

## 7. Datenbankdesign: Entity Relationship Model

7.	Datenbankdesign: Entity Relationship Model .....	1
7.1	Datenmodelle .....	2
7.2	Logisches Datenbankdesign.....	4
7.3	Theoretische Grundlagen der Modellbildung .....	6
7.3.1	Modelle .....	6
7.3.2	Wahrnehmung .....	7
7.3.3	Konstruktion abstrakter Objekte .....	9
7.3.4	Beschränkung auf das Datenmodell .....	12
7.3.5	Datenmodelle und „Beziehungen“ zwischen „Objekten“ .....	13
7.4	Datenbankdesign.....	13
7.5	Das Entity Relationship Model.....	15
7.5.1	Konstruktionselemente .....	15
7.5.2	1:-N Assoziationen im Entity Relationship Modell .....	16
7.5.3	1:-N Assoziationen im Relationenmodell.....	17
7.5.4	M:N Assoziationen.....	17
7.5.5	Darstellung von M:N-Assoziationen im Relationenmodell .....	18
7.5.6	Die Chen-Notation .....	19
7.5.7	Die <min,max> Notation.....	21
7.5.8	Existenzabhängige Entity-Typen .....	22
7.5.9	Modellierungsmethoden und Modellierungstools .....	26
7.6	Aufgaben.....	28

## 7.1 Datenmodelle

*Der Begriff "Modell" ist schon so ausgelutscht, dass man ihn eigentlich nicht mehr gebrauchen sollte.*

Zunächst ist der Begriff "Datenmodell" zu erläutern, da der ihm zugrunde liegende Begriff "Modell" heute in einer kaum überschaubaren Anzahl von Bedeutungen gebraucht wird. Eine Übersicht über die verschiedenen Bedeutungen gibt z.B. Zschocke anhand einer Literaturstichprobe in [ZS 95]. Dort wird der Versuch einer Klassifizierung von Modellen unternommen. Es wird aber auch beklagt, dass die Anwendung von Modellen häufig in einem unausgewogenen Verhältnis zu unserem gesicherten Wissen über Modelle und deren Konstruktion steht:

*"Während sich die Anwendungsmöglichkeiten des Instruments 'Modell' innerhalb einzelner Wissenschaften bisher großer Aufmerksamkeit erfreuen konnten und in einer unüberschaubaren Anzahl wissenschaftlicher Arbeiten Vorschläge und -meist in der Sprache der Mathematik verfasste - Theorien zur Modellanalyse und -umformung ausgebreitet wurden, konnte der Vorgang der Modellbildung, d.h. die Konstruktion des anzuwendenden Werkzeugs, bisher nur ein vergleichsweise geringes wissenschaftliches Interesse auf sich ziehen. Modelle waren oft schnell bei der Hand."<sup>1</sup>*



Datenbanksysteme stellen Funktionen zur reibungslosen Verwaltung umfangreicher und komplex strukturierter Datenmengen zur Verfügung. Der Zweck der Speicherung dieser Datenmengen ist die Repräsentation von Zuständen und Ereignissen, die in der von uns wahrgenommenen Wirklichkeit (in der "Realität") vorkommen und die wir, z. B. zur Beantwortung der Frage, wann welches Ereignis stattgefunden hat, bei Bedarf "rekonstruieren" möchten. "Einfache Daten", z. B. einzelne Zahlenwerte oder auch einfach strukturierte Datenmengen (etwa Arrays), genügen natürlich häufig nicht zur angemessenen Repräsentation der zu beschreibenden Systeme. Man führt zur Beschreibung dann abstrakte Konstruktionen, wie z. B. *Datenobjekte* und die zwischen diesen bestehenden *Assoziationen*<sup>2</sup> (Beziehungen) ein.

Als Ausdrucksmittel zur Beschreibung von Daten dienen graphische Darstellungen oder Sprachen, wie z. B. Programmiersprachen oder spezialisierte *Datenbeschreibungssprachen*<sup>3</sup>.

Die mit diesen Hilfsmitteln erzeugten Darstellungen der Objekte und Beziehungen nennt man dann **Datenmodelle**.

*Wenn wir "etwas" in einer Datenbank speichern wollen, müssen wir uns zuerst über das "etwas" etwas klar werden...*

Wir verwenden den Begriff *Datenmodellierung* hier, wenn es darum geht, eine gegebene "Miniwelt" mit dem Ziel zu beschreiben, eine geeignete Repräsentation dieser Miniwelt im Rechner/Speicher vorzubereiten.

Allerdings soll unsere Beschreibung zunächst möglichst auf einer abstrakten Ebene bleiben und von technischen Realisierungsdetails freigehalten werden. Deshalb trennen wir bewusst mehrere Ebenen der Modellierung. Es geht um die Repräsen-

<sup>1</sup> [ZS 95], S. 2

<sup>2</sup> engl.: Relationship, wohl zu unterscheiden von "Relation"

<sup>3</sup> engl.: Data Definition Language oder Data Description Language (DDL)

tation der Miniwelt auf einer logischen Ebene, deren Ergebnis zunächst eine Beschreibung der Komponenten der Miniwelt und der zwischen diesen bestehenden logischen Zusammenhänge sein wird. Mit anderen Worten, es geht darum, die Miniwelt zu "verstehen" bzw. eine verständliche Beschreibung zu verfertigen.

Diese Aufgabe nennt man semantische Datenmodellierung.

*Beachten Sie, dass  
MODELL im Begriff  
"Entity-Relationship-  
Modell" in einer ande-  
ren Bedeutung ver-  
wendet wird, nämlich  
im Sinne von "Kon-  
zept".*

Eines der bekanntesten Konzepte für die semantische Datenmodellierung ist das **"Entity-Relationship- Modell"** (ERM) von Chen [CH 76] bzw. dessen Weiterentwicklungen.

Alle heute verfügbaren Konzepte haben eine Gemeinsamkeit: *"...sie sind in ihrer Grundkonzeption alle ähnlich, wenn sie auch durch möglichst individuelle Begriffswahl Unterschiedlichkeit zu betonen versuchen. Generell wird nämlich (implizit) der Systembegriff der allgemeinen Systemtheorie als Gerüst zugrunde gelegt: jede Miniwelt wird beschrieben als eine Menge von Gegenständen (unserer Anschauung und unseres Denkens), den Systemelementen, zwischen denen wohldefinierte (System-) Beziehungen bestehen."*<sup>4</sup>

Datenmodellierung hat, wie der Name schon sagt, die Daten (lat. "das Gegebene") eines Systems im Fokus, also eher die statischen Aspekte, z.B. Objekte und Strukturen, die über einen längeren Zeitraum Bestand haben.

Es bestehen aber natürlich enge Verbindungen zum Thema Modellierung im allgemeinen und insbesondere zu den für die Modellbildung im Zusammenhang mit der Programmierung eingesetzten objektorientierten Analyse- und Entwurfsmethoden (OOA, OOD u.a.), in denen Konzepte der objektorientierten Programmierung um Konzepte der semantischen Datenmodellierung ergänzt wurden. Diese Methoden gehen über das Entity-Relationship-Modell hinaus, u.a. indem sie auch die dynamischen Aspekte der Systeme beschreiben. Sie werden im Rahmen von Vorlesungen über Systemplanung bzw. Softwareentwicklung behandelt.

Wir wollen uns einige Grundbegriffe zum Thema Modellierung, soweit sie für die spezielle Aufgabe der Datenmodellierung von Bedeutung sind, etwas genauer ansehen.

Der Grundbegriff überhaupt in diesem Zusammenhang ist der Begriff der Abstraktion. Er wird in einem speziellen Kapitel dieser Vorlesung ausführlich erläutert.

Die folgenden Definitionen sind entnommen aus [KO 96, S. 30]:

*Na ja, irgendwelche  
Modelle kennen Sie  
bestimmt. Überlegen  
Sie sich Beispiele  
dazu!*

*"Ein **Modell** (model) ist eine Zusammenfassung von Merkmalen eines realen (oder empirischen) künstlichen Systems sowie eine Festlegung der Beziehungen zwischen diesen Merkmalen; da ein Modell niemals alle Merkmale eines Systems umfassen kann, ist ein Modell eine Abstraktion eines realen Systems."*

<sup>4</sup> [LO 87], S. 498

*"Ein **Merkmal** ist eine Komponente eines Systems, die verschiedene Ausprägungen annehmen kann. Im Modell werden solche Komponenten auf Einheiten abgebildet, die wir als **Attribute** (attribute) bezeichnen."*

*"Eine **Merkmalsausprägung** ist eine von mehreren möglichen Ausprägungen eines Merkmals, häufig ein Zahlenwert oder ein anderer **Wert** (value) aus einer endlichen oder unendlichen Menge."*

*"Eine **Merkmalsveränderung** ist eine Veränderung einer Ausprägung eines Merkmals; im Modell nennt man eine Funktionalität, die eine Merkmalsveränderung vornimmt, häufig eine Methode (method)."*

Realität		Abstraktion
System	$\leftrightarrow$	Modell
Merkmal	$\leftrightarrow$	Attribut
Merkmalsausprägung	$\leftrightarrow$	Wert
Funktionalität	$\leftrightarrow$	Methode

*Wir haben im Teil I der Vorlesung den Begriff "Logische Ebene der Datenorganisation" kennengelernt.*

*Denken Sie einmal darüber nach, welche Art von Datenbanksein man wohl als "nicht-logisch" bezeichnen könnte.*

*Das Entity-Relationship-Modell ist die "Mutter aller semantischen Datenmodelle".*

## 7.2 Logisches Datenbankdesign

Am Beginn der Entwicklung eines neuen Informationssystems stehen eine Reihe von Designaufgaben. Letztendlich geht es immer darum, ein Modell der sog. "Realität" bzw. eines Ausschnitts derselben zu entwerfen und dieses Modell mit geeigneten Hilfsmitteln (z.B. Programmiersprachen, Datenbanken usw.) in ein passendes Informationssystem umzusetzen.

Ein Teil des Designprozesses betrifft die Daten, genauer die permanent zu speichernden Daten, die Zustände der abgebildeten "realweltlichen" Systeme repräsentieren und dem Anwender bzw. anderen Systemen Auskunft über diese Systeme geben sollen. Diesen Teilbereich des Designprozesses nennen wir "Datendesign" oder auch "Datenbankdesign". In der Vorlesung Datenbanksysteme behandeln wir dieses Thema ausführlich. Wir dürfen jedoch nicht übersehen, dass es sich nur um einen Teil des gesamten Entwurfsprozesses - nämlich eben den Datenaspekt - für ein Informationssystem handelt, der um andere Aspekte (s.u.) zu ergänzen ist.

Wir verwenden für das Datenbankdesign u.a. eine vereinfachte Form des **Entity-Relationship-Models (ERM)** von Chen [CH 76]. Inzwischen gibt es eine Reihe von Weiterentwicklungen dieses Modells, die nach wie vor die Grundelemente des ERM verwenden, jedoch weit darüber hinausgehen. Aktuell wird im Zusammenhang mit der Verbreitung "objektorientierter" Programmiersprachen auch für den Entwurfsprozess eine "objektorientierte" Vorgehensweise angestrebt. In [BA 96] findet sich eine Übersicht über die Konzepte der objektorientierten Systemanalyse. Dort werden drei Teilmodelle unterschieden, nämlich:

- das Basismodell
- das statische Modell
- das dynamische Modell

Das **Basismodell** umfasst die Konzepte

- Objekt
- Klasse
- Attribut
- Operation
- Polymorphismus

Das **statische Modell** beschreibt die Systemstruktur, die Objekte/Klassen und die zwischen ihnen existierenden Beziehungen zusammen mit den Restriktionen, die erfüllt sein müssen:

- Vererbung
- Assoziation
- Aggregation
- Subsystem

Das **dynamische Modell** beschreibt die Verhaltensweise des Systems. Es besteht aus den Konzepten:

- Botschaft und Interaktionsdiagramm
- Spezifikation der Operation
- Objekt-Lebenszyklus (Zustandsautomat)<sup>5</sup>

*Wir werden nur die  
grau unterlegten Teile  
behandeln.*

Obwohl diese Konzepte schon über die Möglichkeiten des ERM hinausgehen, werden auch damit bei weitem nicht alle Aspekte eines Systems erfaßt. So wird es z.B. für die Gestaltung der Benutzungsschnittstelle in Form einer graphischen Oberfläche (GUI) oder die Gestaltung der Architektur eines mehrschichtigen, verteilten Systems weiterer Beschreibungshilfsmittel bedürfen.

*Konzepte und Werk-  
zeuge des objektorien-  
tierten Entwurfs wer-  
den Sie in anderen  
Vorlesungen kennen  
lernen.*

Zur Unterstützung des Designprozesses (allgemeines Design von Programmen und Datenstrukturen) gibt es eine Vielzahl von Methoden und Werkzeugen. Heute existieren mehr als 50 objektorientierte Methoden, und die Experten streiten sich, welche die beste ist. In 1995 haben sich einige Methodenexperten (Booch, Rumbaugh, Jacobson) zusammengetan und den Versuch gemacht, ihre Konzepte in einer einheitlichen Methode zusammenzuführen. Sie nannten diese neue Methode "Unified Method". Im Laufe ihrer Arbeit haben Sie jedoch ihre ursprüngli-

---

<sup>5</sup> zitiert aus [BA 96, Seite 29/30]

chen Ankündigungen selbst etwas zurückgenommen und 1996 an Stelle der "Unified Method" nur eine "Unified Modeling Language" (UML) veröffentlicht.<sup>6</sup>

## 7.3 Theoretische Grundlagen der Modellbildung

### 7.3.1 Modelle

Die für eine betriebliche Anwendung in einem Datenbanksystem gespeicherten Daten dienen zur Beschreibung der für das Unternehmen oder einen Teilbereich des Unternehmens relevanten Zustände und Prozesse.

Man sagt auch, die "Datenbank ist ein Modell des Unternehmens" oder eines Unternehmensteiles. Die Datenbankmanagementsysteme (DBMSe), also die Softwarepakete wie ORACLE, DB/2 und andere stellen uns u.a. die Werkzeuge für die Umsetzung der Modellierung in eine operative Datenbank zur Verfügung.

Im Wesentlichen handelt es sich dabei um Softwarekonstrukte zur "Erzeugung" von Objekten und Strukturen sowie um Sprachkonstrukte zur Manipulation dieser Komponenten. Wir befassen uns im Rahmen anderer Teile der Datenbankvorlesung ausführlich damit.

Ein wichtiger Schritt im Zusammenhang mit dem Entwurf von Unternehmensdatenmodellen bzw. der "Modellierung" überhaupt ist jedoch zunächst die "Gewinnung" der Objekte: Dieser Schritt muss *vor* der Implementierung geleistet werden.

Da die Qualität des gesamten späteren Entwurfsprozesses wesentlich davon abhängt, welche "Sicht der Dinge" am Anfang gewählt wird, widmen wir diesem Problem etwas mehr Aufmerksamkeit als es im aktuellen Informatik- und Wirtschaftsinformatikbetrieb üblich ist.

Die meisten Experten sind der Meinung, dass die Objekte "auf der Straße liegen" oder "vom Himmel fallen". Das heißt, sie sehen sich einen Sachverhalt (z.B. einen Ablauf in einem Unternehmen) an und kreieren die Objekte und Datenstrukturen dann "aus dem Bauch". Manchmal ist dieses Verfahren auch erfolgreich, insbesondere dann, wenn die zu modellierenden Gegenstände eher konkreter Art sind, wie z.B. Maschinen, Autos, Tennisschläger, Fußbälle usw. In der Wirtschaftsinformatik begegnen wir jedoch nicht selten auch "abstrakten" Gegenständen, wie z.B. Organisationsstrukturen, Verträgen, Konten oder Algorithmen.

Die Erfahrung zeigt hier, dass im Verlaufe eines Designprozesses die Objekte sich ständig wandeln. Nicht selten stehen am Ende des Entwurfs einige Objekttypen, die am Anfang niemand "so" gesehen hat.

*Beobachten Sie einmal, wie in der Wirtschaftsinformatik von "Experten" "Modelle gebaut" werden..*

*...und fragen Sie die Experten einmal ganz dumm, wie sie zu ihren Objekten kommen.*

---

<sup>6</sup> Eine Beurteilung der gegenwärtigen und zukünftigen Bedeutung der UML findet sich in [SO 97]. Die Originaldokumente zur UML finden Sie unter <http://www.rational.com>. Informationen zu einer anderen mit UML konkurrierenden Methode, genannt OML, finden Sie unter <http://www.csse.swin.edu.au/cotar/OPEN/OPEN.html>. Weitere Literatur zum Thema: [BO 97, FI 97]. Sowohl UML als auch OML wurden von ihren Schöpfern bei der Object Management Group, einem für die Standardisierung von Objekttechniken maßgeblichen Gremium, zur Anerkennung eingereicht.

Es lohnt sich also, sich mit der Problematik der "Erschaffung" von Objekten etwas intensiver zu befassen.<sup>7</sup>

Wir wollen uns hier folgenden Fragen zuwenden:

- Wie nehmen wir Objekte wahr?
- Warum bilden wir gerade diese Objekte und nicht andere?<sup>8</sup>
- Wie gehen wir mit abstrakten Objekten um?
- Wie kann man die Bildung von Objekten methodisch unterstützen?

Zur Bildung bzw. Beschreibung von Objekten können wir drei Verfahren einsetzen:

- Eigenprädikation
- Abstraktion
- Mereologie

Dies wird im Folgenden näher erläutert.

### 7.3.2 Wahrnehmung

*Ich sehe was, was du  
nicht siehst.*

In unserer täglichen "Realität" nehmen wir ständig äußere Objekte wahr, ohne dass wir dazu bewusste Anstrengungen unternehmen müssen. Wir "erkennen" Personen, Gegenstände, Gebäude, Landschaften usw. in unserer Umgebung. So ist es nicht verwunderlich, dass die wohl älteste Wahrnehmungstheorie, die sog. **Abbildtheorie**, annimmt, dass wir die äußere "Realität" mit Hilfe unserer Sinnesorgane als eine Art Photographie zu einem inneren Abbild verarbeiten. Es wird eine objektive, äußere Realität vorausgesetzt, der Prozess der Wahrnehmung ist weitgehend passiv, und das Ergebnis kann im Prinzip in seiner Qualität beliebig gesteigert werden, "wenn man nur genau genug hinsieht". Das Bild wird dann in irgendeiner Form in unseren grauen Zellen gespeichert und bei Bedarf wieder abgerufen, indem wir uns an etwas erinnern. Diese Theorie ist heute vermutlich noch die vorherrschende Grundlage für viele wissenschaftliche Arbeiten, auch in der Wirtschaftsinformatik. In jedem Fall kann man feststellen, dass wir in unserem Alltag eine der Abbildtheorie entsprechend einfache Vorstellung intuitiv anwenden.

Ein zweiter Blick auf das Problem der Wahrnehmung bringt jedoch eine Reihe von Hinweisen, *dass die Beziehung zwischen Wahrnehmung und Wirklichkeit wesentlich komplexer ist, als wir bisher annahmen* [LE 95]. Wir werden dies im Folgenden anhand einiger Beispiele näher betrachten. Die Infragestellung der Abbildtheorie führt auch zu Neuorientierungen und Verwerfungen in der Wissenschaftstheorie, die in der Vorlesung "Datenbanksysteme" aus Zeitgründen auch

---

<sup>7</sup> Meine Ausführungen dazu beziehen sich vor allem auf die Arbeiten von Prof. Hartmut Wedekind von der Universität Erlangen-Nürnberg. Er stellt hinsichtlich der Behandlung des Themas "Modellierung und Abstraktion" eine rühmliche Ausnahme in der Fachwelt dar. Nicht unerwähnt bleiben soll an dieser Stelle auch Paul Lorenzen, der wesentliche Erkenntnisse zur Klärung des Abstraktionsbegriffs bereitgestellt hat.

<sup>8</sup> Die etwas weitergehende Frage des großen Philosophen Heidegger, "Warum ist überhaupt Seiendes und nicht vielmehr Nichts?" kann ich leider nicht beantworten....

nicht annäherungsweise beschrieben oder diskutiert werden können. Es muss hier genügen, zwei wesentliche Dinge festzustellen, nämlich:

1. Die Abbildtheorie wird durch Forschungsergebnisse der wahrnehmungspsychologischen und -physiologischen Forschung, also auch durch naturwissenschaftliche Disziplinen und Methoden, die gemeinhin dem Ideal der Wissenschaftlichkeit entsprechen, in Zweifel gezogen. Andere Disziplinen, so auch die Philosophie (u.a. Kant), die Soziologie, (u.a. Luhmann) und nicht zuletzt einige Informatik-Außenseiter (Wedekind, Floyd) leisten ebenfalls Beiträge zu einer vermutlich angemesseneren Sichtweise unseres Problems.<sup>9</sup>

2. Gerade die Praxis der Wirtschaftsinformatik, und hier insbesondere jede Aktivität, die mit dem Entwurf von Daten- und Programmstrukturen befasst ist, hat es mit hochstilisierten, abstrakten Objekten zu tun, deren Gestaltung mehr Freiheitsgrade erlaubt als die Gestaltung physischer "realer" Objekte. Zwar müssen sich auch die logischen Konstrukte der Informatik letztendlich in ihrer Implementierung "bewähren", doch hier führen sehr viele Wege nach Rom.

Die durch die Konstrukte der Wirtschaftsinformatik darzustellenden "Objekte der Realität" selbst sind bereits häufig abstrakte Objekte, man denke z.B. an die Objekte und Strukturen eines betrieblichen Rechnungswesens.<sup>10</sup> Nicht selten sind die darzustellenden Systeme schlecht oder nur "in den Köpfen der Anwender" dokumentiert.

Die Feststellung des Anwenders gegenüber dem Entwickler nach einigen Jahren Entwicklungszeit und einigen Millionen Entwicklungskosten, dass er "sich das System aber so nicht vorgestellt" hat, ist leider nicht die Ausnahme, sondern die Regel. Dies wird natürlich von keinem Projektverantwortlichen zugegeben, und es gibt viele Möglichkeiten, gescheiterte Projekte in Erfolgsgeschichten umzudefinieren...

Uns sollten diese Erfahrungen jedoch zu denken geben. Es ist ganz einfach, bereits vor Beginn eines Projekts so gravierende Fehler zu machen, dass dieses garantiert ein Misserfolg wird. Der erste Fehler ist, zu glauben, dass man mit einem ausführlichen "Pflichtenheft" oder einer guten Anforderungsspezifikation beschreiben kann, was der Anwender *wirklich* will.

Als Konsequenz ergibt sich für das Datenbankdesign zumindest die Notwendigkeit der Infragestellung des naiven "Abbildungsverfahrens" und die Einführung eines expliziten, bewusst ablaufenden (Re-)Konstruktionsprozesses für die Objekte des fraglichen Gegenstandsbereiches. Ein interessanter Ansatz dazu ist das sog. "Konstanzer Sprachkritik-Programm"<sup>11</sup>. Ich zitiere dazu:

*"Im Bereich der Softwareentwicklung bedeutet sprachkritisch z.B., dass die Fachbegriffe eines Aufgabenbereichs nicht einfach gesammelt und in den Systementwurf übernommen werden, sondern, dass sie zunächst systematisch geklärt, unternehmensweit abgestimmt und in eine gemeinsam akzeptierte Verwendungsweise überführt werden. ... Von solcherart rekonstruierten Begriffen wird gefordert, dass sie nicht nur dem Verständnis einzelner entsprechen, sondern dass sie kon-*

<sup>9</sup> Eine anspruchsvolle Darstellung zu den physiologischen Grundlagen der Wahrnehmung findet sich in [RE 89].

<sup>10</sup> Hier haben wir es dann mit "Abstraktionen von Abstraktionen" über mehrere Ebenen zu tun.

<sup>11</sup> Ortner 1993.

*Auch hier kann die Praxis zeigen, ob die Theorie stimmt.*

*Wenn Sie jemals als Softwareentwickler arbeiten sollten, werden Sie ihn erleben, den "typischen Anwender", so sicher wie das Amen in der Kirche.*



*sensfähig und verbindlich für eine definierte Gruppe sind (z.B. Branche, Unternehmen, Abteilung). Die zentrale Idee bildet dabei die Annahme, dass alles, was auf dem Weg zu einer Aufgabenlösung eingesetzt wird, zunächst hergestellt bzw. rekonstruiert werden muss.*"<sup>12</sup>

*In einigen Anwendungsbereichen gibt es bereits ausgereifte Modelle. Man nennt diese dann "Referenzmodelle".*

Auch unabhängig von den o.g. wissenschaftlichen Erkenntnissen hat sich in der Praxis der Wirtschaftsinformatik die Vorgehensweise bewährt, standardisierte Modelle für betriebswirtschaftliche Anwendungen als Grundlage der zu implementierenden Softwaresysteme zu verwenden. Hier wären z.B. die "Referenzmodelle" unterschiedlicher Hersteller oder die Daten- bzw. Geschäftsprozessmodelle der Firma SAP einzuordnen. Als Tool für die methodische und technische Unterstützung der Modellierung ist z.B. das ARIS-System von Prof. Scheer [SC 96] auf dem Markt. Diese Systeme sind einerseits sehr erfolgreich, andererseits liegt eine der Hauptschwierigkeiten der Hersteller darin, dass sie oft für den individuellen Anwendungsfall nicht passen bzw. einen hohen Einführungs- und Anpassungsaufwand erfordern. Auch dies ist ein Hinweis darauf, dass die Objekte unserer Anwendungssysteme nicht "einfach auf der Straße liegen" und es (frei nach Kant) "den Kunden an sich" nicht gibt.

Wir werden einige Eigenheiten der menschlichen Wahrnehmung anhand sogenannter "Kipp-Bilder" studieren. Ein auch für den Laien gut lesbares Buch mit Beispielen dazu und einer Darstellung interessanter Ergebnisse der Gehirnforschung ist [HA 92]. Auch der Physiker Prof. H. Haken, der dieses Buch gemeinsam mit seiner Tochter verfasst hat, kommt zu dem Ergebnis, das "Wahrnehmung" zumindest zu einem großen Teil "Wahr-Bildung" ist.

### 7.3.3 Konstruktion abstrakter Objekte

Wir haben uns bereits bewusst gemacht, dass das Phänomen der Wahrnehmung nicht mehr als einfacher Abbildungsprozess im Sinne eines Fotos, sondern eher als komplexer Konstruktionsprozess zu sehen ist.

Wir wollen im Folgenden diesen Konstruktionsprozess noch etwas genauer untersuchen und dazu diese Varianten der Konstruktion unterscheiden:

*Diesen "mental"en Prozesse werden Sie in allen "Designmethoden" unter anderen Namen wiederfinden.*

- Eigenprädikation
- intensionale Abstraktion
- extensionale Abstraktion
- Mereologie

*Die neuen Namen sind teilweise nur deswegen erfunden worden, damit die jeweilige Methode sich besser, weil neu, verkaufen lässt*

Zunächst aber noch einige Gedanken zu der Frage, wie wir mit den einmal wahrgenommenen (erkannten) Objekten umgehen. Eine Art des Umgehens ist sicherlich der unmittelbare physische Umgang mit den Gegenständen unserer Umwelt. Das kann (und muss) jeder ständig selbst ausprobieren. In unserer modernen Zivilisation ist es jedoch in vielen Fällen die intelligentere Lösung, zumindest in Teilbereichen im Vorfeld physischer Aktivitäten mit "Gedankenexperimenten" das Verhalten realer Systeme zu analysieren oder mit "abstrakten Objekten" zu arbeiten. Man nennt das dann "Planung", "Prognose", "Simulation" usw.

---

<sup>12</sup> aus [LE 95], S. 85:

*Was ist ein Crash-  
Test-Dummy?*

*Ich sehe was, was du  
nicht siehst!*

Es gibt hier viele Zwischenformen, so u.a. die Verwendung physischer Objekte, jedoch nicht der Originale, sondern von verkleinerten oder vereinfachten Nachbildungen (Beispiel: "Dummies" zur Verwendung in Crash-Tests in der Automobilindustrie). Als Wirtschaftsinformatiker besteht ein erheblicher Anteil unserer Arbeit im Umgang mit abstrakten Objekten.

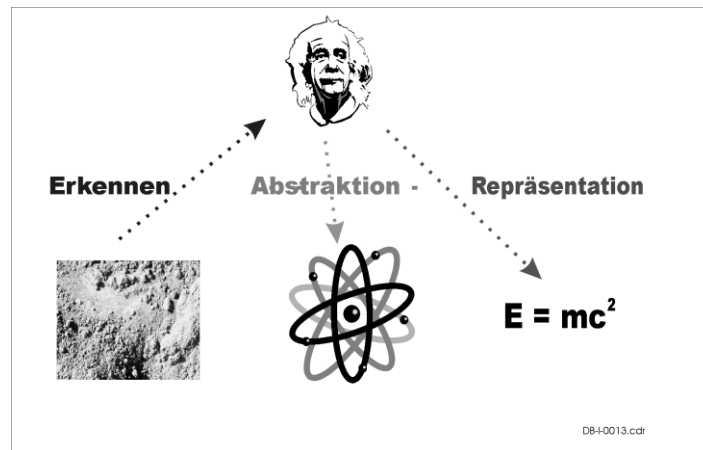


Abbildung 7-1; Vom Erkennen zur Repräsentation

*Sprachen spielen eine  
zentrale Rolle für jede  
Art von Modellierung.*

Das Medium für den abstrakten Umgang mit realen Objekten oder den realen Umgang mit abstrakten Objekten ist die *Sprache* oder besser sind die Sprachen. Wir benötigen Sie zur Beschreibung der Objekte und zur Beschreibung der Operationen, die wir mit diesen Objekten auszuführen gedenken. Ein typisches Beispiel und gleichzeitig ein Spezialfall der Sprachen sind alle Programmiersprachen.

Wie kommen wir nun zu unseren Objekten?

### 7.3.3.1 Eigenprädikation

Der primitivste "Sprachakt" ist die Zuweisung einer sog. Eigenprädikation, auch Eigenaussage genannt, an ein Objekt. Man sagt: "Dies ist ein TISCH", indem man auf ein Objekt zeigt und ihm den Eigenprädikator TISCH zuspricht.

Dadurch wollen wir ausdrücken, dass wir das betrachtete Objekt von anderen Objekten *abgrenzen*, d.h. unterscheiden wollen. Eine Verfeinerung kann durch das Zusprechen sog. Apprädikatoren erfolgen, mit deren Hilfe zusätzliche Unterscheidungen zu anderen Objekten ausgedrückt werden sollen, z.B. "Dies ist ein *kleiner* TISCH." Durch Anwendung dieses Verfahrens erreichen wir eine Gliederung der Gegenstände unserer Welt. Ein anderes Beispiel wäre die Einführung des Begriffs KUNDE durch „Zeigen“ auf eine Person, die in einem Geschäft Waren begutachtet und aussucht.

Das Ergebnis der Festlegung von Eigenprädikatoren und Apprädikatoren für einen bestimmten Gegenstandsbereich, z.B.

KUNDE: KUNDEN-NR, NAME, LIEFERANSCHRIFT, RECHNUNGSANSCHRIFT, ...

nennt man auch Prädikatorenschema oder "relationales Schema". Damit haben wir den direkten Bezug zur Repräsentation von Objekten in relationalen Datenbanksystemen hergestellt.

*DAS kann JEDER.*

Im obigen Beispiel hat auch der Prädikator KUNDEN-NR die Eigenschaft, ein Eigenprädikator zu sein. Er ist nämlich kein weiteres Merkmal im Sinne einer zusätzlichen Unterscheidung, sondern ein Merkmal, das einen bestimmten Kunden "identifiziert". Einen solchen Prädikator nennen wir auch **Primärschlüssel**.

### 7.3.3.2 Intensionale Abstraktion

Wir sagen gerne, dass wir "Kunden in unserer Datenbank speichern" und sind uns hoffentlich darüber einig, dass es nicht die realen Kunden sind, die wir auf die CD pressen. Die Zuweisung des Prädikators KUNDE führt in unserer Vorstellung, insbesondere, wenn wir sie auf mehrere Kundenexemplare anwenden, zu einem Begriff "KUNDE".<sup>13</sup> Wir gewinnen eine Idee dessen, was ein Kunde für uns sein soll. Mit diesem Begriff und über diesen Begriff reden wir dann so, "als ob" wir ein neues Objekt vor uns hätten. Wir können als abstrakten Stellvertreter des realen Kunden auch den Begriff KUNDEN-NR verwenden. Diesen Übergang vom Prädikator zum Begriff nennt man (intensionale ) Abstraktion.

Häufig wird in der Literatur der Vorgang der Abstraktion als "Weglassen" (abstrahere lat.) von unwesentlichen Teilen o.ä. beschrieben. So auch z.B. in (dem ansonsten empfehlenswerten) Buch [KO 96]: S. 30: "Eine Abstraktion ist eine Vereinfachung eines realen Systems durch Reduzierung auf solche Teile, die für die jeweilige Fragestellung wesentlich sind." Solche Definitionen sind natürlich nicht sehr befriedigend, wenn man sich die grundlegende Bedeutung des Abstraktionsvorgangs für die Modellierung von Systemen vor Augen hält.

### 7.3.3.3 Extensionale Abstraktion

Eine weitere Möglichkeit, um zu neuen Objekten/Begriffen zu gelangen, ist die extensionale Abstraktion. Hier wird der Übergang zum Begriff nicht über einen Prädikator vollzogen, sondern über konkrete Folgen  $y$ , also z.B.  $y_1, y_2, y_3, y_1$ , zu abstrakten Mengen. [WE 92]. So tauchen z.B. die Kunden eines Unternehmens (repräsentiert durch Datensätze) in unterschiedlichen Bereichen und unterschiedlichen Geschäftsprozessen mehrfach auf. Es entsteht das Bedürfnis die konkrete, vermutlich stark redundante Ansammlung der Ausprägungen dieser Kunden als abstrakte Menge "UNSERE KUNDEN" aufzufassen. Auch auf diesem Weg kommt man dann zum Begriff "KUNDE".

*Das kann auch FAST jeder.*

Ein gutes Beispiel eines mehrstufigen extensionalen Abstraktionsprozesses bietet die in vielen Wissenschaften übliche *Art-Gattungs-Abstraktion* zur Gliederung des Gegenstandsbereichs der Untersuchung. Hier werden ganze Begriffshierarchien gebildet, die durch Subordination, d.h. durch Zusammenfassen von Prädikatoren ("Arten") zu übergeordneten Prädikatoren ("Gattungen") entstehen.

Eine andere Möglichkeit ist (schon in der antiken Definitionslehre) die Nennung einer nächsthöheren Gattung und die anschließende Angabe eines artbildenden Unterschiedes. Ein Beispiel ist die hierarchische Gliederung des Tierreiches in der Zoologie, siehe folgende Abbildung.

*Solche oder ähnliche Schemata kennen Sie alle. Finden Sie weitere Beispiele!*

### 7.3.3.4 Mereologie

Eine weiterer Weg zur Bildung neuer Objekte ist die Mereologie (Logik der Teil/Ganze-Relation, Montagelogik). Hier werden neue Objekte durch Zusam-

<sup>13</sup> Wir sagen ja auch gelegentlich, dass wir uns von etwas "einen Begriff machen".

mensetzen<sup>14</sup> bereits vorhandener Objekte gewonnen. Ein Beispiel dafür wäre etwa die Montage eines Autos aus den Einzelteilen Karosserie, Motor, Getriebe usw. Auch hier lassen sich wieder ganze Hierarchien bilden, z.B. durch weiteres Auflisten der Getriebeteile, wie Zahnräder, Wellen usw. *"Ob so ein montiertes Getriebe ein neues Teil ist, oder bloß als Zusammensetzung "alter" Teile anzusehen ist, ist nur eine Frage des Bezeichnens. Wichtig ist die Feststellung, dass das neue Objekt auf derselben Sprachstufe entsteht. Mereologie kennt keine Abstraktion und somit keine Sprachebenen. Man sollte aber zur Kenntnis nehmen, dass der mereologische Weg in der Softwarekonstruktion mindestens genauso bedeutsam ist, wie die abstraktive Methode."*<sup>15</sup>

### 7.3.4 Beschränkung auf das Datenmodell

Wir werden uns hier auf das Datenmodell beschränken, d.h. das viel weiter zu fassende Problem des „Informationssystem“-Designs ausklammern. Wir werden auch das Entity Relationship Model in einer sehr einfachen Form verwenden.

*Manchmal ist weniger  
mehr...*

Der Hauptgrund für diese Beschränkung liegt darin, dass wir als Wirtschaftsinformatiker im Rahmen der Entwurfsprozesses zunächst mit Mitteln arbeiten müssen, die auch der Anwender versteht, ohne zuerst ein Informatikstudium absolviert haben zu müssen. Die Beteiligung der Benutzer ist ein ganz wesentlicher kritischer Erfolgsfaktor für das erfolgreiche Design einer Datenbank.. Und wenn wir diese Benutzer mit softwaretechnischen Begriffen wie Klassen oder abstrakten Datentypen konfrontieren oder sie gar fragen, ob sie wissen, wie viele Palindrome sie in der Datenbank haben, dann ...na ja, ich will das nicht weiter ausführen. Im schlechtesten Fall tut er so, als ob er uns verstehen würde. Das Ergebnis sehen wir am Abend des ersten Tages, nachdem das System in Produktion gehen sollte.

Das von uns verwendete *vereinfachte Entity-Relationship-Modell* wird im folgenden erläutert. Es verwendet von den oben genannten Modellkomponenten lediglich die folgenden:

Basismodell:

- Objekt
- Attribut

Statisches Modell:

- Assoziation
- Aggregation

Alle anderen Aspekte werden nicht im Rahmen des Datenbankdesigns behandelt.

Es sei noch darauf hingewiesen, dass Sie sich die Begriffe und Notationen, die Ihnen vielleicht in Büchern, Systembeschreibungen oder anderen Vorlesungen begegnen, jeweils sehr genau ansehen sollten, da einige Begriffe und Darstellungen leider widersprüchlich verwendet werden. Sie werden sich vielleicht auch fragen, weshalb wir eine so fortschrittliche objektorientierte Technik wie die Ver-

---

<sup>14</sup> In der Informatik auch Komposition oder Aggregation genannt

<sup>15</sup> aus [WE 92], S. 34

erbung nicht in unseren Werkzeugkasten aufnehmen. Die Gründe dafür sind folgende:

Vererbung ist weniger für die statische Beschreibung von Systemen, wie wir sie zunächst anstreben, interessant, sondern eher für die Wiederverwendung vererbter Methoden, also "dynamischer" Elemente.

Um die Technik der Vererbung überhaupt anwenden zu können, benötigen wir bereits eine Gliederungslogik, z.B. in Form einer von uns entworfenen Objektstruktur. Letztere ist also die Voraussetzung für den weiteren Entwurf.

Die als Voraussetzung für die Anwendung der Vererbungstechniken erforderliche Struktur schränkt - zumindest wenn man "Mehrfachvererbung" ausschließt - die Anwendungsmannigfaltigkeit der Vererbung ein [WE 92, 32]. Man ist an die einmal festgelegte Gliederungslogik gebunden.

Die Definition der Objekte und Assoziationen eines Datenmodells ist ohnehin eine schwierige Aufgabe. Wenn man nun sieht, dass die einmal "gefundene" Struktur eine solch kritische Bedeutung auch hinsichtlich der späteren Handlungsmöglichkeiten (Klassenhierarchien, Programmiermöglichkeiten) hat, scheint es angebracht, *das Datendesign als eigenständigen Entwurfschritt beizubehalten*. Der Entwurf von Klassenhierarchien für die objektorientierte Programmierung kann sich z. B. daran anschließen oder auch dem Design der Datenbank vorausgehen.

### 7.3.5 Datenmodelle und „Beziehungen“ zwischen „Objekten“

Die Join-Operation bietet uns die Möglichkeit, Tabellen über Ihre Attribute zu verbinden bzw. dem Sachverhalt einer bestehenden *logischen Verbindung* zwischen Tabellen („Objekten“) nachzugehen. Man könnte auch sagen, dass die Join-Operation die in der Datenbank durch die Attribute bzw. Attributwerte repräsentierten Strukturen explizit macht.

## 7.4 Datenbankdesign

In der Praxis anzutreffende Datenbanken enthalten nicht selten sehr komplexe Strukturen, deren Entwurf und Pflege sich in größeren Unternehmen als eine eigenständige Aufgabe im Bereich der IT-Tätigkeiten etabliert hat. Diese Aufgabe nennt man „Datenbankdesign“. Wir wollen an dieser Stelle einen Blick auf das Thema Datenbankdesign werfen.

Unter dem Begriff *Datenbankdesign* versteht man den Entwurf und die Darstellung der in der Datenbank zu repräsentierenden Typen und deren logischer Struktur, d.h. der zwischen ihnen bestehenden logischen Beziehungen. Ein solches Design kann auf unterschiedlichen Abstraktionsebenen stattfinden. Wenn man auf derjenigen Abstraktionsebene arbeitet, die wir bereits als „logische“ Ebene kennen gelernt haben, spricht man von „*logischem Datenbankdesign*“ oder auch von „konzeptuellem Datenbankdesign“.

Das *Ergebnis* des Designs nennt man dann *Schema* oder auch *Datenmodell*. Der Begriff „Datenmodell“ wird aber üblicherweise (leider) auch noch in einer anderen Bedeutung gebraucht, und das soll zum besseren Verständnis im folgenden Absatz kurz erläutert werden.

Für das logische Design von Datenbanken gibt es mehrere Ansätze. Diese unterscheiden sich hinsichtlich der verwendeten Grundlagen und der Gestaltungshilfsmittel zur Darstellung eines Datenbankentwurfs. Man nennt nun die Gesamtheit der Darstellungsmethoden und –hilfsmittel eines speziellen Ansatzes ebenfalls ein „Datenmodell“. Dieses Datenmodell enthält die *Modellierungskonstrukte* (Bausteine) zur Erzeugung eines Informationsmodells desjenigen „Realitätsausschnitts“, den wir in der Datenbank abbilden wollen<sup>16</sup>.

Wenn man vom „*relationalen Modell*“ spricht, können damit also mindestens zwei Dinge gemeint sein, nämlich entweder

- die **Repräsentation** einer Datenbank mit Hilfe des Gestaltungsmittels „Relation“ auf der logischen Ebene oder
- die **Gesamtheit der Methoden und Darstellungsmittel**, also z. B. die relationale Algebra, die man für den Entwurf verwendet.

Wir haben bisher zur Darstellung unserer Datenbankentwürfe ausschließlich das Konzept der Relation und zur Umsetzung in konkrete Datenmodelle das Konzept der Tabelle betrachtet<sup>17</sup>. Es gibt jedoch auch andere Darstellungsmöglichkeiten, wie z. B. das im Folgenden beschriebene „Entity Relationship Model“.

---

<sup>16</sup> Man kann **diesen** Begriff des Datenmodells also in einer gewissen Analogie zu einer Programmiersprache sehen, die ja auch nichts anderes tut als die Typkonstruktoren und Sprachkonstrukte zu definieren, die wir für die Gestaltung eines Anwendungsprogramms einsetzen können.

<sup>17</sup> Wenn wir die beschriebene Doppelbedeutung des Begriffs „Datenmodell“ hier einmal gezielt einsetzen, könnten wir sagen : Wir haben das relationale Modell verwendet und damit relationale Modelle entworfen. Alles klar?

## 7.5 Das Entity Relationship Model

### 7.5.1 Konstruktionselemente

Ebenfalls häufig anzutreffende Darstellungsformen für Datenbankschemata ergeben sich daraus, dass man die Relationen bzw. Tabellen als „Objekte“ oder „Entitäten“ interpretiert und in graphischer Form darstellt. Zur Verdeutlichung und intuitiveren Darstellung von logischen Beziehungen werden in diesen Darstellungsformen Verbindungslinien zwischen den unterschiedlichen Objekttypen eines Schemas verwendet, um die sog. „Relationships“ oder „Assoziationen“ zu repräsentieren<sup>18</sup>.

Bekannte Datenmodelle, die mit solchen Konstrukten arbeiteten, sind das **Entity Relationship Modell (ERM)** von Chen und seine Erweiterungen sowie die damit eng verwandten *objektorientierten Datenmodelle*.

Das ERM bietet uns als Bauelemente zur Konstruktion eines Datenmodells an:

- Entitäten,
- Attribute und
- Beziehungen.

*Entitäten* (Entities) sind reale oder abstrakte Dinge, die für den betrachteten Ausschnitt der Aufgaben einer Unternehmung von Interesse sind, z.B. KUNDE.

*Attribute* sind Eigenschaften von Entitäten, z.B. Kundennummer, Name und Anschrift des Entitätstyps KUNDE.

*Assoziationen* (Beziehungen) sind logische Verbindungen zwischen zwei oder mehreren Entitäten.

Alle drei Begriffe können auf der Ausprägungs- und der Typebene betrachtet werden. Die Typebene repräsentiert Mengen von Exemplaren einer Entität, die Ausprägungsebene die einzelnen Exemplare.

Beziehungstypen werden weiterhin hinsichtlich ihrer **Kardinalität**<sup>19</sup> unterschieden.

Die **Kardinalität** einer Beziehung gibt an, wie viele andere Entitätsexemplare einer gegebenen Entität im Rahmen der betrachteten Beziehung höchstens zugeordnet werden.

Man unterscheidet so z.B. zwischen Beziehungen der Kardinalität 1:N und Beziehungen der Kardinalität M:N. Das müssen wir uns (s.u.) noch etwas genauer ansehen.

---

<sup>18</sup> Um genau zu sein, müssten wir hier immer zwischen Objekttypen und Objektexemplaren, sowie Assoziationstypen und Assoziationsausprägungen unterscheiden. Da wir uns aber vornehmlich auf der „logischen“ Ebene aufhalten, dürfte klar sein, dass i. Allg. „Typen“ gemeint sind, auch wenn dies nicht explizit zum Ausdruck gebracht wird. .

<sup>19</sup> auch „Komplexität“ genannt

Analog zum bereits bekannten Primärschlüssel einer Relation können wir einen *Entitätsschlüssel* einführen: Ein Entitätsschlüssel ist ein Entitätsattribut, mit dessen Werten die einer Entitätsmenge angehörenden Entitäten eindeutig identifiziert werden können.

Die einfachsten graphischen Darstellungsmittel des Entity Relationship Models, und nur diese wollen wir uns hier zunächst ansehen, sind:

- Rechtecke zur Repräsentation von Entitätstypen
- Linien (ggf. Pfeile<sup>20</sup> o.ä.) zur Repräsentation von Assoziationstypen, ggf. mit Angabe der Kardinalität.

Wir wollen uns die oben eingeführten Begriffe an Hand einiger Beispiele verdeutlichen. Wir verwenden dazu einfach das Datenmodell, das uns bereits bekannt ist, nämlich das Datenmodell der Firma eCrash. Dieses Datenmodell liegt in relationaler Form bzw. in Form von Datenbanktabellen vor<sup>21</sup>. Wir werden es lediglich in eine andere Darstellungsform, nämlich ein Entity Relationship Diagramm (ERD) überführen.

### 7.5.2 1:-N Assoziationen im Entity Relationship Modell

Wir gehen so vor, dass wir zunächst ABTEILUNGEN und MITARBEITER auf der Exemplarebene darstellen, ganz einfach, indem wir notieren, welche Exemplare es davon gibt. Genauso machen wir es mit den Assoziationen. Auch hier gibt es „Exemplare“:

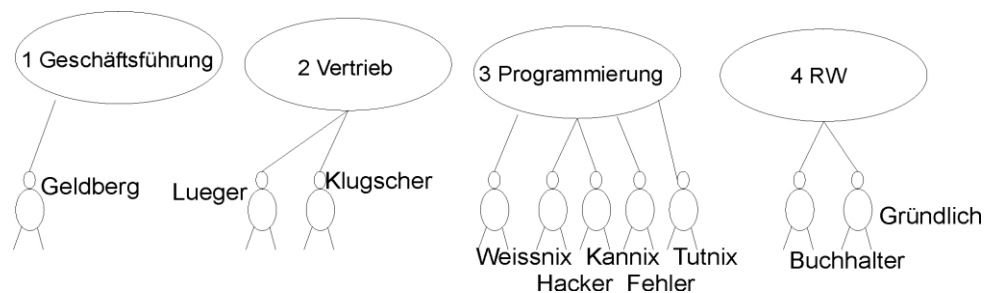


Abbildung 7-2: 1:-N Assoziationen auf der Exemplarebene

Wie man sieht, werden hier jedem Exemplar aus der Menge der Abteilungen ein oder mehrere Exemplare aus der Menge der Mitarbeiter zugeordnet. Umgekehrt trifft dies jedoch nicht zu: Jeder Mitarbeiter wird genau einer Abteilung zugeordnet. Man spricht in diesem Fall von einer Assoziation der Kardinalität 1:N oder einfach von einer 1:N-Beziehung.

Die „Typ-Ebene“ des ERM abstrahiert nun die obige Darstellung, indem sie von den einzelnen Exemplaren absieht und zu den Entitätstypen ABTEILUNG und MITARBEITER übergeht, die stellvertretend für Herrn Lueger, Klugscher, Weissnix usw. stehen.

<sup>20</sup> Dadurch kann z.B. die Kardinalität der Beziehungen kann an den Enden der Verbindungslinien vermerkt werden.

<sup>21</sup> Siehe auch Tabellenbeschreibungen im Anhang.



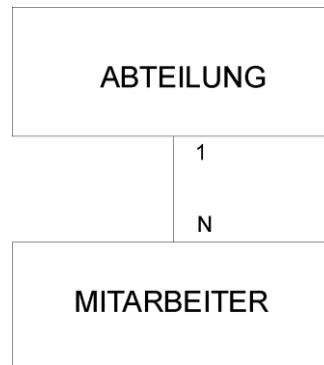


Abbildung 7-3: 1:N-Assoziation im graphischen Modell (ERM)

### 7.5.3 1:-N Assoziationen im Relationenmodell<sup>22</sup>

Im Relationenmodell kann eine 1:N-Assoziation realisiert werden, indem ein Attributwert von der N-Seite aus auf das zugehörige Exemplar („Vater“) auf der 1-Seite verweist. Wir werden das auch bei der ACCESS-Darstellung weiter unten noch sehen.

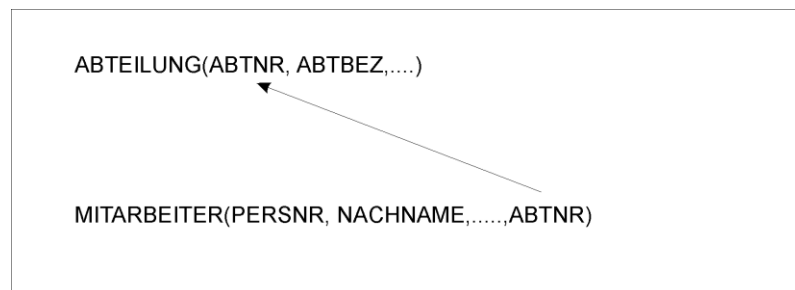


Abbildung 7-4: 1:N Assoziation im Relationenmodell

### 7.5.4 M:N Assoziationen

Stellen wir uns nun vor, Günter Geldberg stellt wieder einmal eine neue Anforderung, nämlich, dass alle Kundenkontakte der Mitarbeiter ebenfalls in der Datenbank erfasst werden sollen. Wir könnten diese Anforderung dadurch erfüllen, dass wir in unserem Datenmodell eine Assoziation KUNDENKONTAKT einführen. Dann haben wir auf der einen Seite der darzustellenden Beziehung die Menge der eCrash-Mitarbeiter und auf der anderen Seite die Menge der eCrash-Kunden. Wir sehen uns das Ganze zunächst wieder auf der Exemplarebene an. Wie? Indem wir einen Blick in unsere relationale Datenbank werfen<sup>23</sup>. Dort gibt es Kunden

<sup>22</sup> Die Darstellung im Relationenmodell ist eigentlich nicht Gegenstand dieses Kapitels, wird jedoch hier zum Vergleich angegeben.

<sup>23</sup> s. Anhang. Beispieltabellen. Bitte beachten: In die graphische Darstellung wurden aus Gründen der Anschaulichkeit einige zusätzliche Kundenkontakte aufgenommen, die sich in der Tabelle KUNDENKONTAKT nicht wiederfinden.

*Aufgabe:* Ergänzen Sie die Verbindungen in Ihrer eigenen Testdatenbank.

und eine Relation Kundenkontakt, in der vermerkt werden kann, welcher Mitarbeiter es mit welchem Kunden zu tun hatte, z. B.

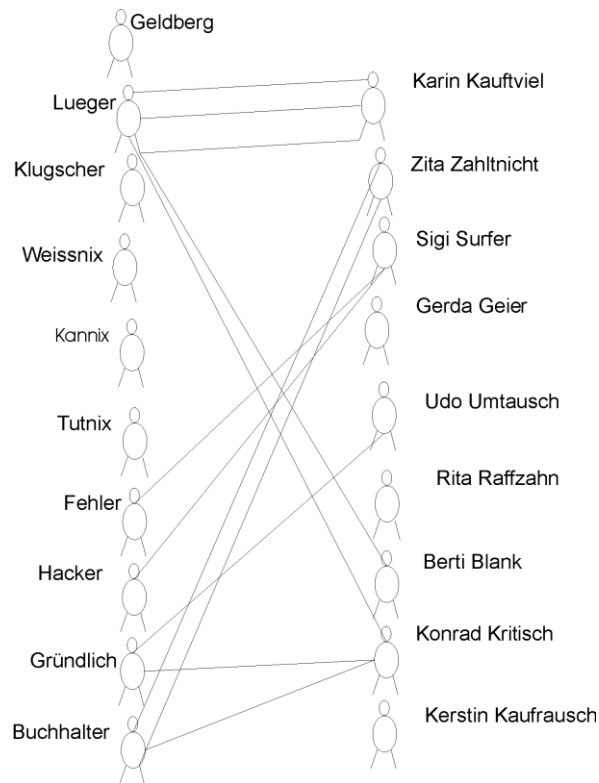


Abbildung 7-5: M:N-Assoziationen auf der Ausprägungsebene

Hier handelt es sich um eine M:N-Assoziation: Ein MITARBEITER kann Kontakt zu den verschiedensten KUNDEN haben und ein KUNDE kann schon mit vielen verschiedenen MITARBEITERn zu tun gehabt haben.

Auf der Typebene des ERD wird dieser Sachverhalt wie folgt dargestellt:



Abbildung 7-6: M:N-Assoziation auf der Typebene

Die Verbindungslinie wird zur Kennzeichnung der M:N-Eigenschaft häufig auch mit Pfeilen, Doppelpfeilen, Punkten, Gabeln (Besen) o.ä. Symbolen versehen.

### 7.5.5 Darstellung von M:N-Assoziationen im Relationenmodell<sup>24</sup>

<sup>24</sup> Die Darstellung im Relationenmodell ist eigentlich nicht Gegenstand dieses Kapitels, wird jedoch hier zum Vergleich angegeben.

Da sich der Sachverhalt der M:N-Beziehung (s.o.) im relationalen Datenmodell nicht so einfach darstellen lässt wie die 1:N-Beziehung<sup>25</sup>, wird die obige Konstruktion bei Verwendung eines relationalen Datenbanksystems durch Einführung einer neuen (Verbindungs-) Entität und zwei neuen Beziehungen ersetzt: Diese Ersatzkonstruktion kann man gewissermaßen als eine Vorbereitung für die Darstellung im Relationenmodell ansehen:

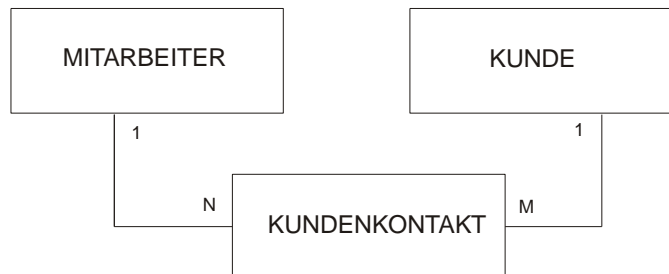


Abbildung 7-7: Auflösung einer M:N-Assoziation

Die M:N-Assoziation lässt sich im Relationenmodell nicht mehr einfach dadurch darstellen, indem die Verweise für die Beziehungen in die verbundenen Relationen aufgenommen werden. Vielmehr benötigen wir eine neue Relation (Verbindungsrelation) zur Repräsentation der Beziehungen.

### 7.5.6 Die Chen<sup>26</sup>-Notation

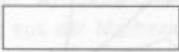
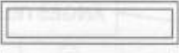





In den vorhergehenden Ausführungen wurden die grundlegenden Konzepte des Entity Relationship Models (ERM) dargestellt. Im Laufe der Zeit sind vielfältige Erweiterungen des ERM in Gebrauch gekommen, verbunden mit ebenso vielfältigen Varianten der verwendeten Notationen.

Die folgende Tabelle gibt (ohne Anspruch auf Vollständigkeit) eine Übersicht über die gängigsten Symbole eines „einfachen“ ERM (Quelle: RRZN Skript).

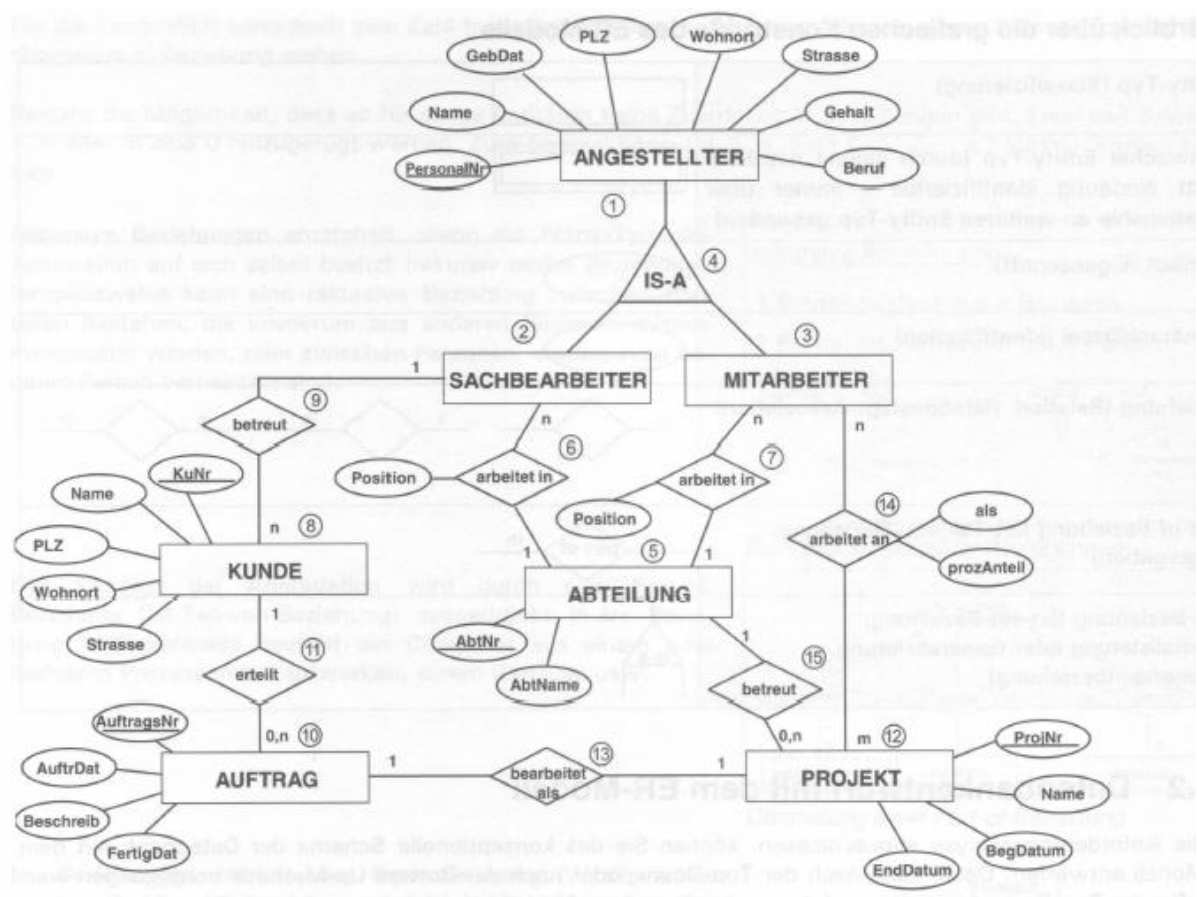
<sup>25</sup> ...nämlich durch Verweis aus dem Objekt der „M-Seite“.

<sup>26</sup> Siehe z.B. [http://de.wikipedia.org/wiki/Peter\\_Chen](http://de.wikipedia.org/wiki/Peter_Chen)

### Überblick über die grafischen Konstrukte des ER-Modells

Entity-Typ (Klassifizierung)	
Schwacher Entity-Typ (durch eigene Attribute nicht eindeutig identifizierbar - immer über Relationship an weiteren Entity-Typ gebunden)	
Attribut (Eigenschaft)	
Primärschlüssel (Identifikation)	
Beziehung (Relation, Relationship, Assoziation)	
Part-of-Beziehung (Ist-Teil-von-Beziehung, Aggregation)	
Is-a-Beziehung (Ist-ein-Beziehung, Spezialisierung oder Generalisierung, Teilmengenbeziehung)	

### Anwendungsbeispiel:



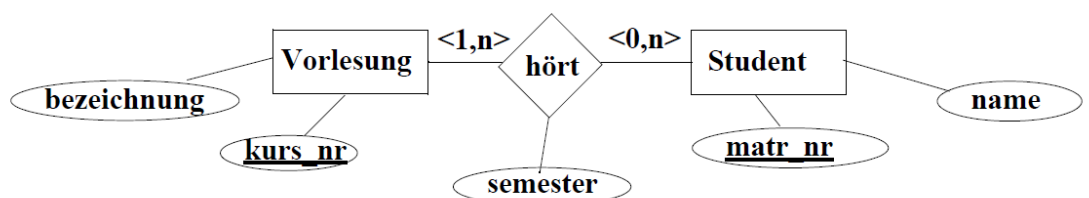
### 7.5.7 Die <min,max> Notation

Hinsichtlich der Angabe der Kardinalitäten gibt es ebenfalls eine breite Palette an Möglichkeiten. Eine Übersicht gibt die folgende Tabelle (Quelle: Wikipedia)

(min,max) [Entity 1]	Multiplizität [UML, Entity 1]	Chen-Notation	MC-Notation	Multiplizität [UML, Entity 2]	(min,max) [Entity 2]
(0,1)	0..1	1:1	c:c	0..1	(0,1)
(0,N)	0..1	1:n	c:mc	0..*	(0,1)
(0,N)	1..1	1:N + total participation	1:mc	0..*	(1,1)
(0,N)	0..*	m:n	mc:mc	0..*	(0,N)
(1,1)	0..1	total participation + 1:1	c:1	1..1	(0,1)
(1,N)	0..1	total participation + 1:N	c:m	1..*	(0,1)
(1,1)	1..1	total part. + 1:1 + total part.	1:1	1..1	(1,1)
(1,N)	1..1	total part. + 1:N + total part.	1:m	1..*	(1,1)
(1,N)	0..*	total participation + M:N	mc:m	1..*	(0,N)
(1,N)	1..*	total part. + M:N + total part.	m:m	1..*	(1,N)

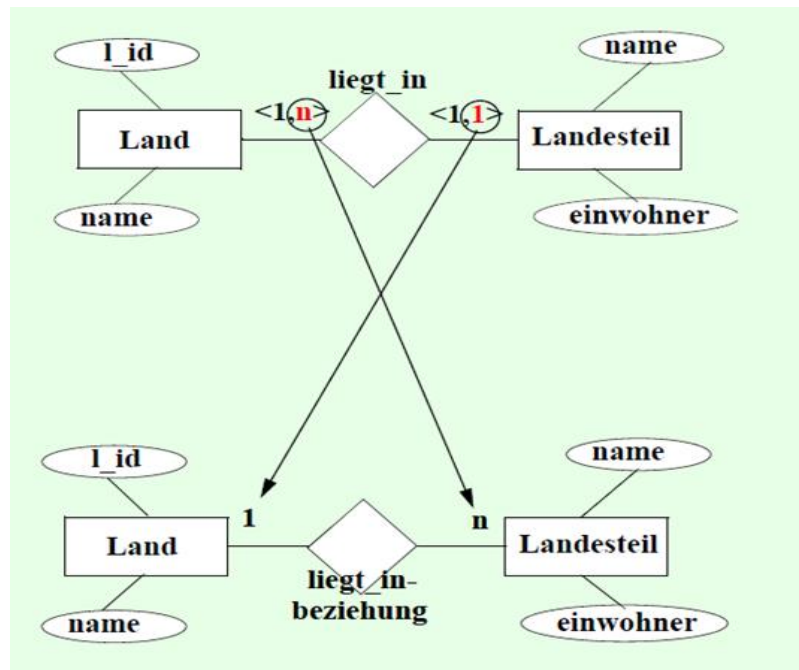
Eine Notation, die häufig anzutreffen ist, ist die in der Tabelle ebenfalls aufgeführte sog. **<min,max> Notation**. Sie arbeitet mit den gleichen graphischen Elementen wie die „einfache“ Chen-Notation.

Der Unterschied liegt in der Ermittlung und Angabe der Kardinalitäten der Beziehungen: Es wird nicht gefragt: „Wie viele Partner hat eine Entität im Zusammenhang mit der betrachteten Beziehung?“, sondern „*An wie vielen Beziehungsausprägungen (des betrachteten Beziehungstyps) nimmt die Entität <mindestens,höchstens> teil?*“ Im Folgenden werden Beispiele dazu angegeben.



- Eine Vorlesung wird in einem Semester von minimal einem Student gehört, i.a. sind es aber n Studenten, die die Vorlesung in einem Semester hören (Minimal: 1; Maximal: n)
- Ein Student hört in einem Semester keine (Praxissemester), eine (fauler Hund) oder mehrere Vorlesungen (Normalfall)

In der folgenden Darstellung wird die  $\langle \min, \max \rangle$  Notation (obere Hälfte) der „einfachen“ Chen Notation gegenübergestellt. Wie man sieht, taucht das „n“ einer 1:n Beziehung jeweils auf der entgegengesetzten Seite der Darstellung auf. Das ist wohl etwas gewöhnungsbedürftig und nicht sehr „intuitiv“. Die  $\langle \min, \max \rangle$  Notation hat jedoch auch Vorteile.

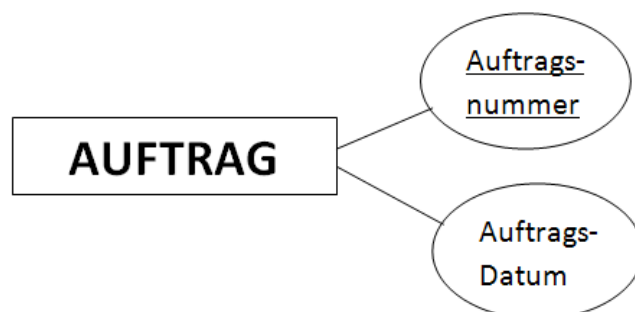


### 7.5.8 Modellierung von starken und schwachen Entitätstypen

Sogenannte *schwache Entities* können nicht autonom existieren, sondern sind in ihrer Existenz von einem anderen, übergeordneten Entity abhängig und nur in Kombination mit dem Schlüssel des übergeordneten Entity eindeutig identifizierbar. Schwache Entities werden durch doppelt gerahmte Rechtecke repräsentiert.

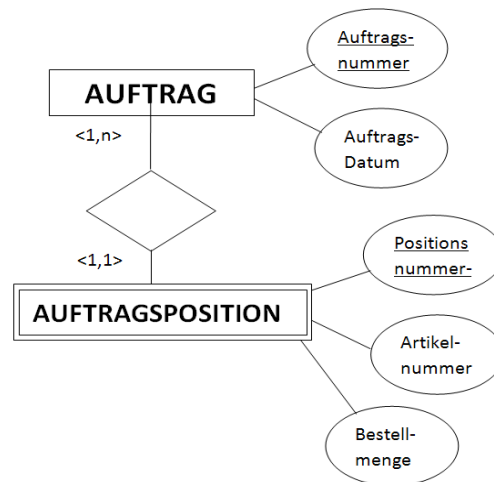
#### (1) Starker Entitätstyp

Man spricht von einer sog. starken Entität A, wenn ihre Identifikation durch ein oder mehrere Werte von Attributen des gleichen Entitätstyps möglich ist. Die Identifikation einer. Andere Entitäten spielen keine Rolle im Zusammenhang mit der Entität A.



## (2) Schwacher Entitätstyp

Zur Identifikation einer schwachen Entität ist ein Attributwert einer anderen mit der schwachen Entität in Beziehung stehenden Entität starken Typs erforderlich; so ist z. B. für die Identifikation des schwachen Entitätstyps Auftragsposition neben der Positionsnummer<sup>27</sup> die Auftragsnummer des starken Entitätstyps Auf-



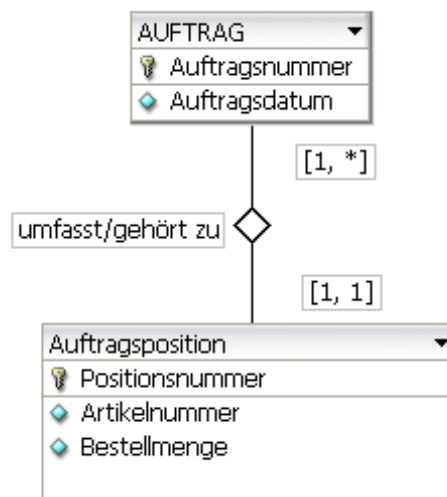
trag erforderlich.

Manche Datenmodelle bzw. Modellierungstools verwenden auch die Begriffe

- Identifizierende Beziehung bzw.
- Nicht identifizierende Beziehung

Der / die Teilschlüssel

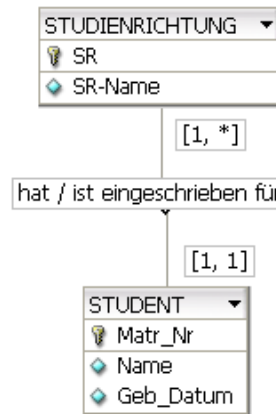
## (3) Identifizierende Beziehung (Darstellung mit dem Tool DBDesigner4)



<sup>27</sup> Die Positionsnummer ist also gewissermaßen ein „Teilschlüssel“ für die Entität AUFTRAGSPOSITION. Aus diesem Grund wird ein solches Attribut in manchen ER-Diagrammen nicht durchgehend, sondern nur gestrichelt unterstrichen.

#### Beispiel 4: NICHT identifizierende Beziehung (DBDesigner4)

Die Tatsache, dass es sich um eine nicht identifizierende Beziehung handelt, ist in DBDesigner4 lediglich an der schwächeren Strichstärke der Beziehungslinien erkennbar.

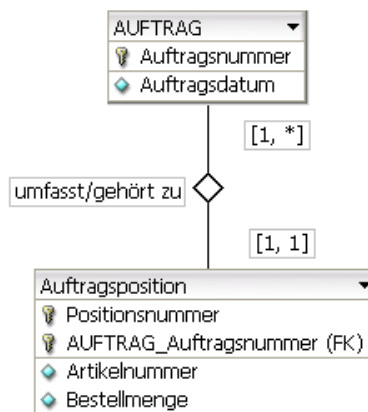


- Bei Verwendung der  $\langle \min, \max \rangle$  Notation findet man auch den Begriff der „**existenziellen Abhängigkeit**“:

So könnte man z.B. im obigen Beispiel die  $\langle \min, \max \rangle$  Angabe auf der Seite der Studienrichtung auch als **[0,1]** angeben, was bedeuten würde, dass es auch Studienrichtungen geben darf, die (noch?) keine Studierenden haben. D.h. die Existenz einer Studienrichtung wird nicht unbedingt vom Vorhandensein von Studierenden abhängig gemacht.

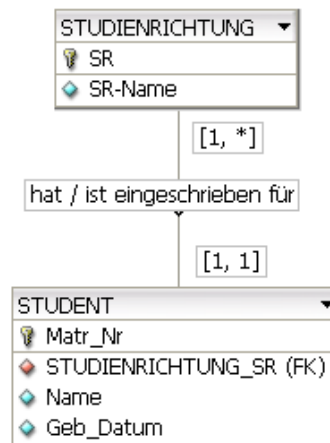
Im Beispiel **AUFTRAG/AUFTRAGSPOSITION** hat man dagegen (sogar beidseitige) existentielle Abhängigkeit. Weder ein Auftrag ohne Positionen noch eine Position ohne Auftrag ist eine besonders sinnvolle Information.

- (5) Wie (3), aber mit Darstellung der Fremdschlüssel:  
(bei uns VERBOTEN!)





**(6) Wie Beispiel (4), aber mit Darstellung der Fremdschlüssel:**  
(bei uns VERBOTEN!)



Und hier noch die **relationale Darstellung der Beispiele** in Form von CREATE TABLE Anweisungen (generiert von DBDesigner4)

Zu (3) bzw. (5):

```

CREATE TABLE AUFTRAG (
    Auftragsnummer      INTEGER    NOT NULL ,
    Auftragsdatum       DATE,
    PRIMARY KEY(Auftragsnummer));

CREATE TABLE Auftragsposition (
    Positionsnummer      INTEGER    NOT NULL ,
    AUFTRAG_Auftragsnummer  INTEGER    NOT NULL ,
    Artikelnummer        INTEGER,
    Bestellmenge          INTEGER,
    PRIMARY KEY(Positionsnummer, AUFTRAG_Auftragsnummer),
    FOREIGN KEY(AUFTRAG_Auftragsnummer)
    REFERENCES AUFTRAG(Auftragsnummer));
  
```

Zu Beispiel (4) bzw. (6):

```

CREATE TABLE STUDIENRICHTUNG (
    SR                  INTEGER      NOT NULL,
    SR-Name             VARCHAR(20)  NOT NULL,
    PRIMARY KEY(SR));

CREATE TABLE STUDENT (
    Matr_Nr             INTEGER      NOT NULL ,
    STUDIENRICHTUNG_SR  INTEGER      NOT NULL ,
    Name                VARCHAR2(25) NOT NULL,
    Geb_Datum           DATE         NOT NULL,
    PRIMARY KEY(Matr_Nr),
    FOREIGN KEY(STUDIENRICHTUNG_SR)
    REFERENCES STUDIENRICHTUNG(SR));
  
```

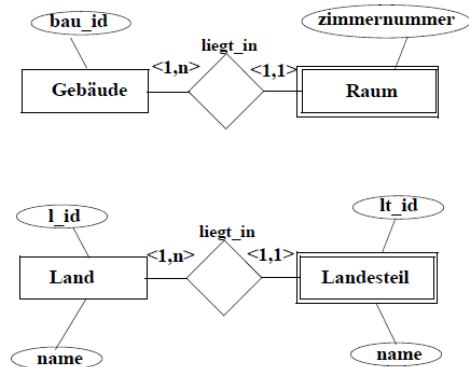
**Weitere Beispiele für schwache Entitäten:**

Die folgende Abbildung verdeutlicht das Konzept anhand von Gebäuden und Räumen. Räume können ohne Gebäude nicht existieren. Die Zimmernummern sind nur innerhalb eines Gebäudes eindeutig.

## Schwache Entitäten

- Existenz abhängig von anderen, übergeordneten Entitätstypen
- Kardinalität zu anderer Entität muss  $\langle 1,1 \rangle$  sein.

**Nur in Kombination mit dem Schlüssel des übergeordneten Entities eindeutig bestimmbar !!**



Andreas Schmidt/Robert Senger

Wiederholung ER-Modell (Teil 2) 5/9

### 7.5.9 Modellierungsmethoden und Modellierungstools

Das Entity Relationship Modell bildet den Ausgangspunkt für eine Vielzahl von Varianten und Erweiterungen von Methoden und von Modellierungsmethoden für das Design, und zwar nicht nur für das Design von Datenbanksystemen, sondern auch für das Design kompletter Anwendungssysteme. Es ist an dieser Stelle nicht möglich, auf alle diese Varianten einzugehen, zumal auch täglich neue hinzukommen können. Leider haben sich im Laufe der Zeit auch viele verschiedene Notationen für die Konstruktionselemente von Datenmodellen herausgebildet.

Mit der Entwicklung einer neuen Methode oder eines darauf aufbauenden Modellierungswerkzeugs ist in der Regel der Versuch verbunden, die entsprechende Nische im Softwaremarkt zu besetzen. So ist es nicht verwunderlich, dass sich in Anlehnung an das schöne Märchen "Des Kaisers neue Kleider" hier ständig etwas tut. Es gibt auch immer wieder Versuche, eine Standardmethode zu etablieren. Die Ausrichtung der Vorlesung an *einer* Standardmethode und *einer* Standardbeschreibungssprache wäre wünschenswert, wenn es denn eine solche gäbe. Leider ist dies jedoch noch nicht der Fall, und eine strenge Ausrichtung erscheint im immerwährenden Stadium des "Methodenkriegs" weder angemessen noch notwendig.

Für die Übungen und Praktika meiner Vorlesung „Datenbanksysteme“ beschränken wir uns deshalb i.allg. auf ein einfaches Entity-Relationship-Modell. Ausschlaggebend für diese Entscheidung sind folgende Überlegungen:

- Ein Ende des Methodenstreits ist nicht abzusehen.
- In Standardwerken der **Wirtschaftsinformatik** wird häufig das ERM verwendet.
- Aus der Sicht der Wirtschaftsinformatik wichtige kommerzielle Anbieter (z. B. SAP) verwenden zur Dokumentation und Publikation der Datenstrukturen Ihrer Systeme eine ebenfalls stark an das ERM angelehnte Darstellung.
- Eine einfache Darstellungstechnik eignet sich zur Verwendung in Vorlesungen und für Entwurfsübungen.
- Anspruchsvollere Modelle mit vielfältigen graphischen Darstellungsmöglichkeiten sind ohne die Unterstützung von Programmen ("Tools") nicht handhabbar.
- Die graphische Darstellung findet für viele Aspekte des Systemdesigns in der Praxis ohnehin ihre Grenzen.

Aufbauend oder in Anlehnung an die verschiedenen Entwurfsmethoden existiert auch eine Vielzahl von Design-Tools für den Datenbank-Entwickler. Diese Tools sind in der Praxis von großer Bedeutung, da sie dem Entwickler zeitraubende und fehleranfällige Tätigkeiten abnehmen oder diese erleichtern.

Die meisten Datenbankmanagementsysteme unterstützen die Arbeit des Datenbankdesigners, indem sie entsprechende Funktionen für die graphische Darstellung von Entwürfen anbieten. Manche Datenbankhersteller oder auch unabhängige Anbieter bieten umfangreiche Methoden- und Softwarepakete speziell für die Aufgabe des Datenbankdesigns.

Heutige Design-Tools bieten typischerweise mindestens den folgenden Funktionsumfang:

- Modellierung von Datenbanken
- Dokumentation vorhandener Datenbanken
- Reengineering-Projekte
- Migrationsprojekte
- Konsolidierung heterogener Datenbank-Umgebungen
- Entwicklung neuer Applikationen (vom Modell zur Applikation)
- Modellierung von Data Warehouse-Architekturen

mit den Leistungsmerkmalen:

- Grafisches Datenmodell mittels Point and Click Technologie, On-Screen Editing
- Vollständige Generierung und Reengineering aller wesentlichen relationalen Datenbanken
- Vollständige Generierung und Reengineering von Triggern und Stored Procedures,
- Gültigkeitsregeln, Vorgabewerte, physikal. Parameter, Kommentare, Indizes, deklarative und referenzielle Integrität, Geschäftsregeln

## 7.6 Aufgaben

### Aufgabe 1

Beantworten Sie folgende Fragen

- 1a) Was verstehen wir unter einem Modell?
- 1b) Was verstehen wir unter einem Datenmodell?
- 1c) Was verstehen wir unter einer Entität? Geben Sie ein Beispiel an.
- 1d) Was verstehen wir unter einer Entitätsausprägung? Geben Sie ein Beispiel an.
- 1e) Was verstehen wir unter einem "Referenzmodell"?
- 1f) Suchen Sie (Literatur/Web) ein Referenzmodell für einen Gegenstandsbereich der Wirtschaftsinformatik.
- 1g) Wozu dient ein "Entity-Relationship-Diagramm" (ERD)?
- 1h) Worin besteht der wesentliche Unterschied zwischen dem „einfachen“ Chen-Modell und dem ERM mit  $\langle \min, \max \rangle$  Notation?

### Aufgabe 2

- 2a) Zeichnen Sie ein bisher in der Vorlesung / im Skript nicht erwähntes Datenmodell mit mindestens 5 Entitätstypen und mindestens einer M:N Beziehung in der einfachen Chen Notation.
- 2b) Zeichnen Sie das Modell aus 2a) in der  $\langle \min, \max \rangle$  Notation.