aber auch auf hierarchische und netzwerkartige Datenmodelle ein.

Inhalt der Kurseinheit

Einige Grundlagen der Datenmodellierung behandelt das Kapitel 3.1, nämlich Grundelemente von Datenmodellen und das Entity Relationship-Modell. Letzteres ist deshalb von Interesse, weil es einen der Ausgangspunkte für die Entwicklung von erweiterten Relationenmodellen darstellt. Zwei klassische Datenmodelle werden in den beiden folgenden Kapiteln vorgestellt, in Kapitel 3.2 das hierarchische Datenmodell und in Kapitel

3.3 das netzwerkartige Datenmodell. Das Kapitel 3.4 geht auf Relationenmodelle ein. Unterschieden wird zwischen dem klassischen Relationenmodell und erweiterten Modellen. Erweiterte Relationenmodelle verbinden und erweitern die dem Entity Relationship-Modell und dem klassischen Relationenmodell zugrundeliegenden Konzepte zu leistungsfähigen Modellierungsinstrumenten. Über die reine Datenmodellierung geht das letzte Kapitel 3.5 hinaus. Es ordnet die Datenmodellierung in den logischen Datenbankentwurf ein, der neben Daten auch Konsistenzbedingungen für Daten und die Manipulation von Daten zum Gegenstand hat. Außerdem wird in diesem Kapitel die Datenmodellierung in einen Software Life Cycle eingebettet, der auf die Entwicklung datenbankgestützter Anwendungssysteme ausgerichtet ist.

3.1 Datenmodellierung

3.1.1 Grundelemente von Datenmodellen

In nahezu allen Datenmodellen treten bestimmte Elemente auf, die hier als Grundelemente bezeichnet seien. Zu den Grundelementen gehören Entitäten, Beziehungen zwischen Entitäten, Attribute zur Charakterisierung von Entitäten und Schlüssel. Nachfolgend werden diese Grundelemente behandelt.

a) Entitäten

Entitäten benötigt man für die datenmäßige Darstellung von Phänomenen eines Realitätsausschnitts. Sie sind gleichsam Abbilder dieser Phänomene in der Datenwelt. Die abgebildeten Phänomene können realer Natur, z.B. Kunden, Lieferanten, Artikel, Maschinen usw., oder gedanklicher Natur, z.B. Kosten, Preise, Termine usw., sein. Für den Entitätsbegriff folgt damit:

Begriff der Entität

> Eine Entität (engl. entity) ist ein Element der Datenwelt, welches ein reales oder ein gedankliches Einzelphänomen in einem betrachteten Realitätsausschnitt repräsentiert.

Beispiele für Entitäten sind der Kunde "ALFRED HUPFER KG", der Fertigungsauftrag "H 240543", das Auslieferungsfahrzeug mit dem Kennzeichen "OHZ-ME 72" usw. Jede dieser Entitäten stellt ein individuelles Exemplar aus einer Klasse von Exemplaren dar.

Objektklassen

Ein zentrales Modellierungskonzept besteht in der Bildung von Klassen von Objekten. Man denke etwa an die Klasse der ganzen Zahlen, die bekanntlich durch den Datentyp INTEGER beschrieben wird. In vollkommen anloger Weise lassen sich Entitäten zu Klassen von Entitäten zusammenfassen. Klassen von Entitäten werden in der Literatur meist als Entitätsmengen bezeichnet. Zum Begriff der Entitätsmenge:

> Eine Entitätsmenge (engl. entity set) fasst alle Entitäten zusammen, die durch gleiche Merkmale, nicht notwendigerweise aber durch gleiche Merkmalsausprägungen, charakterisiert werden.

Beispiele für Entitätsmengen sind *Kunden*, *Fertigungsaufträge*, *Auslieferungsfahrzeuge* usw. Zur Charakterisierung der Entitätsmenge *Auslieferungsfahrzeuge* mögen die Merkmale *Kennzeichen*, *Motorleistung*, *Ladegewicht* und *Höchstgewicht* dienen. Auf die Entitäten dieser Mengen treffen zwar durchweg die genannten Merkmale zu, jedoch können die Merkmalsausprägungen bzw. -werte von Entität zu Entität variieren.

Begriff der Entitätsmenge

Zur grafischen Darstellung von Entitätsmengen werden Rechtecke verwendet. Abb. 3.2 zeigt einige Beispiele.

Darstellung von Entitätsmengen

Abb. 3.2. Grafische Darstellung von Entitätsmengen.

Sinnvoll ist die Bildung von Entitätsmengen deshalb, weil alle individuellen Exemplare einer Entitätsmenge in prinzipiell gleicher Weise behandelt oder manipuliert werden. Statt viele individuelle Entitäten zu betrachten, genügt es daher, sich auf der "Typebene" mit Entitätsmengen zu beschäftigen. Es gelten hier die gleichen Überlegungen, die im Kurs "Algorithmen und Datenstrukturen" bei der Einführung des Begriffs des Datentyps angestellt wurden.

Betrachtung auf Typebene

Alle in Abb. 3.2 angegebenen Entitätsmengen weisen eine Gemeinsamkeit auf. Sie enthalten ausschließlich Entitäten, die nicht gleichzeitig in anderen Entitätsmengen auftreten. Entitätsmengen mit dieser Eigenschaft heißen disjunkt.

Nicht disjunkte Entitätsmengen überlappen sich. Und für überlappende Entitätsmengen kann man immer eine Entitätsmenge angeben, welche die überlappenden Mengen umfasst. Eine umfassende Entitätsmenge kann sich allerdings auch aus nicht überlappenden Entitätsmengen zusammensetzen.

Die genannten Arten von Entitätsmengen seien begrifflich wie folgt präzisiert:

disjunkte

Zwei Entitätsmengen *A* und *B* heißen disjunkt, wenn keine Entität existiert, die sowohl zu *A* als auch zu *B* gehört.

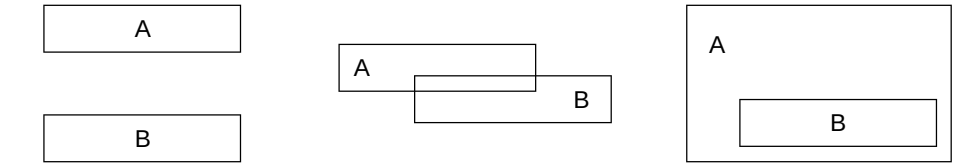
überlappende

Zwei Entitätsmengen *A* und *B* überlappen sich, wenn zumindest eine Entität existiert, die sowohl zu *A* als auch zu *B* gehört.

umfassende Entitätsmengen

Eine Entitätsmenge *A* umschließt eine Entitätsmenge *B* genau dann, wenn sämtliche in *B* auftretenden Entitäten auch zu *A* gehören.

Abbildung 3.3 veranschaulicht diese Definitionen.



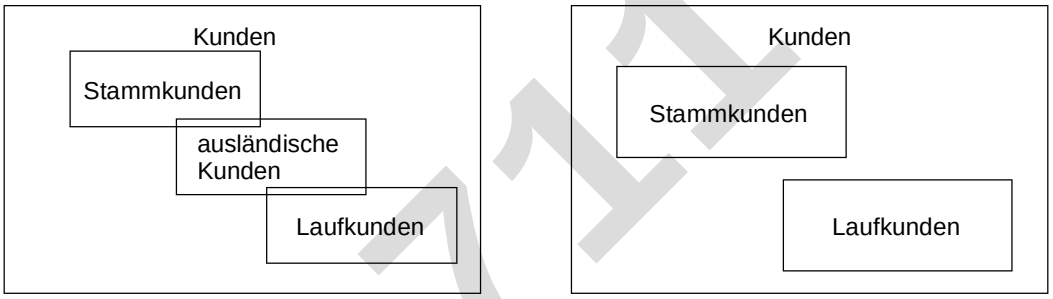
a) Disjunkte Entitätsmengen

b) Überlappende Entitätsmengen

c) Entitätsmenge A umfaßt Menge B

Abb. 3.3. Disjunkte, überlappende und umfassende Entitätsmengen.

Bei der Definition von Entitätsmengen bestehen gewisse Freiheitsgrade, die von den jeweiligen Gegebenheiten abhängen. Zur Verdeutlichung der Freiheitsgrade im Modellierungsprozess sei das in Abb. 3.4 angegebene Beispiel betrachtet.



a) Aufspaltung einer Entitätsmenge mit Überlappung

b) Aufspaltung einer Entitätsmenge ohne Überlappung

Abb. 3.4. Freiheitsgrade im Modellierungsprozess.

Aufspaltung von Entitätsmengen

Beide in Abb. 3.4 gezeigten Fälle entstehen durch die Detaillierung der Entitätsmenge *Kunden* in speziellere Entitätsmengen. Im Fall a) überlappen sie sich und im Fall b) nicht. Ob eine Überlappung vorzusehen ist, hängt von der Bedeutung bestimmter Merkmalseigenschaften ab. Während bei der Aufspaltung ohne Überlappung lediglich die Zugehörigkeit zur Stamm- oder Laufkundschaft interessiert, findet im anderen Fall auch die Auslandszugehörigkeit Berücksichtigung.

Man beachte, dass die Entitätsmenge *Kunden* noch auf viele andere Arten aufgespalten werden könnte. Beispielsweise unter Berücksichtigung von Umsatzklassen, Absatzregionen, Liefer- und Zahlungsmodalitäten usw. Die Frage, welche der möglichen Spezialisierungen angezeigt ist und ob überhaupt eine Spezialisierung vorzunehmen ist, lässt sich anhand der die Entitätsmenge *Kunden* manipulierenden Verarbeitungsprozesse entscheiden. Existieren beispielsweise Prozesse, die ausschließlich auf Stammkunden zugreifen, so könnte die Spezialisierung in eine Entitätsmenge Stammkunden sinnvoll sein.

Übungsaufgabe 3.1

Gegeben sei folgendes Diagramm:

Was drückt dieses Diagramm im Unterschied zu dem in Abb. 3.4. a) gezeigten Diagramm aus?

b) Beziehungen zwischen Entitäten

Zwischen Entitäten, und damit auch zwischen Entitätsmengen, können Beziehungen bestehen, die aus Verträgen, Absprachen, Bestellungen, organisatorischen Maßnahmen usw. hervorgehen. Betrachtet seien zunächst einseitige Beziehungen zwischen je zwei Entitätsmengen; sie heißen auch Assoziationen. Der Begriff der Assoziation drückt ein zahlenmäßiges Verhältnis aus:

Begriff der Assoziation

Eine Assoziation *a*(*E1*, *E2*) gibt an, wie viele Entitäten der Entitätsmenge *E2* einer beliebigen Entität der Entitätsmenge *E1* zugeordnet sein können.

Im Zeitablauf kann die mit einer Assoziation erfasste Zahl schwanken. Bezeichnet *E1* beispielsweise eine Entitätsmenge *Kunden* und *E2* eine Entitätsmenge *Artikel*, so mag bezogen auf eine Bestellperiode von einer Woche gelten *a*(*E1*, *E2*) = [0..500]. Ein beliebiger Kunde ordert in einer Bestellperiode also entweder gar keinen oder bis zu 500 Artikel. Da bei der Modellierung nicht Einzelwerte, sondern prinzipielle Zusammenhänge im Vordergrund stehen, begnügt man sich mit einer groben Einteilung in vier Assoziationstypen *A*(*E1*, *E2*). Diese Typen stellt die Abb. 3.5 vor.

vier Assoziationstypen

Bezeichnung des Assoziationstyps A (E1, E2)	Symbol	Anzahl der Entitäten in E2, die der Entität E1 zugeordnet werden können
einfach	1	genau eine
konditionell	c	keine oder eine, d.h. c=0 oder c=1
multipel	m	mindestens eine, d.h. m >= 1
multipel-konditionell	mc	keine, eine oder mehrere, d.h. mc >=0

Abb. 3.5. In der Datenmodellierung gebräuchliche Assoziationstypen.

Durch das Zusammenfassen der wechselseitigen Assoziationen zwischen zwei Entitäten gelangt man zu einer Beziehung. Für den Begriff der Beziehung gilt:

Begriff der Beziehung

> Eine Beziehung (engl. relationship) zwischen zwei Entitätsmengen *E1* und *E2* besteht aus der Assoziation *a*(*E1*, *E2*) und aus der dieser Assoziation entgegen gerichteten Assoziation *a*(*E2*, *E1*).

Treten in dieser Definition Assoziationstypen *A*(*E1*, *E2*) und *A*(*E2*, *E1*) an die Stelle der Assoziationen *a*(*E1*, *E2*) und *a*(*E2*, *E1*), so soll von Beziehungstyp die Rede sein. Zur grafischen Darstellung eines Beziehungstyps verwendet man die der Abb. 3.6 zu entnehmende Notationsform. Sie geht auf CHEN (1976) zurück.

Darstellung eines Beziehungstyps

Abb. 3.6. Grafische Darstellung eines Beziehungstyps.

Laut Abb. 3.6 wird ein Beziehungstyp grafisch durch eine Raute sowie durch Verbindungslinien zu den angesprochenen Entitätsmengen repräsentiert. Die Raute kann eine Beschreibung der Beziehung aufnehmen. An den Verbindungslinien werden die zugehörigen Assoziationstypen notiert. Man beachte, dass der Ort, an dem ein Assoziationstyp notiert wird, nämlich neben *E1* oder neben *E2*, die Assoziationsrichtung bestimmt. Einige Beispiele für Beziehungstypen sind in Abb. 3.7 zusammengestellt.

Assoziationsrichtung

Erläuterungen zur Abb. 3.7

Einige der Beziehungstypen in Abb. 3.7 seien exemplarisch erläutert:

- Bei der 1-c-Beziehung zwischen Absatzlagern und Absatzregionen geht es um die Existenz von Lagern in Regionen. Da in einer Absatzregion höchstens ein Lager vorhanden ist, aber nicht in jeder Region ein Lager vorhanden sein muss, gilt:

  Assoziation (Absatzregionen, Absatzlager) = c.

  Andererseits befindet sich ein Absatzlager in genau einer Absatzregion, also:

  Assoziation (Absatzlager, Absatzregion) = 1.

- Das Beispiel zu dem c-mc-Beziehungstyp betrifft die Verpachtung von Jagdrevieren an Pachtwillige. Da ein Revier an einen oder gemeinschaftlich an mehrere Interessenten verpachtet werden kann, wegen zu hoher Preisvorstellung eventuell aber auch unverpachtet bleibt, gilt:

  Assoziation (Reviere, Pachtwillige) = mc.

  Bei den von einer Jagdgenossenschaft zu vergebenden Revieren kommt ein Pachtinteressent höchstens einmal zum Zuge; daher folgt:

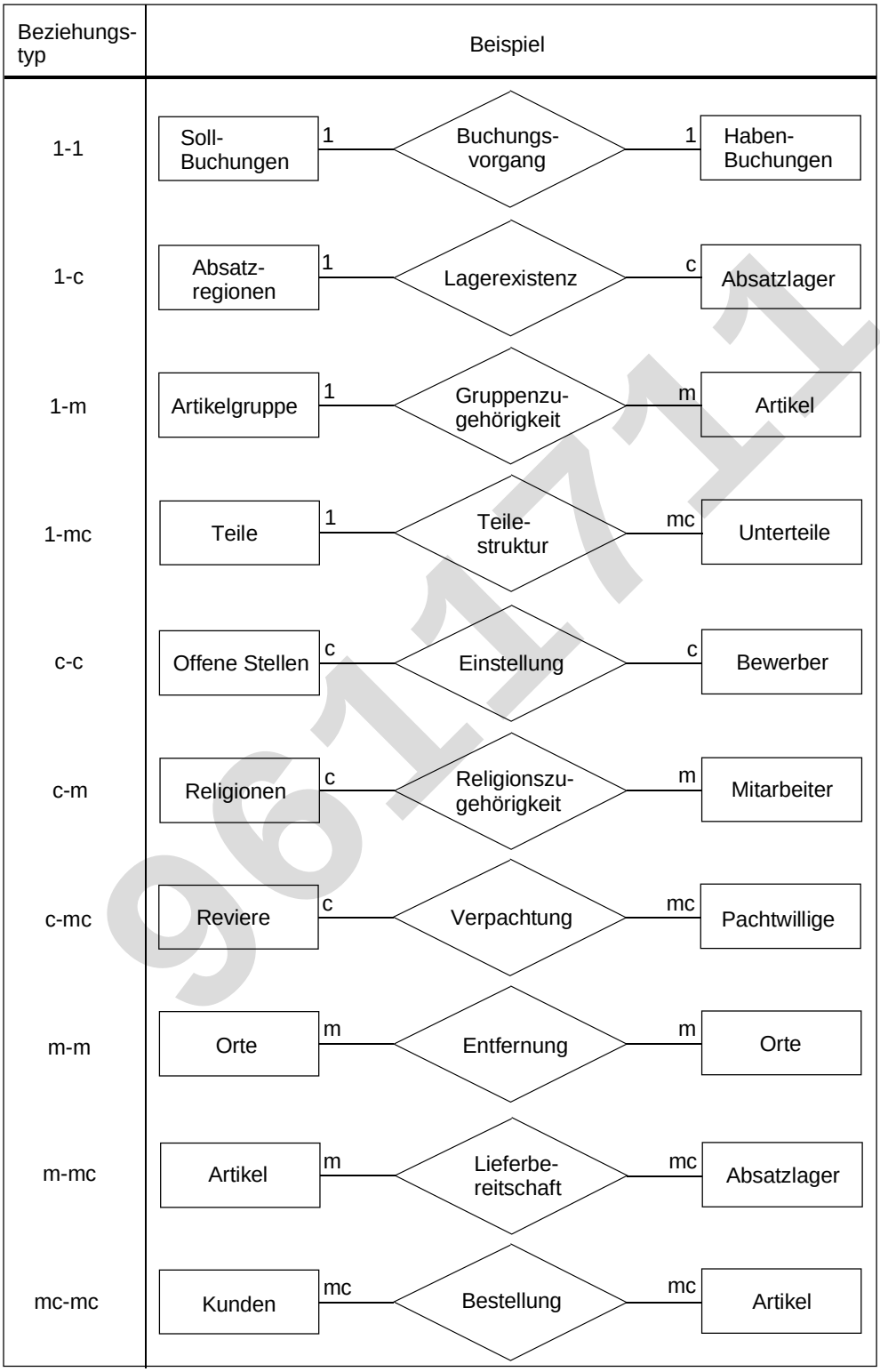
  Assoziation (Pachtwillige, Reviere) = c.

- Bestellungen von Artikeln durch Kunden sind Gegenstand der mc-mc-Beziehung. In einer Bestellperiode kann ein Kunde keinen, einen oder mehrere Artikel bestellen; also muss gelten:

  Assoziation (Kunde, Artikel) = mc.

Umgekehrt kann in einer Bestellperiode ein Artikel von keinem, von einem oder von mehreren Kunden bestellt werden; es gilt daher:

Assoziation (Artikel, Kunde) = mc.



zehn Beziehungstypen

Abb. 3.7. Beispiele für zehn Beziehungstypen.

Übungsaufgabe 3.2

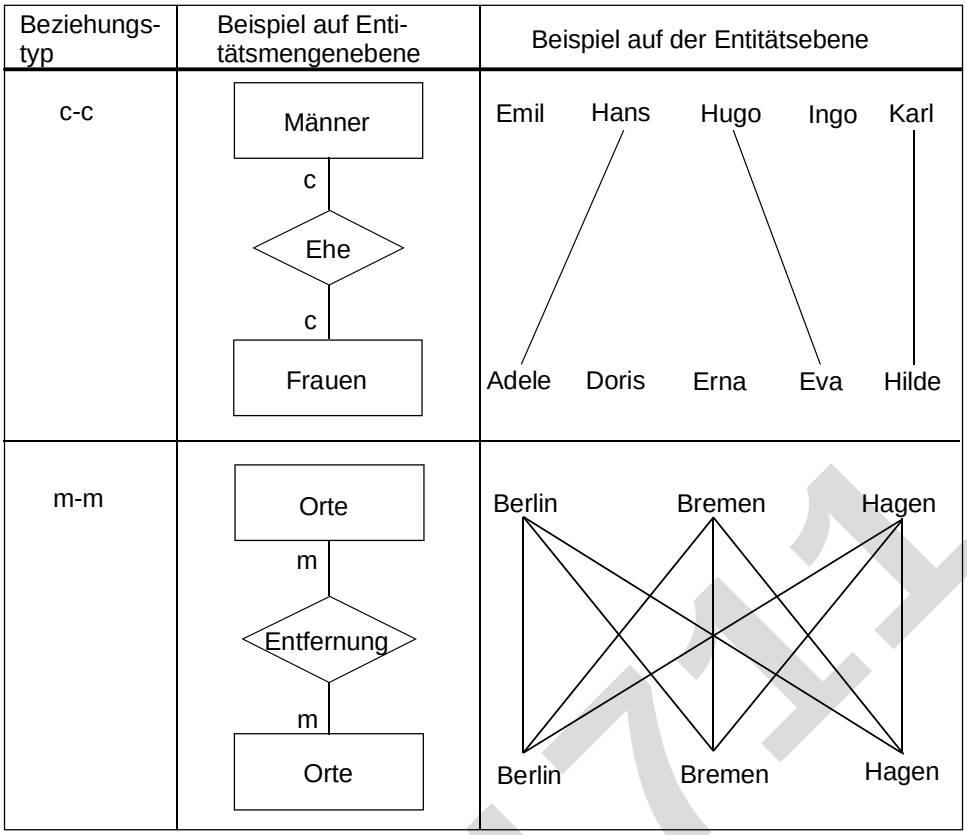
Betrachtet werde die Abb. 3.7. Erläutern Sie die zu den Beziehungstypen 1-m und c-m angegebenen Beispiele.

In Abb. 3.7 wurden nicht zufällig zehn Beziehungstypen angegeben. Vielmehr handelt es sich um genau die zehn Beziehungstypen, die durch Kombination der in Abb. 3.5 eingeführten Assoziationstypen gebildet werden können.

Übungsaufgabe 3.3

Begründen Sie, warum sich durch die Kombination der eingeführten vier Assoziationstypen genau zehn Beziehungstypen ergeben.

Beziehungen lassen sich nicht nur auf der Ebene von Entitätsmengen, sondern auch auf der Ebene individueller Entitäten darstellen. Verdeutlicht sei dies an zwei Beispielen, die der Abb. 3.8 zu entnehmen sind.



Darstellung von Beziehungen auf der Ebene von Entitäten

Abb. 3.8. Beziehungen auf der Ebene von Entitätsmengen und auf der Ebene von Entitäten.

Wie die Beispiele in Abb. 3.8. zeigen, wird eine Beziehung zwischen zwei Entitäten grafisch durch eine Verbindungslinie ausgedrückt. Eine solche Verbindung sei als Einzelbeziehung bezeichnet.

Einzelbeziehung

c) Attribute

Über das Wesen der abgebildeten Phänomene machen Entitätsmengen und Beziehungen nur grobe Aussagen. Interessierende Eigenschaften von Phänomenen lassen sich durch Attribute erfassen, die man den Entitätsmengen und Beziehungen zuordnet.

Attribute zur Charakterisierung von Entitäten

Ein Attribut (engl. attribute) beschreibt eine bestimmte Eigenschaft, die sämtliche Entitäten einer Entitätsmenge oder sämtliche Einzelbeziehungen einer Beziehung aufweisen.

Begriff des Attributs

Relevante Attribute der Entitätsmenge *Kunden* sind beispielsweise *Kundennummer*, *Name*, *Adresse*, *Umsatz* usw. Und die Beziehung *Bestellung* wird inhaltlich durch die Attribute *Kundennummer*, *Artikelnummer*, *Bestellmenge*, *Datum* usw. charakterisiert.

Wie bei einer Variablen ist bei einem Attribut zwischen seinem Namen und seinem Wert zu unterscheiden. Während der Name ein Attribut benennt und identifiziert, gibt der Wert die konkrete Ausprägung des Attributs für eine bestimmte Entität an. Welche diskreten Werte ein Attribut annehmen kann, legt sein Wertebereich fest.

Name und Wert eines Attributs

Wertebereich eines Attributs

Der Wertebereich (engl. domain) eines Attributs besteht aus der Menge der Datenwerte, die das Attribut für die Entitäten der zugrundeliegenden Entitätsmenge annehmen kann.

Die eingeführten Begriffe seien an einem einfachen Beispiel demonstriert. Es ist in Abb. 3.9 dargestellt.

Attribute und Attributwerte

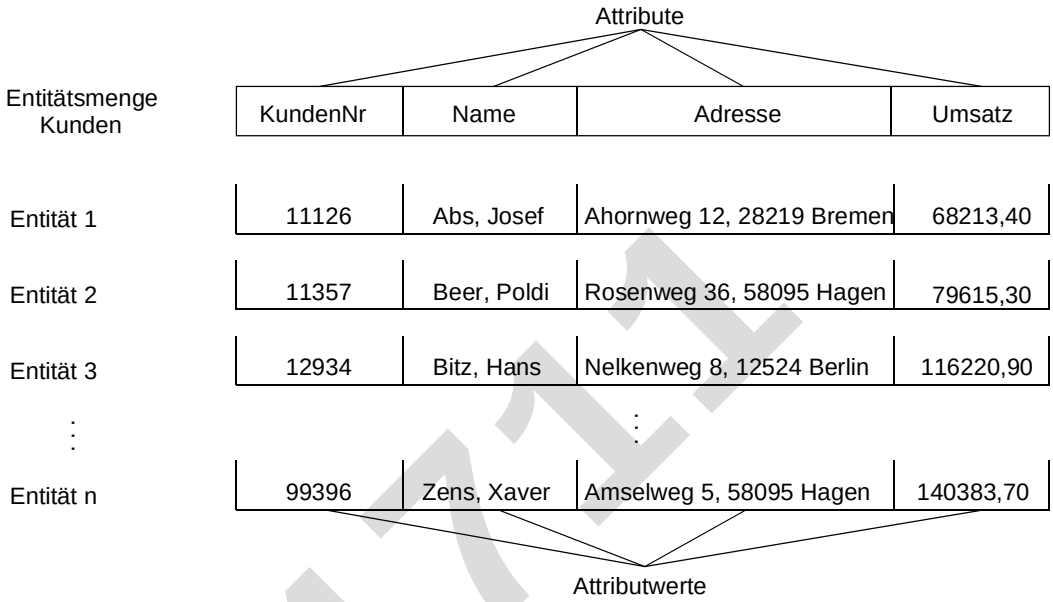


Abb. 3.9. Attribute und Attributwerte einer Entitätsmenge Kunde.

Für die einzelnen Attribute der Entitätsmenge *Kunde* können die Wertebereiche beispielsweise wie folgt vorgegeben sein:

Beispiele für Wertebereiche

Wertebereich von *KundenNr*: [11111..99999],
Wertebereich von *Name*: Zeichenkette der Länge 20,
Wertebereich von *Adresse*: Zeichenkette der Länge 40,
Wertebereich von *Umsatz*: [000000,00..999999,00].

Demnach repräsentiert z.B. der Wertebereich des Attributs *Umsatz* eine Teilmenge der mit dem Standardtyp REAL darstellbaren Zahlenwerte. Während *Umsatz* ein "elementares" Attribut ist, das sich nicht weiter in aussagefähige Submerkmale zerlegen lässt, handelt es sich bei *Adresse* um ein komplexes Attribut. Seine Zerlegung in elementare (Sub-) Attribute behandelt die Übungsaufgabe 3.4.

Übungsaufgabe 3.4

Betrachtet werde das Attribut *Adresse* der in Abb. 3.9 vorgestellten Entitätsmenge *Kunden*. Geben Sie die Zerlegung dieses Attributs in aus logischer Sicht nicht weiter zerlegbare, elementare (Sub-)Attribute sowie die Wertebereiche der elementaren Attribute an.

d) Identifikationsschlüssel

Entität als individuelles Exemplar

Jede Entität einer Entitätsmenge ist ein individuelles Exemplar, das sich von den übrigen Entitäten der Menge unterscheidet. Der Unterschied zwischen zwei Entitäten einer Entitätsmenge drückt sich in unterschiedlichen Attributwerten aus. Zumindest für ein Attribut müssen zwei Entitäten unterschiedliche Werte annehmen, damit sie als individuelle Exemplare unterscheidbar sind. Jede Entität einer Entitätsmenge wird dann eindeutig durch die Werte sämtlicher Attribute identifiziert. Hier stellt sich die Frage, ob nicht wenige Attribute genügen oder sogar nur ein Attribut genügt, um die Entitäten einer Menge eindeutig zu identifizieren. Die Beantwortung dieser Frage führt zu dem Begriff des Identifikationsschlüssels.

Begriff des Identifikationsschlüssels

Ein Identifikationsschlüssel (engl. identification key) besteht aus einem Attribut oder aus einer Kombination von Attributen, welche jede Entität einer Entitätsmenge eindeutig identifiziert.

als Identifikationsschlüssel geeignete Attribute

Bei der in Abb. 3.9 angegebenen Entitätsmenge *Kunde* ist beispielsweise das Attribut *KundenNr* oder die Attributkombination aus *Name* und *Adresse* grundsätzlich als Identifikationsschlüssel geeignet. Nicht als Identifikationsschlüssel eignet sich dagegen das Attribut *Umsatz*, da es für verschiedene Entitäten den gleichen Wert annehmen kann.

einfacher und zusammengesetzter Identifikationsschlüssel

Besteht ein Identifikationsschlüssel aus nur einem Attribut, so heißt er einfach. Dagegen bezeichnet man einen aus mehreren Attributen bestehenden Identifikationsschlüssel als zusammengesetzten Schlüssel (engl. concatenated key). Aus Gründen der Speicherplatzersparnis ist man bestrebt, möglichst wenig Attribute in einen zusammengesetzten Identifikationsschlüssel einzubeziehen. Ein aus der geringst möglichen Anzahl von Attributen zusammengesetzter Identifikationsschlüssel heißt minimal.

minimaler Identifikationsschlüssel

Ein minimaler Identifikationsschlüssel ergibt sich stets dann, wenn man die Entitäten einer Entitätsmenge fortlaufend durchnummeriert. Fortlaufend vergebene Nummern charakterisieren nicht das Wesen von Entitäten. Ein zur Durchnummerieren verwendetes Attribut dieser Art heißt daher auch künstlich. Beispielsweise stellt die *KundenNr* der in Abb. 3.9 genannten Entitätsmenge Kunden einen künstlichen Identifikationsschlüssel dar.

Primärschlüssel und Sekundärschlüssel

Hinzuweisen ist noch auf zwei weitere Schlüsselbegriffe, den Primärschlüssel und den Sekundärschlüssel. Ein Primärschlüssel identifiziert eindeutig die Entitäten einer Entitätsmenge und ist daher zugleich auch ein Identifikationsschlüssel. Dagegen ist ein Sekundärschlüssel ein Attribut, welches für mehrere Entitäten den gleichen Wert annehmen kann. Ein Sekundärschlüssel dient also der Zusammenfassung von Entitäten zu Teilmengen mit einer gleichen Eigenschaft. Beispielsweise könnte das Attribut *Ort* einer Entitätsmenge *Kunden* als Sekundärschlüssel Verwendung finden.

Übungsaufgabe 3.5

Betrachtet werde eine Entitätsmenge *Bestellungen* mit folgenden Attributen:

Entitätsmenge Bestellungen	KundenNr	ArtikelNr	Artikelgruppe	Menge	Datum

Geben Sie den minimalen Identifikationsschlüssel für die Entitäten dieser Entitätsmenge an. Nennen Sie für jedes Attribut den Grund für das Einbeziehen oder Nichteinbeziehen in den Identifikationsschlüssel.

3.1.2 Entity Relationship-Modell

Auf die besondere Bedeutung des Entity Relationship-Modells (ER-Modells) als Vorläufer erweiterter Relationenmodelle wurde bereits hingewiesen. CHEN (1976) griff bei der Entwicklung des ER-Modells auf mehrere Konzepte zurück. Insbesondere auf den als Entity Set Model bekannt gewordenen Vorschlag von SENKO et al. (1973), der in einem ersten Schritt die Abgrenzung von Entitäten bzw. Entitätsmengen und danach deren datenmäßige Darstellung vorsieht, sowie auf die Beschreibung der Beziehungen zwischen Entitätsmengen nach ABRIAL (1974). Die von ABRIAL eingeführten Zuordnungskardi-

nalitäten stellen zahlenmäßig ausgedrückte Assoziationen dar und bildeten den Ausgangspunkt für die Definition von Assoziations- und Beziehungstypen.
Grundlegende Elemente des ER-Modells sind:

Elemente des ER-Modells

- Entitäten, d.h. eindeutig abgrenzbare Dinge wie z.B. Personen, Abteilungen und Maschinen.
- Beziehungen (engl. relationships), d.h. Zuordnungen zwischen Entitäten wie z.B. die Beziehung *Mitarbeiter* zwischen Abteilungs- und Personenentitäten.
- Rollen (engl. roles), d.h. Funktionen, welche Entitäten in einer Beziehung erfüllen wie z.B. die Rollen *Meister*, *Geselle* und *Auszubildender* von Personen in der Beziehung *Mitarbeiter*.

ER-Diagramm

Zur grafischen Darstellung eines ER-Modells als sogenanntes Entity Relationship-Diagramm (ER-Diagramm) verwendet CHEN die bereits bekannten Symbole:

- Rechtecke für Entitätsmengen und
- Rauten für Beziehungsmengen.

Ursprünglich wurden nur 1-1-, 1-mc- und mc-mc-Beziehungen verwendet. Heute sind die in Kapitel 3.1.1 eingeführten zehn Beziehungstypen gebräuchlich (vgl. Abb. 3.7).

Ein Beispiel für ein ER-Diagramm, in dem die Beschränkung auf die ursprünglich vorgesehenen Beziehungen aufgehoben ist, zeigt die Abb. 3.10. Dargestellt wird ein Ausschnitt aus einem ER-Modell eines hypothetischen Unternehmens.

Beschreibung von Beziehungsmengen

Beziehungsmengen werden in diesem ER-Diagramm durch die Benennung der Entitätsmengen beschrieben, zwischen denen die Beziehungsmenge definiert ist. Meist liegt die Bedeutung einer Beziehung auf der Hand. So drückt beispielsweise die Beziehungsmenge *Kunden/Vertreter* die Betreuung einer Teilmenge von Kunden durch einen Vertreter aus. In bestimmten Fällen versagt diese Beschreibungsform jedoch. Beispielsweise können zwischen den Entitätsmengen *Projekte* und *Personal* die Beziehungsmengen *Projektleiter* und *Projektmitarbeiter* definiert werden. Diese Unterscheidung ist mit einer Beziehungsmenge *Projekte/Personal* nicht erfassbar. Was die Bezeichnung von Beziehungsmengen betrifft, sei daher festgehalten:

Namensregelung für Beziehungsmengen

Eine Beziehungsmenge sollte nur dann durch die Nennung eines Paars von Entitätsmengen beschrieben werden, wenn die Gefahr der Mehrdeutigkeit nicht besteht. Sollen inhaltlich verschiedene Beziehungen zwischen zwei Entitätsmenge berücksichtigt werden, so sind sie explizit einzuführen und aussagefähig zu bezeichnen.

Gelegentlich wird vorgeschlagen, Entitätsmengen mit Substantiven und Beziehungsmengen mit Verben bzw. verbalen Formulierungen zu beschreiben. Da verbale Formulierungen häufig langatmig sind, hat sich dieser Vorschlag nicht zu einem allgemeinen Gebrauch entwickelt. Beispielsweise kann man die verbale Formulierung *ist verheiratet mit*, die eine bestimmte Beziehungsform zwischen den Entitätsmengen *Männer* und *Frauen* beschreibt, kurz und prägnant durch *Ehe* ausdrücken.

ER-Diagramm für ein hypothetisches Unternehmen

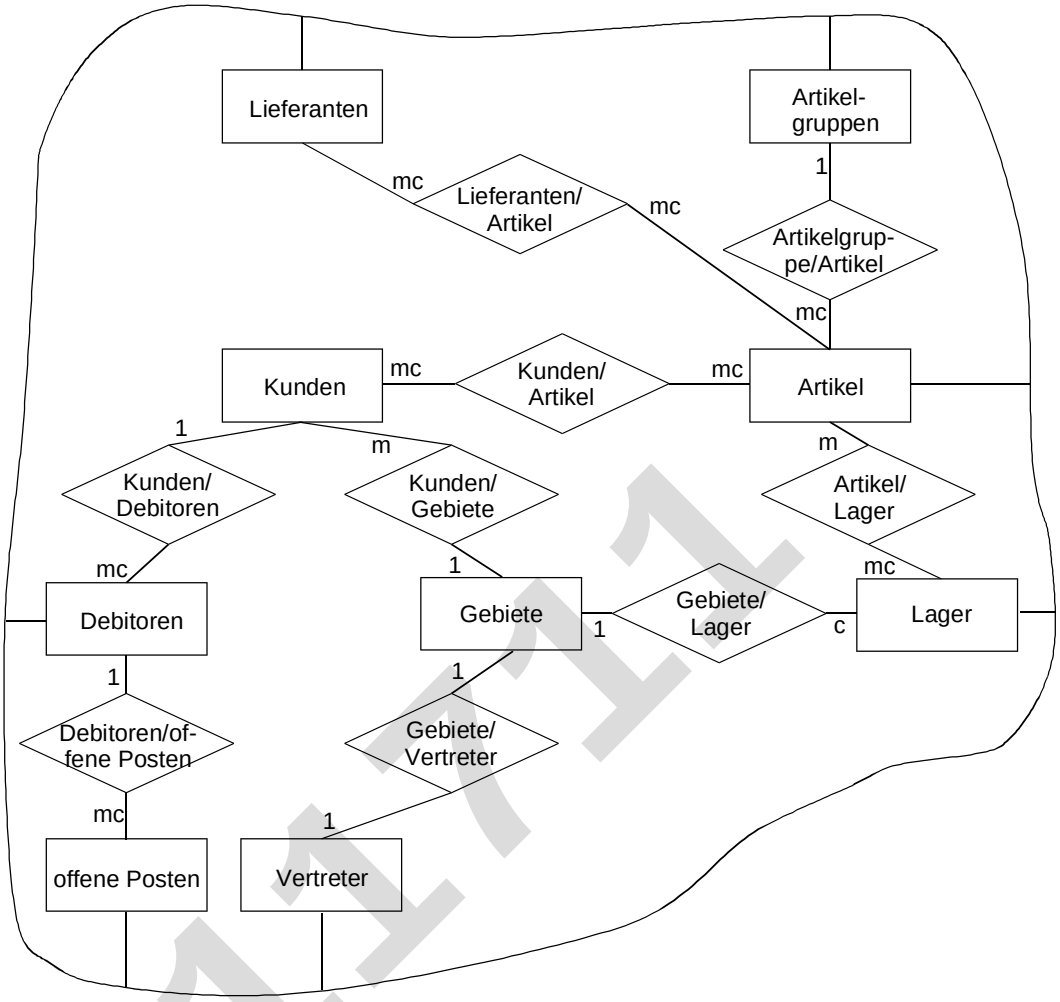


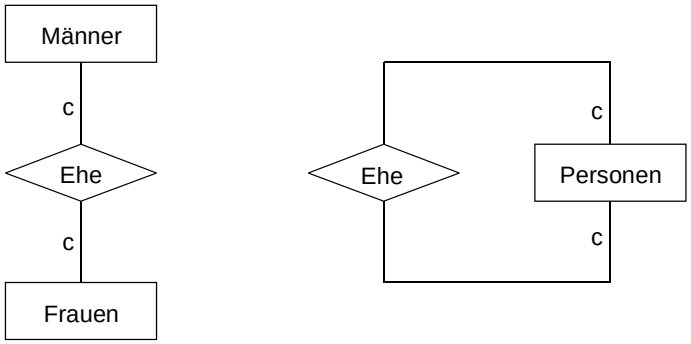
Abb. 3.10. Ausschnitt aus einem ER-Diagramm für ein hypothetisches Unternehmen.

Übungsaufgabe 3.6

Betrachtet werde das in Abb. 3.10 gezeigte ER-Diagramm. Erläutern Sie kurz, was die Beziehungsmengen *Debitoren/offene Posten*, *Gebiete/Vertreter* und *Artikel/Lager* inhaltlich ausdrücken.

rekursive Beziehungen

Bislang wurden nur Beziehungen zwischen verschiedenen Entitätsmengen betrachtet. Beziehungen können jedoch auch zwischen Entitäten aus einer Entitätsmenge auftreten. Solche Beziehungen heißen rekursiv. Eine rekursive Beziehung auf einer Entitätsmenge lässt sich stets durch eine nichtrekursive Beziehung zwischen zwei Entitätsmengen ausdrücken. Die Abb. 3.11 verdeutlicht dies.



a) Nichtrekursive Modellierung

b) Rekursive Modellierung

rekursive und nicht-rekursive Modellierung

Abb. 3.11. Nichtrekursive und rekursive Modellierung.

Die beiden Modellierungsformen in Abb. 3.11 besitzen die gleiche Aussagekraft. Aus der nichtrekursiven Darstellungsform geht die rekursive Form durch das Zusammenfassen der Entitätsmengen *Männer* und *Frauen* hervor. Umgekehrt gelangt man von der rekursiven Darstellung zu der nichtrekursiven durch die Aufspaltung der Entitätsmenge *Personen*.

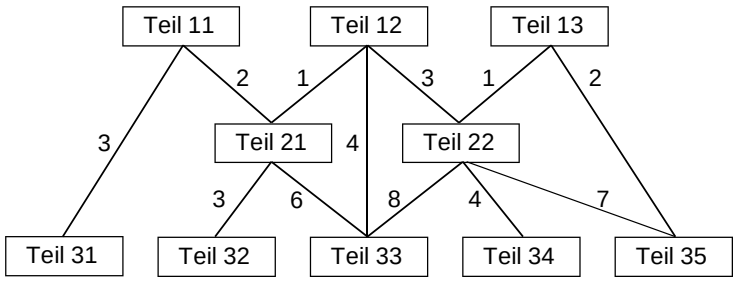
Ein weiteres Beispiel zur rekursiven und nicht rekursiven Modellierung behandelt die Übungsaufgabe 3.7.

Übungsaufgabe 3.7

Betrachtet werde ein kleiner Ausschnitt aus dem Fertigungsbereich eines Unternehmens. Er besteht aus Maschinen und Teilen sowie aus Beziehungen zwischen diesen Entitäten.

Sämtliche Teile werden auf den Maschinen des Unternehmens gefertigt oder aus gefertigten Teilen zusammengesetzt. Teile, die in ein anderes Teil eingehen, seien als Unterteile bezeichnet und Oberteile seien Teile, die sich durch das Zusammenfügen von Unterteilen ergeben. Insgesamt liegt also eine hierarchische Teilestruktur vor, die mehrere Stufen umfassen kann.

Zur Verdeutlichung sei folgendes einfache Beispiel einer Teilestruktur betrachtet:



Teilestruktur

In dieser Darstellung bezeichnen die Nummern 11, 12, 13 usw. unterschiedliche Teile. Das Oberteil 12 setzt sich z.B. aus den Unterteilen 21, 22 und 33 zusammen. Das Unterteil 21 beispielsweise ist zugleich auch Oberteil, da es aus den Teilen 32 und 33 besteht. Die Verbindungskanten zwischen den Teilen drücken die Teileverwendung aus. Der einer Kante zugeordnete positive ganzzahlige Parameter gibt an, wie viele Unterteile in das zugehörige Oberteil eingehen.

Entwickeln Sie für diesen Ausschnitt zwei ER-Diagramme und zwar ein Diagramm mit rekursiver Darstellung der Teilehierarchie sowie eine nichtrekursive Darstellung.

3.2 Hierarchisches Datenmodell

hierarchische Strukturen

Viele Phänomene der Realität stellen Hierarchien dar. Man denke etwa an die Aufbauorganisation von Betrieben, die Zusammensetzung von Produkten aus Teilen, die Untergliederung von Absatzgebieten in Teilgebiete, die Strukturierung von Aufträgen in Auftragspositionen usw. Wie noch gezeigt wird, lassen sich hierarchische Strukturen

sequentiell verarbeiten

sequentiell darstellen und damit auch sequentiell verarbeiten. In der Anfangsphase der betrieblichen Datenverarbeitung, die durch sequentielle Verarbeitungsmuster geprägt war, spielten hierarchisch strukturierte Daten daher eine zentrale Rolle. Hierarchische Beziehungen zwischen den Datenelementen eines Datensatzes erfasste man mittels sogenannter Mehrfachsätze. Zur Veranschaulichung diene das in Abb. 3.12 gezeigte Beispiel.

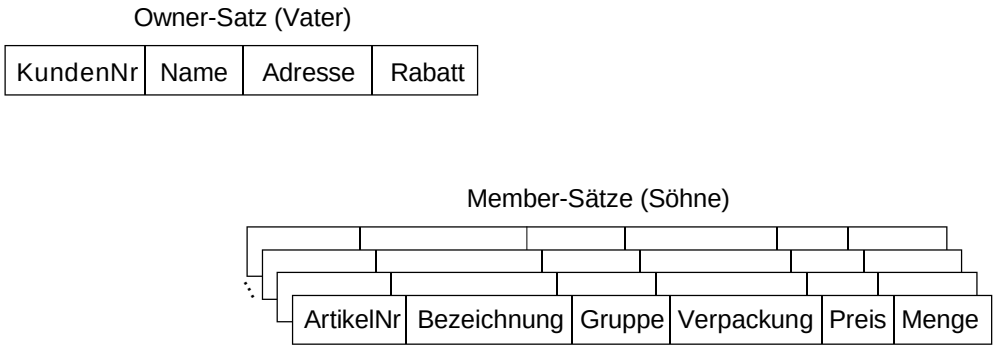


Abb. 3.12. Owner-Satz und Member-Sätze.

Owner- und Member-Sätze

Zu dem Owner-Satz in Abb. 3.12 gehören mehrere Member-Sätze. Ein Owner beschreibt einen Kunden und die Member-Sätze beschreiben die Bestellungen dieses Kunden. Die Anzahl der Bestellungen ist variabel. Die gesamte Struktur wird sequentiell gespeichert. Zuerst ein Owner gefolgt von den zugehörigen Member-Sätzen, danach der nächste Owner mit seinem Member-Sätzen usw.

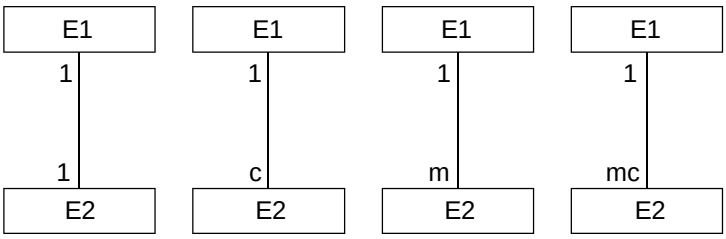
Inhalt des Kapitels

Aus dem Konzept der Mehrfachsätze ging das hierarchische Datenmodell hervor, wie es z.B. dem früher sehr verbreiteten Datenbanksystem IMS der Firma IBM zugrunde liegt. Hierarchische Datenbanksysteme werden auch heute noch eingesetzt. Im vorliegenden Kapitel wird daher das hierarchische Datenmodell in seinen Grundzügen behandelt. Im Einzelnen wird auf die Struktur des hierarchischen Datenmodells, die hierarchische Datenmodellierung, die Darstellung von Hierarchien und die Vor- und Nachteile des hierarchischen Modells eingegangen.

a) Struktur des hierarchischen Datenmodells

Strukturelemente

Strukturelemente des hierarchischen Datenmodells sind Entitätsmengen und hierarchische Beziehungen zwischen Entitätsmengen. Mittels dieser Strukturelemente werden Hierarchien gebildet. Als hierarchische Beziehungen $h(E1, E2)$ zwischen zwei Entitätsmengen *E1* und *E2* kommen die ersten vier der in Abb. 3.7 zusammengestellten Beziehungstypen in Frage. Eine Übersicht über die hierarchischen Beziehungen zeigt die Abb. 3.13. Einfachheitshalber werden bei dieser Darstellung keine Rauten als Beziehungssymbole verwendet. Beziehungen werden also lediglich (implizit) durch Verbindungslinien wiedergegeben.



hierarchische Beziehungstypen

Abb. 3.13. Vier hierarchische Beziehungstypen.

Eine Hierarchie besteht aus einer übergeordneten Entitätsmenge, dem Vater, und hierarchisch untergeordneten Entitätsmengen. Die Unterordnung kann sich über mehrere hierarchische Ebenen erstrecken und somit Söhne, Enkel, Urenkel usw. einschließen. Ein allgemeines Beispiel ist in Abb. 3.14 dargestellt.

hierarchisches Datenmodell

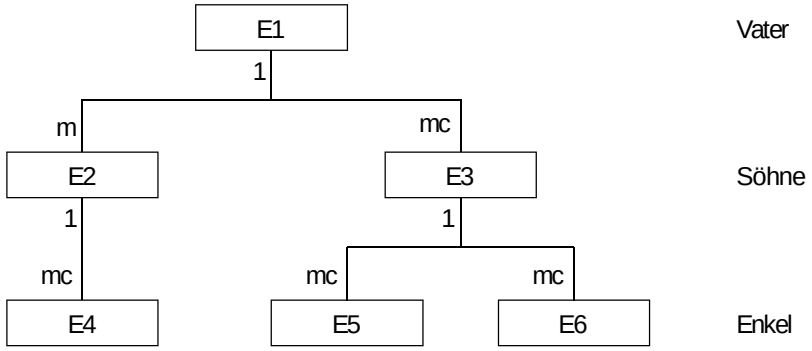


Abb. 3.14. Einfaches hierarchisches Datenmodell.

Anhand von Abb. 3.14 lassen sich die Merkmale des hierarchischen Datenmodells nachvollziehen.

Merkmale des hierarchischen Datenmodells

Merkmale des hierarchischen Datenmodells:

- Genau eine Entitätsmenge, der Vater, bildet die Spitze der Hierarchie. Dem Vater ist also keine andere Entitätsmenge übergeordnet.
- Jeder Entitätsmenge mit Ausnahme des Vaters ist genau eine andere Entitätsmenge übergeordnet.
- Jeder Entitätsmenge können beliebig viele andere Entitätsmengen direkt untergeordnet sein.

Eine Hierarchie stellt also einen Baum dar und der Vater als Spitze der Hierarchie ist die Wurzel des Baums.

Ausprägungen von Entitätsmengen

Abhängig vom Datenaufkommen entsprechen einer Hierarchie auf der Ebene von Entitätsmengen mehr oder weniger Hierarchien auf der Ebene einzelner Entitäten. Bei jeder Hierarchie auf Entitätsebene handelt es sich um eine datenmäßige Ausprägung der Hierarchie auf Entitätsmengenebene. Die strukturelle Vielfalt der einzelnen Ausprägungen wird durch die Beziehungstypen zwischen den Entitätsmengen eingegrenzt. Einige mit dem in Abb. 3.14 angegebenen Datenmodell verträgliche Ausprägungen sind in Abb. 3.15 zu sehen. Einzelne Entitäten werden in dieser Darstellung mit Kleinbuchstaben bezeichnet.

Hierarchien auf Entitätsebene

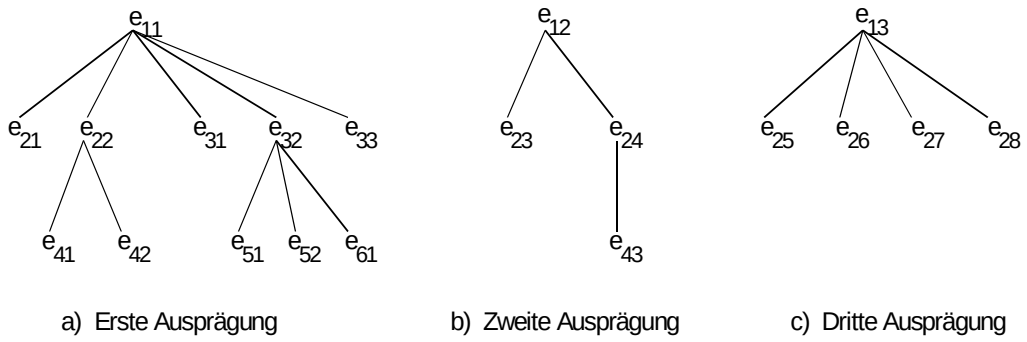


Abb. 3.15. Hierarchien auf der Ebene von Entitäten.

Für die Indizierung der Entitäten e_{ij} in Abb. 3.15 gilt:

- Der erste Index stimmt mit dem Index der entsprechenden Entitätsmenge überein; beispielsweise gehören die Entitäten e_{2j}, j=1,2,3,..., der Entitätsmenge *E2* an.
- Der zweite Index dient der fortlaufenden Nummerierung der Entitäten einer Entitätsmenge.

Einzelbeziehungen in einem hierarchischen Datenmodell

Sämtliche in Abb. 3.15 gezeigten Ausprägungen sind mit dem zugrunde liegenden Datenmodell verträglich. Betrachtet werde beispielsweise die dritte Ausprägung. Da zwischen *E1* und *E2* eine 1-m-Beziehung besteht, muss e_{13} mindestens eine Entität der Menge *E2* nachgeordnet sein. Hier sind vier Entitäten nachgeordnet, nämlich e_{25}, e_{26}, e_{27} und e_{28}. Zwischen den Entitätsmengen *E2* und *E4* besteht eine 1-mc-Beziehung. Einer Entität aus *E2* entspricht folglich keine, genau eine oder mehrere Entitäten aus der Menge *E4*. Hier entsprechen den Entitäten e_{25} bis e_{28} keine Entitäten aus *E4*. Neben den Entitäten e_{25} bis e_{28} besitzt die Entität e_{13} keine weiteren direkten Nachfolger. Insbesondere keine Nachfolger aus der Menge *E3*. Dies ist zulässig, denn *E1* ist mit *E3* über eine 1-mc-Beziehung verbunden.

Übungsaufgabe 3.8

Betrachtet werde das in Abb. 3.14 gezeigte hierarchische Datenmodell. Skizieren Sie eine mit diesem Modell verträgliche Hierarchie auf Entitätsebene, die folgende Eigenschaft aufweist: Sie besteht aus der minimalen Anzahl von Entitäten.

mehrere Hierarchien in einer hierarchischen Datenbank

Aus dem in Abb. 3.14 dargestellten Beispiel eines hierarchischen Datenmodells darf nicht die Schlussfolgerung gezogen werden, dass ein hierarchisches Datenmodell bzw. eine hierarchische Datenbank stets aus nur einer auf Entitätsmengen definierten Hierarchie besteht. Je nach den vorgesehenen Entitätsmengen, den Beziehungen zwischen den Entitätsmengen und den auf den Entitätsmengen abzuwickelnden Verarbeitungsprozessen wird man geeignete Gruppen von Entitätsmengen bilden und jede dieser Gruppen hierarchisch strukturieren. In der Regel umfasst eine hierarchische Datenbank folglich mehrere, voneinander völlig unabhängige Hierarchien. Welche Hierarchien im Einzelnen gebildet werden, ist ein Modellierungsproblem.

b) Hierarchische Modellierung

Betriebliche Gegebenheiten lassen sich auf unterschiedliche Weise in Hierarchien abbilden. Dies sei an einem hypothetischen Beispiel gezeigt. Ausgegangen werde von einem Realitätsausschnitt, wie ihn das in Abb. 3.10 dargestellte ER-Diagramm zeigt. Auf diesen Realitätsausschnitt kann sich eine Vielzahl von Verarbeitungsprozessen beziehen. Beispielsweise die Erfassung von Kundenaufträgen, die Fortschreibung von Lagerbeständen, die Erstellung von Umsatzstatistiken nach Artikeln, Kunden und Gebieten usw. Im Folgenden werde die Erfassung von Kundenaufträgen zugrunde gelegt und es werde davon ausgegangen, dass die zu erfassenden Daten sich auf die Datenwelt beschränken, die durch das ER-Diagramm in Abb. 3.16 abgegrenzt ist.

Beispiel für ein ER-Diagramm

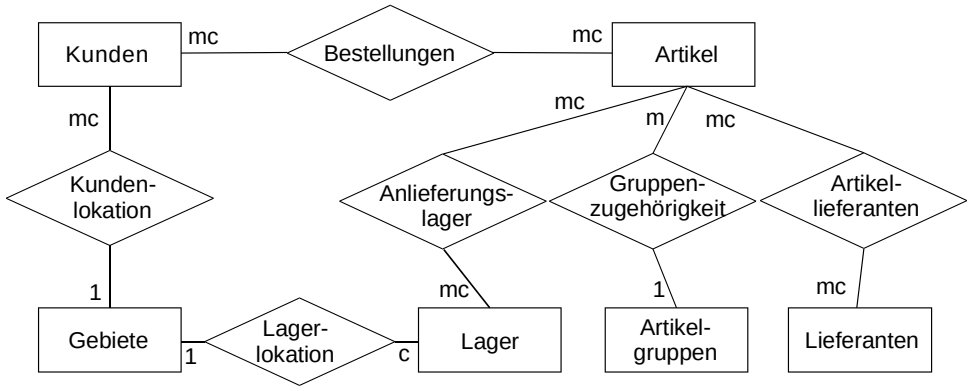
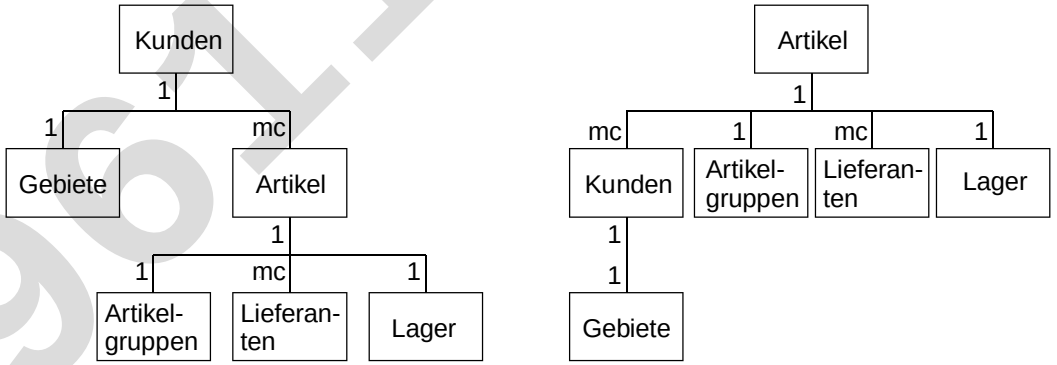


Abb. 3.16. ER-Diagramm zur Auftragserfassung mit Kontext.

Neben Daten über Kunden und Artikel sind auch bestimmte Daten über Gebiete, Lager, Artikelgruppen und Lieferanten zu erfassen. Dies hängt mit den Verarbeitungsprozessen zusammen, die der Auftragserfassung nachgelagert sind. Angenommen sei, dass im Anschluss an die Auftragserfassung Aufgaben wie Erstellung von Begleitpapieren für die Auslieferung von Kundenaufträgen, Erzeugung von Kunden- und Lieferantenstatistiken und Schreiben von Rechnungen abzuwickeln sind.

Abbildung 3.17 zeigt zwei mögliche hierarchische Datenmodelle, die ausgehend von dem ER-Diagramm in Abb. 3.16 erstellt werden können. Beide Modelle sind mit einem Informationsverlust verbunden, der durch die strukturellen Restriktionen des hierarchischen Ansatzes bedingt ist.

alternative hierarchische Modelle



a) Modellierung mit *Kunden* als Wurzel

b) Modellierung mit *Artikel* als Wurzel

Abb. 3.17. Zwei hierarchische Datenmodelle.

Betrachtet werde zunächst die Hierarchie mit der Entitätsmenge *Kunden* als Wurzel. Zu diesem Modell ist folgendes anzumerken:

- Aufgrund der strukturellen Restriktionen des hierarchischen Modells muss der ursprüngliche Beziehungsreichtum zwischen *Kunden* und *Gebieten* auf eine 1-1-Beziehung abgemagert werden. Der in Kauf zu nehmende Informationsverlust besteht darin, dass sich aus dem Modell nun nicht mehr unmittelbar entnehmen lässt, welche Kunden zu einem bestimmten Gebiet gehören. Nach wie vor wird dagegen erfasst, welchem Gebiet ein bestimmter Kunde angehört.

- Auch hinsichtlich der Beziehung zwischen *Kunden* und *Artikeln* muss ein Informationsverlust hingenommen werden. Da Söhne nur einen Vater haben dürfen, muss nun gelten:

  Assoziation(*Artikel*, *Kunden*) = 1

  Das Modell drückt zwar noch aus, welche Artikel ein bestimmter Kunde geordert hat, aber nicht mehr, von welchen Kunden ein bestimmter Artikel bestellt wurde.

- Die Beziehung zwischen *Gebieten* und *Lagern* wurde gänzlich eliminiert, und die Zuordnungen von *Artikelgruppen*, *Lieferanten* und *Lagern* zu *Artikeln* musste aus dem oben genannten Grund wie folgt eingeschränkt werden:

  Assoziation (*Artikelgruppen*, *Artikel*) = 1,
  Assoziation (*Lieferanten*, *Artikel*) = 1,
  Assoziation (*Lager*, *Artikel*) = 1.

In analoger Weise ließe sich der Informationsverlust, der mit dem hierarchischen Datenmodell in Abb. 3.17 b) hinzunehmen ist, begründen.

Untersucht man die Aussagefähigkeit der beiden Datenmodelle in Abb. 3.17, so kann man folgenden wesentlichen Unterschied feststellen:

Unterschiede in der Aussagefähigkeit

Die Hierarchie mit *Kunden* als Wurzel gestattet für einen gegebenen Kunden den unmittelbaren Zugriff zu sämtlichen Artikeln, die der Kunde bestellt hat. Dagegen erlaubt die Hierarchie mit *Artikel* als Wurzel den unmittelbaren Zugriff auf die Kunden, welche einen gegebenen Artikel bestellt haben.

Welches der beiden hierarchischen Datenmodelle verwendet werden sollte, hängt von den Informationsverarbeitungsaufgaben ab, die in Verbindung mit der modellierten Datenwelt abzuwickeln sind.

Übungsaufgabe 3.9

Gegeben seien die beiden in Abb. 3.17 gezeigten hierarchischen Datenmodelle. Angenommen werde, dass in Verbindung mit diesen Modellen lediglich die Informationsverarbeitungsaufgaben Auftragserfassung, Rechnungsschreibung und Erstellung von Kundenumsatzstatistiken zu betrachten sind. Welches Datenmodell ist das geeignetere? Begründen Sie ihre Antwort.

Wie die Übungsaufgabe 3.9 verdeutlicht, begünstigt das hierarchische Datenmodell bestimmte Verarbeitungsprozesse und erschwert andere Verarbeitungsprozesse. Bei der hierarchischen Datenmodellierung sind daher stets die abzuwickelnden Informationsverarbeitungsaufgaben im Auge zu behalten. Dies gilt nicht für die Datenmodellierung mit dem ER-Modell, da dieses Modell den Beziehungsreichtum zwischen zwei Entitätsmengen nicht einseitig einschränkt.

c) Darstellung von Hierarchien

Darstellungsformen von Beziehungen

Auf die Darstellung von Hierarchien wird hier eingegangen, weil das hierarchische Datenmodell eine Besonderheit aufweist. Grundsätzlich können Beziehungen zwischen Entitäten auf zwei Arten dargestellt werden:

- Direkt, d.h. durch explizite Verweise von einer gegebenen Entität auf andere Entitäten mit Hilfe von Zeigern (engl. pointer).
- Indirekt, d.h. durch die Verwendung von gleichen Attributen mit übereinstimmenden Attributwerten für Entitäten, die miteinander in Beziehung stehen.

Bei dem hierarchischen Datenmodell, und nur bei diesem Modell, kommt noch eine weitere Darstellungsform hinzu:

- Die sequentielle Speicherung von Entitäten.

sequentielle Speicherung von Hierarchien

Eine Hierarchie kann man umkehrbar eindeutig auf eine sequentielle Datenstruktur abbilden, wenn eine Ordnung über den Elementen der Hierarchie existiert, die eindeutig festlegt, in welcher Reihenfolge die Elemente der Hierarchie anzusprechen sind. Solch eine Ordnung kann beispielsweise durch folgende Regeln gegeben sein:

- Vater kommt vor Sohn.
- Sohn kommt vor Bruder.
- Die Söhne eines Vaters werden von links nach rechts angesprochen.

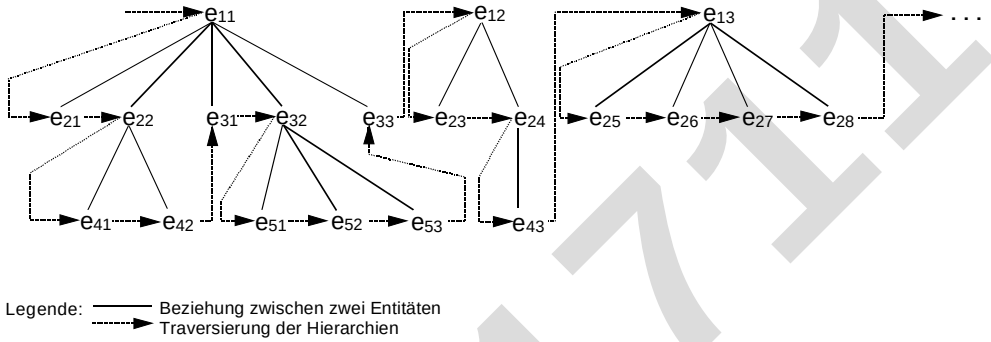
Traversierung von Hierarchien

Diese Regeln beschreiben nichts anderes als das Traversieren eines Baumes in symmetrischer Ordnung (engl. inordner). Geeignete Algorithmen für diesen Zweck, sogenannte Traversierungsalgorithmen, werden im Kurs "Algorithmen und Datenstrukturen" behandelt. Welche sequentielle Speicherungsfolge sich für eine bestimmte Hierarchie ergibt, sei anhand von Abb. 3.18 demonstriert.

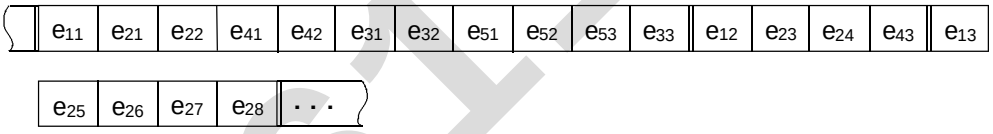
Ablauf der Traversierung

Der Abb. 3.18 liegen das in Abb. 3.14 gezeigte hierarchische Datenmodell mit den in Abb. 3.15 angegebenen Ausprägungen auf Entitätsebene zugrunde. Einige Schritte des Ablaufs der Traversierung seien nun erläutert:

- Da Vater vor Sohn kommt und Brüder von links nach rechts anzusprechen sind, beginnt die Traversierung mit dem Zugriff zu der Entität e_{11}.
- Von e_{11} wird nach e_{21} und nicht etwa nach e_{12} verzweigt, weil Sohn vor Bruder kommt. Aus der Menge der Söhne wird e_{21} gewählt, weil e_{21} am weitesten links liegt.
- Die Entität e_{21} besitzt keinen Sohn. Verzweigt wird daher zu dem rechts folgenden Bruder e_{22}.



a) Traversierung hierarchisch strukturierter Daten in symmetrischer Ordnung



b) Sequentielle Speicherung der Entitäten

Abb. 3.18. Abbildung hierarchisch strukturierter Daten in eine sequentielle Speicherungsfolge.

Die sequentielle Speicherung hierarchisch strukturierter Daten sei auch an einem konkreten Beispiel verdeutlicht. Es ist in Abb. 3.19 dargestellt und umfasst lediglich die Entitätsmengen *Kunden* und *Bestellungen* sowie eine 1-mc-Beziehung zwischen diesen Entitätsmengen. Den Entitätsmengen wurden einige wenige Attribute zugeordnet.

Definiert man auf dem in Abb. 3.19 a) gezeigten hierarchischen Datenmodell eine symmetrische Ordnung, so können datenmäßige Ausprägungen der Hierarchie in eine sequentielle Datei abgebildet werden. Die Abb. 3.19 b) veranschaulicht dies exemplarisch.

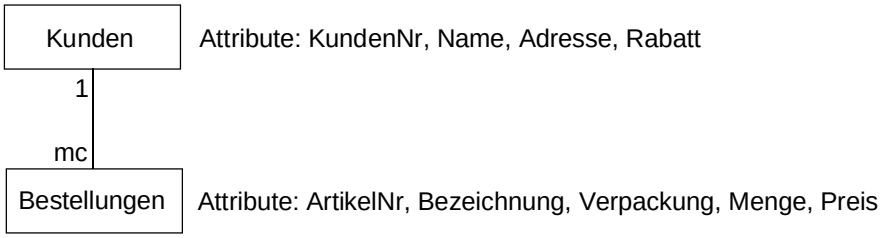
Gruppenwechsel

Bei der Verarbeitung einer als sequentielle Datei dargestellten Hierarchie müssen die Übergänge zwischen den hierarchischen Ebenen beachtet werden. Sind die Entitäten einer Gruppe abgearbeitet, so findet ein Übergang zur nächsthöheren Ebene statt. In der klassischen betrieblichen Datenverarbeitung spricht man in diesem Fall von einem Gruppenwechsel. Bei dem Beispiel in Abb. 3.19 b) erfolgt ein Gruppenwechsel jeweils an den durch einen Pfeil gekennzeichneten Stellen.

Datenredundanz

Hinzuweisen ist noch auf einen Nachteil des hierarchischen Datenmodells, der in Abb.

3.19 b) deutlich zum Ausdruck kommt: die beträchtliche Datenredundanz. Relativ viele Daten müssen mehrfach gespeichert werden. Bei dem Beispiel in Abb. 3.19 sind die Attribute *Bezeichnung, Verpackung* und *Preis* der Entitätsmenge *Bestellungen* betroffen. Für jeden von mehreren Kunden bestellten Artikel müssen die entsprechenden Attribute jeweils erneut gespeichert werden.



a) Hierarchisches Datenmodell

Darstellung einer Hierarchie als sequentielle Datei

K1	Abs, Josef	Ahornweg 12, 28219 Bremen		05
A01	Bier	Kiste	20	14,00
A07	Wein	Karton	50	50,00
A12	Wasser	Kiste	120	11,00
K2	Amt, Xaver	Tulpenweg 20, 52062 Aachen		03
A03	Bier	Fass	10	80,00
A05	Wodka	Karton	15	65,00
K3	Beer, Poldi	Rosenweg 36, 58095 Hagen		05
A05	Wodka	Karton	10	65,00
K4	Bitz, Hans	Nelkenweg 8, 12524 Berlin		10
A03	Bier	Fass	20	80,00
A04	Gin	Karton	30	70,00
A05	Wodka	Karton	40	65,00
A12	Wasser	Kiste	30	12,00
K5	Buhl, Herbert	Amselweg 90, 28219 Bremen		03
A03	Bier	Fass	10	80,00
		⋮		

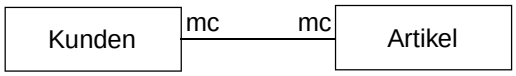
sequentielle Speicherung

b) Speicherung in einer sequentiellen Datei

Abb. 3.19. Speicherung einer Hierarchie als sequentielle Datei.

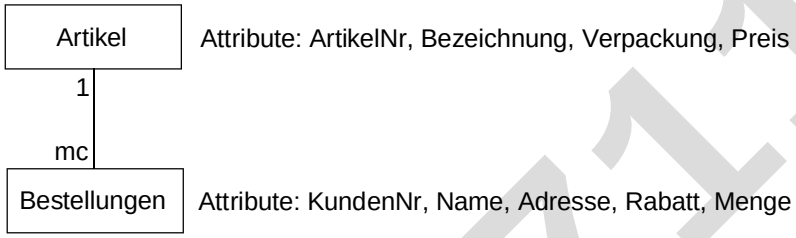
Übungsaufgabe 3.10

Betrachtet werde das in Abb. 3.19 gezeigte Beispiel eines hierarchischen Datenmodells. Ausgangspunkt dieses Modells ist eine mc-mc-Beziehung zwischen den zwei Entitätsmengen *Kunden* und *Artikel*:



Informationsverlust im hierarchischen Datenmodell

Da diese Struktur mit dem hierarchischen Datenmodell nicht verträglich ist, wurde - unter Inkaufnahme eines Informationsverlustes - das in Abb. 3.19 a) dargestellte Modell entwickelt. Eine alternative Modellierung, die ebenfalls wesentliche Zusammenhänge ausdrückt, ist die folgende:



Bilden Sie diese Hierarchie unter Verwendung der der Abb. 3.19 b) zu entnehmenden konkreten Daten in eine sequentielle Datei ab. Benennen Sie außerdem eine Informationsverarbeitungsaufgabe, die sich mittels dieser Datei besonders effizient abwickeln lässt.

d) Vor- und Nachteile des hierarchischen Datenmodells

Einige Nachteile des hierarchischen Datenmodells wurden bereits erwähnt. Sie seien hier kurz zusammengefasst:

- Semantische Armut, d.h. es sind nicht sämtliche Beziehungen zwischen Daten darstellbar und bei der Modellierung betrieblicher Phänomene muss daher ein Informationsverlust hingenommen werden.

Nachteile

- Beträchtliche Redundanz, d.h. eine Mehrfachspeicherung von Daten ist aus strukturellen Gründen unumgänglich.

- Geringe Auswertungsflexibilität, d.h. Auswertungen, die nicht entlang den durch die Hierarchie gegebenen Zugriffspfaden verlaufen, sind mit einem sehr hohen Aufwand verbunden.

und

An Vorteilen des hierarchischen Datenmodells sind zu nennen:

Vorteile des hierarchischen Datenmodells

- Hierarchisch strukturierte Daten können auf sequentielle Dateien abgebildet und auf einfache Weise sequentiell verarbeitet werden.

- Auswertungen, die entlang den durch die Hierarchie gegebenen Zugriffspfaden verlaufen, können auf sehr effiziente Weise durchgeführt werden.

Vor allem das Unvermögen, die Reichhaltigkeit der betrieblichen Datenwelt realitätsnah zu modellieren, und die daraus resultierende geringe Auswertungsflexibilität haben dazu geführt, dass das hierarchische Datenmodell neueren Datenbankverwaltungssystemen nicht mehr zugrunde gelegt wird.

3.3 Netzwerkartiges Datenmodell

Sämtliche Beziehungstypen darstellbar

Beschränkungen, wie sie das hierarchische Datenmodell auferlegt, fallen bei dem netzwerkartigen Datenmodell weg. So lassen sich mit dem netzwerkartigen Datenmodell alle Beziehungen zwischen Entitätsmengen darstellen. Schon bald nach dem Aufkommen hierarchischer Datenbankverwaltungssysteme wurden daher Datenbankverwaltungssysteme für netzwerkartig strukturierte Daten entwickelt. Eine Führungsrolle kam hierbei der Data Base Task Group (DBTG) des CODASYL-Komitees zu. Dem CODASYL-Komitee gehören einige der wichtigsten US-amerikanischen Computeranwender und -hersteller an. Eine der Aufgaben der CODASYL-Komitees besteht in der Entwicklung von Normen für den Computerbereich, so beispielsweise für die Programmiersprache COBOL. Im Jahre 1971 legte die CODASYL-DBTG einen Entwurf für eine netzwerkartige Datenbankarchitektur vor, welche eine DDL zur Schemabeschreibung sowie eine in COBOL eingebettete DML zur Datenmanipulation einschloss. Auf der Grundlage dieses CODASYL-DBTG-Modells entwickelten viele Computerhersteller netzwerkartige Datenbankverwaltungssysteme. Beispiele sind die Systeme IDS von Honeywell, DMS-1100 von Univac, UDS von Siemens und IMF von CDC. Zwischenzeitlich ist die praktische Bedeutung dieser Systeme zurückgegangen. Da das netzwerkartige Datenmodell - im Folgenden auch kurz als Netzwerk bezeichnet - eine wichtige Stufe in der Entwicklung von Datenbanksystemen repräsentiert, wird es im vorliegenden Kapitel in seinen Grundzügen behandelt. Angesprochen werden die Struktur von Netzwerken, die Modellierung mit Netzwerken und die Darstellung von Netzwerken.

Datenbankverwaltungssysteme für Netzwerke

a) Struktur des netzwerkartigen Datenmodells

Strukturelemente

Netzwerke bestehen aus den gleichen Strukturelementen wie hierarchische Datenmodelle, nämlich aus Entitätsmengen und hierarchischen Beziehungen zwischen Entitätsmengen. Anders als bei dem hierarchischen Datenmodell darf in einem Netzwerk eine Entitätsmenge mehreren Hierarchien und nicht nur genau einer Hierarchie angehören. Ein Netzwerk entsteht somit durch die Verknüpfung mehrerer Hierarchien und es weist - wenn man von trivialen Fällen absieht - mehrere Wurzeln auf. Ein allgemeines Beispiel für ein Netzwerk zeigt die Abb. 3.20.

mehrere Wurzeln möglich

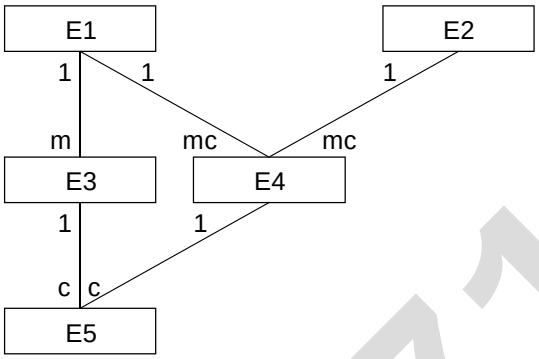


Abb. 3.20. Einfaches netzwerkartiges Datenmodell.

Vorgänger und Nachfolger

In einem Netzwerk kann eine Entitätsmenge mehrere *Väter* besitzen. Das bisherige Vater-Sohn-Schema verliert daher seine Gültigkeit. An die Stelle von Vater-Sohn-Beziehungen treten nun Vorgänger-Nachfolger-Beziehungen. Die in Netzwerken verwendbaren hierarchischen Beziehungen *h*(*E1*, *E2*) legen eine Richtung bzw. einen Vorgänger und einen Nachfolger fest. Bei dem Netzwerk in Abb. 3.20 ist beispielsweise die Entitätsmenge *E4* ein Nachfolger von *E1* und *E2*. Andererseits ist *E4* ein Vorgänger von *E5*. Mittels der eingeführten Begriffe lässt sich die Struktur des netzwerkartigen Datenmodells nun wie folgt charakterisieren:

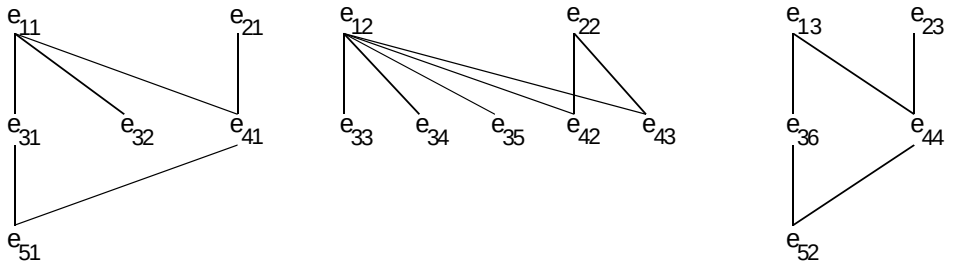
Merkmale des netzwerkartigen Datenmodells

Merkmale des netzwerkartigen Datenmodells:

- Es existieren mehrere Wurzeln, d.h. Entitätsmengen, welche keine Vorgänger besitzen.
- Mit Ausnahme der Wurzeln kann jede Entitätsmenge einen oder mehrere Vorgänger besitzen.
- Jede Entitätsmenge kann beliebig viele, d.h. keinen, einen oder mehrere Nachfolger besitzen.

Zu einem Netzwerk auf der Ebene von Entitätsmengen können beliebig viele datenmäßige Ausprägungen des Netzwerks gehören. Jede datenmäßige Ausprägung stellt seinerseits ein Netzwerk dar, jedoch ein Netzwerk auf der Ebene einzelner Entitäten. Einige mit dem Netzwerk in Abb. 3.20 verträgliche Netzwerke auf Entitätenebene sind in Abb. 3.21 zusammengestellt.

Netzwerke auf Entitätsebene



a) Erste Ausprägung b) Zweite Ausprägung c) Dritte Ausprägung

Abb. 3.21. Netzwerke auf der Ebene von Entitäten.

Für die Indizierung der Entitäten e_{ij} in Abb. 3.21 gilt:

- Der erste Index stimmt mit dem Index der entsprechenden Entitätsmenge in Abb. 3.20 überein; beispielsweise gehören die Entitäten e_{3j}, j = 1,2,..., zu der Entitätsmenge E_3.
- Der zweite Index dient der fortlaufenden Nummerierung der Entitäten einer Entitätsmenge.

Übungsaufgabe 3.11

Begründen Sie, warum das in Abb. 3.21 a) dargestellte Netzwerk auf Entitätsebene eine zulässige Ausprägung des in Abb. 3.20 gezeigten allgemeinen Netzwerks ist. Gehen Sie hierbei insbesondere auf die zwischen den Entitätsmengen bestehenden Beziehungen ein.

b) Modellierung mit Netzwerken

Zu Beginn dieses Kapitels wurde erwähnt, dass sich mit dem netzwerkartigen Datenmodell alle Beziehungen zwischen Entitätsmengen darstellen lassen. Andererseits wurden aber bei der Nennung der Strukturelemente von Netzwerken lediglich hierarchische Beziehungen berücksichtigt. Hier scheint ein Widerspruch vorzuliegen. Ein Widerspruch besteht jedoch nicht, da man mit Hilfe hierarchischer Beziehungstypen alle übrigen Beziehungstypen ausdrücken kann.

Überführung von netzwerkartigen Beziehungen in hierarchische Beziehungen

Es ist nicht nur möglich, sondern vielmehr sinnvoll alle übrigen Beziehungstypen auf hierarchische Beziehungen zurückzuführen. In Kapitel 3.4.2, bei der Behandlung eines erweiterten Relationenmodells, wird dies im Detail erläutert und begründet. Hier werde exemplarisch, und zwar für den mc-mc-Beziehungstyp, gezeigt, wie ein Übergang zu hierarchischen Beziehungen ohne Informationsverlust möglich ist und warum der Übergang sinnvoll ist. Betrachtet werde die bereits in Übungsaufgabe 3.10 angesprochene mc-mc-Beziehung zwischen Kunden und Artikeln. Die Beziehung ist in Abb. 3.22 nochmals dargestellt.

mc-mc-Beziehung

Abb. 3.22. Beispiel für eine mc-mc-Beziehung.

Beziehungen vom Typ mc-mc treten in der betrieblichen Praxis häufig auf. Jede mc-mc-Beziehung ist durch eine Matrix darstellbar. So erhält man z.B. für die mc-mc-Beziehung in Abb. 3.22 die in Abb. 3.23 gezeigte Matrixdarstellung.

Matrixdarstellung der mc-mc-Beziehung

Artikel / Kunden	A1	A2	A3	A4	A5	A6	A7	A8	...
K1		X				X			
K2	X								
K3			X						
K4							X		
K5									
K6					X				
K7		X							
K8								X	
⋮									

Abb. 3.23. Matrixdarstellung einer mc-mc-Beziehung.

In Abb. 3.23 repräsentieren die mit *K1*, *K2*, *K3*,... bezeichneten Matrixzeilen einzelne Kunden und die mit *A1*, *A2*, *A3*,... bezeichneten Spalten einzelne Artikel. Die Markierungen in der Matrix drücken Bestellungen aus. Sie geben an, welche Artikel die einzelnen Kunden bestellt haben bzw. von welchen Kunden die einzelnen Artikel bestellt wurden. Beispielsweise hat der Kunde *K1* die Artikel *A2* und *A6* geordert und der Kunde *K5* keinen Artikel.

Erfahrungsgemäß sind derartige Matrizen sehr dünn besetzt. Vertreibt beispielsweise ein Süßwarenhersteller 2000 Artikel an 6000 Kunden und ordert ein Kunde pro Bestellperiode im Mittel 40 Artikel, so beträgt die Besetzungsdichte 2 %. Von den insgesamt möglichen 12 000 000 Einzelbeziehungen bzw. Bestellungen werden in einer Bestellperiode im Mittel lediglich 240 000 realisiert. Es ist daher sinnvoll, nur die tatsächlichen Bestellungen zu erfassen. Zu wählen ist daher eine Modellierung, die

(1) für einen beliebigen Kunden die bestellten Artikel erfasst und

(2) für einen beliebigen Artikel die ordernden Kunden erfasst.

Einführung einer Hilfsentitätsmenge

Genau dies leistet die in Abb. 3.24 gezeigte Modellierung, die sich einer Hilfsentitätsmenge *Bestellungen* bedient und die an die Stelle der ursprünglichen mc-mc-Beziehung zwei (hierarchische) 1-mc-Beziehungen setzt.

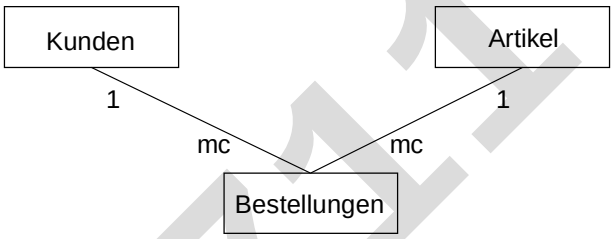


Abb. 3.24. Modellierung mit einer Hilfsentitätsmenge.

Einer Entität der Menge *Kunden* können nun beliebig viele bestellte Artikel zugeordnet sein und jede Bestellung bzw. jeder bestellte Artikel ist genau einem Kunden zugeordnet. Analoge Verhältnisse liegen aus der Sicht der Entitätsmenge *Artikel* vor.

Mit Hilfe mehrerer Hilfsentitätsmengen kann auch das ER-Diagramm in Abb. 3.16 so umformuliert werden, dass - bei gleichem Informationsgehalt - nur noch hierarchische Beziehungen auftreten. Das Ergebnis dieser Umformulierung findet sich in Abb. 3.25.

Netzwerk mit hierarchischen Beziehungen

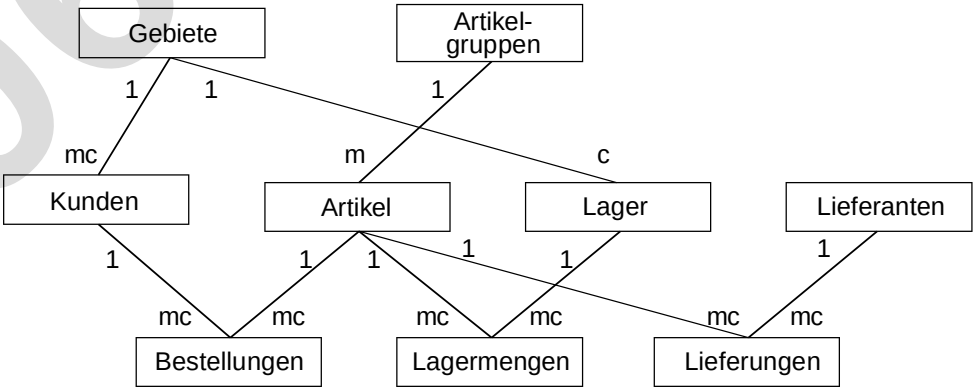


Abb. 3.25. Mit hierarchischen Beziehungen dargestelltes Netzwerk.

Auf die explizite Darstellung von Beziehungen mit Hilfe des Symbols der Raute wurde in Abb. 3.25 verzichtet. Aus der Bezeichnung der Hilfsentitätsmengen, nämlich *Bestellungen*, *Lagermengen* und *Lieferungen*, geht der Inhalt der neu eingeführten Beziehungen deutlich hervor.

Übungsaufgabe 3.12.

Betrachtet werde ein Ausschnitt aus einem hypothetischen Fertigungsbetrieb, der aus mehreren Fertigungsabteilungen besteht. In jeder Abteilung produzieren mehrere Werker mit Hilfe von Maschinen unterschiedliche Produkte. Jedes der Produkte wird in einem Arbeitsgang von einem Werker auf einer Maschine produziert. Einige Produkte können nur auf einer Maschine, andere auf mehreren Maschinen hergestellt werden. Jede Maschine kann mehrere Produkte fertigen und von verschiedenen Werkern bedient werden. Jeder Werker ist in der Lage, mehrere Maschinen zu bedienen.

Modellieren Sie die skizzierten Zusammenhänge mit Hilfe eines Netzwerks, in dem nur hierarchische Beziehungen auftreten. Gehen Sie hierbei in zwei Schritten vor: Erstellen Sie zunächst ein ER-Diagramm, in dem auch nichthierarchische Beziehungen auftreten, und transformieren Sie dieses Diagramm dann in die gewünschte Darstellungsform.

c) Darstellung von Netzwerken

zwei Darstellungsformen für Netzwerke

Da die sequentielle Speicherung nur für Hierarchien in Frage kommt, verbleiben für die Darstellung von Netzwerken nur zwei Alternativen:

- die direkte Darstellung mit Hilfe von Zeigern und
- die indirekte Darstellung mit Hilfe von gemeinsamen Attributen.

Beide Darstellungsformen seien nachfolgend kurz erläutert.

Betrachtet werde das der Abb. 3.19 und der Übungsaufgabe 3.10 zugrunde liegende einfache Beispiel einer mc-mc-Beziehung zwischen den Entitätsmengen *Kunden* und *Artikel*. Durch Einführung einer Hilfsentitätsmenge *Bestellungen* erhält man das in Abb. 3.24 gezeigte Netzwerk. Legt man die der Abb. 3.19 b) zu entnehmenden konkreten Daten zugrunde, so ergibt sich die in Abb. 3.26 angegebene indirekte Darstellung auf Entitätsebene.

indirekte Darstellung mit gemeinsamen Attributen

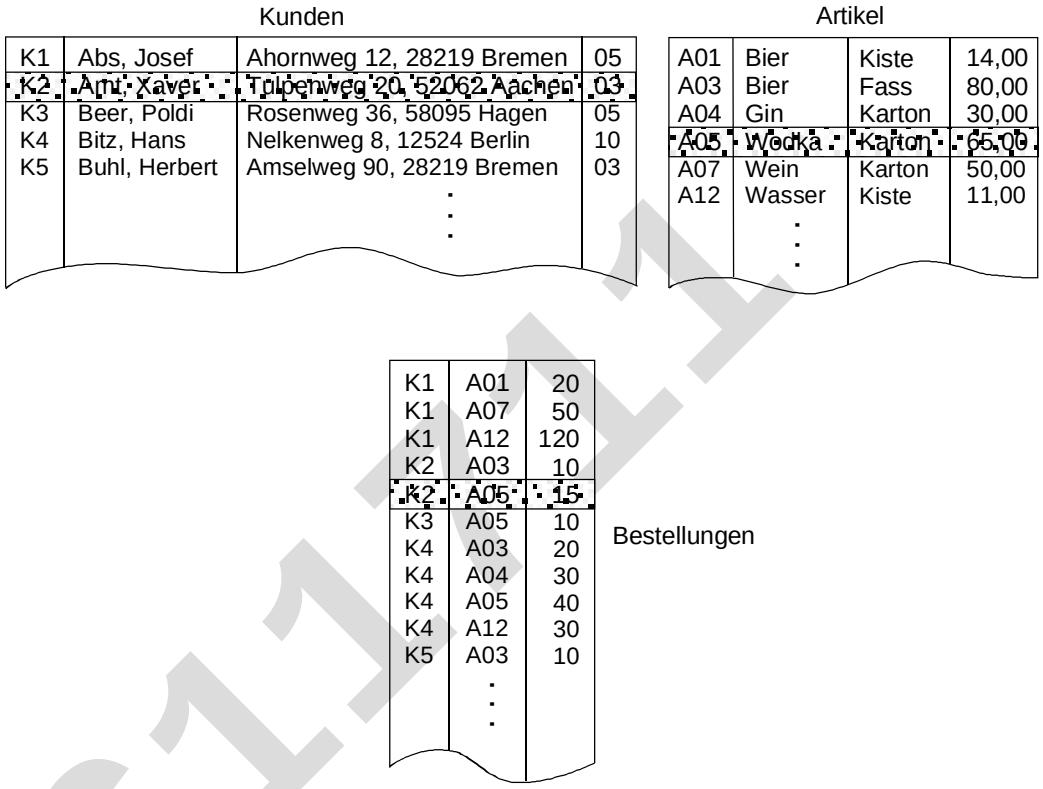


Abb. 3.26. Indirekte Darstellung der Beziehungen zwischen Entitäten in einem Netzwerk mit Hilfe gemeinsamer Attribute.

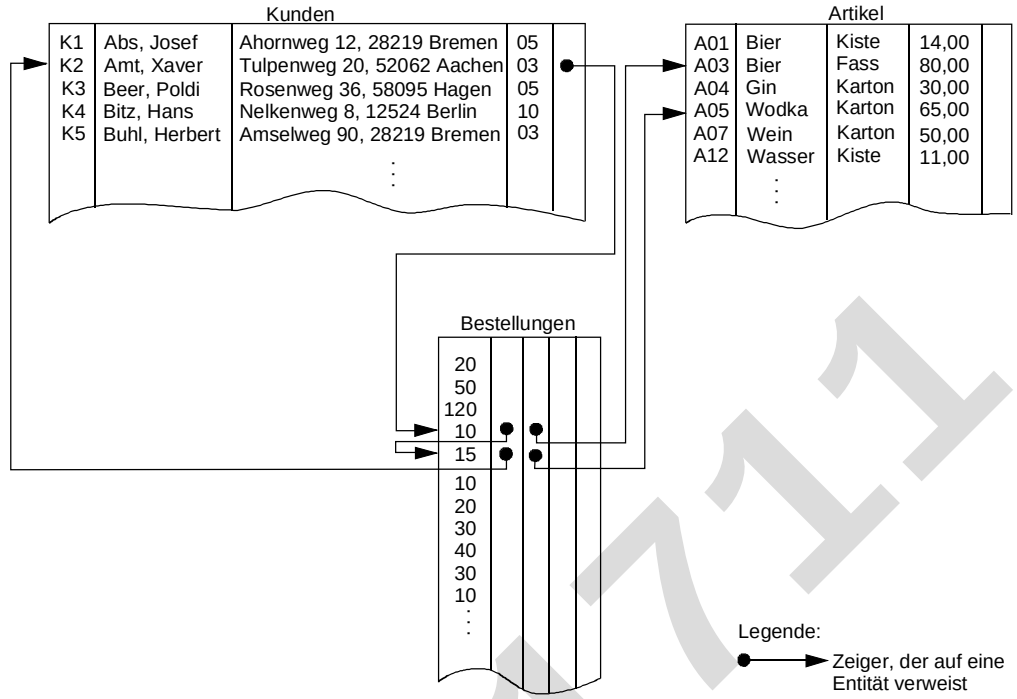
Durch das Unterlegen eines Rasters wird in Abb. 3.26 ein Beziehungszusammenhang hervorgehoben. Über die gemeinsamen Attributwerte *K2* in den Entitätsmengen *Kunden* und *Bestellungen* bzw. *A05* in den Entitätsmengen *Artikel* und *Bestellungen* wird ausgedrückt, dass der Kunde *Amt* 15 Kartons Wodka bestellt hat.

direkte Darstellung mit Zeigern

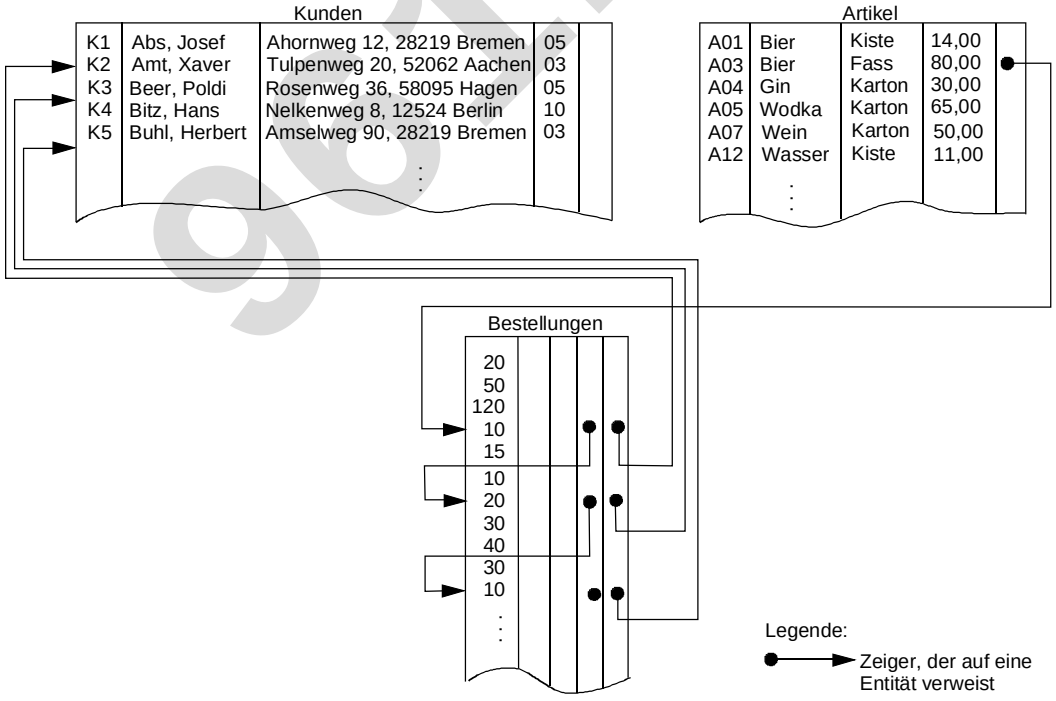
Bei der direkten Darstellungsform werden Beziehungen zwischen Entitäten durch Verweise mit Zeigern hergestellt. Neben Attributen, die den Charakter von Nutzdaten haben, müssen den Entitäten daher Hilfsdaten - nämlich Zeiger - zugeordnet werden. Für das der Abb. 3.26 zugrunde liegende Beispiel ist eine mögliche Form der direkten Darstellung mit Zeigern in Abb. 3.27 wiedergegeben. Um die Übersicht zu wahren, wurde eine Aufspaltung in zwei Teildarstellungen vorgenommen und außerdem die Menge der Beziehungen auf je ein Beispiel begrenzt:

- Die Abb. 3.27 a) zeigt die Verkettung der Artikel, die der Kunde *Amt* bestellt hat. Von diesem Kunden geht eine Ringkettung aus, die über die Bestellmengen 10 und 15 zum Ausgangspunkt zurückkehrt. Zwei weitere Zeiger ordnen diese Bestellmengen den Artikeln *A03* und *A05* zu. Der Kunde *Amt* hat also 10 Einheiten des Artikels *A03*, d.h. Bier im Fass, und 15 Einheiten des Artikels *A05*, d.h. Wodka im Karton, geordert.

- Die Abb. 3.27 b) zeigt die Verkettung der Kunden, die den Artikel *A03* bestellt haben. Von diesem Artikel wurden von den Kunden *Amt*, *Bitz* und *Buhl* 10, 20 und 10 Mengeneinheiten bestellt.



a) Verkettung der Artikel, die von einem bestimmten Kunden geordert wurden



b) Verkettung der Kunden, die einen bestimmten Artikel geordert haben

Abb. 3.27. Direkte Darstellung der Beziehungen zwischen Entitäten in einem Netzwerk mit Hilfe von Zeigern.

Laut Abb. 3.27 wird pro Entität der Entitätsmengen *Kunden* und *Artikel* ein Zeiger benötigt. Pro Entität der Entitätsmenge *Bestellungen* sind es dagegen vier benötigte Zeiger. Die direkte Darstellung von Netzwerken erfordert folglich mehr Speicherplatz als die indirekte Darstellung. Andererseits beansprucht die Auswertung von Beziehungen bei der direkten Darstellung wesentlich weniger Zeit als bei der indirekten Darstellungsform.

Übungsaufgabe 3.13

Betrachtet werde das in Abb. 3.27 gezeigte Beispiel für die direkte Darstellung von Beziehungen in einem Netzwerk. Aus welchem Grund ist es sinnvoll, in die Entitäten der Entitätsmenge *Bestellungen* zusätzlich die Attribute *KundenNr* und *ArtikelNr* einzubeziehen?

3.4 Relationenmodelle

3.4.1 Klassisches Relationenmodell

Relationen als redundanzfreie Datenanordnungen

Bereits im Jahre 1970 erschien die grundlegende Arbeit von CODD über das Relationenmodell. Als Mathematiker hat sich CODD in formaler Weise mit Datenmengen auseinandergesetzt. Das von ihm entwickelte Modell sieht die Bildung sogenannter Relationen vor. Darunter sind Anordnungen von Daten zu verstehen, die sich durch bestimmte Eigenschaften auszeichnen. CODD zielt insbesondere auf redundanzfreie Relationen ab, also auf Strukturen, die keine überflüssigen Daten enthalten. Erreicht werden solche Strukturen in einem mehrstufigen Prozess, den CODD als Normalisierung bezeichnet.

Normalisierung

Analyse von Abhängigkeiten

Der Prozess der Normalisierung besteht in einer Analyse der Abhängigkeiten zwischen den Daten einer Relation und in der Elimination unerwünschter Abhängigkeiten durch die sukzessive Aufspaltung der Relation in mehrere neue Relationen. Hierdurch werden zugleich Redundanzen beseitigt. Außerdem zeichnen sich die schließlich entstandenen Relationen dadurch aus, dass bei ihrer Manipulation keine sogenannten Update- oder Mutationsanomalien auftreten. Darunter sind Inkonsistenzen zu verstehen, die durch die Mutation von Relationen - d.h. durch das Einfügen, Ändern oder Löschen von Daten - bewirkt werden.

Mutationsanomalien

Auf diese grob umrissenen Zusammenhänge geht das vorliegende Kapitel in folgenden Schritten ein:

- Relationsbegriff,
- Abhängigkeiten und
- Normalisierungsprozess.

Mutationsanomalien werden im Zusammenhang mit dem Normalisierungsprozess behandelt.

Relationenmodell als formales Datenmodell

Hingewiesen sei noch darauf, dass CODD Anordnungen von Daten in formaler Weise betrachtet, ohne solche Anordnungen als Entitäten bzw. Entitätsmengen zu bezeichnen. Ihm geht es auch nicht primär um die Abbildung realer Phänomene durch Entitäten und deren Charakterisierung mit zugeordneten Attributen. Im Vordergrund steht vielmehr die Aufspaltung einer Datenmenge derart, dass Relationen mit bestimmten Eigenschaften entstehen. Erst Jahre später, nachdem CHEN sein "Entity Relationship-Modell" präsentiert hatte, wurde die Verbindung zwischen Relationen und Entitätsmengen hergestellt.

a) Begriff der Relation

In die nachfolgende mengentheoretische Definition des Begriffs "Relation" gehen die bereits bekannten Begriffe "Attribut" (engl. attribute) und "Wertebereich" (engl. domain) ein. Bezeichne A_i, $i = 1,..., n$, ein Attribut aus einer Menge von n Attributen und W_i, $i = 1,..., n$, den Wertebereich des i-ten Attributs. Bezeichne außerdem w_i einen beliebigen Wert aus der endlichen Menge von Werten, die der Wertebereich W_i umfasst, also $w_i \in W_i$, $i = 1,..., n$. Dann gilt:

mengentheoretische Definition der Relation

Eine Relation R ist eine Teilmenge des kartesischen Produkts der nicht notwendigerweise disjunkten Wertebereiche W_i von n Attributen A_i, $i = 1,..., n$:

$$R \subset W_1 \times W_2 \times ... \times W_n$$

Die Relation R stellt somit eine Menge von n-Tupeln dar:

$$R = \{(w_1, w_2,..., w_n) \mid w_i \in W_i, i = 1,..., n\}$$

Grad einer Relation

Hierbei bezeichnet der Begriff "n-Tupel" eine Anordnung von n Einzelwerten, die je den Wertebereichen der n Attribute der Relation entnommen sind. Der Wert n bestimmt den Grad einer Relation, d.h. die Anzahl der in der Relation enthaltenen Attribute.

Beispiel 3.1

Als Beispiel werde eine Relation 2-ten Grades mit den Attributen *Geschlecht* und *Religion* betrachtet. Angenommen seien folgende Wertebereiche für diese Attribute:

Geschlecht: $W_1 = \{m, w\}$,
Religion: $W_2 = \{ev, rk, sonst\}$.

Hierbei stehen die Werte "m" für männlich, "w" für weiblich, "ev" für evangelisch, "rk" für katholisch und "sonst" für sonstige Konfessionen.

Das kartesische Produkt W_1 x W_2 besteht aus der Menge aller möglichen 2-Tupel bzw. Paare (w_1, w_2):

W_1 x W_2 ={(m, ev), (m, rk), (m, sonst),
(w, ev), (w, rk), (w, sonst)}.

Eine Relation R als Teilmenge des kartesischen Produkts W_1 x W_2 ist dann beispielsweise gegeben durch:

R = {(m, ev), (w, rk), (w, sonst)}.

Durch den Vergleich der Ausdrücke für R und W_1 x W_2 lässt sich leicht feststellen, dass gilt: $R \subset W_1$ x W_2.

Relation als Teilmenge eines kartesischen Produkts

Bei dem sehr einfach gehaltenen Beispiel 3.1 mag nicht einleuchten, dass eine Relation als Teilmenge eines kartesischen Produkts definiert ist. Dies liegt daran, dass die Anzahl der 2-Tupel des kartesischen Produkts sehr klein ist. In weniger einfachen Fällen ist die Anzahl der möglichen Tupel so groß, dass die Vereinbarung einer Relation als eine Teilmenge der möglichen Tupel sinnvoll ist.

Übungsaufgabe 3.14

Mächtigkeit eines kartesischen Produkts

Gegeben sei eine Relation 4-ten Grades $R \subset W_1$ x W_2 x W_3 x W_4, die aus den Attributen *ArtikelNr*, *Artikelgruppe*, *Menge* und *Preis* bestehe. Für die Wertebereiche gelte:

ArtikelNr : W_1 = [111..999],
Artikelgruppe : W_2 = [A, B, C],
Menge : W_3 = [00..99],
Preis : W_3 = [00000..99999].

Schätzen Sie ab, aus wie vielen 4-Tupeln das kartesische Produkt W_1 x W_2 x W_3 x W_4 besteht.

Die in einer Relation enthaltenen Tupel repräsentieren jeweils ein konkretes Datenaufkommen. In Verbindung mit dem Ergebnis der Übungsaufgabe 3.14 leuchtet daher die Definition einer Relation *n*-ten Grades als Teilmenge aller möglichen n-Tupel ein.

Relation als Tabelle

Eine Relation lässt sich auch in tabellarischer Form darstellen. Die Abb. 3.28 zeigt eine Relation 8-ten Grades in Tabellenform. Die Relation besteht aus den Attributen *KundenNr*, *Name*, *Adresse*, *Umsatz* und *Jahr*. Exemplarisch sind vier 5-Tupel angegeben. Jedes 5-Tupel ist eine wertmäßige Ausprägung der Relation.

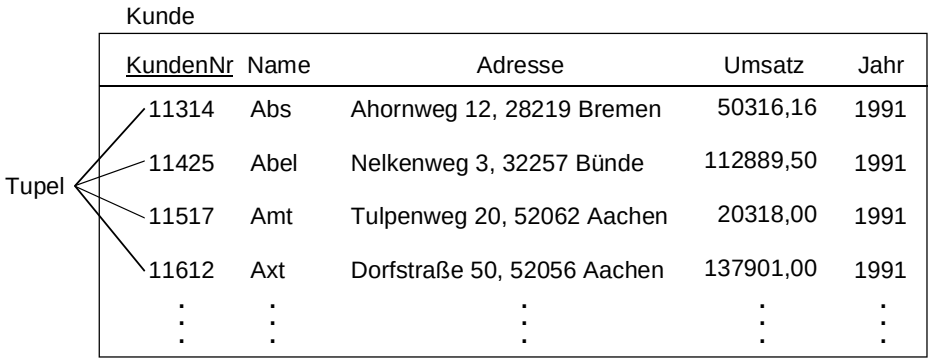


Abb. 3.28. Tabellarische Darstellung einer Relation 5-ten Grades.

Eigenschaften von Relationen

Zur weiteren Verdeutlichung des Wesens einer Relation seien einige ihrer Eigenschaften genannt:

- Da die Attribute durch Namen identifiziert werden, ist die Anordnung der Attribute nicht signifikant; die Attribute können also beliebig angeordnet werden.
- Sämtliche Tupel sind völlig gleichberechtigt; die Anordnung der Tupel einer Relation ist folglich ebenfalls nicht signifikant.
- Identische Tupel treten nicht auf; zwei beliebige Tupel unterscheiden sich daher in mindestens einem Attributwert.
- Aus der letztgenannten Eigenschaft folgt: Es existieren stets ein oder mehrere Attribute, deren Werte jeden Tupel der Relation eindeutig identifizieren.
- Die Werte der Attribute sind atomar, d.h. ein Attribut nimmt in einer Relation genau einen Wert und nicht etwa eine Folge von Werten an.

Identifikation von Tupeln

Um die Tupel einer Relation gezielt ansprechen und manipulieren zu können, bedarf es eines Identifikationsmittels. Jeder Tupel einer Relation ist durch die Kombination seiner Attributwerte eindeutig bestimmt. Eine eindeutige Identifikation kann meist aber auch durch eine Teilmenge von Attributen erreicht werden. Zielt man auf eine möglichst geringe Anzahl von identifizierenden Attributen ab, so kommt man zu dem von CODD geprägten Begriff des Schlüsselkandidaten:

Begriff des Schlüsselkandidaten

> Ein Schlüsselkandidat (engl. candidate key) ist ein Attribut oder eine minimale Kombination von Attributen, welche jeden Tupel einer Relation eindeutig identifiziert.

Bekanntlich besagt die Eigenschaft der Minimalität, dass im Fall einer Attributkombination kein Attribut aus der Kombination entfernt werden kann, ohne dass die Eindeutigkeit der Identifizierung verloren geht.

Beispielsweise ist in der in Abb. 3.28 gezeigten Relation das Attribut *KundenNr* ein Schlüsselkandidat. Würde in dieser Relation das Attribut *KundenNr* fehlen, so wäre die Attributkombination *Name, Adresse* ein Schlüsselkandidat.

Im weiteren Verlauf dieses Kurses werden Relationen ausschließlich zur inhaltlichen Charakterisierung von Entitätsmengen verwendet. Es erscheint daher vertretbar, den Begriff Schlüsselkandidat mit dem im Kapitel 3.1 eingeführten Begriff des Identifikationsschlüssels gleichzusetzen und im Folgenden nur noch einen dieser beiden Begriffe - nämlich Identifikationsschlüssel - zu verwenden.

atomare Attributwerte

Besonders hinzuweisen ist noch auf die Eigenschaft atomarer Attributwerte. Eine Relation befindet sich daher grundsätzlich in der weiter unten eingeführten 1. Normalform. Auf diesen Sachverhalt wird bei der Behandlung des Normalisierungsprozesses noch eingegangen.

b) Abhängigkeiten

Nachfolgend werden einige Abhängigkeitsbegriffe eingeführt. Sie beschreiben bestimmte Zusammenhänge zwischen den Attributen einer Relation und werden bei der Formulierung der verschiedenen Normalformen von Relationen benötigt.

Bezeichne *R* eine Relation und bezeichnen *A* und *B* je ein Attribut oder eine Attributkombination der Relation *R*. Dann gilt für den Begriff der funktionalen Abhängigkeit:

Begriff der funktionalen Abhängigkeit

Das Attribut bzw. die Attributkombination *B* ist genau dann funktional abhängig von dem Attribut bzw. der Attributkombination *A*, wenn zu einem beliebigen Wert von *A* in der Relation *R* höchstens ein Wert von *B* existiert.

Formal wird dieser Zusammenhang wie folgt ausgedrückt:

$$R.A \rightarrow R.B$$

Um diesen Abhängigkeitsbegriff an einem Beispiel verdeutlichen zu können, sei folgende Schreibweise einer Relation eingeführt:

Schreibweise einer Relation

Relationsname(Attribut1, Attribut2, Attribut3,...)

Eine Relation wird also durch einen Relationsnamen identifiziert. Dem Relationsnamen folgen in einer runden Klammer und getrennt durch Kommata die Attribute der Relation.

Gegeben sei nun die Relation:

Artikel(ArtikelNr, Bezeichnung, Gruppe, Preis)

In dieser Relation tritt folgende funktionale Abhängigkeit auf:

$$Artikel.ArtikelNr \rightarrow Artikel.(Bezeichnung, Gruppe, Preis)$$

Die Attributkombination (*Bezeichnung, Gruppe, Preis*) ist von dem Attribut *ArtikelNr* funktional abhängig, weil zu jedem Wert von *ArtikelNr* stets nur eine Wertekombination der Attribute *Bezeichnung, Gruppe* und *Preis* auftritt.

Übungsaufgabe 3.15

Gegeben sei eine Relation *Bestellung* mit folgenden Attributen:

Bestellung(KundenNr, ArtikelNr, Datum, Menge)

Geben Sie an, zwischen welchen Attributen bzw. Attributkombinationen funktionale Abhängigkeiten bestehen.

Eine noch stärkere Form des Zusammenhangs zwischen Attributen liegt bei der vollfunktionalen Abhängigkeit vor. Die vollfunktionale Abhängigkeit ist wie folgt definiert:

Begriff der vollfunktionalen Abhängigkeit

Ein Attribut bzw. eine Attributkombination *B* einer Relation *R* ist vollfunktional abhängig von dem Attribut bzw. der Attributkombination *A* derselben Relation *R*, wenn:

- *B* von *A* funktional abhängig ist, d.h. $R.A \rightarrow R.B$, und
- *B* nicht schon allein von einem Teil von *A* funktional abhängig ist.

Formale Schreibweise:

$$R.A \Rightarrow R.B$$

Im Fall der Relation *Artikel(ArtikelNr, Bezeichnung, Gruppe, Preis)* gilt:

Artikel.ArtikelNr $\Rightarrow$ *Artikel.(Bezeichnung, Gruppe, Preis)*

Ist nämlich eine Attributkombination funktional von einem einzelnen Attribut abhängig, so liegt zugleich auch vollfunktionale Abhängigkeit vor.

Nun zu einem weniger einfachen Beispiel: Für die Relation

Lagerbestand(LagerNr, ArtikelNr, Bezeichnung, Datum, Bestand)

gilt:

Lagerbestand.(*LagerNr*, *ArtikelNr*, *Datum*) $\Rightarrow$ *Lagerbestand.Bestand,*

denn der Bestand pro Artikel hängt von dem betrachteten Lager und dem Betrachtungszeitpunkt ab. Dagegen hängt die Artikelbezeichnung nur von der Artikelnummer ab. *Be-*

zeichnung ist deshalb von der Attributkombination (*LagerNr, ArtikelNr, Datum)* nicht vollfunktional abhängig.

Übungsaufgabe 3.16

Gegeben sei folgende, gegenüber der Übungsaufgabe 3.15 erweiterte Relation *Bestellung*:

Bestellung(KundenNr, Name, ArtikelNr, Bezeichnung, Datum, Menge)

Geben Sie an, zwischen welchen Attributen bzw. Attributkombinationen vollfunktionale Abhängigkeit besteht.

Die funktionale und die vollfunktionale Abhängigkeit weisen eine Gemeinsamkeit auf: Sie beschreiben unmittelbare Zusammenhänge zwischen Attributen bzw. Attributkombinationen. Bei der transitiven Abhängigkeit geht es dagegen um mittelbare Zusammenhänge zwischen Attributen bzw. Attributkombinationen. Die transitive Abhängigkeit ist wie folgt definiert:

Begriff der transitiven Abhängigkeit

Seien *A, B* und *C* Attribute bzw. Attributkombinationen einer Relation *R*, dann heißt *C* transitiv abhängig von *A*, wenn gilt:

- $R.A \rightarrow R.B$, d.h. *B* ist funktional abhängig von *A*, und
- $R.B \rightarrow R.C$, d.h. *C* ist funktional abhängig von *B*.

C ist also mittelbar, über *B*, von *A* abhängig.

Betrachtet werde die Relation *Mitarbeiter*:

Mitarbeiter(MitarbeiterNr, Name, Gehalt, AbteilungsNr, Abteilungsname)

In dieser Relation ist *Abteilungsname* transitiv von *MitarbeiterNr* abhängig, denn es gilt:

- *AbteilungsNr* ist funktional abhängig von *MitarbeiterNr* und
- *Abteilungsname* ist funktional abhängig von *AbteilungsNr*.

Bei der Einführung der obigen Abhängigkeitsbegriffe war es nicht notwendig, zwischen Schlüsselattributen, d.h. zu Identifikationsschlüssel gehörigen Attributen, und Nichtschlüsselattributen zu unterscheiden. Diese Unterscheidung spielt bei der Vorgehensweise der Normalisierung eine Rolle. Wie sich noch zeigen wird, basiert der Ansatz der Normalisierung insbesondere auf der Analyse der Abhängigkeiten zwischen Schlüssel- und Nichtschlüsselattributen.

c) Normalisierungsprozess

Zu Beginn dieses Kapitels wurde darauf hingewiesen, dass durch die Normalisierung von Daten Mutationsanomalien vermieden werden, die als Folge von Redundanzen in nicht normalisierten Datenanordnungen oder als Folge nicht normalisierter Datenanordnungen auftreten können. Zu unterscheiden ist zwischen Anomalien, die beim Einfügen, Ändern und Löschen von Daten auftreten. Bevor nun der Normalisierungsprozess behandelt wird, sollen die verschiedenen Arten von Anomalien an einem Beispiel demonstriert werden.

Mutationsanomalien

Beispiel 3.2

Gegeben seien drei Relationen, die sich auf die Forschungsförderung durch ein Unternehmen der öffentlichen Verwaltung beziehen:

Programme(ProgrammNr, Fachgebiet, Budget, Laufzeit)
Institute(InstitutsNr, Name, Mitarbeiterzahl, Forschungsgebiet)
Vorhaben(ProgrammNr, InstitutsNr, Institutsadresse, Fördermittel)

Von Zeit zu Zeit legt das Unternehmen neue Förderprogramme auf, deren Zweck in der Unterstützung der Forschung in bestimmten Fachgebieten dient. Ein Programm erstreckt sich über eine bestimmte Laufzeit und umfasst ein bestimmtes Budget.

Um Mittel aus den Förderprogrammen können sich universitäre und nichtuniversitäre Forschungsinstitute bewerben. Nur Institute, deren Forschungsgebiet eines der geförderten Fachgebiete überdeckt und die Fördermittel für ein erfolgversprechendes Forschungsvorhaben beantragt haben, können mit einer Förderung rechnen.

In der Regel übersteigen die zu einem Programm beantragten Fördermittel das verfügbare Budget. In die Förderung werden daher solange förderungswürdige Vorhaben in der zeitlichen Reihenfolge der Antragstellung einbezogen, bis das Budget erschöpft ist.

Bei der Manipulation der angegebenen Relationen, d.h. bei dem Einfügen, Löschen und Ändern von Tupeln, können

- Einfügeanomalien,
- Löschanomalien und
- Änderungsanomalien

auftreten.

Arten von Mutationsanomalien

(1) Einfügeanomalie (engl. insertion anomaly)

Jedes Institut, das einen Förderantrag stellt, wird in der Relation *Institute* verzeichnet und zwar unabhängig davon, ob eine Förderung gewährt wird oder nicht. Stellt nun ein Institut, das noch nicht in der Relation *Institute* verzeichnet ist, einen Förderantrag und wird dieser Antrag abgelehnt, so kommt es zu einer Einfügeanomalie:

Da ein Eintrag in die Relation *Institute* vorgenommen wird, nicht aber in die Relation *Vorhaben*, kann die Institutsadresse nicht eingefügt werden.

Einfügeanomalie

(2) Löschanomalie (engl. deletion anomaly)

Sobald ein Förderungsprogramm ausgelaufen ist, werden die entsprechenden Tupel aus der Relation *Vorhaben* entfernt. Dies hat eine Löschanomalie zur Folge:

Löschanomalie

Die Adressen sämtlicher Institute, die nur an dem ausgelaufenen Förderungsprogramm beteiligt waren, sind nun nicht mehr verfügbar.

(3) Änderungsanomalie (engl. update anomaly)

Änderungsanomalie

Ändert sich die Adresse eines Instituts, so müssen alle Tupel der Relation *Vorhaben* aufgesucht und die betroffenen Tupel geändert werden. Obwohl also nur *ein* Tatbestand geändert wird müssen *mehrere* Änderungen vorgenommen werden.

Die beschriebenen Anomalien sind die Folge ungeeignet gebildeter Relationen. Sie treten nicht mehr auf, falls man das Attribut *Institutsadresse* der Relation *Institute* zuordnet.

Durch die Normalisierung von Relationen lassen sich Mutationsanomalien verhindern. Der Prozess der Normalisierung besteht in der stufenweisen Bildung sogenannter Normalformen. Im vorliegenden Kurs werden die in Abb. 3.29 genannten Normalisierungsstufen behandelt.

Normalisierungsprozess

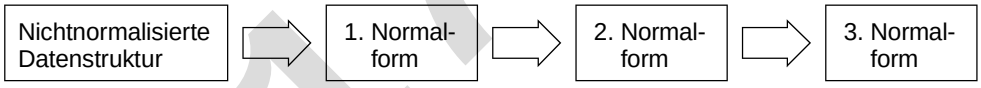


Abb. 3.29. Normalisierungsprozess.

In der Literatur werden noch weitere Normalisierungsstufen, die zur 4. Normalform und weiteren Normalformen führen, betrachtet. Für die Modellierung wirtschaftswissenschaftlicher Problemstellungen reichen die hier betrachteten Normalisierungsschritte völlig aus. Die Darstellung weiterer Normalformen erübrigt sich daher.

1. Normalform

Definitionsgemäß darf eine Relation keine sogenannten wiederholenden Gruppen (engl. repeating groups), d.h. mehrere Attributwerte pro Attribut, enthalten. Eine Relation im hier eingeführten CODD'schen Sinne befindet sich daher automatisch in 1. Normalform. Festzuhalten ist also:

Begriff der 1. Normalform

Eine Relation *R* ist in 1. Normalform, da sie nur atomare Attributwerte und keine wiederholenden Gruppen enthält.

An einem Beispiel sei gezeigt, wie der Übergang von einer nichtnormalisierten Datenstruktur zu einer Relation in 1. Normalform zu vollziehen ist.

Beispiel 3.3

Betrachtet werde eine Datenstruktur, welche die Bestellung von Büchern durch Kunden einer Buchhandlung beschreibt:

Datenstruktur mit wiederholenden Gruppen

KundenNr	Datum	ISBN	Autor	Menge	Preis	Betrag
107	92 04 01	25; 76; 93	Abs; Beer; Hein	2; 1; 3	8,50; 12,30; 5,60	17,00; 12,30; 16,80
126	92 04 01	76; 81	Beer; Mann	1; 2	12,30; 16,10	12,30; 32,20
215	92 04 01	81	Mann	3	16,10	48,30
219	92 04 01	25; 93	Abs; Hein	1; 1	8,50; 5,60	8,50; 5,60
⋮						

In den Elementen dieser Datenstruktur treten wiederholende Gruppen bzw. nicht atomare Attribute auf. Beispielsweise hat der Kunde mit der Kundennummer 219 zwei Bücher bestellt. Wiederholte Werte liegen daher für die Attribute *ISBN*, *Autor*, *Menge*, *Preis* und *Betrag* vor.

Elimination wiederholender Gruppen

Eliminiert man die wiederholenden Gruppen, so geht die Datenstruktur in eine Relation über, die sich in 1. Normalform befindet. Die tabellarische Darstellung dieser Relation lautet:

Kundenorder

KundenNr	Datum	ISBN	Autor	Menge	Preis	Betrag
107	92 04 01	25	Abs	2	8,50	17,00
107	92 04 01	76	Beer	1	12,30	12,30
107	92 04 01	93	Hein	3	5,60	16,80
126	92 04 01	76	Beer	1	12,30	12,30
126	92 04 01	81	Mann	2	16,10	32,20
215	92 04 01	81	Mann	3	16,10	48,30
219	92 04 01	25	Abs	1	8,50	8,50
219	92 04 01	93	Hein	1	5,60	5,60
⋮						

Die wiederholenden Gruppen wurden je in mehrere Tupel aufgelöst. Zur Identifikation der einzelnen Tupel dient ein zusammengesetzter Schlüssel, der aus den Attributen *KundenNr*, *Datum* und *ISBN* besteht. Um den Identifikationsschlüssel zu kennzeichnen, wurden die Schlüsselattribute in der Kopfzeile der obigen Tabelle unterstrichen.

Als Ergebnis des ersten Normalisierungsschrittes erhält man damit folgende Relation *Kundenorder*:

Kundenorder(KundenNr, Datum, ISBN, Autor, Menge, Preis, Betrag)

Relation in 1. Normalform

Man beachte, dass durch den Übergang von der nichtnormalisierten Datenstruktur in Beispiel 3.3 zu einer Relation in 1. Normalform kein Informationsverlust eingetreten ist. Die Relation enthält jedoch Redundanzen, die in der tabellarischen Darstellungsform besonders deutlich werden: So ist der Autorenname eines Buches durch die ISBN eindeutig festgelegt. Bei einem mehrmals bestellten Buch müsste der Autorenname eigentlich nicht wiederholt werden. Außerdem kann bei dieser Relation eine

Mutationsanomalie auftreten: Wird in dem ersten Tupel der Relation der Preis von 8,50 auf 9,80 geändert und existieren weitere dieses Buch betreffende Tupel, so liegen widersprüchliche Daten vor.
Die angesprochenen Redundanzen und Anomalien sind darauf zurückzuführen, dass die Relation gleichzeitig mehrere voneinander unabhängige Phänomene beschreibt. In der Tat stellen *Kundenbestellungen* und *Bücher* in der Relation enthaltene eigenständige Phänomene dar, die sich zu unterschiedlichen Zeitpunkten ändern können. Anzustreben ist daher eine Aufspaltung der Relation im nächsten Normalisierungsschritt. Er führt zur 2. Normalform.

2. Normalform

Für eine Relation in 2. Normalform gilt:

Begriff der 2. Normalform

> Eine Relation *R* ist in 2. Normalform, wenn sie sich bereits in 1. Normalform befindet und wenn jedes Nichtschlüsselattribut vom Identifikationsschlüssel vollfunktional abhängig ist.

Unter einem Nichtschlüsselattribut ist hierbei ein Attribut zu verstehen, welches nicht zu dem gegebenenfalls zusammengesetzten Identifikationsschlüssel gehört.

Gefordert wird also, dass jedes Nichtschlüsselattribut vom ganzen Identifikationsschlüssel abhängt. Existieren in einer Relation Nichtschlüsselattribute, die nur von einem Teil des zusammengesetzten Schlüssels abhängen, so lassen sie sich in eine neue, abgespaltene Relation einbringen. Demonstriert sei dies im Rahmen der Fortsetzung des Beispiels 3.3.

Fortsetzung Beispiel 3.3

Betrachtet werde die in 1. Normalform befindliche Relation *Kundenorder*. In dieser Relation sind vollfunktionale Abhängigkeiten vorhanden, die sich wie folgt darstellen lassen:

Kundenorder.(*KundenNr*, *Datum*, *ISBN*) ⇒ *Kundenorder*.(*Menge*, *Betrag*)

K*undenorder*.*ISBN* ⇒ *Kundenorder*.(*Autor*, *Preis*)

Die Darstellung drückt aus, dass

vollfunktionale und funktionale Abhängigkeiten

- die Attribute *Menge* und *Betrag* vom ganzen zusammengesetzten Schlüssel vollfunktional abhängig sind und
- die Attribute *Autor* und *Preis* lediglich von dem Schlüsselattribut *ISBN* voll- funktional abhängig sind.

Die angegebenen Abhängigkeiten gestatten eine Aufspaltung der Relation *Kundenorder* in zwei Relationen *Buch* und *Bestellung*:

Buch

ISBN	Autor	Preis
25	Abs	8,50
76	Beer	12,30
93	Hein	5,60
81	Mann	16,10
⋮		

Bestellung

KundenNr	Datum	ISBN	Menge	Betrag
107	920401	25	2	17,00
107	920401	76	1	12,30
107	920401	93	3	16,80
126	920401	76	1	12,30
126	920401	81	2	32,20
215	920401	81	3	48,30
219	920401	25	1	8,50
219	920401	93	1	5,60
⋮				

Aufspaltung in zwei Relationen

Die beiden Relationen

Buch(ISBN, Autor, Preis)

Bestellung(KundenNr, Datum, ISBN, Menge, Betrag)

Relationen in 2. Normalform

befinden sich in 2. Normalform, da sämtliche Nichtschlüsselattribute in diesen Relationen vom jeweiligen Identifikationsschlüssel vollfunktional abhängig sind.

Das Bilden der zweiten Normalform hat keinen Informationsverlust zur Folge. Zwar führt das in Beispiel 3.3 demonstrierte Aufspalten einer Relation zu neuen Relationen, doch bleiben die zuvor vorhandenen inhaltlichen Zusammenhänge erhalten und zwar in indirekter Form über gemeinsame Attribute. Zwischen den Tupeln der Relationen *Buch* und *Bestellung* in Beispiel 3.3 werden inhaltliche Verbindungen nämlich über das in beiden Relationen enthaltene Attribut *ISBN* hergestellt.

innere Redundanzen

Im Gegensatz zur Relation *Buch* weist die Relation *Bestellung* noch innere Redundanzen auf, denn mit der Menge liegt in jedem Tupel auch der Betrag fest. Mutationsanomalien können daher durchaus noch auftreten. Beispielsweise dann, wenn in einem Tupel lediglich der Wert von *Menge* geändert wird. Um solche Anomalien zu vermeiden, ist ein weiterer Normalisierungsschritt erforderlich, der die 3. Normalform herstellt.

3. Normalform

Die Definition der 3. Normalform lautet:

Begriff der 3. Normalform

Eine Relation *R* ist in 3. Normalform, wenn sie sich bereits in 2. Normalform befindet und wenn kein Nichtschlüsselattribut transitiv vom Identifikationsschlüssel abhängt.

Aus dem Verbot transitiver Abhängigkeiten resultiert ein weiteres Kriterium für das Aufspalten von Relationen. Die Handhabung dieses Kriteriums sei an dem bekannten Beispiel demonstriert.

Fortsetzung Beispiel 3.3

Ausgangspunkt der Betrachtung sind die beiden in 2. Normalform befindlichen Relationen *Buch* und *Bestellung*. In der Relation *Buch* hängt kein Nichtschlüsselattribut transitiv vom Identifikationsschlüssel ab. Für die Relation *Bestellung* gilt jedoch:

Bestellung (*KundenNr*, *Datum*, *ISBN*) → *Bestellung*.(*ISBN*, *Menge*)

Bestellung.(*ISBN*, *Menge*) → *Bestellung.Betrag*

transitive Abhängigkeit

Das Attribut *Betrag* hängt somit transitiv vom Identifikationsschlüssel ab. Beheben lässt sich diese unerwünschte Abhängigkeit durch das Abspalten des Attributs *Betrag* und das Bilden einer neuen Relation *Rechnung*, die neben dem Attribut *Betrag* als Identifikationsschlüssel das neu eingeführte Attribut *RechnungsNr* enthält. Damit ergeben sich folgende in 3. Normalform befindliche Relationen:

Buch(ISBN, Autor, Preis),

Bestellung(KundenNr, Datum, ISBN, Menge, RechnungsNr)

Rechnung(RechnungsNr, Betrag).

Die tabellarische Darstellung dieser Relationen lautet:

Relationen in 3. Normalform

Buch

ISBN	Autor	Preis
25	Abs	8,50
76	Beer	12,30
93	Hein	5,60
81	Mann	16,10
:		

Bestellung

KundenNr	Datum	ISBN	Menge	RechnungsNr
107	920401	25	2	27
107	920401	76	1	28
107	920401	93	3	29
126	920401	76	1	30
126	920401	81	2	31
215	920401	81	3	32
219	920401	25	1	33
219	920401	93	1	34
:				

Rechnung

RechnungsNr	Betrag
27	17,00
28	12,30
29	16,80
30	12,30
31	32,20
32	48,30
33	8,50
34	5,60
:	

Offensichtlich hat der Übergang zur 3. Normalform keinen Informationsverlust zur Folge, denn über das Attribut *RechnungsNr* werden die inhaltlichen Bezüge zwischen den Relationen *Bestellung* und *Rechnung* hergestellt.

Übungsaufgabe 3.17

Gegeben sei eine nichtnormalisierte Datenstruktur mit folgenden Attributen und beispielhaften Attributwerten:

KundenNr	Name	Kundenart	Rabatt	ArtikelNr	Bezeichnung	Verpackung	Preis	Menge
23	Just	A	0	228; 419	Mehl; Gries	Sack; Kiste	49,00;	3; 5
41	Held	C	5	419	Gries	Kiste	52,00	12
36	Mag	B	2	346; 419	Kleie; Gries	Sack; Kiste	52,00	10; 8
76	Hug	A	0	510	Zucker	Sack	36,00;	18
19	Bast	A	0	228	Mehl	Sack	52,00	2
.							64,00	
.							49,00	
.								

Offensichtlich beschreibt die Struktur die Bestellung von einem oder mehreren Artikeln durch Kunden. Die Bestellungen mögen alle am gleichen Tage erfolgen; auf das Attribut Datum wurde daher verzichtet. Die Kunden sind in die Kundenarten A, B und C eingeteilt; Kunden der Art A erhalten 0% Rabatt, der Art B 2% Rabatt und der Art C 5% Rabatt. Die übrigen Attribute bedürfen keiner Erläuterung.

Normalisieren Sie diese Datenstruktur, d.h. überführen Sie die Struktur schrittweise in die 3. Normalform. Prüfen Sie die Struktur vor dem ersten Normalisierungsschritt auf wiederholende Gruppen und stellen Sie vor den beiden folgenden Normalisierungsschritten fest, ob unerwünschte Abhängigkeiten vorliegen. Bezeichnen Sie die durch Aufspaltung entstehenden Relationen in geeigneter Weise und stellen Sie die Relationen auch in tabellarischer Form dar.

Verknüpfung von Relationen über redundante Attribute

Wie das Beispiel 3.3 und die Übungsaufgabe 3.17 zeigen, werden durch den Normalisierungsprozess unerwünschte Redundanzen eliminiert. Andererseits entstehen neue Redundanzen im Zuge der Aufspaltung von Datenstrukturen in Relationen. Beispielsweise durch die mehrfache Verwendung bestimmter Attribute als Schlüsselattribute. So ist das Attribut *RechnungsNr* in Beispiel 3.3 sowohl in der Relation *Bestellung*, als auch in der Relation *Rechnung* enthalten. Diese Mehrfachspeicherung ist unumgänglich, denn sie ermöglicht die inhaltliche Verknüpfung von Relationen. Inhaltliche Verknüpfungen von Relationen stellen aber nichts anderes dar, als die zwischen den Relationen bestehenden Beziehungen. Festzuhalten ist daher:

Beziehung über gemeinsame Attribute

Eine Beziehung zwischen zwei Relationen *R1* und *R2* wird durch ein sowohl in *R1*, als auch in *R2* auftretendes, gemeinsames Attribut bzw. eine in beiden Relationen auftretende gemeinsame Attributkombination gebildet.

explizite Relationenschreibweise bzw. Relationendiagramm

Die inhaltlichen Beziehungen zwischen Relationen lassen sich auch grafisch darstellen. Auf diese Weise erhält man ein relationales Datenmodell in expliziter Relationenschreibweise, auch Relationendiagramm genannt. Für die in Beispiel 3.3 angegebenen Relationen beispielsweise ergibt sich das in Abb. 3.30 gezeigte Datenmodell in expliziter Relationenschreibweise.

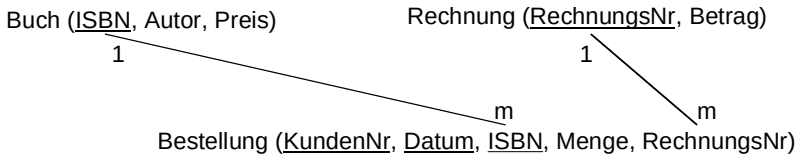


Abb. 3.30. Einfaches relationales Datenmodell in expliziter Relationenschreibweise.

Verzicht auf Beziehungssymbol

An die in Abb. 3.30 eingeführten Verbindungslinien sind beidseitig Assoziationssymbole angefügt worden. Die Verbindungslinien repräsentieren damit Beziehungsmengen - allerdings wurde hierbei einfachheitshalber auf die Raute als Beziehungssymbol verzichtet.

Wie das Relationendiagramm in Abb. 3.30 zeigt, wird die zwischen den Relationen *Buch* und *Bestellung* bestehende Beziehung durch ein gemeinsames Attribut, die *ISBN*, hergestellt. Die *ISBN* bildet in der Relation *Buch* den Identifikationsschlüssel; in der Relation *Bestellung* ist die *ISBN* dagegen nur Bestandteil des Identifikationsschlüssels. Die Beziehung zwischen den Relationen *Rechnung* und *Bestellung* wird, wie bereits erwähnt, durch das gemeinsame Attribut *RechnungsNr* hergestellt. Während die *RechnungsNr* in der Relation *Rechnung* den Identifikationsschlüssel bildet, ist sie in der Relation Bestellung ein Nichtschlüsselattribut.

Für das Datenmodell in Abb. 3.30 lässt sich unmittelbar das dazugehörige ER-Diagramm angeben. Es ist in Abb. 3.31 dargestellt, wobei ebenfalls auf die Raute als Beziehungssymbol verzichtet wurde.

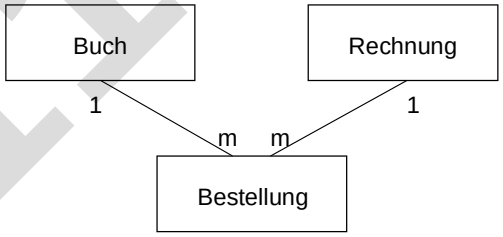


Abb. 3.31. ER-Diagramm für ein einfaches relationales Datenmodell.

Ergebnisse des relationalen Modellierungsansatzes sind beide Darstellungen: Das in Abb. 3.30 gezeigte relationale Datenmodell und das dazugehörige ER-Diagramm gemäß Abb. 3.31. Während das ER-Diagramm eine Übersicht über die gebildeten Relationen und die zwischen ihnen bestehenden Beziehungen gibt, gehen aus dem Datenmodell in expliziter Relationenschreibweise zusätzlich die die Relationen charakterisierenden Attribute und die Identifikationsschlüssel hervor.

3.4.2 Unternehmensdatenmodellierung mit einem erweiterten Relationenmodell

Verschiedene Schwachstellen schränken die Eignung des eben in Kapitel 3.4.1 vorgestellten relationalen Ansatzes zur unternehmensweiten Datenmodellierung ein. Die Schwachstellen betreffen:

Schwachstellen des Relationenmodells

- Redundanzen bei überlappenden Entitätsmengen,
- die Integrität der Daten und
- die Genauigkeit der Modellierung.

Ein relationales Datenmodell, dessen Relationen sich durchweg in 3. Normalform befinden, kann durchaus noch unerwünschte Redundanzen enthalten. Solche Redundanzen treten beispielsweise auf, wenn zwei verschiedene Relationen eine gleiche Eigenschaft beschreiben.

Beziehungen zwischen Entitätsmengen bzw. Relationen werden, wie aus Kapitel 3.4.1 hervorgeht, durch gleiche Attribute in zwei Relationen ausgedrückt (vgl. hierzu Abb. 3.30). Nach welchen Regeln dies zu geschehen hat, bleibt offen. Es kann daher zu Verletzungen der Integrität der gemäß dem Datenmodell angeordneten Daten kommen; die Daten sind dann nicht mehr widerspruchsfrei.

Die Modellierungsgenauigkeit des vorgestellten Relationenmodells ist begrenzt. So lassen sich z.B. Ober- und Untermengenbeziehungen zwischen Entitätsmengen, wie sie in der betrieblichen Praxis nicht selten auftreten, nicht immer präzise modellieren.

erweiterte Relationenmodelle

Die genannten Schwachstellen führten zur Entwicklung von erweiterten Relationenmodellen. Letztere bauen auf dem klassischen Relationenmodell von CODD und auf dem Entity Relationship-Modell von CHEN auf. Darüber hinaus verfügen erweiterte Relationenmodelle über Konzepte zur Vermeidung oder Milderung der obigen Schwachstellen. Zu nennen sind u.a.:

Konzepte in erweiterten Relationenmodellen

- Das Konzept globaler und lokaler Attribute in Verbindung mit statischen und dynamischen Wertebereichen,
- das Verbot nichthierarchischer Beziehungen,
- das Konzept der referentiellen Integrität,
- die Generalisierung und Spezialisierung von Entitätsmengen und
- indirekte Beziehungen zwischen Entitätsmengen.

Inhalt des Kapitels

Im vorliegenden Kapitel werden diese Konzepte behandelt. Die Ausführungen lehnen sich an ein erweitertes Relationenmodell an, das auf THURNHERR und ZEHNDER zurückgeht (vgl. hierzu THURNHERR und ZEHNDER 1979, THURNHERR 1980, ZEHNDER 1985). Der Rest des Kapitels ist wie folgt gegliedert: Abschnitt a) behandelt die Beschreibung von Entitätsmengen mit Relationen unter Verwendung globaler und lokaler Attribute und unter Berücksichtigung statischer und dynamischer Wertebereiche. Die ausschließliche Verwendung hierarchischer Beziehungen wird in Abschnitt b) begründet; Fragen der Datenintegrität und insbesondere die referentielle Integrität spielen hierbei eine ausschlaggebende Rolle. Eine Konsequenz des Verbots nichthierarchischer Beziehungen ist die Notwendigkeit der Umwandlung nichthierarchischer in hierarchische Beziehungen; in Abschnitt c) wird gezeigt, wie dies geschehen kann. Der Abschnitt d) legt dar, wie sich mit der Generalisierung und Spezialisierung von Entitätsmengen eine genauere Modellierung bestimmter betrieblicher Gegebenheiten erreichen lässt. Schließlich wird in Abschnitt e) gezeigt, wie sogenannte "indirekte Beziehungen" zwischen Entitätsmengen zu modellieren sind.

Struktur- und Integritätsregeln

Als Hilfestellung bei der Datenmodellierung sind einige Struktur- und Integritätsregeln gedacht, die in den einzelnen Kapiteln angegeben werden. Die Regeln sollen die Umsetzung der genannten Konzepte erleichtern.

a) Beschreibung von Entitätsmengen mit Relationen

Beschreibung von Entitätsmengen mit Relationen

Wie in anderen Relationenmodellen wird auch bei dem hier beschriebenen relationalen Datenmodell die Verbindung zwischen den Ansätzen von CODD und CHEN dadurch hergestellt, dass Entitätsmengen als Relationen aufgefasst und mit Relationen beschrieben werden. Die einzelnen Tupel einer Relation repräsentieren also die einzelnen zu einer Entitätsmenge gehörigen Entitäten.

Anforderungen an Relationen

Die Beschreibung von Entitätsmengen mit Relationen hat mehrere Konsequenzen:

- Zum einen muss in jede Relation ein Identifikationsschlüssel eingeführt werden.
- Zum zweiten dürfen Relationen nur aus globalen und lokalen Attributen gebildet werden.
- Zum dritten sind für Attribute in der Regel statische Wertebereiche und nur in bestimmten Fällen dynamische Wertebereiche vorzusehen.

Die Konsequenzen und die sie begründeten Konzepte werden nachfolgend behandelt.

Identifikationsschlüssel

Damit jede einzelne Entität einer Entitätsmenge eindeutig identifiziert und angesprochen werden kann, wird ein Identifikationsschlüssel benötigt. Dies drückt die folgende Strukturregel 1 aus (vgl. auch ZEHNDER 1985).

Strukturregel 1

> Strukturregel 1:
> Jede Relation, die eine Entitätsmenge beschreibt, muss einen Identifikationsschlüssel aufweisen.

Begriff und Wesen des Identifikationsschlüssels wurden bereits in Kapitel 3.1.1 behandelt. Hier erübrigen sich daher weitere Ausführungen.

Globale und lokale Attribute

Verbindung von Relationen durch globale Attribute

In Kapitel 3.4.1 wurde gezeigt, wie inhaltliche Beziehungen zwischen Relationen hergestellt werden. Als Verbindungselemente dienen nämlich Attribute, die zugleich in mehreren Relationen auftreten. Betrachtet sei in diesem Zusammenhang nochmals das in Abb. 3.30 gezeigte einfache relationale Datenmodell. Dort stellt das in den Relationen *Buch* und *Bestellung* zugleich auftretende Attribut *ISBN* eine inhaltliche Verknüpfung zwischen diesen Relationen her.

Die bisher für die Beschreibung inhaltlicher Beziehungen verwendeten unspezifizierten Formulierungen "zugleich auftretendes Attribut" oder "gemeinsames Attribut" bedarf einer Präzisierung:

Beziehungen zwischen Relationen sind mit Hilfe sogenannter globaler Attribute herzustellen. Von den globalen Attributen sind die lokalen Attribute zu unterscheiden. Für beide Arten von Attributen gilt:

Ein Attribut heißt global, wenn es mindestens in einer Relation im Identifikationsschlüssel auftritt.

Begriff des globalen Attributs

Ein Attribut heißt lokal, wenn es nur in einer Relation und in dieser nicht im Identifikationsschlüssel auftritt.

Begriff des lokalen Attributs

Als Beispiel sei wiederum das relationale Datenmodell in Abb. 3.30 betrachtet: Globale Attribute sind *ISBN*, *RechnungsNr*, *KundenNr* und *Datum* während *Autor*, *Preis*, *Betrag* und *Menge* lokale Attribute darstellen. Außerdem werden Beziehungen, wie oben gefordert, durch globale Attribute hergestellt; zwischen den Relationen *Buch* und *Bestellung* durch das globale Attribut *ISBN* und zwischen den Relationen *Rechnung* und *Bestellung* durch das globale Attribut *RechnungsNr*.

Im Gegensatz zu lokalen Attributen können globale Attribute also in mehreren Relationen vorkommen. Ein globales Attribut, welches definitionsgemäß in einer Relation zum Identifikationsschlüssel gehört und außerdem in einer weiteren Relation auftritt, stellt zwangsläufig eine Beziehung zwischen den beiden Relationen her.

Mit lokalen Attributen verfolgt man ebenfalls einen speziellen Zweck. Er besteht in der Verhinderung von unerwünschten Redundanzen, die mehrere Relationen trotz Normalisierung noch gemeinsam aufweisen können. Das Auftreten solcher Redundanzen und ihre Vermeidung durch die ausschließliche Verwendung globaler und lokaler Attribute seien an einem einfachen Beispiel demonstriert.

relationsübergreifende Redundanzen

Beispiel 3.4

Betrachtet werde ein Unternehmen, welches einige Mitarbeiter sowohl im Innendienst als auch zeitweise im Außendienst als Vertreter einsetzt. Die Modellierung dieser Situation führt zu den überlappenden Entitätsmengen *Mitarbeiter* und *Vertreter*, denn die erwähnten Mitarbeiter gehören beiden Entitätsmengen an. Beide Entitätsmengen lassen sich beispielsweise wie folgt als Relationen in 3. Normalform darstellen:

Mitarbeiter(<u>MitarbeiterNr</u>, Name, Adresse, Abteilung, Gehalt)

Vertreter(<u>VertreterNr</u>, Name, Adresse, Bezirk, Provision)

Die beiden Attribute *Name* und *Adresse* in diesen Relationen sind weder global noch lokal und außerdem redundant. Solche relationsübergreifenden Redundanzen können Mutationsanomalien verursachen.

Um diese Redundanzen zu beheben, definiert man eine übergreifende Entitätsmenge und ordnet dieser die gemeinsamen bzw. redundanten Attribute zu. Es ergeben sich dann die drei folgenden Relationen:

Personal(<u>PersonalNr</u>, Name, Adresse)

Mitarbeiter(<u>PersonalNr</u>, MitarbeiterNr, Abteilung, Gehalt)

Vertreter(<u>PersonalNr</u>, VertreterNr, Bezirk, Provision)

Alle drei Relationen befinden sich in 3. Normalform. Attribute, die weder global noch lokal sind, treten nun nicht mehr auf und das Redundanzproblem ist behoben.

Begriff der Datenbasis

Das Konzept der globalen und lokalen Attribute findet seinen Niederschlag in der zweiten Strukturregel. Bezeichne der Begriff "Datenbasis" eine Menge von Relationen, welche die Entitätsmengen eines betrachteten Realitätsausschnitts darstellen, dann fordert die zweite Strukturregel folgende Beschaffenheit der Datenbasis (vgl. ZEHNDER 1985):

Strukturregel 2

Strukturregel 2:
Eine Datenbasis muss aus Relationen in 3. Normalform bestehen, welche ausschließlich aus globalen und lokalen Attributen gebildet werden.

Zur Verdeutlichung des Sinns der zweiten Strukturregel sei an das Konzept der Normalisierung erinnert. Die Normalisierung verfolgt insbesondere das Ziel, Redundanzen innerhalb von Relationen zu vermeiden. Mit dem Konzept der globalen und lokalen Attribute strebt man dagegen die Beseitigung von Redundanzen auf relationsübergreifender, globaler Ebene an.

Übungsaufgabe 3.18

Betrachtet werde das Beispiel 3.4. Die zu Beginn eingeführten Relationen *Mitarbeiter* und *Vertreter* enthalten die beiden Attribute *Name* und *Adresse*, die weder global noch lokal sind. Beschreiben Sie eine mögliche Mutationsanomalie, welche sich durch die Verletzung der Strukturregel 2 ergeben kann.

Statische und dynamische Wertebereiche

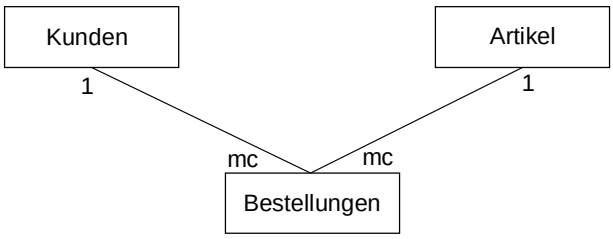
Zur präzisen Definition der Relationen eines relationalen Datenmodells gehört neben der Festlegung der globalen und lokalen Attribute der Relationen auch die Vorgabe von Wertebereichen für die Attribute. Der Begriff des Wertebereichs eines Attributs wurde bereits in Kapitel 3.4.1 eingeführt. Dort wurden auch beispielhaft Wertebereiche für einige Attribute angegeben (vgl. Übungsaufgabe 3.14 und zugehörige Erläuterungen).

Die spezielle Rolle, die globale Attribute als Verbindungselemente zwischen den Relationen spielen können, legt eine Unterscheidung zwischen statischen und dynamischen Wertebereichen nahe. Beide Arten von Wertebereichen seien an einem Beispiel eingeführt und verdeutlicht.

Beispiel 3.5

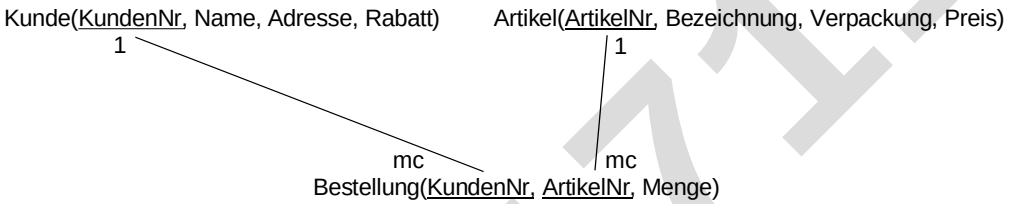
Gegeben sei folgendes einfache Datenmodell, welches die Bestellung von Artikeln durch Kunden beschreibt:

einfaches Datenmodell



Um die Attribute der Entitätsmengen sichtbar zu machen, stellt man die Entitätsmengen durch Relationen dar. Das Datenmodell geht dann in folgendes Relationendiagramm über:

zugehöriges Relationendiagramm



Die Attributwerte werden sichtbar, wenn man die Relationen tabellarisch wiedergibt:

tabellarische Darstellung von Relationen

Kunde

Kunden-Nr	Name	Adresse	Rabatt
K1	Abs, Josef	Ahornweg 12, Bremen	05
K2	Amt, Xaver	Tulpenweg 20, Aachen	03
K3	Beer, Poldi	Rosenweg 36, Hagen	05
K5	Bitz, Hans	Nelkenweg 8, Berlin	10
K7	Buhl, Herbert	Amselweg 90, Bremen	03
		⋮	

Artikel

Artikel-Nr	Bezeich-nung	Ver-packung	Preis
A01	Bier	Kiste	14,00
A03	Bier	Fass	80,00
A04	Gin	Karton	30,00
A05	Wodka	Karton	65,00
A07	Wein	Karton	50,00
A12	Wasser	Kiste	11,00
		⋮	

Bestellung

Kunden-Nr	Artikel-Nr	Menge
K1	A01	20
K1	A07	50
K1	A12	120
K2	A03	10
K2	A05	15
K3	A05	10
K5	A03	20
K5	A04	30
K5	A05	40
K5	A12	30
K7	A03	10
	⋮	

Anhand dieser tabellarischen Darstellung lassen sich folgende Sachverhalte nachvollziehen:

Sachverhalt (1)

(1) In der Relation *Bestellung* dürfen nur solche Tupel existieren, für die gilt: Dem Wertepaar *(KundenNr, ArtikelNr)* eines Tupels ist genau ein Tupel in der Relation *Kunde* mit der in dem Wertepaar enthaltenen Kundennummer und genau ein Tupel in der Relation *Artikel* mit der in dem Wertepaar enthaltenen Artikelnummer zugeordnet. Beispielsweise gibt das mit einem Raster unterlegte fünfte Tupel einen Sinn ab, weil für die Kundennummer K2 genau ein Tupel in der Relation *Kunde* und für die Artikelnummer A05 genau ein Tupel in der Relation *Artikel* existiert. Dagegen wäre in der Relation *Bestellung* z.B. ein Tupel (K6, A08, 25) sinnlos, weil ein Kunde mit der Kundennummer K6 und ein Artikel mit der Artikelnummer A08 offensichtlich nicht existieren.

Sachverhalt (2)

(2) Die Werte die das Attribut bzw. der Identifikationsschlüssel *KundenNr* in der Relation *Kunde* annehmen darf, sei per Definition auf den Bereich [K1..K9] festgelegt. Ebenso sei der Wertebereich des Attributs bzw. Identifikationsschlüssels *ArtikelNr* in der Relation *Artikel* per Definition auf [A01..A99] festgelegt. Beide Wertebereiche stellen ein Ergebnis der Datenmodellierung dar.

Sachverhalt (3)

(3) Anders verhält es sich mit den Werten, welche die Attribute *KundenNr* und *ArtikelNr* in der Relation *Bestellung* annehmen können. In *Bestellung* dürfen nur solche Werte des Attributs *KundenNr* auftreten, wie sie in der Relation *Kunde* aktuell vorhanden sind, und außerdem dürfen in *Bestellung* nur die Werte des Attributs *ArtikelNr* vorkommen, die in der Relation *Artikel* aktuell vorhanden sind.

Für die in Beispiel 3.5 formulierten Sachverhalte werden im Folgenden umschreibende Begriffe angegeben.

Zu (1):
Zwischen der Relation *Kunde* und der Relation *Bestellung* wird eine inhaltliche Beziehung durch die Übernahme der aktuellen Werte des globalen Attributs *KundenNr* der Relation *Kunde* in die Relation *Bestellung* hergestellt. Da das globale Attribut *KundenNr* in der Relation *Kunde* Identifikationsschlüssel ist und von der Relation *Bestellung* gleichsam importiert wird, bezeichnet man das Attribut *KundenNr* in der Relation Bestellung als Fremdschlüssel. In analoger Weise stellt das Attribut *ArtikelNr* in der Relation *Bestellung* einen Fremdschlüssel dar. Allgemein gilt:

Begriff des Fremdschlüssels

Ein Fremdschlüssel (engl. foreign key) in einer Relation *R2* ist ein globales Attribut oder eine Attributkombination, welches oder welche auch in einer anderen Relation *R1* vorkommt und in *R1* Identifikationsschlüssel ist.

Aus dem Sachverhalt (1) folgt für den Wertebereich eines Fremdschlüssels:

Wertebereich eines Fremdschlüssels

Der Wertebereich eines Fremdschlüssels in einer Relation *R2* umfasst genau die Menge der Werte, welche als aktuelle Identifikationsschlüsselwerte in genau der Relation *R1* auftreten, aus der der Fremdschlüssel importiert wird.

Zu (2):
Da der unter (2) beschriebene Wertebereich per Definition fixiert ist, bezeichnet man ihn als statisch. Es gilt (vgl. auch ZEHNDER 1985):

Ein statischer Wertebereich besteht aus einer Menge von Werten, welche bei der Datenmodellierung per Definition festgelegt wurde. Ein statischer Wertebereich ändert sich im Zeitablauf nicht.

Begriff des statischen Wertebereichs

Betrachtet sei das Beispiel 3.5. Statische Wertebereiche könnten z.B. [K1..K9] für den Identifikationsschlüssel *KundenNr* und [000.00..999.99] für das Attribut *Preis* sein.

Zu(3):
Der unter (3) beschriebene Wertebereich heißt dynamisch. Es gilt (vgl. auch ZEHNDER 1985):

Ein dynamischer Wertebereich besteht aus einer Menge von Identifikationsschlüsselwerten oder, im Falle eines zusammengesetzten Identifikationsschlüssels, aus Wertekombinationen, welche für einen Fremdschlüssel aktuell zur Verfügung stehen. Ein dynamischer Wertebereich kann sich im Zeitablauf ändern.

Begriff des dynamischen Wertebereichs

Betrachtet sei wiederum das Beispiel 3.5. Dynamische Wertebereiche sind die Mengen der aktuell vorkommenden Werte {K1, K2, K3, K5, K7, ...} des Identifikationsschlüssels *KundenNr* der Relation *Kunde* bzw. {A01, A03, A04, A05, A07, A12, ...} des Identifikationsschlüssels *ArtikelNr* der Relation *Artikel*.

Übungsaufgabe 3.19

Begründen Sie, warum sich ein dynamischer Wertebereich im Zeitablauf ändern kann. Nennen Sie insbesondere auch mögliche Änderungsursachen.

Die mit Hilfe der Begriffe "Fremdschlüssel", "dynamischer Wertebereich" und "statischer Wertebereich" formulierten Zusammenhänge sind für die Datenmodellierung von zentraler Bedeutung. Sie werden daher in der dritten Strukturregel zusammengefasst (vgl. ZEHNDER 1985):

Strukturregel 3:
Für jedes lokale Attribut ist ein statischer Wertebereich zu definieren, in dem sich die Attributwerte bewegen können.
Jedem globalen Attribut darf nur in genau einer Relation ein statischer Wertebereich zugrunde liegen und in dieser Relation muss das Attribut Identifikationsschlüssel sein. In andere Relationen darf das Attribut nur als Fremdschlüssel mit einem dynamischen Wertebereich eingebracht werden.

Strukturregel 3

Sämtliche in diesem Kapitel vorgestellten Konzepte werden nochmals in der Übungsaufgabe 3.20 aufgegriffen.

Übungsaufgabe 3.20

Betrachtet werde das Ergebnis des in Kapitel 3.4.1, Beispiel 3.3 vorgenommenen Normalisierungsprozesses. Es besteht aus den drei in 3. Normalform befindlichen Relationen

Buch(ISBN, Autor, Preis)
Bestellung(KundenNr, Datum, ISBN, Menge, RechnungsNr)

Rechnung(RechnungsNr, Betrag)

einschließlich der tabellarischen Darstellung dieser Relationen.

a) Entwerfen Sie für diese Relationen das zugehörige ER-Diagramm und das zugehörige Relationendiagramm.

b) Geben Sie an, welche in den Relationen vorkommenden Attribute global und welche lokal sind.

c) Definieren Sie für zwei lokale Attribute sinnvolle statische Wertebereiche.

d) Geben Sie für ein globales Attribut an, in welcher Relation es Identifikationsschlüssel ist und definieren Sie einen sinnvollen statischen Wertebereich für den Identifikationsschlüssel. Benennen Sie außerdem eine Relation, in der dieses Attribut als Fremdschlüssel vorkommt, und spezifizieren Sie den dynamischen Wertebereich der dem Fremdschlüssel zugrunde liegt.

b) Hierarchische Beziehungen und referentielle Integrität

Bekanntlich werden Beziehungen zwischen Relationen mittels globaler Attribute hergestellt. Aus Gründen der Datenintegrität sollten hierbei nur hierarchische Beziehungen verwendet werden. Die Erläuterung und Begründung dieser Aussage erfordern es, auf

- das Wesen der hierarchischen Beziehung und
- den Begriff der referentiellen Integrität

einzugehen.
Zunächst sei der Begriff der hierarchischen Beziehung eingeführt:

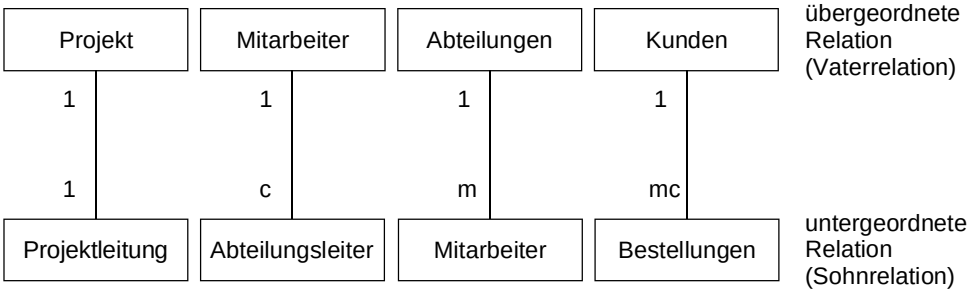
hierarchische Beziehungen

Hierarchische Beziehungen sind die 1-1-, die 1-c-, die 1-m- und die 1-mc-Beziehung.

Eine hierarchische Beziehung stellt ein Verhältnis der Über-/Unterordnung zwischen zwei Relationen her. Es wird also eine übergeordnete Relation mit einer untergeordneten Relation verbunden. Dabei gilt:

- Für ein Tupel der übergeordneten Relation existieren genau (1), kein oder ein (c), ein oder mehrere (m) oder kein, ein oder mehrere Tupel (mc) in der untergeordneten Relation.
- Für ein Tupel der untergeordneten Relation existiert genau ein Tupel in der übergeordneten Relation.

Beispiele hierarchischer Beziehungen zeigt die Abb. 3.32.



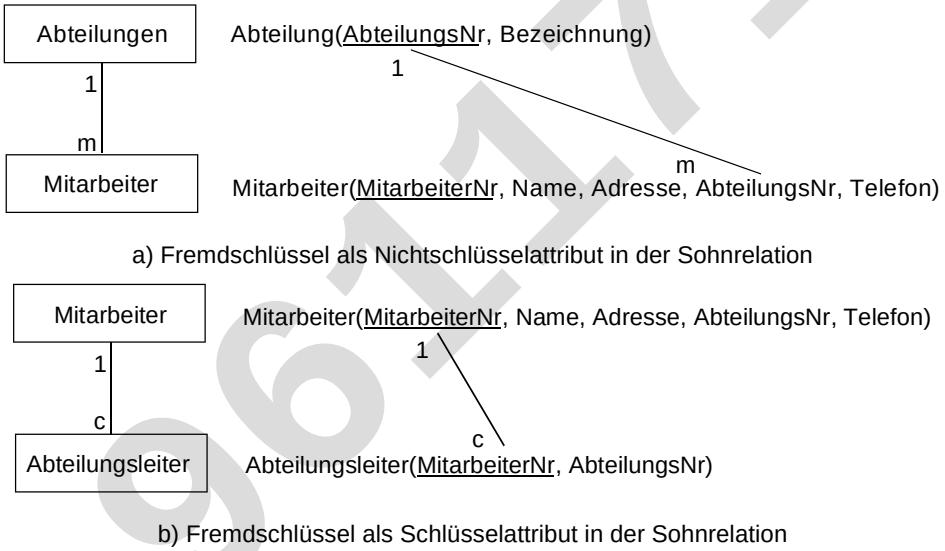
Beispiele für hierarchische Beziehungen

Abb. 3.32. Hierarchische Beziehungen.

Betrachtet man hierarchische Beziehungen auf der Ebene von Attributen, so kommt der Begriff des Fremdschlüssels ins Spiel. Ein Identifikationsschlüssel, der aus einer übergeordneten Relation (Vaterrelation) als Fremdschlüssel in eine untergeordnete Relation (Sohnrelation) übernommen wird, stellt nämlich genau eine hierarchische Beziehung zwischen Vater- und Sohnrelation her.

Vaterrelation und Sohnrelation

In der Sohnrelation kann der Fremdschlüssel Schlüsselattribut oder Nichtschlüsselattribut sein. Die in Abb. 3.33 angegebenen Beispiele verdeutlichen dies.



Rollen von Fremdschlüsseln in Sohnrelationen

Abb. 3.33. Rollen von Fremdschlüsseln.

Fordert man für einen Fremdschlüssel in einer Sohnrelation nun einen dynamischen Wertebereich, so treten in der Sohnrelation nur Tupel auf, zu denen ein Tupel in der Vaterrelation existiert. Nur solche Referenzen von einer Sohnrelation auf eine Vaterrelation sind sinnvoll. Es gäbe dagegen keinen Sinn ab, wenn ein Tupel einer Sohnrelation ein Tupel einer Vaterrelation referenzieren würde, das gar nicht existiert. Übertragen auf das Beispiel 3.5 würde dies z.B. bedeuten, dass zu einer Bestellung kein Kunde existieren würde.

Zur Gewährleistung korrekter Referenzen wird für hierarchische Beziehungen folgende Bedingung der "referentiellen Integrität" formuliert:

Die Bedingung der referentiellen Integrität fordert, dass ein Fremdschlüssel in einer Sohnrelation *R2* nur Tupel in der Vaterrelation *R1* referenzieren darf, die in *R1* auch

referentielle Integrität

tatsächlich vorhanden sind.

Datenintegrität

Die Bedingung der referentiellen Integrität gehört zu den sogenannten Integritätsregeln. Diese zielen auf die Erhaltung der Datenintegrität ab. Darunter ist die Korrektheit und die Konsistenz der in den Relationen einer Datenbank abgelegten Daten zu verstehen. Während mit "Korrektheit" die sachliche Richtigkeit von Daten gemeint ist, bezieht sich "Konsistenz" auf die Verträglichkeit von Daten.

Korrektheit und Konsistenz

Übungsaufgabe 3.21

Betrachtet werde das linke der in Abb. 3.32 gezeigten einfachen ER-Diagramme. Es besteht aus den Entitätsmengen *Projekt* und *Projektleitung* sowie einer die beiden Entitätsmengen verbindenden 1-1-Beziehung.

a) Entwerfen Sie ein Relationenmodell für den durch das ER-Diagramm beschriebenen Realitätsausschnitt. Beschränken Sie sich hierbei auf die wichtigsten Attribute.

b) Begründen Sie, warum die Entitätsmenge *Projekt* die Vaterrelation und die Entitätsmenge *Projektleitung* die Sohnrelation darstellt.

c) Behandlung nichthierarischer Beziehungen

Im vorliegenden Kapitel wird zunächst begründet, warum nichthierarchische Beziehungen einer weiteren Behandlung bedürfen, und es wird dargelegt, wie dies prinzipiell geschehen kann. Anschließend wird die Behandlung der verschiedenen Arten von nichthierarchischen Beziehungen dargestellt. Dies geschieht in folgenden Abschnitten:

Inhalt des Kapitels

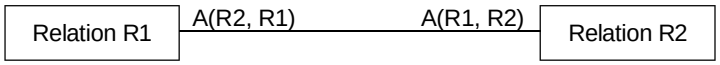
- Hilfsrelationen als informationstragende Beziehungen,
- Behandlung konditioneller Beziehungen,
- Behandlung netzwerkartiger Beziehungen,
- Behandlung rekursiver Beziehungen.

Hilfsrelationen als informationstragende Beziehungen

Bekanntlich begünstigen hierarchische Beziehungen die Erhaltung der Integrität einer relationalen Datenbank. Andererseits führen nichthierarchische Beziehungen zu bestimmten Problemen, auf die je separat hingewiesen wird. Man ist daher bestrebt, relationale Datenmodelle ausschließlich aus hierarchischen Beziehungen zu konstruieren. Zu diesem Zweck wandelt man nichthierarchische Beziehungen in hierarchische Beziehungen um. Dies geschieht durch die Einführung sogenannter Hilfsrelationen. In die Hilfsrelationen werden dabei die Informationen übernommen, welche die umzuwandelnden Beziehungen charakterisieren.

Umwandlung nichthierarchischer Beziehungen

Zunächst sind die verschiedenen Arten von nichthierarchischen Beziehungen abzugrenzen. Die Menge der Beziehungstypen, die zwischen zwei Relationen *R1* und *R2* auftreten können, ist in Abb. 3.34 zusammengestellt.



A(R2, R1) \ A(R1, R2)	1	c	m	mc
1	1-1	1-c	1-m	1-mc
c	c-1	c-c	c-m	c-mc
m	m-1	m-c	m-m	m-mc
mc	mc-1	mc-c	mc-m	mc-mc

hierachische Beziehungen
konditionelle Beziehungen
netzwerkartige Beziehungen

Systematik der Beziehungstypen

Abb. 3.34. Mögliche Beziehungstypen zwischen zwei Relationen.

Von den 16 in Abb. 3.34 angegebenen Beziehungstypen sind drei hierarchische, zwei konditionelle und eine netzwerkartige Beziehung redundant. Beispielsweise geht der Typ c-1 in den Typ 1-c über, wenn man die Reihenfolge der Relationen vertauscht. Die nach der Vernachlässigung von redundanten Beziehungen übrig bleibenden 10 Beziehungstypen sind bereits aus Abb. 3.7 bekannt.

Neben konditionellen und netzwerkartigen Beziehungen bedürfen auch rekursive Beziehungen einer besonderen Behandlung. Festgehalten sei daher:

Hilfsrelationen

> Konditionelle, netzwerkartige und rekursive Beziehungen sind durch das Einführen von Hilfsentitätsmengen bzw. Hilfsrelationen auf hierarchische Beziehungen zurückzuführen.

Bei der Einführung einer Hilfsrelation werden Informationen, welche die umgewandelte Beziehung inhaltlich beschreiben, in die Hilfsrelation übernommen. Umzuwandelnde Beziehungen sind also informationstragend in dem Sinne, dass ihr Informationsgehalt über den bloßen, durch zwei gegenseitige Assoziationen beschriebenen Beziehungszusammenhang hinausgeht.

informationstragende Beziehungen

Einer Hilfsrelation werden daher Attribute wie einer Relation zugeordnet. Darüber hinaus wird eine Hilfsrelation wie eine Relation behandelt und im ER-Diagramm als Rechteck dargestellt.

Darstellung von Hilfsrelationen

Nach der Behandlung sämtlicher nichthierarchischer Beziehungen besteht ein relationales Datenmodell aus Relationen und Hilfsrelationen sowie aus die Relationen verbindenden hierarchischen Beziehungen. Im ER-Diagramm werden hierarchischen Beziehungen als einfache Verbindungslinien dargestellt. Die Raute als Beziehungssymbol erübrigt sich daher und man erhält einfacher zu zeichnende und klarer strukturierte ER-Diagramme.

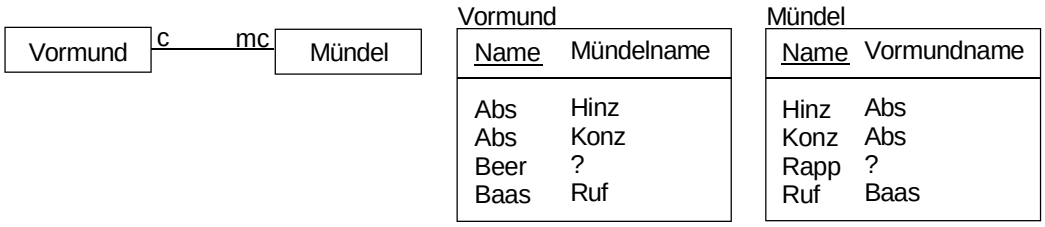
Verzicht auf die Raute als Beziehungssymbol

Behandlung konditioneller Beziehungen

Bei der Darstellung konditioneller Beziehungen auf der Ebene von Tupeln bereitet das Auftreten sogenannter Nullwerte Probleme. Ein Nullwert besagt, dass im Fall einer As-

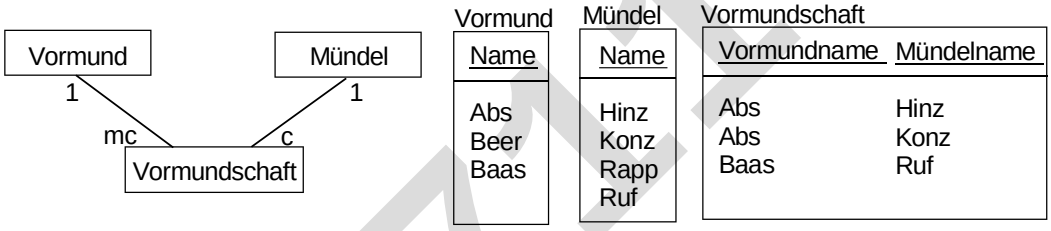
Nullwerte

soziation c=0 dem verbindenden Attribut kein Wert zugeordnet werden kann. In der tabellarischen Darstellung von Relationen drückt man einen Nullwert durch das Symbol "?" aus. Die Abb. 3.35 a) zeigt ein Beispiel. Wie der Abb. 3.35 b) zu entnehmen ist, kommen nach der Einführung einer Hilfsentitätsmenge bzw. Hilfsrelation keine Nullwerte mehr vor.



a) Konditionelle Beziehung mit Nullwerten

Elimination von Nullwerten durch Einführung einer Hilfsrelation



b) Einführung einer Hilfsentitätsmenge bzw. einer Hilfsrelation

Abb. 3.35. Rückführung einer konditionellen Beziehung auf zwei hierarchische Beziehungen.

Bei dem in Abb. 3.35 gezeigten Beispiel sei unterstellt, dass eine bisher von der Person *Beer* ausgeübte Vormundschaft nicht mehr besteht und dass für den Mündel *Rapp* ein Vormund noch bestellt werden muss. Beide Fälle führen zu Nullwerten. Die Nullwerte verschwinden, wenn die c-mc-Beziehung durch zwei hierarchische Beziehungen ersetzt wird, welche die Relationen *Vormund* und *Mündel* mit der Hilfsrelation *Vormundschaft* verbinden.

Übungsaufgabe 3.22

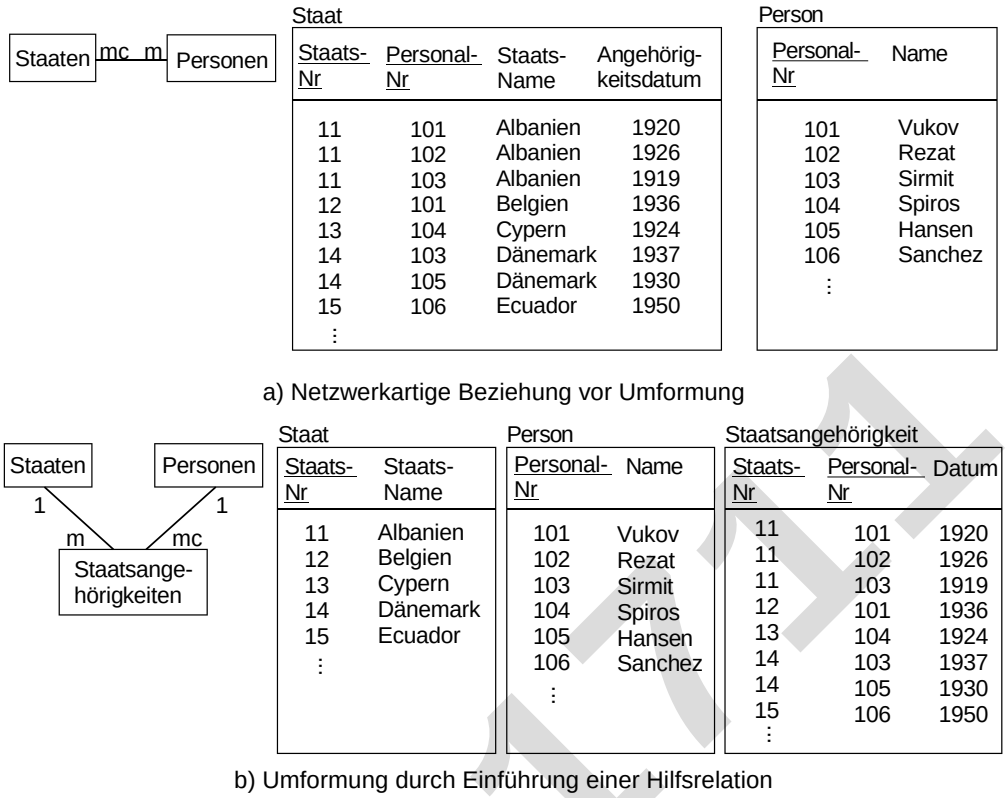
Betrachtet werde die Transformation der c-mc-Beziehung der Vormundschaft in Abb. 3.35 in die Hilfsrelation *Vormundschaft*. Die Beziehung der Vormundschaft ist informationstragend. Dies zeigt sich daran, dass man ihr mehr bedeutsame Attribute zuordnen kann, als die von den Relationen *Vormund* und *Mündel* übernommenen Fremdschlüssel *Vormundname* und *Mündelname*. Geben Sie zumindest ein weiteres Attribut an, um das man die Hilfsrelation Vormundschaft erweitern kann.

Behandlung netzwerkartiger Beziehungen

Normalisierung durch Umformung

Netzwerkartige Beziehungen bestehen häufig zwischen Relationen, welche sich nicht in 3. Normalform befinden. Ersetzt man eine netzwerkartige Beziehung durch zwei auf eine Hilfsrelation verweisende hierarchische Beziehungen, so gehen die ursprünglichen Relationen in die 3. Normalform über. Dies wird dadurch bewirkt, dass diejenigen Attribute aus den ursprünglichen Relationen in die Hilfsrelation übernommen werden, die der 3.

Normalform entgegenstanden. Die normalisierende Wirkung dieses Transformationsprozesses sei anhand von Abb. 3.36 erläutert.



Umformung einer netzwerkartigen Beziehung durch Einführung einer Hilfsrelation

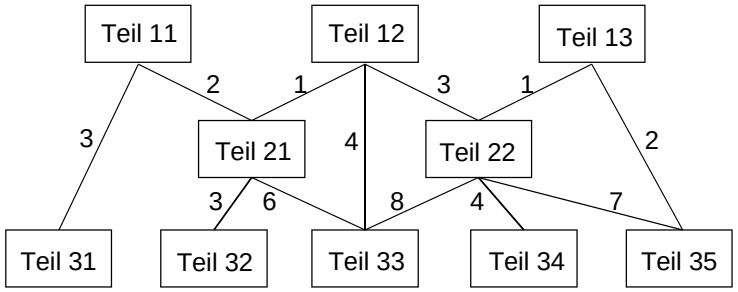
Abb. 3.36. Rückführung einer netzwerkartigen Beziehung auf zwei hierarchische Beziehungen.

Die Relation *Staat* der netzwerkartigen Beziehung in Abb. 3.36 a) beschreibt zwei Phänomene, nämlich Staaten und Personen. Sie befindet sich daher nicht in 3. Normalform. Durch das Aufbrechen der Relation *Staat* und durch das Einbeziehen der die Relationen *Staat* und *Person* verbindenden Attribute in die Hilfsrelation *Staatsangehörigkeit* ergeben sich zwei hierarchische Beziehungen zwischen den Relationen *Staat* bzw. *Person* und der Hilfsrelation. Außerdem sind nun, wie die Abb. 3.36 b) zeigt, sämtliche Relationen in 3. Normalform.

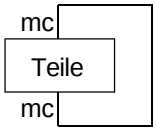
Behandlung rekursiver Beziehungen

Rekursive Beziehungen werfen z.B. das Problem auf, dass ein globales Attribut, welches in einer Relation als Identifikationsschlüssel dient, in der gleichen Relation die Basis für einen Fremdschlüssel bildet. Verdeutlicht sei dies an der in Abb. 3.37 dargestellten netzwerkartigen Beziehung zwischen Teilen, die bereits aus Übungsaufgabe 3.7 bekannt ist.

Teilestruktur



a) Beispiel für eine hypothetische Teilestruktur



Teil

TeileNr	Bezeichnung	Preis	OUTeileNr	Teileart	Menge
11	Produkt 1	7014,00	21	Unterteil	2
11	Produkt 1	7014,00	31	Unterteil	3
12	Produkt 2	12880,00	21	Unterteil	1
12	Produkt 2	12800,00	22	Unterteil	3
12	Produkt 2	12800,00	33	Unterteil	4
13	Produkt 3	6530,00	22	Unterteil	1
13	Produkt 3	6530,00	35	Unterteil	2
21	Baugruppe 1	630,00	32	Unterteil	3
21	Baugruppe 1	630,00	33	Unterteil	6
22	Baugruppe 2	490,00	33	Unterteil	8
22	Baugruppe 2	490,00	34	Unterteil	4
22	Baugruppe 2	490,00	35	Unterteil	7
31	Einzelteil 1	80,00	11	Oberteil	3
32	Einzelteil 2	66,00	21	Oberteil	3
33	Einzelteil 3	25,00	21	Oberteil	6
33	Einzelteil 3	25,00	22	Oberteil	8
33	Einzelteil 3	25,00	12	Oberteil	4
34	Einzelteil 4	19,00	22	Oberteil	4
35	Einzelteil 5	21,00	22	Oberteil	7
35	Einzelteil 5	21,00	13	Oberteil	2
21	Baugruppe 1	630,00	11	Oberteil	2
21	Baugruppe 1	630,00	12	Oberteil	1
22	Baugruppe 2	490,00	12	Oberteil	3
22	Baugruppe 2	490,00	13	Oberteil	1

b) Modellierung mit einer rekursiven Oberteil- und Unterteilbeziehung

Abb. 3.37. Beispiel für eine rekursive Beziehung.

rekursive Darstellung einer Teilestruktur

Die Abb. 3.37 a) beschreibt die Zusammensetzung von drei Produkten aus zwei Baugruppen und fünf Einzelteilen. Einzelteile können sowohl in die Baugruppen als auch in die Produkte eingehen. Die Parameter an den Verbindungslinien zwischen den Teilen stellen Mengenangaben dar. Beispielsweise geht das Teil 33, ein Einzelteil, mit der Anzahl 6 in das Teil 21, eine Baugruppe, ein und mit der Anzahl 4 in das Teil 12, ein Produkt, ein.

Oberteil und Unterteil

Teil 33 heißt daher Unterteil in Bezug auf die Teile 21 und 12; umgekehrt heißen die Teile 21 und 12 Oberteil in Bezug auf Teil 33.

Beziehung einer Entitätsmenge auf sich selbst

Wie sich die Oberteil- und Unterteilbeziehungen einer Teilestruktur auf rekursive Weise modellieren lassen, soll die Abb. 3.37 b) verdeutlichen. Definiert wird eine Entitätsmenge *Teile* mit einer mc-mc-Beziehung auf sich selbst. Welche Einzelbeziehungen dieses Modell umfasst, geht aus der Relation *Teil* hervor. Die ersten drei Attribute der Relation *Teil* beschreiben ein Teil, während die drei folgenden Attribute strukturelle Zusammenhänge erfassen:

- Das Attribut *OUTeileNr* bezeichnet ein Unter- oder Oberteil in Bezug auf ein gegebenes Teil mit der Nummer *TeileNr*.
- Das Attribut *Teileart* gibt an, ob das Teil mit der Nummer *OUTeileNr* Ober- oder Unterteil ist.
- Das Attribut *Menge* beschreibt das Mengenverhältnis der Ober- bzw. Unterteilzuordnung.

Problematisch ist bei dieser Modellierung, dass das globale Attribut *TeileNr* in der Relation *Teil* nicht nur den Identifikationsschlüssel bildet, sondern mit seinem dynamischen Wertebereich dem Attribut *OUTeileNr* in der **gleichen** Relation zugrunde liegt. Diese Rolle eines globalen Attributs verträgt sich nicht mit der dritten Strukturregel. Aus diesem Grunde fordert die vierte Strukturregel (vgl. ZEHNDER 1985):

Strukturregel 4

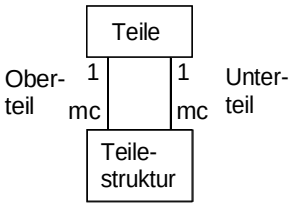
Strukturregel 4:
Rekursive Beziehungen zwischen Relationen sind unzulässig. In einer Relation *R1* darf ein globales Attribut nur mit einem Fremdschlüssel gebildet werden, dessen Ursprungsrelation *R2* unabhängig von *R1* definiert werden kann.

Im Fall der betrachteten Teilestruktur kann in der Tat eine von der Relation *Teil* unabhängige Relation *Teilestruktur* definiert werden, in die sich das globale Attribut *TeileNr* als Fremdschlüssel einbringen lässt. Hierzu ist die rekursive mc-mc-Beziehung durch zwei hierarchische Beziehungen zwischen der Relation *Teil* und der neuen Relation *Teilestruktur* zu ersetzen. Die Abb. 3.38 zeigt das Ergebnis dieser Umformung.

Man beachte, dass die Relation *Teilestruktur* in Abb. 3.38 gleichzeitig Tupel enthält, welche zur Oberteilbeziehung gehören, und Tupel, welche zur Unterteilbeziehung gehören. Erwähnenswert ist außerdem die normalisierende Wirkung der vorgenommenen Umformung.

Beide Relationen, *Teil* und *Teilestruktur*, befinden sich in 3. Normalform.

nichtrekursive Darstellung einer Teilestruktur



Teil

TeileNr	Bezeichnung	Preis
11	Produkt 1	7014,00
12	Produkt 2	12800,00
13	Produkt 3	6530,00
21	Baugruppe 1	630,00
22	Baugruppe 2	490,00
31	Einzelteil 1	80,00
32	Einzelteil 2	66,00
33	Einzelteil 3	25,00
34	Einzelteil 4	19,00
35	Einzelteil 5	21,00

Teilestruktur

TeileNr	OUTeileNr	Teileart	Menge
11	21	Unterteil	2
11	31	Unterteil	3
12	21	Unterteil	1
12	22	Unterteil	3
12	33	Unterteil	4
13	22	Unterteil	1
13	35	Unterteil	2
21	32	Unterteil	3
21	33	Unterteil	6
22	33	Unterteil	8
22	34	Unterteil	4
22	35	Unterteil	7
31	11	Oberteil	3
32	21	Oberteil	3
33	21	Oberteil	6
33	22	Oberteil	8
33	12	Oberteil	4
34	22	Oberteil	4
35	22	Oberteil	7
35	13	Oberteil	2
21	11	Oberteil	2
21	12	Oberteil	1
22	12	Oberteil	3
22	13	Oberteil	1

Abb. 3.38. Ergebnis der Umformung einer rekursiven Beziehung.

Übungsaufgabe 3.23

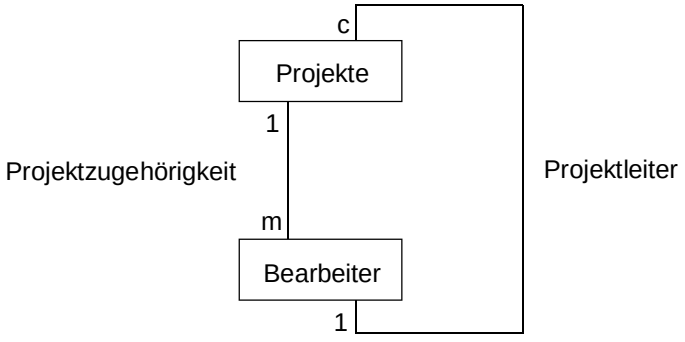
Betrachtet werde das in Abb. 3.38 gezeigte Ergebnis der Umformung einer rekursiven Teilebeziehung. Geben Sie für das dort dargestellte einfache Datenmodell das zugehörige Relationendiagramm an.

indirekte rekursive Beziehung

Das in Abb. 3.37 angegebene Beispiel beschreibt eine direkte rekursive Beziehung. Denkbar sind nun aber auch indirekte rekursive Beziehungen, bei denen die Beziehung einer Relation auf sich selbst nicht unmittelbar hergestellt wird, sondern auf dem Umweg über eine andere Relation. Die Übungsaufgabe 3.24 greift einen solchen Fall auf.

Übungsaufgabe 3.24

Betrachtet werde folgendes Datenmodell, das die Bearbeitung und Leitung von Projekten durch Personen beschreibt:



Einem Projekt sind ein Bearbeiter oder mehrere Bearbeiter zugeordnet. Durch eine indirekte rekursive 1-c-Beziehung wird außerdem ausgedrückt, dass sich der Leiter eines Projekts aus dem Kreis der Bearbeiter rekrutiert.

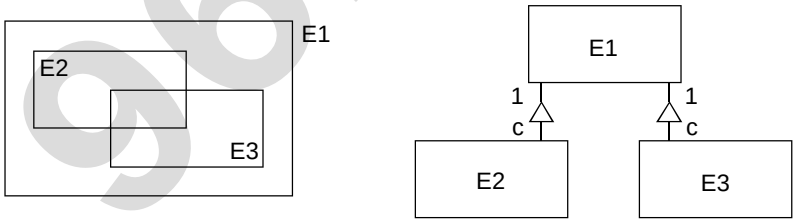
Definieren Sie zunächst zwei Relationen *Projekt* und *Bearbeiter*, welche die Entitätsmengen des obigen Datenmodells beschreiben und geben Sie für beide Relationen einige beispielhafte Tupel an. Verwenden Sie hierbei die tabellarische Darstellungsform. Transformieren Sie dann das rekursive Modell in eine nach der vierten Strukturregel erlaubte Form und beschreiben Sie die sich ergebenden Entitätsmengen mit tabellarisch dargestellten Relationen, welche einige beispielhafte Tupel enthalten.

d) Generalisierung und Spezialisierung von Relationen

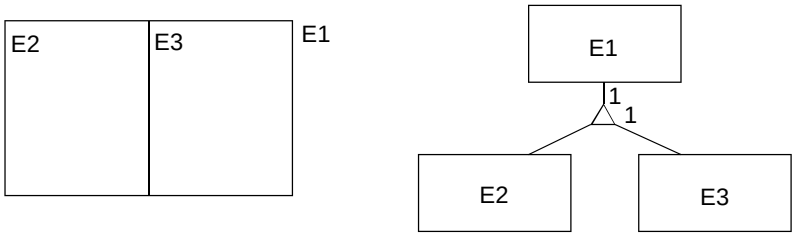
Generalisierung und Spezialisierung

Wie in Kapitel 3.1.1 bemerkt wurde, können zwischen Entitätsmengen Ober- und Untermengenbeziehungen existieren. Die Ursache liegt darin, dass sich Entitätsmengen überlappen oder umfassen, oder dass sie sich in Teilmengen aufspalten oder zu Obermengen zusammenfassen lassen. Diese Fälle der sogenannten Generalisierung oder Spezialisierung von Entitätsmengen schlagen sich in entsprechend generalisierten oder spezialisierten Relationen nieder.

Zur präzisen Modellierung von Generalisierungen und Spezialisierungen wird, abweichend von THURNHERR und ZEHNDER, die der Abb. 3.39 zu entnehmende Notation verwendet. Die verwendete Notation ist bei der Darstellung und Spezialisierung von Objekten bzw. Klassen in der objektorientierten Systementwicklung gebräuchlich. Angegeben werden jeweils ein Mengendiagramm und die dazugehörige Notation im ER-Diagramm.

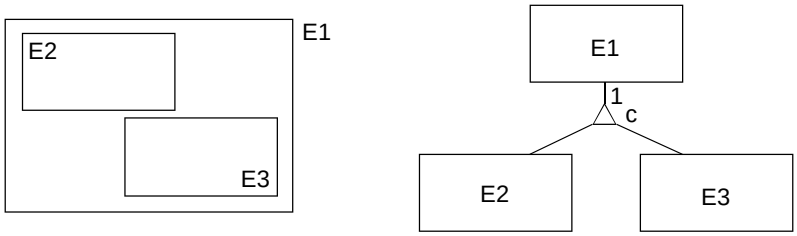


a) Generalisierung/Spezialisierung bei überlappenden Entitätsmengen



Notation von Generalisierungen/ Spezialisierungen

b) Generalisierung/Spezialisierung bei disjunkten, die Obermenge E1 genau überdeckenden Entitätsmengen



c) Generalisierung/Spezialisierung von disjunkten Entitätsmengen

Abb. 3.39. Generalisierung und Spezialisierung von Entitätsmengen.

überlappende Entitätsmengen

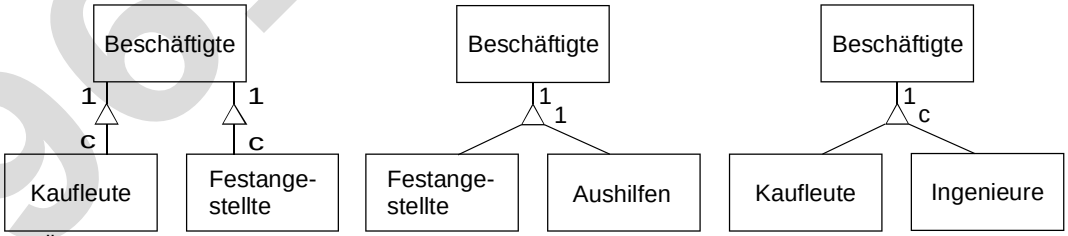
Im Fall nichtdisjunkter bzw. überlappender Untermengen *E2* und *E3* sind zwei separate 1-c-Beziehungen ausgehend von der Obermenge *E1* vorzusehen. Auf diese Weise kann ausgedrückt werden, ob eine Entität der Entitätsmenge *E1* auch zu *E2* oder zu *E3* oder zu *E2* und *E3* gehört, oder ob sie weder zu *E2* noch zu *E3* gehört.

disjunkte Entitätsmengen

Zur Darstellung disjunkter Untermengen *E2* und *E3* wird eine sich in einem kleinen Dreieck aufgabelnde Linie verwendet. Je nachdem ob die beiden Untermengen die Obermenge *E1* vollständig überdecken oder nicht, wird am Gabelungspunkt 1 oder c notiert. Im Fall c=1 ist jede in *E1* vorkommende Entität zwingend in *E2* oder in *E3* enthalten. Dagegen können im Fall c=0 Entitäten in *E1* existieren, die weder zu *E2* noch zu *E3* gehören.

Für die in Abb. 3.39 unterschiedenen Fälle der Modellierung von Generalisierungen und Spezialisierungen seien einige Beispiele betrachtet. Sie sind in Abb. 3.40 zusammengefasst.

Beispiele für Generalisierungen/ Spezialisierung



a) Überlappende Entitätsmengen

b) Disjunkte, die Obermenge genau überdeckende Entitätsmengen

c) Disjunkte Entitätsmengen

Abb. 3.40. Beispiel für die Generalisierung und Spezialisierung von Entitätsmengen.

In Abb. 3.40 wird jeweils die gleiche Obermenge in verschiedene Untermengen spezialisiert:

- Im Fall a) liegen überlappende Untermengen vor, weil ein Beschäftigter ein nicht fest beschäftigter Kaufmann oder ein festangestellter Nicht-Kaufmann oder ein festangestellter Kaufmann sein kann.
- Im Fall b) setzen sich die festangestellten und die nicht festangestellten Beschäftigten genau zur Gesamtmenge der Beschäftigten zusammen.
- Im Fall c) gibt es neben Kaufleuten und Ingenieuren noch weitere zu den Beschäftigten zählenden Personengruppen.

Wie die Beispiele in Abb. 3.40 verdeutlichen, lassen sich Obermengen in verschiedener Weise spezialisieren. Welche Untermengen zu bilden sind, hängt von den abzuwickelnden Informationsverarbeitungsprozessen ab.
Beziehungen zwischen Ober- und Untermengen sollten in einem Datenmodell explizit und genau festgehalten werden. Die fünfte Strukturregel fordert daher (vgl. ZEHNDER 1985):

Strukturregel 5

Strukturregel 5:
Vorhandene Ober- und Untermengenbeziehungen zwischen Relationen sind präzise darzustellen. Die Zuordnung einer Entität zu disjunkten spezialisierten Untermengen wird durch ein diskriminierendes Attribut in der generalisierten Relation ausgedrückt.

diskriminierendes Attribut

Gemäß der Strukturregel 5 kommt bei der Modellierung disjunkter spezialisierter Untermengen ein diskriminierendes Attribut ins Spiel. Attribute werden sichtbar, wenn man sich bei der Modellierung auf die Ebene des Relationendiagramms begibt. Dies soll in der folgenden Übungsaufgabe geschehen.

Übungsaufgabe 3.25

Betrachtet werde der Fall b) der in Abb. 3.40 angegebenen Beispiele für die Generalisierung und Spezialisierung von Entitätsmengen.

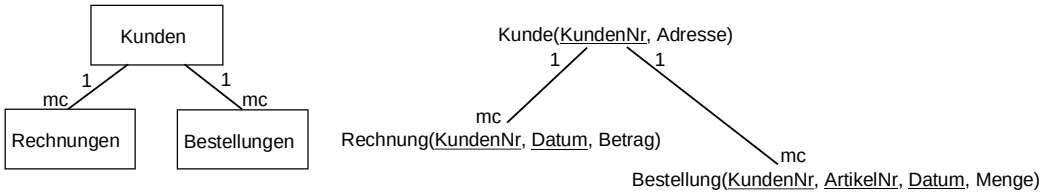
a) Entwerfen Sie das zugehörige Relationendiagramm. Beschränken Sie sich hierbei auf einige wenige Attribute und geben Sie an, welches Attribut dem zu modellierenden Diskriminierungszweck dient.

b) Demonstrieren Sie die Rolle des diskriminierenden Attributs durch die Angabe einiger beispielhafter Tupel für jede Relation des Relationendiagramms.

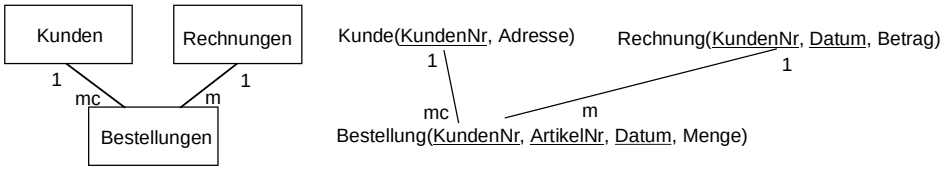
e) Indirekte Beziehungen zwischen Entitätsmengen

direkte und indirekte Beziehungen

Bislang wurden lediglich direkte Beziehungen zwischen Entitätsmengen betrachtet. Denkbar sind jedoch auch indirekte Beziehungen, die sich über mehrere Stufen von Entitätsmengen hinweg ergeben. Indirekte Beziehungen lassen sich wiederum in direkte Beziehungen transformieren. Welche Darstellungsform in einem konkreten Fall zu wählen ist, sei anhand von Abb. 3.41 erörtert.



a) Modellierung mit direkten Beziehungen



b) Mehrstufige Modellierung

Abb. 3.41. Direkte und indirekte Beziehungen zwischen Entitätsmengen.

Modellierung mit direkten Beziehungen

In Abb. 3.41 a) bestehen jeweils direkte Beziehungen zwischen den Entitätsmengen *Kunden* und *Rechnungen* sowie *Kunden* und *Bestellungen*. Diese Form der Modellierung drückt u.a. aus, dass zu einem Kunden keine, eine oder mehrere Rechnungen bzw. Bestellungen existieren können. Entsprechend basiert in der Relation *Kunde* der Identifikationsschlüssel *KundenNr* auf einem statischen Wertebereich und im Einklang mit der dritten Strukturregel wird dieses globale Attribut als Fremdschlüssel in die Relationen *Rechnung* und *Bestellung* eingebracht.

indirekte Modellierung bewirkt...

Anders als das Modell in Abb. 3.41 a) berücksichtigt die in Abb. 3.41 b) gezeigte mehrstufige Modellierung, dass Rechnungen nur für die Kunden erstellt werden dürfen, die auch Bestellungen getätigt haben. Dies gewährleistet die durch das Modell bewirkte Einschränkung des Wertebereichs des Fremdschlüssels *KundenNr* in der Relation *Rechnung* auf die Werte, welche dieses globale Attribut in der Relation *Bestellung* annimmt. Insgesamt liegt also folgende Situation vor: Das globale Attribut *KundenNr* wird direkt in die Relation *Bestellung* als Fremdschlüssel eingebracht und indirekt - über die Relation *Bestellung* - in die Relation *Rechnung* übernommen.

zusätzliche Einschränkung des Wertebereichs

Während in Abb. 3.41 a) der Wertebereich des Fremdschlüssels *KundenNr* der Relation *Rechnung* auf die Kundennummern eingeschränkt wird, die in der Relation *Kunde* tatsächlich auftreten, liegt in Abb. 3.41 b) eine stärkere Einschränkung vor: Der Fremdschlüssel wird auf die Kundennummern eingeschränkt, die in der Relation *Kunde* auftreten und für die Bestellungen vorliegen.

Zweifellos ist der präziseren Modellierung in Abb. 3.41 b), die den Wertebereich des Fremdschlüssels *KundenNr* in der Relation *Rechnung* am stärksten einschränkt, der Vorzug zu geben. Dies drückt auch die sechste Strukturregel aus.

sechste Strukturregel

> Sechste Strukturregel (SR 6):
> Die globalen Attribute einer Relation, die nicht auf statischen Wertebereichen basieren, sind als Fremdschlüssel aus denjenigen Relationen einzuführen, welche die größtmögliche Einschränkung des zulässigen Wertebereichs bewirken.

global normalisierte Datenbasis

THURNHERR und ZEHNDER bezeichnen eine Datenbasis, deren Relationen allen sechs Strukturregeln genügen, als global normalisiert. Der Prozess der globalen Normalisierung schließt den in Kapitel 3.4.1 behandelten Normalisierungsprozess ein. Denn die Beachtung der sechs Strukturregeln führt zu einer Datenbasis, die nur aus Relationen in 3. Normalform besteht. Charakteristisch für eine global normalisierte Datenbasis ist außerdem, dass sie ausschließlich hierarchische Beziehungen zwischen Relationen enthält, die mit Hilfe globaler Attribute bzw. Fremdschlüssel realisiert werden.

Das erweiterte Relationenmodell von THURNHERR und ZEHNDER eignet sich in besonderer Weise zur Datenmodellierung im betrieblichen Bereich. Es wird daher dem im nächsten Kapitel behandelten logischen Datenbankentwurf zugrunde gelegt.

3.5 Logischer Datenbankentwurf

Definition der Datenbasis und der Datenbankoperationen

Gegenstand des logischen Datenbankentwurfs ist die Definition einer Datenbasis für einen betrachteten Realitätsausschnitt einschließlich der diese Datenbasis manipulierenden Datenbankoperationen. Die Struktur der Datenbasis wird durch das verwendete Datenmodell bestimmt. Das vorliegende Kapitel geht von einer global normalisierten Datenbasis aus. Mit den zuvor formulierten Strukturregeln stehen damit einige Richtlinien für den Entwurf der Datenbasis zur Verfügung.

Konsistenz der Datenbasis

Transaktionen

Bei dem Entwurf der Datenbankoperationen ist der Konsistenz der Datenbasis besondere Beachtung zu schenken. Von Interesse sind nur solche Operationen, welche eine konsistente Datenbasis nicht in einem inkonsistenten Zustand hinterlassen. Datenbankoperationen mit dieser Eigenschaft bezeichnet man als Transaktionen.

Konsistenzbedingungen

Transaktionen können nur definiert werden, wenn die Bedingungen für die Gewährleistung der Konsistenz einer Datenbasis bekannt sind. Die explizite Formulierung dieser sogenannten Konsistenzbedingungen ist daher ein wesentlicher Bestandteil des logischen Datenbankentwurfs.
Die umrissenen Entwurfsziele und -aufgaben geht man sinnvollerweise in Schritten an. Denkbar sind folgende, in einem iterativen Prozess mehrfach zu durchlaufenden Entwurfsschritte (vgl. auch ZEHNDER 1985, S.68):

Entwurfsschritte

- Problemabgrenzung,
- Definition von Entitätsmengen und Beziehungen,
- Definition von Relationen,
- Umwandlung nichthierarchischer Beziehungen,
- Definition von Konsistenzbedingungen,
- Definition von Transaktionen.

Inhalt des Kapitels

Im Kapitel 3.5.1 werden die Schritte des logischen Datenbankentwurfs anhand eines praktischen Beispiels erläutert. Auf die Verzahnung des logischen Datenbankentwurfs bzw. der Entwicklung von Datenbankanwendungen mit dem allgemeinen Prozess der Softwareentwicklung geht das Kapitel 3.5.2 ein.

3.5.1 Schritte des logischen Datenbankentwurfs

Ziele des logischen Datenbankentwurfs

Der logische Datenbankentwurf zielt auf folgende Entwurfsergebnisse ab:

- Eine global normalisierte Datenbasis, welche die Datenwelt eines gegebenen Realitätsausschnitts auf der logischen Ebene beschreibt.
- Eine Menge von Konsistenzbedingungen, welche die inhaltliche Korrektheit der Datenbasis gewährleisten.
- Eine Menge von Transaktionen, welche geeignet sind, die Datenbasis unter Einhaltung der Konsistenzbedingungen zu manipulieren.

Mit den nachfolgend dargestellten Entwurfstätigkeiten lassen sich diese Ergebnisse erreichen.

a) Problemabgrenzung

Analogien zur Problemanalyse

Bei dem logischen Datenbankentwurf geht es stets darum, eine aus Daten und auf den Daten definierten Operationen bestehende Grundlage zu schaffen, die gleichermaßen für die verschiedensten Datenbankanwendungen zur Verfügung steht. Grundsätzlich sollte daher der Problemabgrenzung der gleiche Stellenwert zukommen wie der Problemanalyse im Zyklus der Softwareentwicklungstätigkeiten. Auch inhaltlich ergeben sich weitgehende Analogien zur Problemanalyse. Innerhalb der Tätigkeiten

- Abgrenzung des relevanten Realitätsausschnitts,
- Beschreibung des Realitätsausschnitts,
- Darstellung der aktuellen Informationsverarbeitungsprozesse,
- Analyse der Schwachstellen der gegenwärtigen Situation,

Datenobjekte und manipulierende Prozesse

sollte jedoch den durch Daten abzubildenden Objekten und den die Objekte bzw. Daten manipulierenden Prozessen besondere Aufmerksamkeit gewidmet werden, denn aus diesen Daten und Verarbeitungsprozessen ist die Datenbasis abzuleiten. Dem nachfolgend dargestellten Beispiel einer Problemabgrenzung liegt ein praktischer Problemfall zugrunde, der hier allerdings nur vereinfacht wiedergegeben werden kann.

Beispiel 3.6

praktischer Problemfall

Ein Mittelbetrieb der Elektrobranche stellt Schaltgeräte zur Steuerung elektrischer Systeme und Anlagen her. Die Geräte bzw. Artikel sind von unterschiedlicher Komplexität. Komplexe Geräte können einfachere Artikel als Komponenten enthalten. Aus welchen Artikeln und Teilen ein Gerät besteht, geht aus der Stückliste hervor.

Geräte bzw. Artikel werden im Kundenauftrag gefertigt. Häufiger als Komponenten verwendete Artikel werden auch auf Vorrat bzw. Lager produziert. Die Lagerhaltung erstreckt sich außerdem auf die in die Geräte eingehenden (Einzel-)Teile. Teile werden nicht im Hause gefertigt, sondern durchweg von Lieferanten bezogen.

Aus Gründen der rationelleren Fertigung wird die Fertigung für mehrere Kunden zusammengelegt. Dies hat den Vorteil, dass Geräte und Teile, die für unterschiedliche komplexere Geräte benötigt werden, nicht separat abgewickelt werden müssen.
Mit einem zu entwickelnden Datenbanksystem sollen insbesondere die Auftragsabwicklung, die Lagerhaltung und die Artikelkalkulation unterstützt werden. Die Datenbasis ist so zu konzipieren, dass sie im Einzelnen folgende Informationsverarbeitungsaufgaben unterstützt:

zu unterstützende Informationsverarbeitungsaufgaben

- Erfassung von Artikelbewegungen, d.h. von Zu- und Abgängen von Geräten und Teilen, und Führen eines Artikelbestandes. Ein Kundenauftrag kann die Bestellung mehrerer Artikel umfassen. Eine Artikelkalkulation wird kundenauftragsbezogen durchgeführt. Das heißt, für jeden der im Rahmen eines Kundenauftrages bestellten Artikel kann eine Kalkulation durchgeführt werden. Eine Kalkulation erübrigt sich dann, falls auf eine bereits vorhandene Kalkulation zurückgegriffen wird. Umgekehrt wird die Kalkulation für einen Artikel im Rahmen eines bestimmten Kundenauftrages durchgeführt
- Verwaltung von Stücklisten für sämtliche Artikel, die als Geräte im Hause gefertigt werden.

- Verwaltung von Artikelkalkulationen, d.h. bereits früher erstellten Preiskalkulationen für Geräte.
- Verwaltung von Kunden- und Fertigungsaufträgen, wobei der Zusammenhang zwischen beiden Auftragsarten jederzeit deutlich sein muss. Ein Kundenauftrag löst für jeden der bestellten Artikel eine Artikelkalkulation aus. Eine Artikelbewegung ist genau einem bestimmten Kundenauftrag zuzuordnen.
- Erfassung des Fertigungsaufwandes nach Material, d.h. Verbrauch an Geräten und Teilen, sowie nach bewerteter Zeit, d.h. bewertetem Verbrauch an Mitarbeiter-Fertigungszeiten; bei der Zeiterfassung ist zugleich eine Zuordnung zu Mitarbeitern, Lohnarten und Kostenstellen zu ermöglichen.
- Ermittlung von Fertigungskosten und Kalkulation von Kundenaufträgen bzw. Geräten unter Berücksichtigung früherer Gerätekalkulationen.
- Verwaltung von "Zeitlisten" für Geräte, aus denen der bewertete Fertigungszeitbedarf - gegebenenfalls aufgeschlüsselt nach als Komponenten verwendeten einfacheren Geräten - hervorgeht.
- Kontrolle des zeitlichen Fertigungsaufwandes durch die Gegenüberstellung von kumulierten Fertigungszeiten und kumulierten Fertigungszeitbedarfen auf der Ebene von einem oder mehreren Kundenaufträgen.

Die in den Zeitlisten für Geräte auszuweisenden bewerteten Soll-Fertigungszeiten stellen Erfahrungswerte dar, die sich im Laufe der Zeit herausgebildet haben. Die Bewertung von Fertigungszeiten hat mit Verrechnungslöhnen bzw. Personalkostensätzen zu erfolgen. Bewertete Fertigungszeiten werden im Weiteren auch als Arbeitswerte bezeichnet.

zum Begriff des Artikels

Mit dem Begriff "Artikel" werden im obigen Text sowohl Geräte als auch Teile bezeichnet. Im Einzelnen fallen unter diesen Begriff somit:

- End- bzw. Fertigprodukte, d.h. komplexe Geräte, die selbst wieder (einfachere) Geräte als Komponenten enthalten.
- Halbfertigprodukte, d.h. einfachere Geräte, die als Komponenten in Endprodukte eingehen.
- (Einzel-)Teile, die in Endprodukte sowie in Halbfertigerzeugnisse eingehen.

Stücklisten existieren nur für Fertig- und Halbfertigprodukte und Zeitlisten nur für Fertigprodukte.

b) Definition von Entitätsmengen und Beziehungen

Bei der Definition von Entitätsmengen ist auf Überlappungen zu achten und es sind geeignete Generalisierungen oder Spezialisierungen vorzunehmen. Die Notwendigkeit von Generalisierungen und Spezialisierungen kann sich jedoch auch im Falle disjunkter Entitätsmengen ergeben.

Die Beziehungen zwischen den Entitätsmengen werden zunächst nicht eingeschränkt. Neben hierarchischen Beziehungen können also auch konditionelle, netzwerkartige und

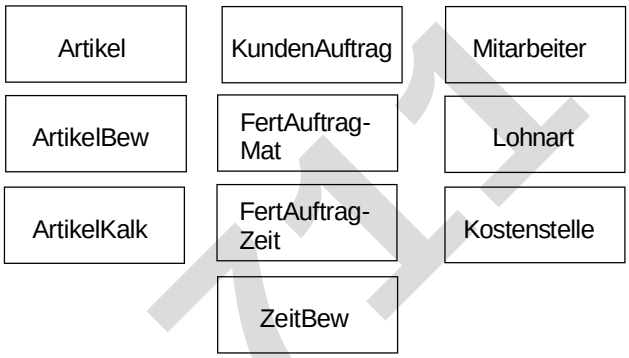
rekursive Beziehungen verwendet werden. Die Elimination unerwünschter Beziehungsarten erfolgt später.

Zur Darstellung von Beziehungen wird bekanntlich kein eigenständiges Symbol verwendet. Vielmehr drücken die Verbindungslinien zwischen den Entitätsmengen Beziehungen aus. Falls erforderlich, werden nichttriviale Beziehungen, z.B. Mehrfachbeziehungen oder Rekursionen, durch eine Beschriftung gekennzeichnet und hervorgehoben.

Fortsetzung Beispiel 3.6

aus der Analyse resultierende Entitätsmengen

Aus dem Problemanalysetext ergeben sich etwa folgende Entitätsmengen:

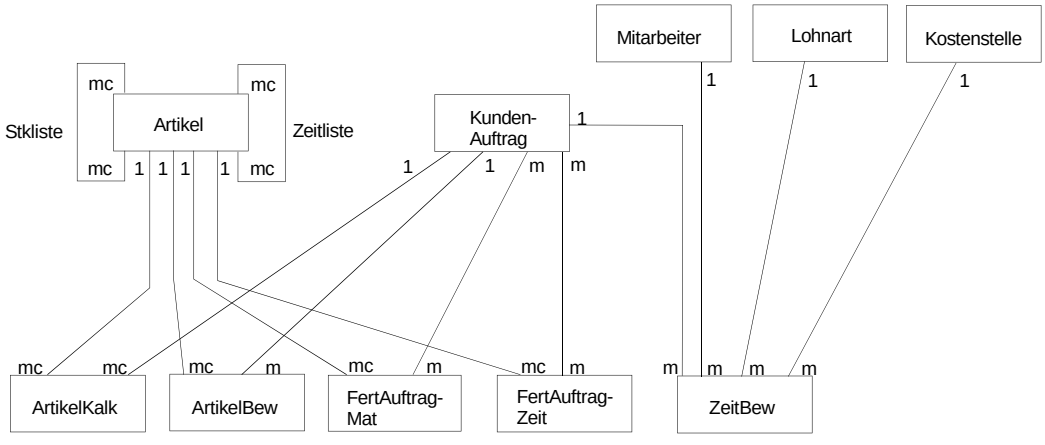


Zum Unterschied zwischen *FertAuftragMat* und *FertAuftragZeit* bedarf es eines Hinweises. Eine Entität der Menge *FertAuftragMat* spezifiziert ein zu fertigendes End- oder Halbfertigprodukt hinsichtlich des einzusetzenden Materials. Bei der Entitätsmenge *FertAuftragZeit* geht es dagegen um die gesamte bewertete Fertigungszeit pro Gerät im Sinne einer bewerteten Vorgabezeit, der - nach erfolgter Fertigung - die tatsächliche bewertete Fertigungszeit gegenübergestellt werden kann. Angemerkt sei außerdem:

- Die Entitätsmenge *ArtikelBew* beschreibt Artikelbewegungen,
- die Entitätsmenge *ArtikelKalk* beinhaltet bereits früher erstellte Artikelkalkulationen,
- die Entitätsmenge *ZeitBew* dient dem Erfassen von Fertigungszeiten und deren Bewertung. Im Unterschied zur Entitätsmenge *FertAuftragZeit* werden Ist-Zeiten beschrieben.

Die Bedeutung der übrigen Entitätsmengen ergibt sich aus ihrer Bezeichnung.

Die Beziehungen zwischen den Entitätsmengen gehen aus dem folgenden Datenmodell hervor:



Das vorläufige Datenmodell enthält zehn hierarchische Beziehungen und vier nichthierarchische Beziehungen.
Bei den hierarchischen Beziehungen handelt es sich um

hierarchische Beziehungen

- vier 1:mc-Beziehungen zwischen *Artikel* sowie je *ArtikelKalk*, *ArtikelBew*, *FertAuftragMat*, *FertAuftragZeit*,
- eine 1:mc-Beziehung zwischen *KundenAuftrag* und *ArtikelKalk*,
- vier 1:m-Beziehungen zwischen je *KundenAuftrag*, *Mitarbeiter*, *Lohnart*, *Kostenstelle* sowie *ZeitBew*
- und eine 1:m-Beziehung zwischen *KundenAuftrag* und ArtikelBew.

Zu den vier nichthierarchischen Beziehungen gehören

nichthierarchische Beziehungen

- zwei auf der Entitätsmenge *Artikel* definierte rekursive mc-mc-Beziehungen, nämlich *Stkliste* und *Zeitliste*,
- zwei m-m-Beziehungen zwischen *FertAuftragMat* und *KundenAuftrag* sowie zwischen *FertAuftragZeit* und *KundenAuftrag*.

Die beiden rekursiven Beziehungen modellieren Ober-/Unterteilbeziehungen zwischen Artikeln und zwar

rekursive Beziehungen

- im Falle der Beziehung *Stkliste* hinsichtlich Artikelmengen und
- im Falle der Beziehung *Zeitliste* hinsichtlich bewerteten Fertigungszeiten bzw. Arbeitswerten.

Übungsaufgabe 3.26

Betrachtet seien die beiden rekursiven Beziehungen des in Beispiel 3.6 vorgestellten vorläufigen Datenmodells. Begründen Sie, warum in beiden Fällen eine rekursive mc-mc-Beziehung vorliegt.

Übungsaufgabe 3.27

Betrachtet seien die beiden m-m-Beziehungen des in Beispiel 3.6 vorgestellten vorläufigen Datenmodells. Begründen Sie je den gewählten Typ der m-m-Beziehung.

c) Definition von Relationen

Hier stehen vor allem folgende Tätigkeiten im Vordergrund:

- Zuordnung einer Relation zu jeder Entitätsmenge,
- Festlegung eines Identifikationsschlüssels für jede Relation und
- Ergänzung der Relationen um weitere bzw. lokale Attribute.

natürliche oder künstliche Schlüssel

Als Identifikationsschlüssel kommen natürliche oder künstliche Schlüssel in Frage. Natürliche Identifikationsschlüssel bestehen aus Attributen, welche das Wesen von Entitätsmengen charakterisieren. Künstliche Schlüssel sind dagegen Mengen von fortlaufend vergebenen Nummern.

Zuordnung lokaler Attribute

Bereits jetzt können den einzelnen Entitätsmengen lokale Attribute zugeordnet werden. Hinweise auf relevante Attribute sind den Formularen, Berichten und sonstigen Unterlagen zu entnehmen, mit denen man sich im Rahmen der Problemabgrenzung auseinandersetzt.

Fortsetzung Beispiel 3.6

Den im ER-Diagramm ausgewiesenen Entitätsmengen werden folgende vorläufige Relationen zugeordnet:

vorläufige Relationen

Artikel(ArtikelNr, Bezeichnung, TechnDaten, Preis, Bestand, MindBestand, DispoBestand, AuftrBestand)

ArtikelBew(ArtikelNr, AuftrNr, BewDatum, Menge, DispoMenge)

ArtikelKalk(ArtikelNr, AuftrNr, FertMenge, MatKosten, ZeitKosten, SondKosten, HerstellKosten)

KundenAuftrag(AuftrNr, KundenNr, Name, AuftrDatum)

FertAuftragMat(StklisteNr, Menge, FertDatum)

FertAuftragZeit(ZeitlisteNr, ArbWert, FertDatum)

ZeitBew(AuftrNr, MitarbNr, BewDatum, Zeit, LohnartNr, KoststellNr, ArbWert, Bemerkung)

Mitarbeiter(MitarbNr, Name, Vorname, Straße, PLZ, Ort, TelefonNr, Eintritt)

Lohnart(LohnartNr, Bezeichnung, Bemerkung, ArbWertMonat, ArbWertJahr)

Kostenstelle(KostenstNr, Bezeichnung, Bemerkung, KalkBasis, VerrechLohn, ArbWertMonat, ArbWertJahr)

Bei den meisten Attributen lässt sich die Bedeutung an der Bezeichnung ablesen. Zu einigen Attributen der Relation *Artikel* sind Hinweise erforderlich:

Hinweise zu Attributen

- *Bestand* bezeichnet den aktuellen Bestand.
- *MindBestand* steht für "Mindestbestand" und bezeichnet eine möglichst nicht zu unterschreitende Bestandsuntergrenze.
- *DispoBestand* steht für "Dispositionsbestand" und gibt an, wie viele Einheiten aufgrund angenommener Aufträge bereits "gebunden" bzw. nicht frei verfügbar sind.
- *AuftrBestand* steht für "Auftragsbestand" und bezieht sich hier auf die in der Fertigung befindliche Einheiten, mit deren Zugang bereits gerechnet werden kann.

d) Umwandlung nichthierarchischer Beziehungen

Einführung von Hilfsentitätsmengen

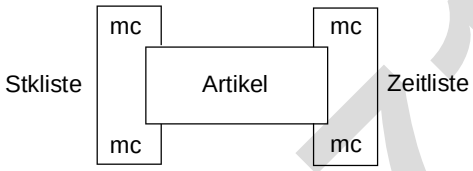
Alle nichthierarchischen Beziehungen werden in diesem Schritt durch die Einführung von Hilfsentitätsmengen in hierarchische Beziehungen transformiert. Graphisch werden sämtliche Beziehungen so arrangiert, dass sich übergeordnete Entitätsmengen stets oben und untergeordnete stets unten befinden. Dadurch kommt auch optisch deutlicher zum

Ausdruck, dass ein global normalisiertes Datenmodell durch einen Graphen repräsentiert werden kann, der eine Halbordnung darstellt.

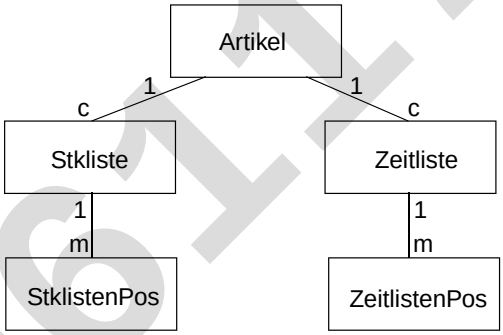
Im Zuge der globalen Normalisierung ergeben sich zusätzliche Entitätsmengen, die nun ebenfalls durch Relationen zu beschreiben sind. Die globale Normalisierung führt quasi automatisch zu Relationen in 3. Normalform. Dennoch sollten sämtliche Relationen nochmals überprüft werden. Insbesondere auch hinsichtlich der korrekten Definition von globalen und lokalen Attributen sowie hinsichtlich der Verwendung globaler Attribute als Fremdschlüssel.

Fortsetzung Beispiel 3.6

Das in Schritt b) entworfene Datenmodell weist zwei rekursive und zwei netzwerkartige Beziehungen auf. Beide rekursive Beziehungen können in gleicher Weise transformiert werden. Die Darstellung



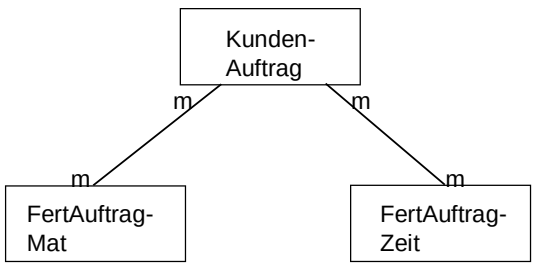
wird transformiert in:



Transformation rekursiver Beziehungen

Die beiden Beziehungen *Stkliste* und *Zeitliste* werden als Entitätsmengen dargestellt und um die Entitätsmengen *StklistenPos* und *ZeitlistenPos* ergänzt. Eine Entität der Entitätsmenge *Stkliste* repräsentiert genau ein End- oder ein Halbfertigprodukt. Zu einem Endprodukt finden sich in der Entitätsmenge *StklistenPos* die Entitäten, welche in das Endprodukt als Halbfertigprodukte eingehen. Entsprechend existieren zu einer Halbfertigprodukt-Entität die dazugehörigen Artikel-Entitäten in der Entitätsmenge *StklistenPos*. Ein analoger Zusammenhang besteht zwischen *Zeitlisten* und *ZeitlistenPos*, jedoch nur in Bezug auf Endprodukte und Halbfertigprodukte.

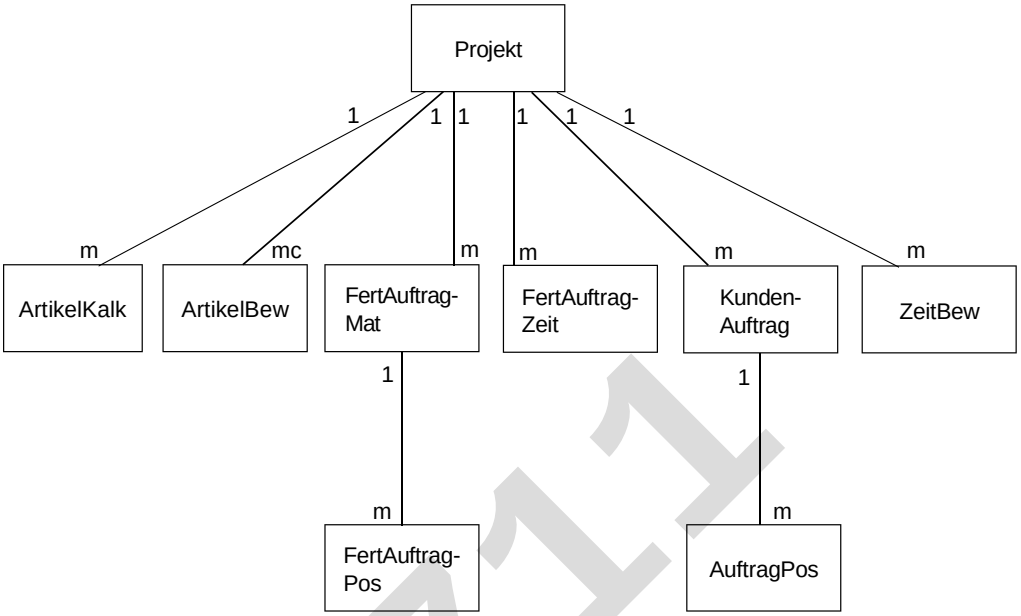
Die beiden netzwerkartigen Beziehungen



Transformation netzwerkartiger Beziehungen

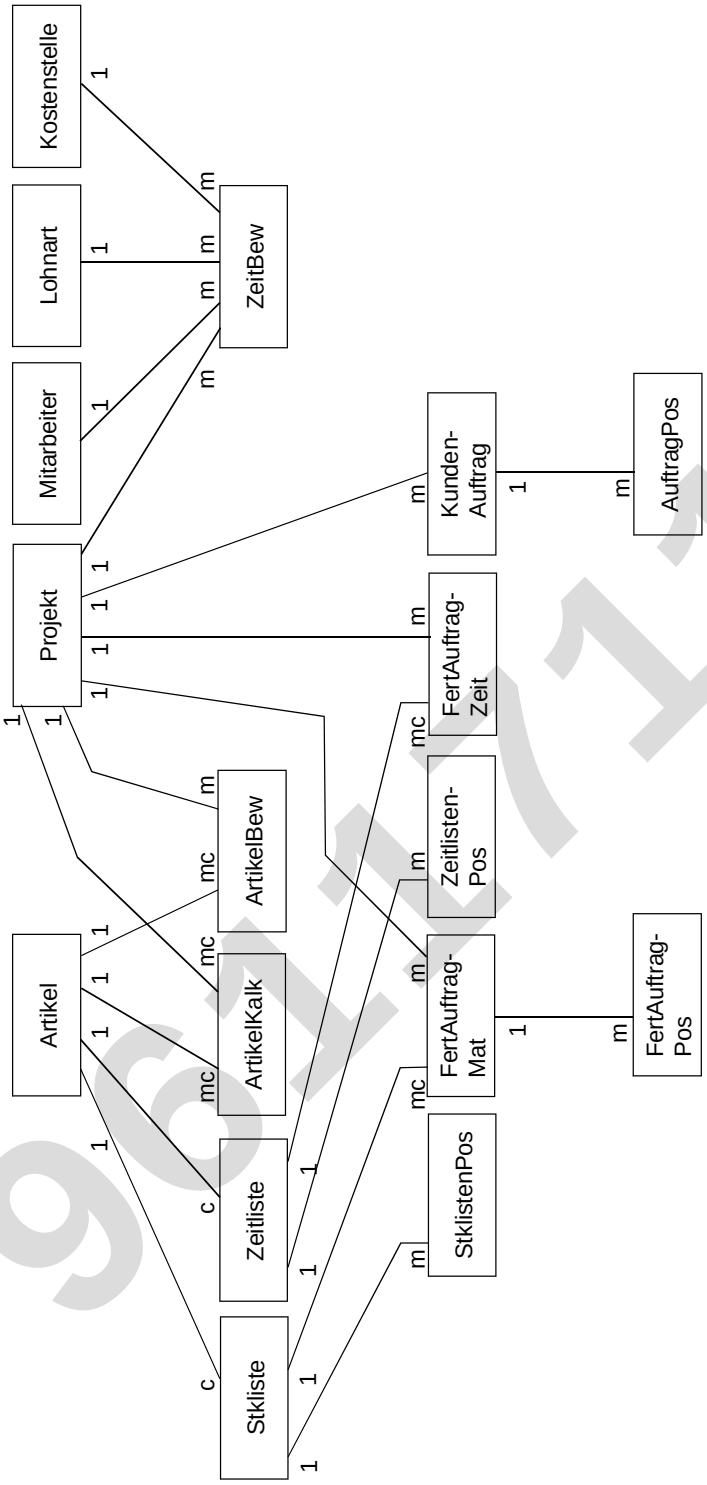
werden unter Einführung einer Hilfsentitätsmenge *Projekt*, an die auch die Entitätsmengen *ZeitBew* angehängt werden, umgewandelt in:

Transformationsergebnis



Mit dieser Transformation ist eine organisatorische Maßnahme gekoppelt. Es werden Projekte definiert, die eine bessere Auftragsverfolgung ermöglichen sollen und zugleich berücksichtigen, dass in der Regel mehrere Kundenaufträge und mehrere Fertigungsaufträge gemeinsam abgewickelt werden. Ein Projekt umfasst somit mehrere Kundenaufträge und mehrere Fertigungsaufträge. Für jeden Kundenauftrag werden die zugehörigen Halbfertigerzeugnisse mittels der Entitätsmenge *AuftragPos* erfaßt, für jeden Fertigungsauftrag die zu fertigenden Halbfertigerzeugnisse mittels der Entitätsmenge *FertAuftragPos*. Da Zeitbewegungen nun einzelnen Projekten zugeordnet werden, erfolgt die Zeitkontrolle auf der Ebene von Projekten. Darüber hinaus wird ebenfalls die Artikelbewegung und die Artikelkalkulation projektbezogen verfolgt bzw. durchgeführt

Insgesamt ergibt sich damit das folgende relationale Datenmodell:



Da zusätzliche Entitätsmengen hinzugekommen sind und sich bei einigen Relationen außerdem Änderungen ergeben haben, seien sämtliche Relationen im Zusammenhang dargestellt:

endgültiges Datenmodell

Artikel(<u>ArtikelNr</u>, Bezeichnung, TechnDaten, Preis, Bestand, MindBestand, DispoBestand, AuftrBestand)

ArtikelBew(<u>ArtikelNr</u>, <u>ProjektNr</u>, <u>BewDatum</u>, Menge, DispoMenge)

ArtikelKalk(<u>ArtikelNr</u>, <u>ProjektNr</u>, FertMenge, MatKosten, ZeitKosten, SondKosten, HerstellKosten)

endgültige Relationen

Stkliste(StklisteNr, ArtikelNr, StklisteDatum)

StklistenPos(StklisteNr, Position, ArtikelNr, Menge)

FertAuftragMat(ProjektNr, StklisteNr, Menge, FertDatum)

FertAuftragPos(ProjektNr, StklisteNr, Position, ArtikelNr, Menge)

Zeitliste(ZeitlisteNr, ArtikelNr, Datum)

ZeitlistenPos(ZeitlisteNr, Position, ArtikelNr, Menge, ArbWert)

FertAuftragZeit(ProjektNr, ZeitlisteNr, ArbWert, FertDatum)

Projekt(ProjektNr, FertDatum, EinkaufWoche, LagerWoche, SollZeit, SollMat, SollMenge, IstMenge)

KundenAuftrag(AuftrNr, ProjektNr, KundenNr, Name, AuftrDatum)

AuftragPos(AuftrNr, Position, ArtikelNr, Menge, Preis)

ZeitBew(ProjektNr, MitarbNr, BewDatum, Zeit, LohnartNr, KoststellNr, ArbWert, Bemerkung)

Mitarbeiter(MitarbNr, Name, Vorname, Straße, PLZ, Ort, TelefonNr, Eintritt)

Lohnart(LohnartNr, Bezeichnung, Bemerkung, ArbWertMonat, ArbWertJahr)

Kostenstelle(KoststellNr, Bezeichnung, Bemerkung, KalkBasis, Verrechnungslohn, ArbWertMonat, ArbWertJahr)

Aufgrund der graphischen Übersichtlichkeit sind in dem links abgebildeten Entity Relationship Modell die vier 1:mc-Beziehungen zwischen *Artikel* sowie *StklistenPos*, *FertAuftragPos*, *ZeitlistenPos* und *AuftragPos* nicht eingezeichnet

e) Definition von Konsistenzbedingungen

Konsistenzbedingungen sind Vorschriften, welche die inhaltliche Korrektheit einer Datenbasis bei ihrer Erzeugung und während ihres Betriebs gewährleisten sollen. Die Formulierung von Konsistenzbedingungen betrifft die logische Ebene einer Datenbank und fällt daher unter den logischen Datenbankentwurf.

modellinhärente und modellexterne Konsistenzbedingungen

Man unterscheidet zwischen modellinhärenten und modellexternen Konsistenzbedingungen. Modellinhärente Konsistenzbedingungen werden durch ein benutztes Datenbankmodell gleichsam automatisch sichergestellt. Dagegen müssen modellexterne Konsistenzbedingungen explizit formuliert werden.

Beispiele für modellinhärente Konsistenzbedingungen sind die Forderung der Eindeutigkeit von Identifikationsschlüsseln, die Vorgabe statischer Wertebereiche für Attribute und die Bedingung der referentiellen Integrität. Als Beispiel für eine modellexterne Konsistenzbedingung sei die Forderung genannt, dass bei der Modellierung einer Teilehierarchie die Anzahl der Oberteilbeziehungen gleich der Anzahl der Unterteilbeziehungen ist.

In Kapitel 6 werden verschiedene Arten von Konsistenzbedingungen und die Formulierung von Konsistenzbedingungen eingehender behandelt. An dieser Stelle mögen daher die obigen, etwas allgemein gehaltenen Ausführungen genügen.

f) Definition von Transaktionen

Transaktionen sind konsistenzerhaltende Operationen

Sämtliche bisherigen Entwurfsschritte bezogen sich auf Daten und Beziehungen zwischen Daten. Im vorliegenden Schritt geht es um Operationen, und zwar solche Operationen, welche unmittelbar auf die Daten einer Datenbasis zugreifen. Von Interesse sind allerdings nur die Operationen, die eine konsistente Datenbasis nach ihrer Ausführung in einem konsistenten Zustand hinterlassen. Operationen mit dieser Eigenschaft heißen Transaktionen. Transaktionen sind folglich Operationen, die den vorgegebenen modellinhärenten und modellexternen Konsistenzbedingungen in vollem Umfang genügen. Die ausschließliche Verwendung von Transaktionen zur Manipulation einer Datenbasis stellt sicher, dass Inkonsistenzen erst gar nicht entstehen.

Leseoperationen und Mutationen

Leseoperationen ändern keine Dateninhalte und sind daher grundsätzlich Transaktionen. Anders verhält es sich mit Mutationen. Sie bewirken Änderungen von Dateninhalten und können daher zu Inkonsistenzen führen. Diese Gefahr besteht nicht bei Mutationen, welche die Einhaltung der gegebenen Konsistenzbedingungen gewährleisten. Mutationen dieser Art stellen folglich Transaktionen dar.

Zugang zur Datenbasis über Transaktionen

Ohne Zweifel handelt es sich bei der Definition von Transaktionen um eine Entwurfsaufgabe, die dem logischen Datenbankentwurf und nicht etwa dem Entwurf von Anwendungssystemen zuzurechnen ist. Eine Datenbasis und die sie manipulierenden Transaktionen bilden eine Einheit, die man sich als einen aus der Datenbasis bestehenden Kern und den diesen Kern umgebenden bzw. einhüllenden Transaktionen vorstellen kann. Der aus Datenbasis und Transaktionen bestehenden Einheit stehen die verschiedenen Anwendungen gegenüber. Kein Anwendungssystem hat direkten Zugang zu der Datenbasis. Vielmehr muss sich jedes Anwendungssystem der Transaktionen bedienen.
Transaktionen kann man somit als Dienste begreifen, welche für die verschiedenen Anwender einer Datenbasis zur Verfügung stehen. Bei der Definition von Transaktionen sind daher auch die Anforderungen der verschiedenen Anwender zu berücksichtigen.

Kriterien für den Transaktionsentwurf

Bei der Formulierung von Transaktionen bestehen erhebliche Freiheitsgrade. So kann beispielsweise die Reichweite einer Änderungsoperation mehrere Tupel oder nur einen Tupel einschließen, oder sich gar nur auf einzelne Attribute beschränken - was unter Effizienzgesichtspunkten allerdings fragwürdig wäre. Neben Konsistenzbedingungen und Anwenderanforderungen bilden Effizienzüberlegungen das dritte bei dem Transaktionsentwurf zu berücksichtigende Kriterium.
Nachfolgend werden einige Beispiele für Transaktionen angegeben.

Fortsetzung Beispiel 3.6

Betrachtet werde das im Schritt d) entworfene global normalisierte Datenmodell. In Verbindung mit der Relation *Zeitliste* sind u.a. folgende Transaktionen auf der Ebene je eines Tupels denkbar:

Beispiele für Transaktionen

- Lesen einer Zeitliste,
- Hinzufügen einer Zeitliste,
- Ändern einer Zeitliste,
- Löschen einer Zeitliste.

Diese Transaktionen sind unvollständig formuliert. Sie wären je um Parameter zu ergänzen, welche die auszuführenden Operationen genau spezifizieren. So müsste z.B. qualifiziert werden, welcher Tupel gelesen werden soll.

Was die Konsistenzproblematik betrifft, sei beispielhaft die Transaktion *Hinzufügen einer Zeitliste* betrachtet. Zu berücksichtigen sind u.a. folgende Konsistenzbedingungen:

Beispiele für Konsistenzbedingungen

- Eine Zeitliste darf in der Relation *Zeitliste* nur hinzugefügt werden, wenn das zugehörige Endprodukt in der Relation *Artikel* existiert.
- Falls in der Relation *Zeitliste* eine Zeitliste hinzugefügt wird, müssen die dazugehörigen Zeitlistenpositionen in die Relation *ZeitlistenPos* eingetragen werden.

3.5.2 Datenbankentwurf und Softwareentwicklungsprozess

Die Notwendigkeit der Einbeziehung des Datenbankentwurfs in den Prozess der Softwareentwicklung ergibt sich zwangsläufig, falls ein datenbankgestütztes Anwendungssystem entwickelt werden soll. Das auf der folgenden Seite dargestellte Produktmuster zeigt exemplarisch, wie die Einbindung des Datenbankentwurfs und weiterer datenbankbezogener Entwicklungstätigkeiten in den Softwareentwicklungsprozess organisiert werden kann.

grobes Produktmuster für die Entwicklung von Datenbankanwendungen

PRODUKT-MUSTER FÜR DIE ENTWICKLUNG DATENBANKGESTÜTZTER ANWENDUNGSSYSTEME

1 PROBLEMANALYSE

1.1 Beschreibung des Ist-Zustandes
1.2 Schwachstellenanalyse
1.3 Entwicklung eines Soll-Konzepts
1.4 Durchführbarkeitsstudie

2 ANFORDERUNGSDEFINITION

2.1 Definition des logischen Datenmodells
2.2 Definition der Transaktionen
2.3 Definition der Datenbankanwendungen
2.4 Definition der Benutzerschnittstellen

3 EXTERNER ENTWURF UND SYSTEMENTWURF

Softwareentwicklungsmethoden lassen sich durchaus so modifizieren, dass sie speziell auch bei der Entwicklung datenbankgestützter Anwendungssysteme genutzt werden können. Dies sei am Beispiel der Methode "Structured Systems Analysis", einer Methode zur Anforderungsanalyse, demonstriert. Die Abb. 3.42 zeigt beispielhaft, wie das Zusammenwirken von Datenbasis, Transaktionen und Prozessen modelliert werden kann.

Einbeziehung von Transaktionen in ein Datenflussdiagramm

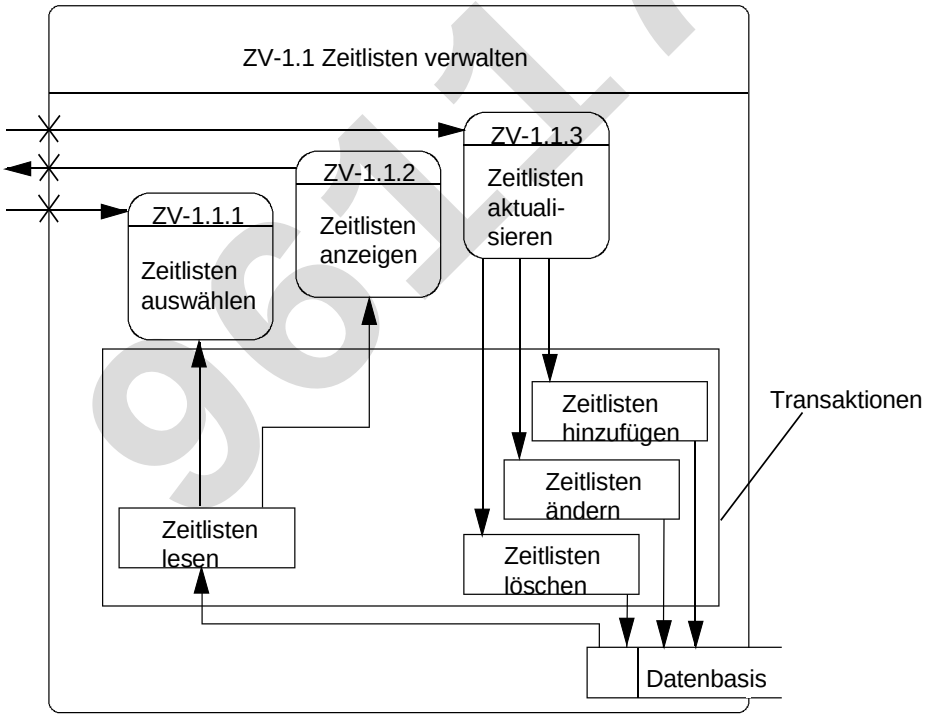


Abb. 3.42. Beispiel für die Darstellung von Transaktionen in einem Datenflussdiagramm.

Zusätzlich zu den aus dem Kurs "Software Engineering" bekannten Symbolen tritt in Abb. 3.42 das Rechteck als Symbol für eine Transaktion auf. Im gegebenen Fall haben die Prozesse *Zeitlisten auswählen*, *Zeitlisten anzeigen* und *Zeitlisten aktualisieren* nur über die ausgewiesenen Transaktionen Zugang zu der Datenbasis.

Lösungen zu den Übungsaufgaben

Übungsaufgabe 3.1

Das Diagramm aus der Übungsaufgabe 3.1 zeigt, dass die Entitätsmenge *Stammkunden* vollständig in der Entitätsmenge *Kunden* enthalten ist. Dieser Sachverhalt ist auch in der Abb. 3.4 a) dargestellt.

Hingegen werden in der Abb. 3.4 a) weitere Spezialisierungen vorgenommen: Es werden ebenfalls die in *Kunden* enthaltenen Entitätsmengen *Laufkunden* und *ausländische Kunden* dargestellt, wobei sich die Entitätsmenge *ausländische Kunden* mit den Entitätsmengen *Stammkunden* und *Laufkunden* überlappt.

Der Verzicht auf diese Spezialisierungen in Übungsaufgabe 3.1 drückt aus, dass bei der Verarbeitung von Kundendaten eine separate Betrachtung von Laufkunden und ausländischen Kunden nicht erforderlich ist.

Übungsaufgabe 3.2

Bei der 1-m-Beziehung handelt es sich um die Darstellung des Verhältnisses zwischen Artikelgruppen und Artikeln. Da eine Artikelgruppe mindestens einen Artikel enthält; gilt:

Assoziation (Artikelgruppe, Artikel) = m.

Außerdem ist jeder Artikel genau einer Artikelgruppe zugeordnet. Es gilt also:

Assoziation (Artikel, Artikelgruppe) = 1.

Die c-m-Beziehung beschreibt den Zusammenhang zwischen Mitarbeitern und Religionen. Da einer berücksichtigten Religionsgemeinschaft ein oder mehrere Mitarbeiter angehören werden, gilt:

Assoziation (Religion, Mitarbeiter) = m.

Andererseits sind jedoch konfessionslose Mitarbeiter nicht auszuschließen. Daher folgt:

Assoziation (Mitarbeiter, Religion) = c.

Übungsaufgabe 3.3

Ausgehend von den vier Assoziationstypen (1,c,m,mc) sind zwischen zwei Entitätsmengen 16 verschiedene Beziehungstypen möglich:

A(R2, R1) \ A(R1, R2)	1	c	m	mc
1	1-1	1-c	1-m	1-mc
c	c-1	c-c	c-m	c-mc
m	m-1	m-c	m-m	m-mc
mc	mc-1	mc-c	mc-m	mc-mc

Allerdings unterscheiden sich einige Beziehungstypen nur durch die Richtung ihrer Assoziationen, wie z. B. c-1 und 1-c. Damit bleiben 10 tatsächlich voneinander verschiedene Beziehungstypen übrig:

A(R2, R1) \ A(R1, R2)	1	c	m	mc
1	1-1	1-c	1-m	1-mc
c		c-c	c-m	c-mc
m			m-m	m-mc
mc				mc-mc

Übungsaufgabe 3.4

Das Attribut *Adresse* der Entitätsmenge *Kunden* lässt sich z.B. in die (Sub-)Attribute *Straße*, *PLZ* und *Ort* zerlegen. Dabei können die Wertebereiche beispielsweise wie folgt angegeben werden:

Straße: Zeichenkette der Länge 25

PLZ: [1000..9999]

Ort: Zeichenkette der Länge 20

Übungsaufgabe 3.5

Ein möglicher minimaler Identifikationsschlüssel besteht aus der Attributkombination *KundenNr*, *ArtikelNr* und *Datum*.

Würde man lediglich ein einzelnes Attribut als Identifikationsschlüssel verwenden, so würde das z.B. für das Attribut *KundenNr* bedeuten, dass ein Kunde nur einmal in der Entitätsmenge vertreten sein darf. Folglich könnte ein Kunde maximal einen Artikel innerhalb der erfassten Zeitperiode bestellen. Analoges gilt für das Attribut *ArtikelNr*: Es könnte ein Artikel nur einmal bestellt werden.

Im Falle der Attributkombination *KundenNr, ArtikelNr* könnte ein Kunde einen bestimmten Artikel nur einmal bestellen. Erst die Hinzunahme des Attributs *Datum* gestattet es, dass ein Kunde einen Artikel mehrmals bestellt, jedoch nicht am gleichen Tag.

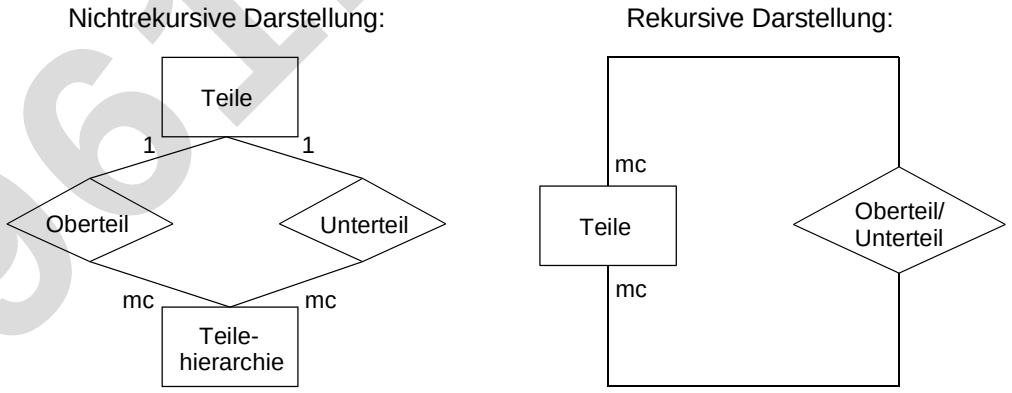
Übungsaufgabe 3.6

Die Beziehungsmenge *Debitoren/offene* Posten gibt an, welche offenen Posten für einen Debitor vorliegen. Für einen Debitor können kein, ein oder mehrere offene Posten existieren. Umgekehrt wird festgehalten, zu welchem Debitor ein offener Posten gehört.

Die Beziehungsmenge *Gebiete/Vertreter* gibt darüber Auskunft, welcher Vertreter in einem Gebiet tätig ist bzw. welches Gebiet ein Vertreter bedient. Ein Vertreter ist immer nur in einem Gebiet tätig und dieses Gebiet wird ausschließlich von diesem Vertreter betreut. Es gibt keinen Vertreter ohne Gebiet, und umgekehrt existiert auch kein Gebiet ohne Vertreter.

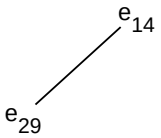
Mit der Beziehungsmenge *Artikel/Lager* wird die Verfügbarkeit von Artikeln in Lägern ausgedrückt. Ein Artikel kann in keinem, einem oder mehreren Lägern verfügbar sein. In einem Lager werden ein oder mehrere Artikel geführt.

Übungsaufgabe 3.7



Übungsaufgabe 3.8

Hierarchie auf Entitätsebene mit minimaler Anzahl von Entitäten:

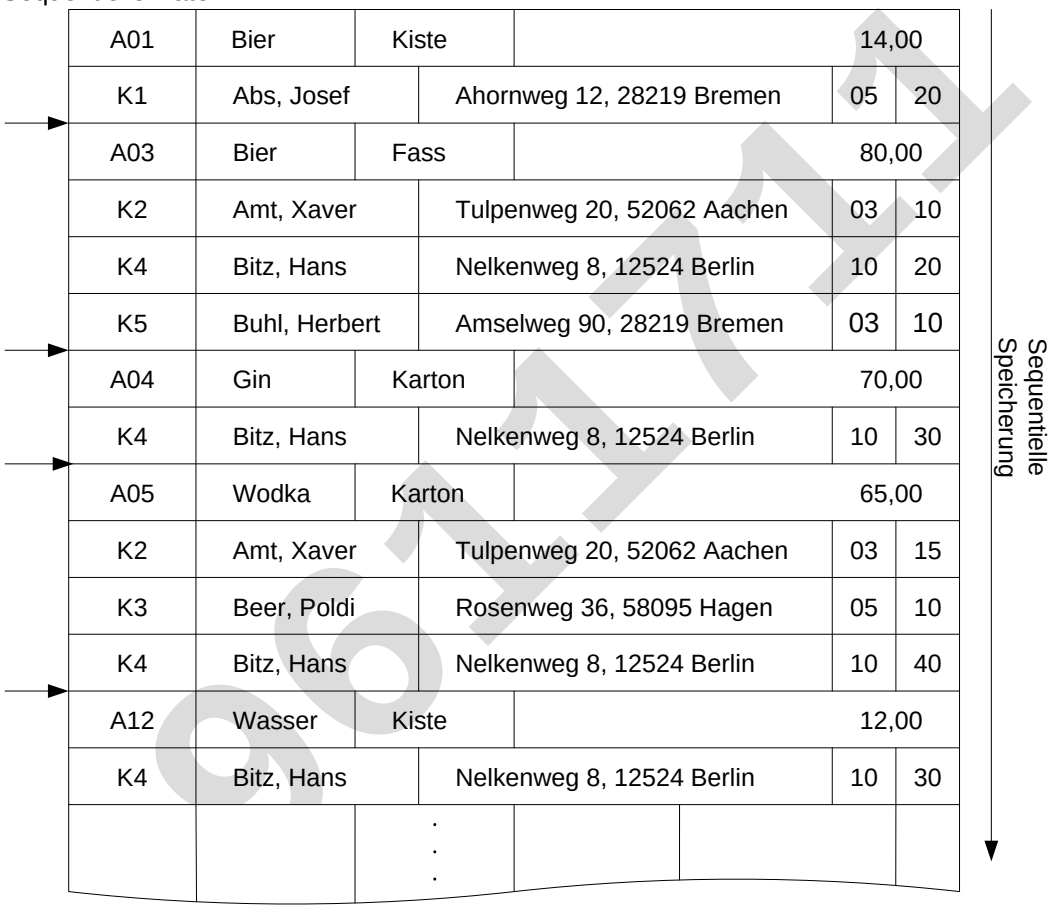


Übungsaufgabe 3.9

Für die Informationsverarbeitungsaufgaben Auftragserfassung, Rechnungsschreibung und Erstellung von Kundenumsatzstatistiken ist ein Datenzugriff über die Entitätsmenge *Kunden* notwendig. Nur die Hierarchie mit Kunden als Wurzel gestattet für einen gegebenen Kunden den unmittelbaren Zugriff zu den zu diesem Kunden gehörigen Artikeln. Das Datenmodell in Abb. 3.17 a) ist folglich das geeignetere.

Übungsaufgabe 3.10

Sequentielle Datei:



Besonders effizient lässt sich mit einer solchen sequentiellen Datei z.B. die Informationsverarbeitungsaufgabe der Erstellung von Umsatzstatistiken abwickeln.

Übungsaufgabe 3.11

Für die fraglichen Beziehungen gilt:

- Einer Entität der Entitätsmenge *E1* müssen eine oder mehrere Entitäten der Entitätsmenge *E3* zugeordnet sein und außerdem können ihr eine oder mehrere Entitäten aus der Entitätsmenge *E4* zugeordnet sein. Dies gilt speziell auch für die Entität e_{11}

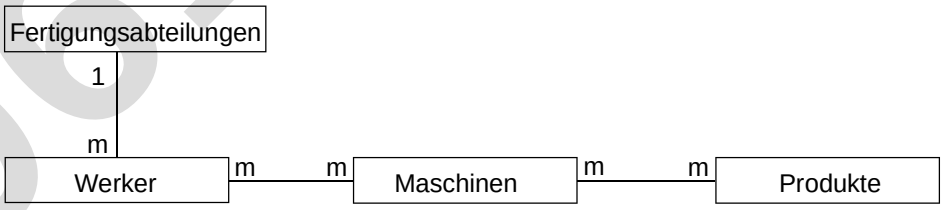
der Menge *E1*, die den beiden Entitäten e_{31},e_{32} der Menge *E3* sowie der Entität e_{41} der Menge *E4* zugeordnet ist.

- Einer Entität der Entitätsmenge *E2* können eine oder mehrere Entitäten der Menge *E4* zugeordnet sein. In Abb. 3.21 a) ist der Entität e_{21} der Menge *E2* eine Entität e_{41} der Entitätsmenge *E4* zugeordnet.
- Einer Entität aus der Entitätsmenge *E3* muss eine Entität der Menge *E1* und kann maximal eine Entität der Menge *E5* zugeordnet sein. Die Entitäten e_{31} und e_{32} der Menge *E3* sind jeweils der Entität e_{11} der Menge *E1* zugeordnet. Die Entität e_{31} ist außerdem auch der Entität e_{51} der Menge *E5* zugeordnet.
- Einer Entität der Menge *E4* muss jeweils eine Entität der Menge *E1* und der Menge *E2* zugeordnet sein. Der Entität e_{41} ist die Entität e_{11} der Menge *E1* und e_{21} der Menge *E2* zugeordnet. Weiterhin kann einer Entität der Menge *E4* maximal eine Entität der Menge *E5* zugeordnet sein. Der Entität e_{41} ist die Entität e_{51} der Menge *E5* zugeordnet.
- Einer Entität der Menge *E5* muss eine Entität der Menge *E3* und eine Entität der Menge *E4* zugeordnet sein. Weitere Beziehungen bestehen nicht. Der Entität e_{51} der Menge *E5* ist genau eine Entität e_{31} der Menge *E3* und eine Entität e_{41} der Menge *E4* zugeordnet.

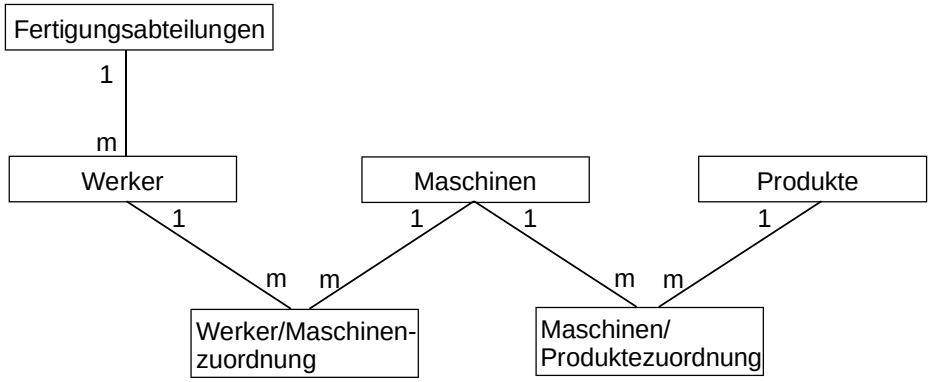
Damit stellt das in Abb. 3.21 a) dargestellte Netzwerk auf Entitätsebene eine zulässige Ausprägung des in Abb. 3.20 gezeigten allgemeinen Netzwerks dar.

Übungsaufgabe 3.12

ER-Diagramm mit netzwerkartigen Beziehungen:



Durch die Einführung von Hilfsentitätsmengen werden die netzwerkartigen Beziehungen zwischen den Entitätsmengen *Werker*, *Maschinen* und *Produkte* eliminiert.



Übungsaufgabe 3.13

Nimmt man in die Entitätsmenge *Bestellungen* zusätzlich die Attribute *KundenNr* und *ArtikelNr* mit auf, so ist es möglich, die Zeigerorganisation auch dann wieder korrekt aufzubauen, wenn die Zeiger beispielsweise aufgrund eines Systemabsturzes nicht ordnungsgemäß gespeichert werden konnten.

Übungsaufgabe 3.14

Das kartesische Produkt der Wertebereiche der gegebenen Attribute umfasst:

889 * 3 * 100 * 100000 = 26670000000 Elemente.

Übungsaufgabe 3.15

Unterstellt sei, dass ein Kunde je Tag höchstens eine Bestellung pro Artikel aufgibt. Dann gilt:

$$Bestellung.(KundenNr, ArtikelNr, Datum) \rightarrow Bestellung.Menge$$

Kann dagegen ein Kunde einen Artikel an einem Tag mehrmals in beliebigen Mengen bestellen, so treten in der Relation *Bestellung* keine funktionalen Abhängigkeiten auf.

Übungsaufgabe 3.16

Es gilt:

$$Bestellung.KundenNr \Rightarrow Bestellung.Name$$
$$Bestellung.ArtikelNr \Rightarrow Bestellung.Bezeichnung$$
$$Bestellung.(KundenNr, ArtikelNr, Datum) \Rightarrow Bestellung.Menge$$

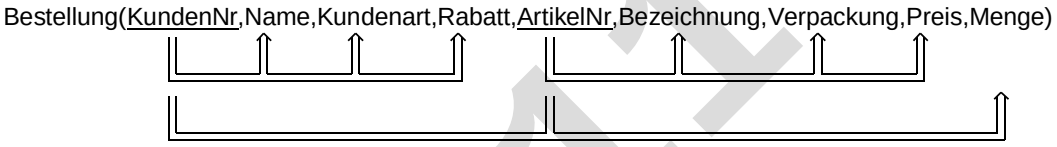
Übungsaufgabe 3.17

Die Struktur "Bestellung" befindet sich nicht in 1. Normalform, da nicht sämtliche Attribute atomar sind. Die Attribute *ArtikelNr, Verpackung, Preis* und *Menge* enthalten wiederholende Gruppen. Folglich muss die Struktur zunächst in eine Relation überführt werden:

Bestellung

KundenNr	Name	Kundenart	Rabatt	ArtikelNr	Bezeichnung	Verpackung	Preis	Menge
23	Just	A	0	228	Mehl	Sack	49,00	3
23	Just	A	0	419	Gries	Kiste	52,00	5
41	Held	C	5	419	Gries	Kiste	52,00	12
36	Mag	B	2	346	Kleie	Sack	36,00	10
36	Mag	B	2	419	Gries	Kiste	52,00	8
76	Hug	A	0	510	Zucker	Sack	64,00	18
19	Bast	A	0	228	Mehl	Sack	49,00	2
.								
.								
.								

Die Relation *Bestellung* befindet sich zumindest in der 1. Normalform. In der Relation treten folgende vollfunktionalen Abhängigkeiten auf:



Da nicht sämtliche Nichtschlüsselattribute vollfunktional vom Schlüssel abhängig sind, befindet sich die Relation nicht in der 2. Normalform. Durch Aufspalten der Relation *Bestellung* in die Relationen *Kunde, Artikel* und *Bestellung* kann der Übergang zur 2. Normalform vollzogen werden:

Kunde

KundenNr	Name	Kundenart	Rabatt
23	Just	A	0
41	Held	C	5
36	Mag	B	2
76	Hug	A	0
19	Bast	A	0
.			
.			
.			

Artikel

ArtikelNr	Bezeichnung	Verpackung	Preis
228	Mehl	Sack	49,00
419	Gries	Kiste	52,00
346	Kleie	Sack	36,00
510	Zucker	Sack	64,00
.			
.			
.			

Order

KundenNr	ArtikelNr	Menge
23	228	3
23	419	5
41	419	12
36	346	10
36	419	8
76	510	18
19	228	2
.		
.		
.		

Die durch die Aufspaltung gebildeten Relationen *Kunde, Artikel* und *Bestellung* befinden sich zumindest in der 2. Normalform. In den Relationen *Artikel* und *Order* treten darüber hinaus keine weiteren unerwünschten Abhängigkeiten auf; sie befinden sich damit bereits in der 3. Normalform. Für die Relation *Kunde* gilt jedoch:

Das Nichtschlüsselattribut *Rabatt* hängt über das Nichtschlüsselattribut *Kundenart* vom Identifikationsschlüssel *KundenNr* ab. Durch Aufspalten der Relation *Kunde* kann diese indirekte Abhängigkeit eliminiert werden:

Kunde

KundenNr	Name	Kundenart
23	Just	A
41	Held	C
36	Mag	B
76	Hug	A
19	Bast	A
.		
.		
.		

Rabattstufe

Kundenart	Rabatt
A	0
B	2
C	5
.	
.	
.	

Insgesamt erhält man damit folgendes Normalisierungsergebnis:

Kunde(KundenNr, Name, Kundenart)
Artikel(ArtikelNr, Bezeichnung, Verpackung, Preis)
Bestellung(KundenNr, ArtikelNr, Menge)
Rabattstufe(Kundenart, Rabatt)

Übungsaufgabe 3.18

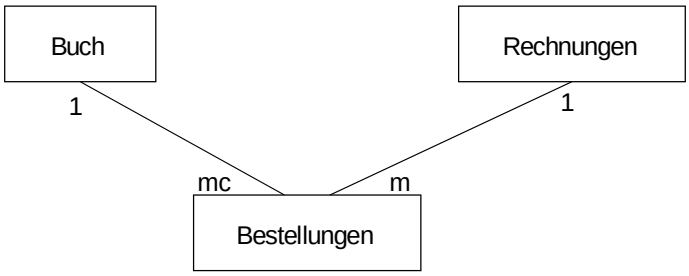
Die Verletzung der Strukturregel 2 kann beispielsweise zu einer Änderungsanomalie führen: Ändert sich die Adresse eines Mitarbeiters, so muss je ein Tupel in den Relationen *Mitarbeiter* und *Vertreter* geändert werden. Es sind also zwei Änderungen erforderlich, obwohl sich nur ein Sachverhalt ändert.

Übungsaufgabe 3.19

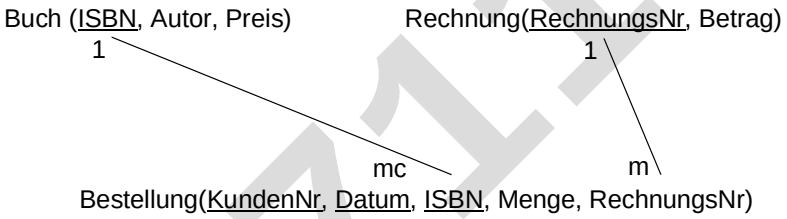
Ein dynamischer Wertebereich ändert sich, weil er durch die konkreten Ausprägungen eines Attributs in einer Relation gegeben ist. So sind z.B. für das Attribut *ArtikelNr* in der Relation *Bestellung* aus dem Beispiel 3.5 nur die Werte zugelassen, welche das Attribut *ArtikelNr* in der Relation *Artikel* auch konkret annimmt. Mit dem Einfügen von Artikeln in die Relation *Artikel* oder dem Löschen von Artikeln aus der Relation *Artikel* ändert sich der Wertebereich im Zeitablauf.

Übungsaufgabe 3.20

a) Darstellung als ER-Diagramm:



Darstellung als Relationendiagramm:



b) Die Attribute *ISBN* und *RechnungsNr* sowie die Attributkombination *KundenNr, Datum, ISBN* sind global. Lokale Attribute sind *Autor, Preis, Menge* und *Betrag.*

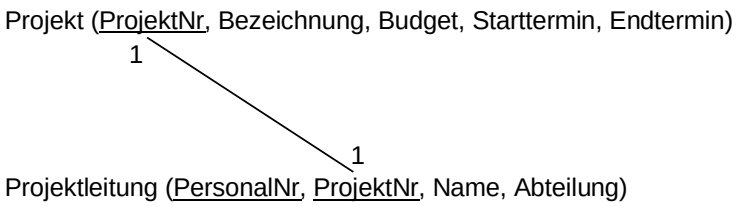
c) Sinnvolle Wertebereiche für zwei lokale Attribute sind z.B.:

Menge: [1..9999]
Betrag: [-99999,99..99999,99]

d) Für das Attribut *RechnungsNr* in der Relation *Rechnung* kann man den Wertebereich statisch auf [0..9999] festlegen. Das Attribut *RechnungsNr* ist in der Relation *Bestellung* Fremdschlüssel. Der dynamische Wertebereich ist auf die konkreten Ausprägungen des Attributs *RechnungsNr* in der Relation *Rechnung* festgelegt. Gemäß dem Beispiel 3.3 ist der dynamische Wertebereich auf die Werte 27, 28, 29, 30, 31, 32, 33, 34 beschränkt.

Übungsaufgabe 3.21

a) Relationenmodell:



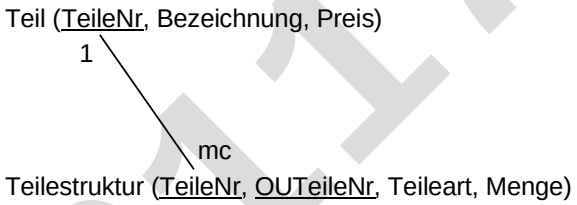
b) Bei der hier vorgenommenen Modellierung wird unterstellt, dass die Projektleitung aus einem Projektleiter besteht. Während der Dauer eines Projekts kann sich die Person des Projektleiters ändern. Es erscheint daher sinnvoll, die Attributkombination (*PersonalNr*, *ProjektNr*) als Identifikationsschlüssel der Relation *Projektleitung* vorzusehen, und die *ProjektNr* als Fremdschlüssel aus der Relation *Projekt* zu importieren. Damit ergibt sich *Projekt* als übergeordnete und *Projektleitung* als untergeordnete Relation.

Übungsaufgabe 3.22

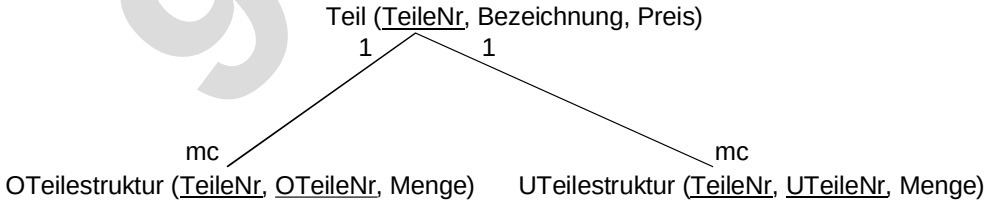
Von inhaltlicher Bedeutung für die Hilfsrelation *Vormundschaft* sind z.B. Attribute, welche die Übertragung der Vormundschaft und das Datum der Übertragung beschreiben.

Übungsaufgabe 3.23

Fasst man, wie in der Relation *Teilestruktur* geschehen, Ober- und Unterteile-Tupel in einer Relation zusammen, so trifft folgendes Relationendiagramm zu:



Bringt man jedoch, anders als dies in Abb. 3.38 geschehen ist, die Ober- und Unterteile-Tupel je in separate Relationen ein, so ergibt sich folgendes Relationendiagramm:



Da Ober- und Unterteilbeziehungen nun separat erfasst werden, erübrigt sich das unterscheidende Attribut *Teileart*. Statt einer Relation Teilestruktur erhält man dann die beiden folgenden Relationen:

OTeilestruktur

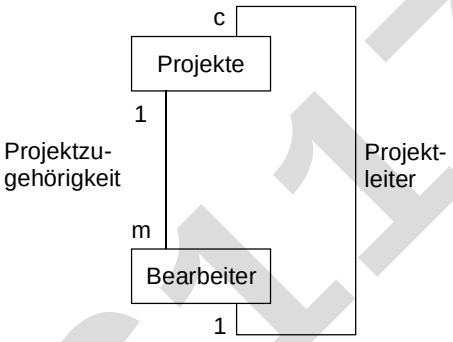
TeileNr	OTeileNr	Menge
31	11	3
32	21	3
33	21	6
33	22	8
33	12	4
34	22	4
35	22	7
35	13	2
21	11	2
21	12	1
22	12	3
22	13	1

UTeilestruktur

TeileNr	UTeileNr	Menge
11	21	2
11	31	3
12	21	1
12	22	3
12	33	4
13	22	1
13	35	2
21	32	3
21	33	6
22	33	8
22	34	4
22	35	7

Übungsaufgabe 3.24

Gegebenes Datenmodell:



Beschreibung der Entitätsmengen mit den beiden folgenden Relationen:

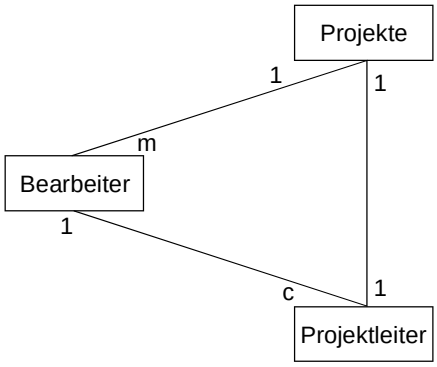
Projekt

ProjektNr	Bezeichnung
1	Rechnungsschreibung
2	Marktanalyse
3	Vertretereinsatz
4	Fuhrpark
.	
.	
.	

Bearbeiter

ProjektNr	BearbeiterNr	Name	Leiter
1	11	Abs	1
1	12	Zorn	
2	14	Beer	2
3	17	Grimm	
4	15	Raab	4
3	16	Zapf	3
	.		
	.		
	.		

Durch die Einführung einer Hilfsrelation *Projektleiter* wird das gegebene rekursive Modell in eine nach der vierten Strukturregel erlaubten Form transformiert:



Projekt

Projekt Nr	Bezeichnung
1	Rechnungsschreibung
2	Marktanalyse
3	Vertretereinsatz
4	Fuhrpark
.	
.	
.	

Bearbeiter

Projekt Nr	Bearbeiter Nr	Name
1	11	Abs
1	12	Zorn
2	14	Beer
3	17	Grimm
4	15	Raab
3	16	Zapf
	.	
	.	
	.	

Projektleiter

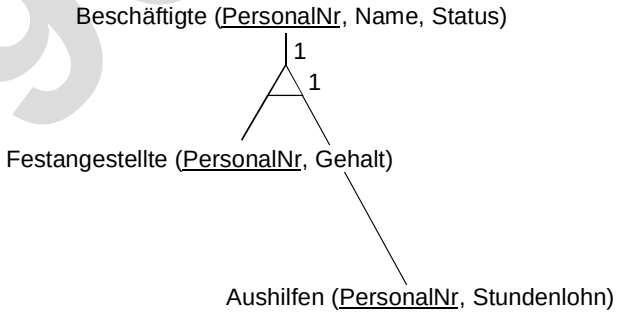
Projekt Nr	Bearbeiter Nr
1	11
2	14
3	16
4	15
.	
.	
.	

Man beachte:

Die Relation *Bearbeiter* ist nicht vollständig normalisiert, da das Attribut *Name* nur von einem Teil des Identifikationsschlüssels abhängt. Dennoch ist die Strukturregel 4 erfüllt.

Übungsaufgabe 3.25

a) Relationendiagramm:



In der generalisierten Relation *Beschäftigte* dient das Attribut *Status* der Zuordnung einer Entität zu den spezialisierten Untermengen. Das Attribut *Status* kann z.B. die Werte "F" für Festangestellte und "A" für Aushilfen annehmen und so die erforderliche Diskriminierung leisten.

b) Darstellung der Relationen:

Beschäftigte

PersonalNr	Name	Status
10010	Abs	F
10012	beer	F
10017	Doll	A
10018	Dürr	F
10023	Ebert	A
.	.	.
.	.	.
.	.	.

Festangestellte

PersonalNr	Gehalt
10010	90000
10012	120000
10018	60000
.	.
.	.
.	.

Aushilfen

PersonalNr	Stundenlohn
10017	15
10023	19
.	.
.	.
.	.

In der generalisierten Relation *Beschäftigte* lässt sich die Rolle des diskriminierenden Attributes *Status* nun deutlich an den Attributwerten ablesen.

Übungsaufgabe 3.26

Die Beziehung *Stkliste* beschreibt eine Teilestruktur mit den drei Ebenen Fertigprodukte (komplexe Geräte), Halbfertigprodukte (einfache Geräte) und Einzelteile. Allgemein gilt daher für ein Teil:

(1) Es kann in keinem oder in einem übergeordneten Teil oder in mehreren übergeordneten Teilen enthalten sein.

(2) Es kann kein oder ein untergeordnetes Teil oder mehrere untergeordnete Teile enthalten.

Die Eigenschaften (1) und (2), die man sich auch an der allgemeinen in Übungsaufgabe 3.7 veranschaulichten Teilestruktur verdeutlichen kann, begründen eine mc-mc-Beziehung zwischen Teilen. Dies ist nichts anderes als eine rekursive mc-mc-Beziehung auf der Entitätsmenge *Artikel*.

Für die Beziehung *Zeitliste* gelten vollkommen analoge Überlegungen wie für die Beziehung *Stkliste*. Zu beachten sind jedoch zwei Unterschiede:

- Bei der Beziehung *Zeitliste* werden Über-/Unterordnungen von Teilen nur für die Ebenen Fertig- und Halbfertigprodukte betrachtet.
- Inhaltlich stellt die Beziehung *Zeitliste* auf bewertete Fertigungszeiten bzw. Arbeitswerte ab; während bei der Beziehung *Stkliste* über ein später noch einzuführendes Attribut z.B. festhalten wird, wie viele Halbfertigprodukte einer bestimmten Art in ein bestimmtes Fertigprodukt eingehen, interessiert es bei der Beziehung *Zeitliste* darüber hinaus, mit welchem Arbeitswert das Halbfertigprodukt zu dem Arbeitswert des Endprodukts beiträgt.

Unabhängig von diesen Unterschieden gelten jedoch auch für die Beziehung *Zeitliste* die oben formulierten Eigenschaften (1) und (2). *Zeitliste* ist somit eine rekursive mc-mc-Beziehung auf der Entitätsmenge *Artikel*.

Übungsaufgabe 3.27

Für die Beziehung zwischen *FertAuftragMat* und *KundenAuftrag* gilt:

(1) Ein Kundenauftrag bewirkt das Auflegen eines oder mehrere Fertigungsaufträge und zwar abhängig davon, ob ein Artikel (Fertigprodukt) oder mehrere Artikel (Fertig-/Halbfertigprodukte) zu fertigen ist oder sind.

(2) Ein Fertigungsauftrag geht auf einen Kundenauftrag zurück oder auf mehrere Kundenaufträge, wenn diese zwecks rationellerer Fertigung zusammengefasst werden können.

Aus den Eigenschaften (1) und (2) resultiert eine m-m-Beziehung zwischen den Entitätsmengen *FertAuftragMat* und *KundenAuftrag*.

Analoge Überlegungen gelten für die Beziehung zwischen *FertAuftragZeit* und *KundenAuftrag*. Zu beachten sind die bereits in der Lösung zur Übungsaufgabe 3.26 angegebenen Unterschiede.

Diese Seite bleibt aus technischen Gründen frei.

Literaturverzeichnis

a) Im Lehrtext zitierte Literatur

ABRIAL J.R.: Data Semantics. In: Data Base Management. North Holland Verlag, Amsterdam 1974.

CHEN, P.P.: Entity-Relationship Approach to Information Modelling and Analysis. Proceedings of the 2nd International Conference on Entity-Relationship Approach. Washington D.C., Oct. 12-14, 1981, North-Holland Verlag, Amsterdam 1981.

CODD, E.F.: The Relational Model for Database Management: Version 2. Addison-Wesley Verlag, Reading 1990.

SENKO M.E. u. a.: Conceptual Schemas, Data Structures and Accessing in Data Base Systems. In: IBM Systems Journal (1973) 1, S. 30-93.

THURNHERR B.: Konzepte und Sprachen für den Entwurf konsistenter Datenbanken. Diss. ETH Nr. 6526, Zürich 1980.

THURNHERR, B. und ZEHNDER, C.A.: Global Data Base Aspects, Consequences for the Relational Model and a Conceptual Schema Language. ETH Zürich, Institut für Informatik, Bericht No. 30, Zürich 1979.

ZEHNDER, C.A.: Informationssysteme und Datenbanken. 3. Auflage. Teubner Verlag, Stuttgart 1985.

b) Weitere Literaturangaben

DATE, C.J.: An Introduction to Database Systems, Vol. I. 5. Auflage. Addison-Wesley Verlag, Bonn 1986.

DAVIS, C.G. und JAJODIA, S.: Entity-Relationship Approach to Software Engineering. Proceedings of the 3rd International Conference on Entity-Relationship Approach. North-Holland Verlag, Amsterdam 1983.

HOGAN, R.: A Practical Guide to Data Base Design. Prentice-Hall, Englewood Cliffs 1990.

OLLE, T.W.: Das CODASYL-Datenbankmodell. Springer Verlag, Berlin 1981.

OLLE, T.W.: The CODASYL Approach to Data Base Management. Wiley Verlag, Chichester 1978.

SCHÄFER, G.: Datenstrukturen und Datenbanken. Vieweg Verlag, Wiesbaden 1989.

WIBORNY, W.: Datenmodellierung, CASE-Management. Addison-Wesley Verlag, Bonn 1991.

WIEDERHOLD, G.: Database Design. McGraw-Hill Verlag, Singapur 1983.

WIEDERHOLD, G.: Dateiorganisation in Datenbanken. McGraw-Hill Verlag, Hamburg 1989.

Index

Diese Seite bleibt aus technischen Gründen frei.