

Datenbanken

Datenbank-Modellierung

Prof. Dr. Michael Kaufmann

Dozent für Datenbanken

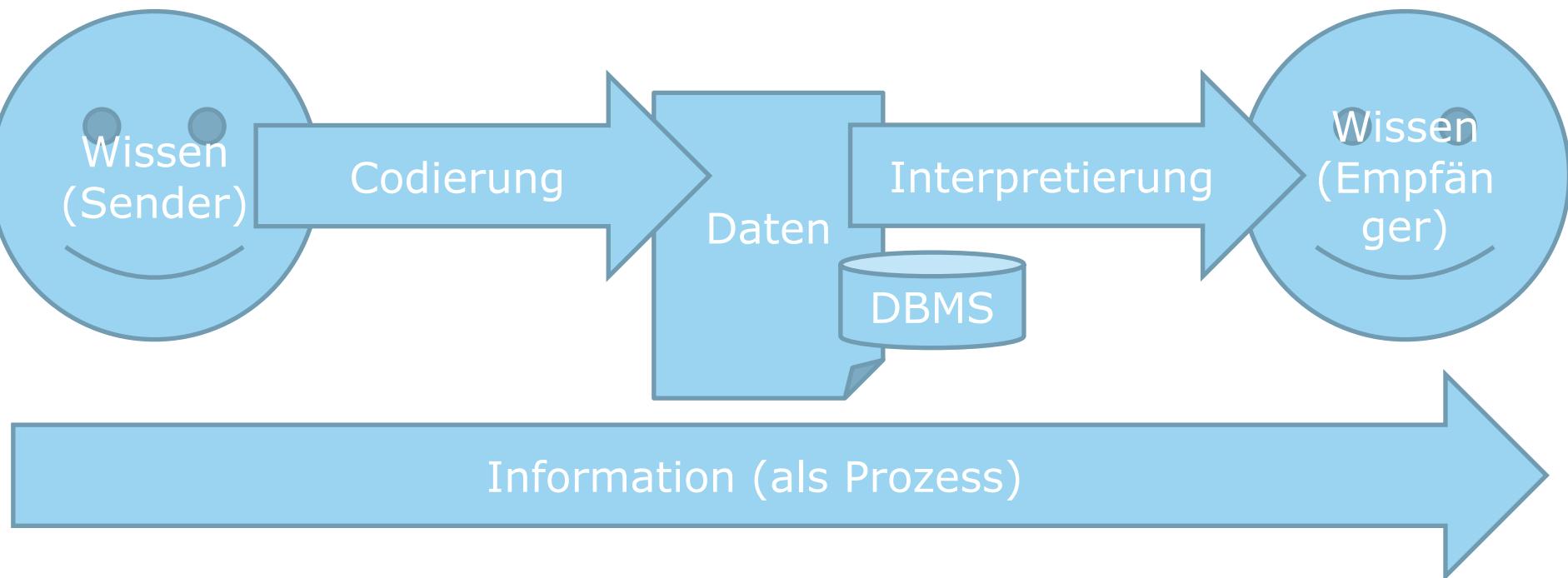
Hochschule Luzern – Informatik, Rotkreuz

T direkt +41 41 757 68 48

m.kaufmann@hslu.ch

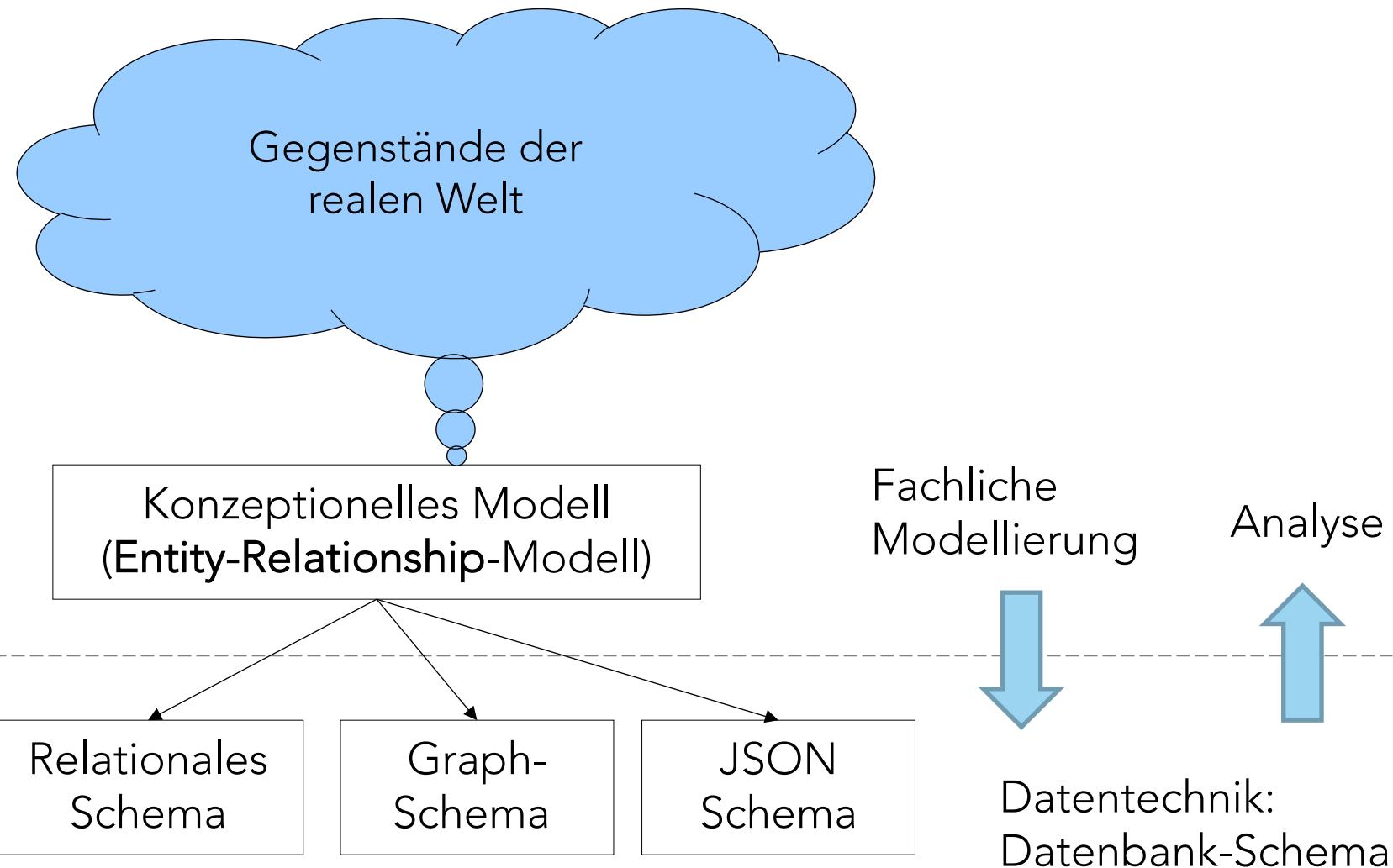


ÜBUNGSBESPRECHUNG



Codierung von Wissen in Daten

Was sind die Voraussetzungen, das Wissen in Daten effektiv so gespeichert werden kann, dass eine spätere Interpretation wieder möglich ist?

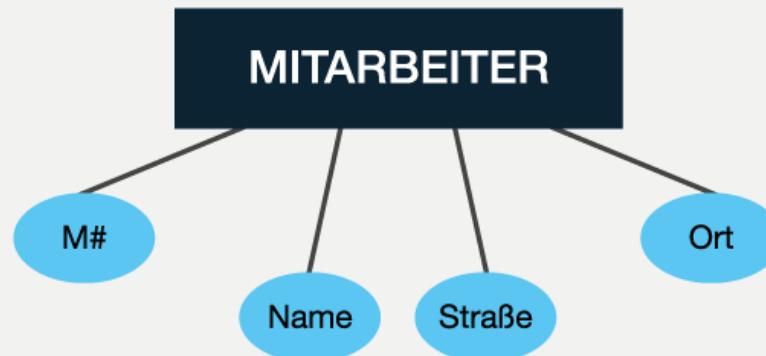


DAS ENTITY RELATIONSHIP MODELL

Entitäten und Entitätsmengen

- Entität:** Mitarbeiter Meier, wohnhaft in der Lindenstrasse in Liestal
- Entitätsmenge:** Menge aller Mitarbeiter mit Merkmalen Name, Straße und Ort
- Identifikationsschlüssel:** Mitarbeiternummer als künstlicher Schlüssel

Darstellung im Entitäten-Beziehungsmodell



Beziehungen und Beziehungsmengen

Beziehung:

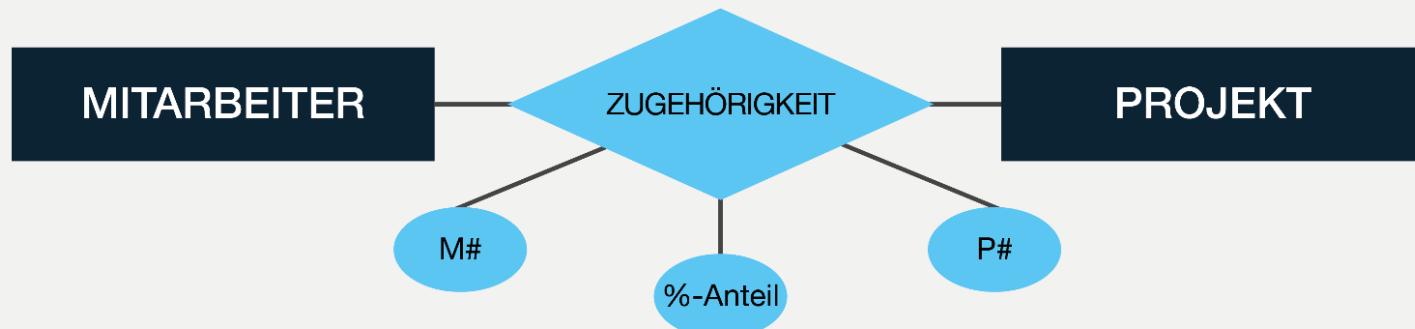
Mitarbeiter Meier arbeitet zu 70% am Projekt P17

Beziehungsmenge:

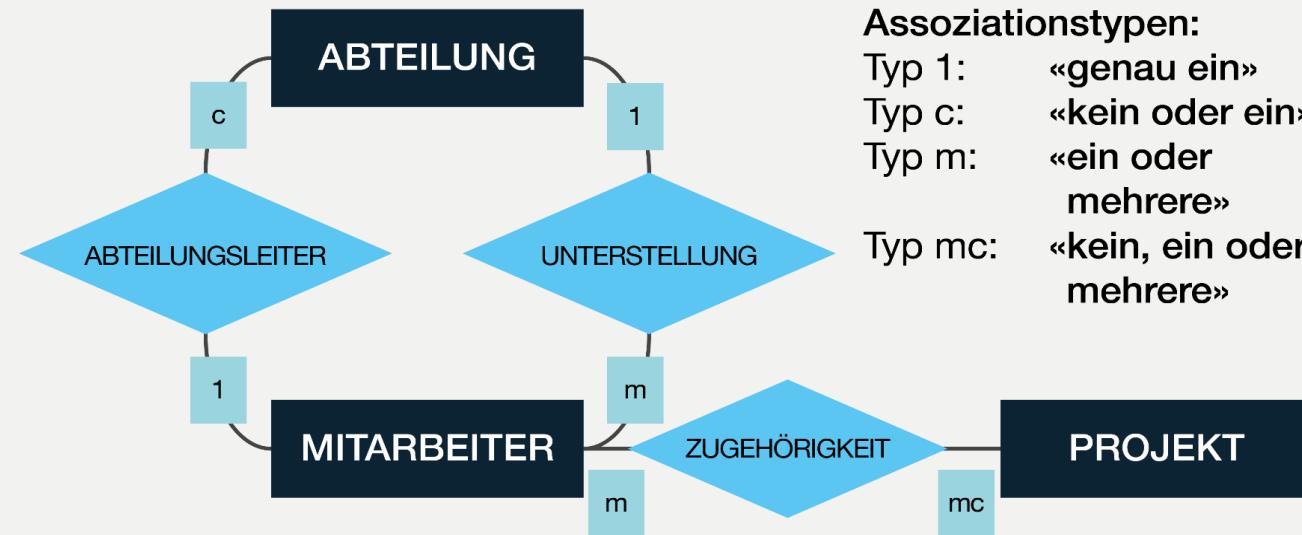
Menge aller Mitarbeiter-Projektzugehörigkeiten mit Merkmalen Mitarbeiternummer, Projekt-nummer und %-Anteil

Identifikationsschlüssel: zusammengesetzter Schlüssel aus Mitarbeiter- und Projektnummer

Darstellung im Entitäten-Beziehungsmodell



Assoziationstypen: Beispiel

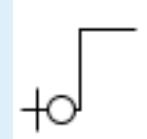
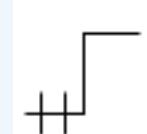
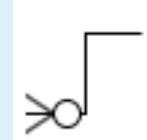
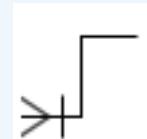


Beispiel zum Abteilungsleiter:

- Typ c: «Jeder Mitarbeiter kann bedingt Abteilungsleiter sein.»
- Typ 1: «Jede Abteilung hat genau einen Abteilungsleiter.»

Assoziationstypen: Notationen

Assoziationstypen (auch Kardinalitäten genannt), sagen, wie oft ein Element der jeweiligen Entitätsmenge in der Beziehung vorkommen kann

Bedeutung	Meier & Kaufmann	UML	Martin
Keins bis eins	c	0..1	
Genau eins	1	1	
Keins bis viele	mc	0..*	
Eins bis viele	m	*	

Der Grad von Beziehungen

$B_j := (A_1, A_2)$ Grad der Beziehungen mit Assoziationstypen A_1 und A_2

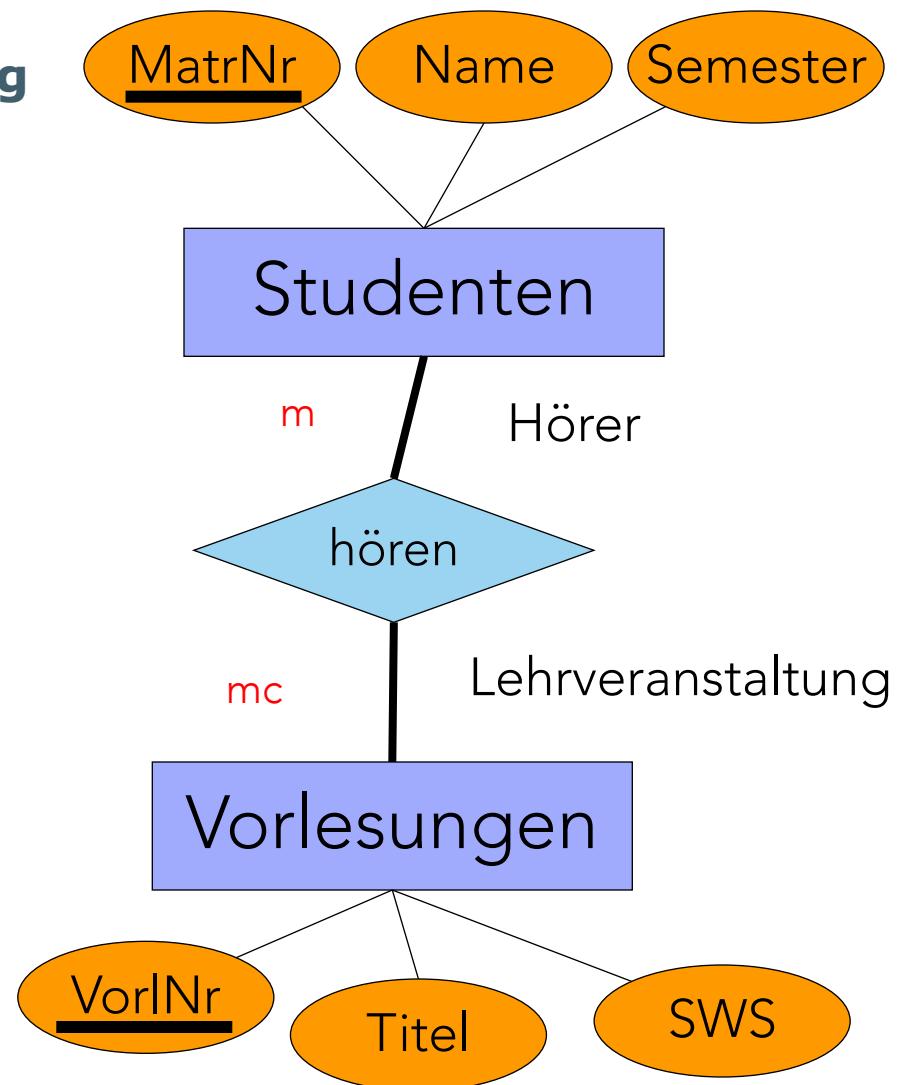
		A1	A2	1	c	m	mc
		1		(1,1)	(1,c)	(1,m)	(1,mc)
		c		(c,1)	(c,c)	(c,m)	(c,mc)
		m		(m,1)	(m,c)	(m,m)	(m,mc)
		mc		(mc,1)	(mc,c)	(mc,m)	(mc,mc)
		B1		B2		B3	

- B1: einfach-einfache Beziehungen
- B2: einfach-komplexe Beziehungen
- B3: komplex-komplexe Beziehungen

Zusammenfassung Entity/Relationship-Modellierung nach Kemper

- Entity Set
(Gegenstandstyp)
- Relationship Set
(Beziehungstyp)
- Assoziationstyp
(Kardinalität)
- Attribut
(Eigenschaft)
- Schlüssel
(Rolle)

Theorie: P. P. Chen (1976).
The entity-relationship
model—toward a unified
view of data. ACM
Transactions on Database
Systems (TODS), 1(1) 9-36



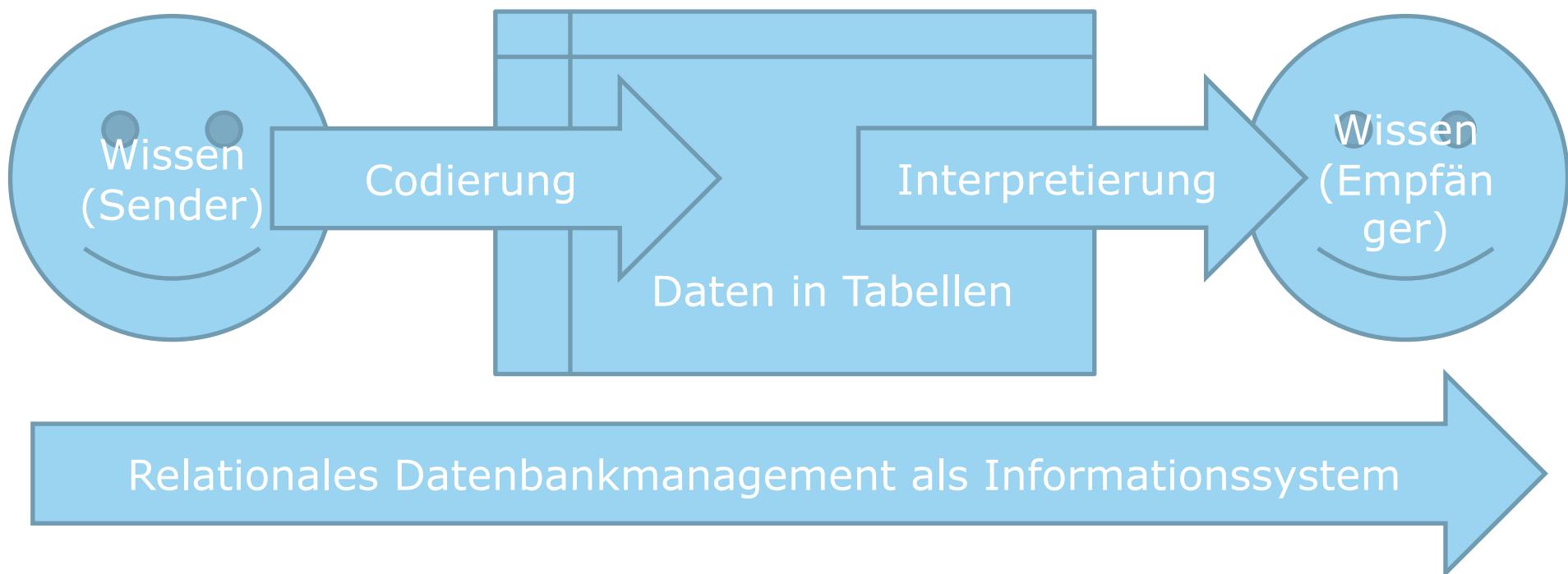


Abbildung von Wissen in Datenbank-Tabellen

Was sind wesentliche Voraussetzungen, das Wissen in Tabellen effektiv gespeichert werden kann?

Was sind die Ziele? Worauf muss besonders geachtet werden?

Tabellendefinition

Unter einer *Tabelle* oder *Relation* verstehen wir eine Menge von Tupeln, die tabellenförmig dargestellt werden.

- **Tabellenname:** Eine Tabelle besitzt einen eindeutigen Tabellennamen.
- **Merkmalsname:** eindeutig; bezeichnet eine bestimmte Spalte mit der gewünschten Eigenschaft.
- **Keine Spaltenordnung:** Die Anzahl der Merkmale ist beliebig, die Ordnung der Spalten ist bedeutungslos.
- **Keine Zeilenordnung:** Die Anzahl der Tupel einer Tabelle ist beliebig, die Ordnung der Tupel ist bedeutungslos.
- **Identifikationsschlüssel:** Eines der Merkmale oder eine Kombination identifiziert eindeutig die Tupel innerhalb der Tabelle und wird als Identifikationsschlüssel deklariert.

MITARBEITER		
M#	Name	Ort
M19	Schweizer	Frenkendorf
M4	Becker	Liestal
M1	Meier	Liestal
M7	Huber	Basel

Identifikationsschlüssel



Ein *Schlüssel* einer Tabelle ist ein Merkmal oder eine minimale Merkmalskombination, welche einzelne Datensätze eindeutig identifizieren.

Schlüsseleigenschaften:

- **Eindeutigkeit:** Jeder Schlüsselwert identifiziert eindeutig einen Datensatz innerhalb der Tabelle, d.h. verschiedene Tupel dürfen keine identischen Schlüssel aufweisen.
- **Minimalität:** Falls der Schlüssel eine Kombination von Merkmalen darstellt, muss diese minimal sein. Mit anderen Worten: Kein Merkmal der Kombination kann gestrichen werden, ohne dass die Eindeutigkeit der Identifikation verlorengeht.

Relationenmodell: Speicherung von Daten in Tabellen

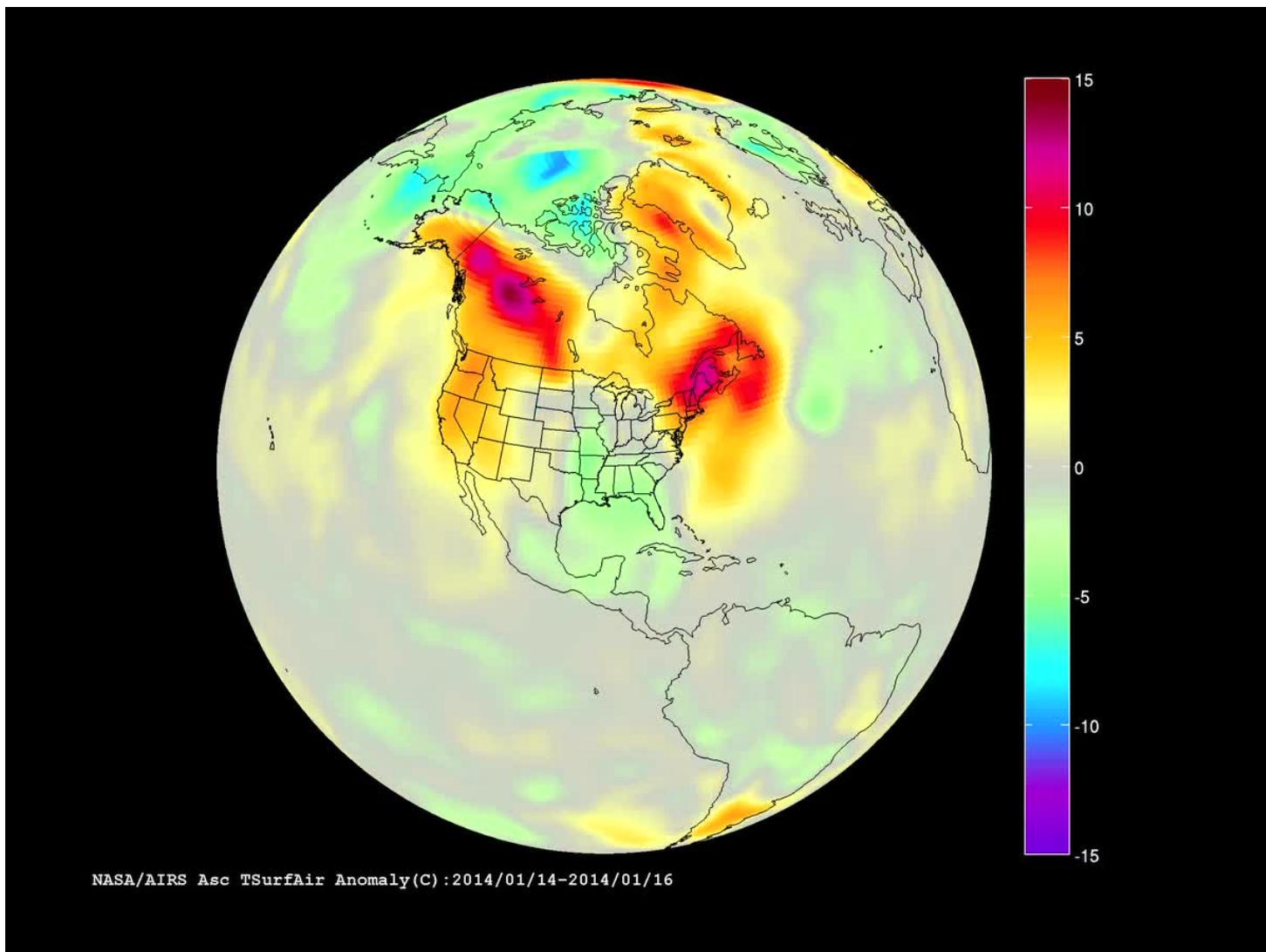
- **Relationale Datenbanken** speichern Daten in Tabellen mit Zeilen und Spalten. Tabellen werden mathematisch als Relationen beschrieben.
- Das Relationenmodell drückt Daten wie Beziehungen zwischen den Daten tabellarisch aus. Mathematisch ist eine **Tabelle** oder **Relation** R nichts anderes als eine *Teilmenge aus dem kartesischen Produkt von Wertebereichen*: $R \subseteq D_1 \times D_2 \times \dots \times D_n$ mit D_i als Wertebereich (engl. *domain*) des i-ten Attributs resp. Merkmals.
- Ein **Tupel** r selber ist demnach eine Menge konkreter Datenwerte resp. Ausprägungen, $t = (d_1, d_2, \dots, d_n)$. Zu beachten ist, dass aufgrund dieses Mengenbegriffs in einer Relation ein und dasselbe Tupel nur einmal vorkommen darf, d.h. $R = \{t_1, t_2, \dots, t_m\}$.
- Das Relationenmodell wurde Anfang der siebziger Jahre durch die Arbeiten von Edgar Frank Codd begründet. Darauf aufbauend, entstanden in Forschungslabors erste *relationale Datenbanksysteme*, die SQL oder ähnliche Datenbanksprachen unterstützten.

TABELLEN IN NORMALFORMEN

Die Normalform der Schlafenszeiten



Was ist eine Anomalie?



Redundanz in Tabellen

- Redundanz in Tabellen kann zu *Anomalien* in Datenbanken führen
- Ein Merkmal ist *redundant*, wenn einzelne Werte dieses Merkmals ohne Informationsverlust weggelassen werden können.
- Normalformen dienen der Vermeidung von Redundanz.
- Anomalien:
 - Einfügeanomalie, Mutationsanomalie, Löschanomalie

Frage: Was
ist hier
Redundant?

ABTEILUNGSMITARBEITER

M#	Name	Strasse	Ort	A#	Bezeichnung
M19	Schweizer	Hauptstrasse	Frenkendorf	A6	Finanz
M1	Meier	Lindenstrasse	Liestal	A3	Informatik
M7	Huber	Mattenweg	Basel	A5	Personal
M4	Becker	Wasserweg	Liestal	A6	Finanz

Normalformen



<https://openclipart.org/detail/17170/>

Erste Normalform (1NF)

Zweite Normalform (2NF)

Dritte Normalform (3NF)

Boyce-Codd-Normalform (BCNF)

Vierte Normalform (4NF)

Fünfte Normalform (5NF)

Nur triviale Verbund-abhängigkeit

Keine Mehrwertabhängigkeiten

Nur Abhängigkeiten vom Schlüssel zugelassen

Es bestehen keine transitiven Abhängigkeiten

Nichtschlüsselmerkmale sind voll vom Schlüssel abhängig

Alle Merkmalswerte sind atomar (keine Wiederholungsgruppen zugelassen)

Tabelle in beliebiger (unnormalisierter) Form

Die erste Normalform (1NF)

- => Atomare Daten in den Tabellenzellen
- Keine Arrays, Mengen, Aufzählungen, Listen etc. als Merkmalswerte (im Kontrast z.B. zu OOP)

PROJEKTMITARBEITER (unnormalisiert)

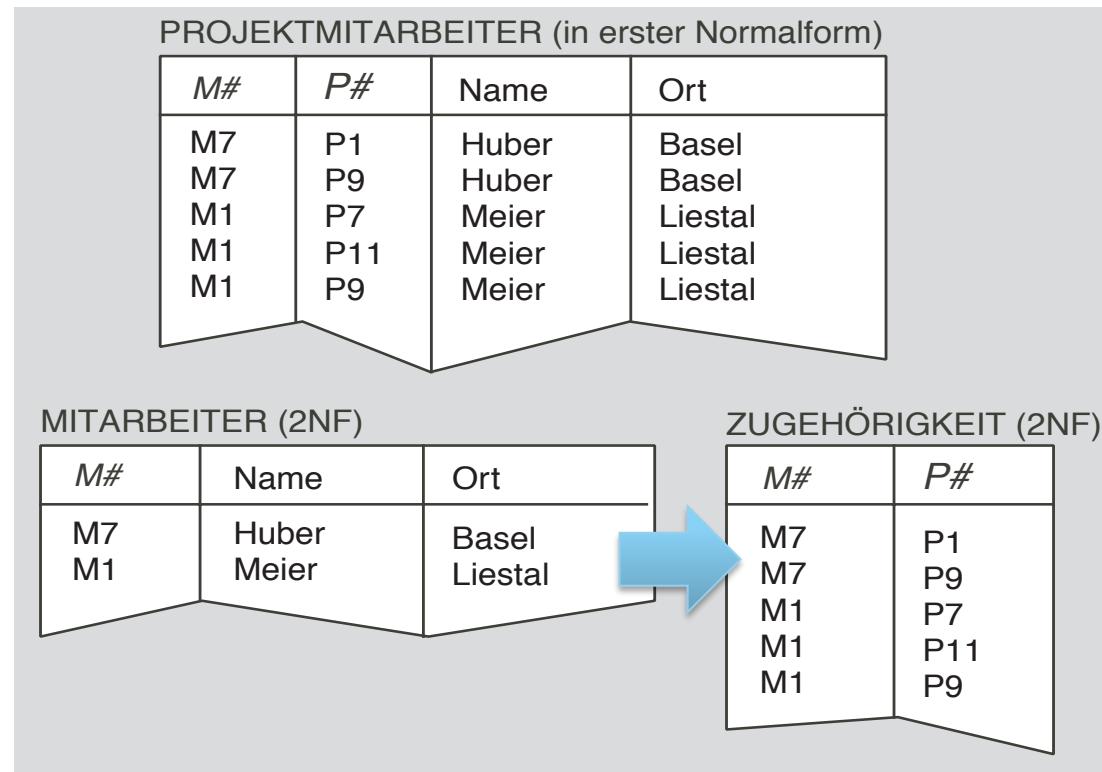
M#	Name	Ort	P#
M7	Huber	Basel	{ P1, P9 }
M1	Meier	Liestal	{ P7, P11, P9 }

PROJEKTMITARBEITER (in erster Normalform)

M#	P#	Name	Ort
M7	P1	Huber	Basel
M7	P9	Huber	Basel
M1	P7	Meier	Liestal
M1	P11	Meier	Liestal
M1	P9	Meier	Liestal

Die zweite Normalform (2NF)

- 1.) Erste Normalform ok
- 2.) Alle Nicht-schlüsselmerkmale sind voll funktional abhängig von den Schlüsselmerkmalen.
- **Funktionale Abhängigkeit A**
→ B: Jeder Wert von A bestimmt eindeutig den Wert von B
- Wenn A ein zusammengesetzter Schlüssel ist: **Volle funktionale Abhängigkeit** heisst, dass B nur von A, aber nicht von Untermengen von A (Teilschlüssel) abhängig ist



Die dritte Normalform (3NF)

- 1.) 1NF ok.
- 2.) 2NF ok.
- 3.) Die Tabelle enthält keine transitiven Abhängigkeiten.

- Transitive Abhängigkeit

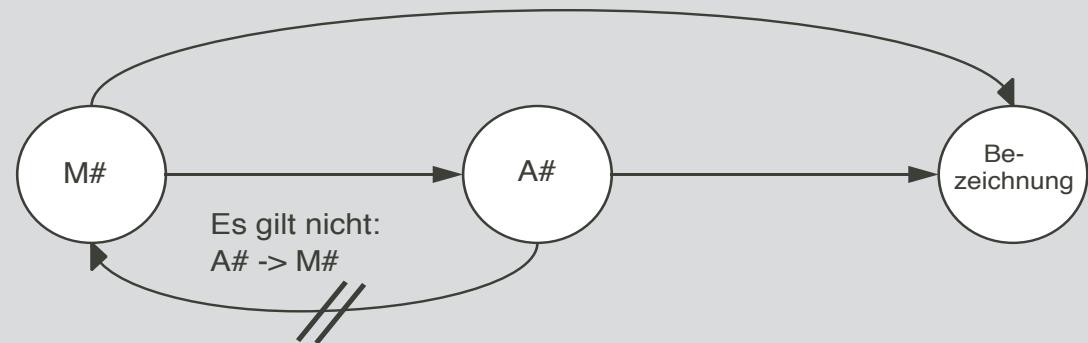
$A \rightarrow B \rightarrow C$:

$A \rightarrow B$, $B \rightarrow C$; aber nicht $B \rightarrow A$

ABTEILUNGSMITARBEITER (in zweiter Normalform)

M#	Name	Strasse	Ort	A#	Bezeichnung
M19	Schweizer	Hauptstrasse	Frenkendorf	A6	Finanz
M1	Meier	Lindenstrasse	Liestal	A3	Informatik
M7	Huber	Mattenweg	Basel	A5	Personal
M4	Becker	Wasserweg	Liestal	A6	Finanz

transitive Abhängigkeit:



MITARBEITER (in dritter Normalform)

M#	Name	Strasse	Ort	A#_Unt	A#	Bezeichnung
M19	Schweizer	Hauptstrasse	Frenkend.	A6	A3	Informatik
M1	Meier	Lindenstrasse	Liestal	A3	A5	Personal
M7	Huber	Mattenweg	Basel	A5	A6	Finanz
M4	Becker	Wasserweg	Liestal	A6		

ABTEILUNG (3NF)

RELATIONALE DATENBANKSCHEMAS

Vom Datenmodell zum Datenbankschema

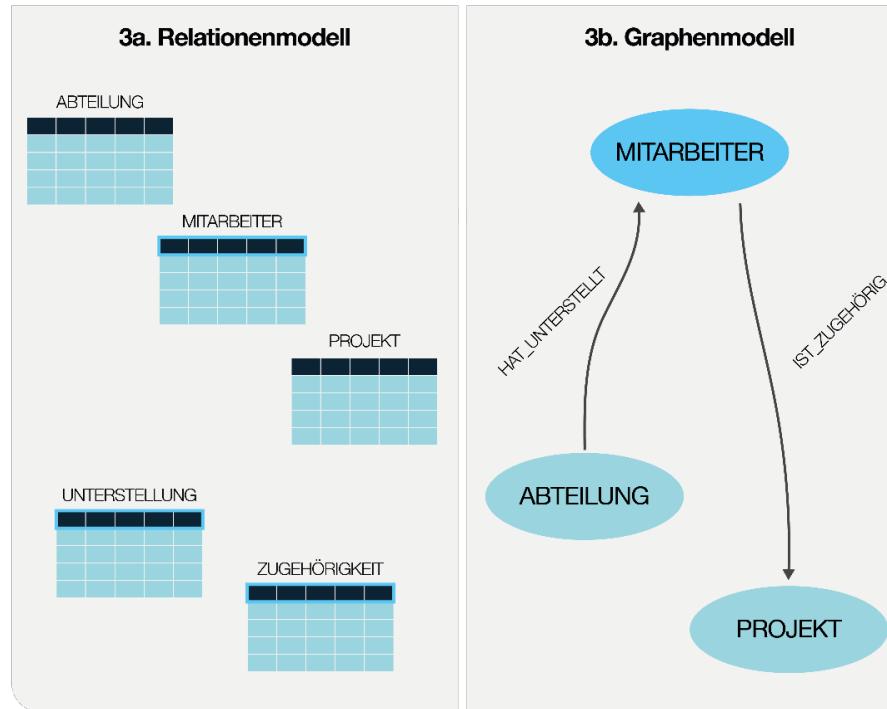
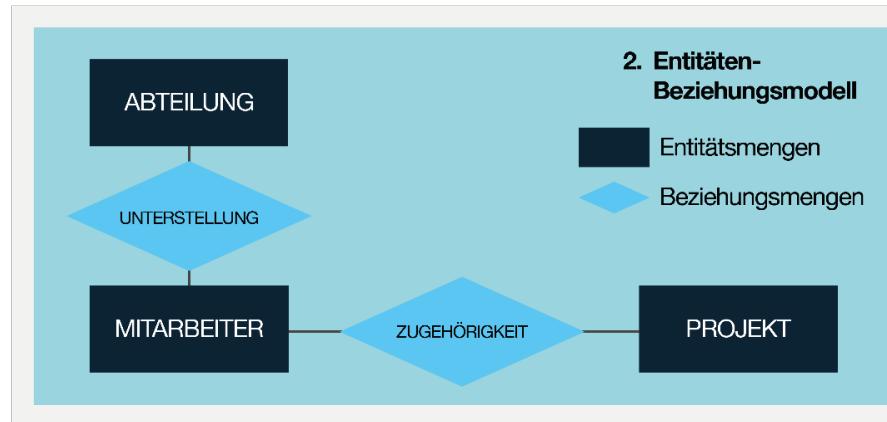
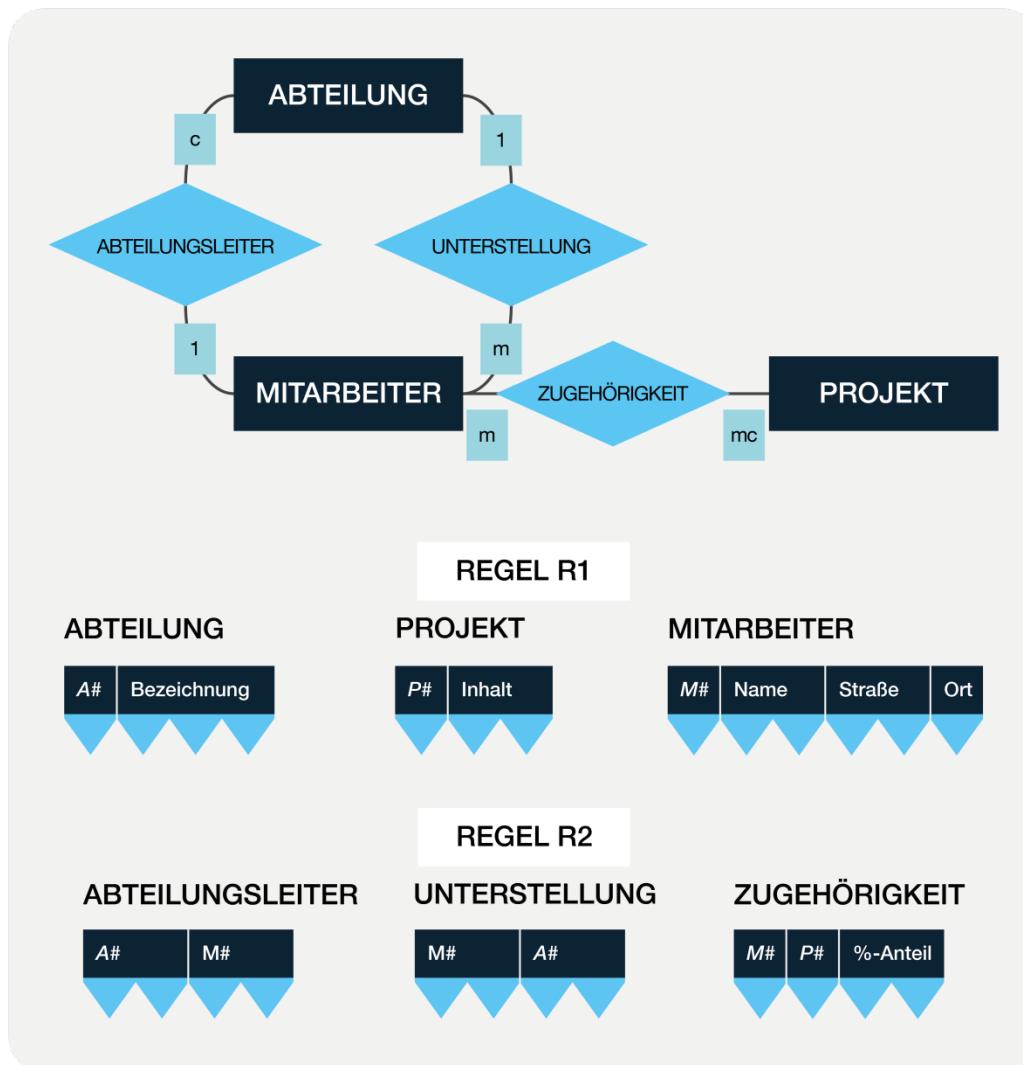


Abbildung von Entitäts- und Beziehungsmengen in Tabellen



-- SQL Code:
CREATE TABLE Abteilung (A# INTEGER,
 Bezeichnung VARCHAR (30))

Umsetzung von ER-Diagrammen als Relationale Schemas

- Ein *Schema* gibt Struktur für Daten und Datenbeziehungen vor.
- Im Unterschied zu einem ER-Modell besteht ein *relationales Schema* nicht aus Entitäten und Beziehungen, sondern nur aus Tabellen (Relationen) mit entsprechenden Spalten.
- Für die Abbildung von Beziehungen zwischen Tabellen müssen diese mit *Schlüsselbeziehungen* referenziert werden.

Patentrezept

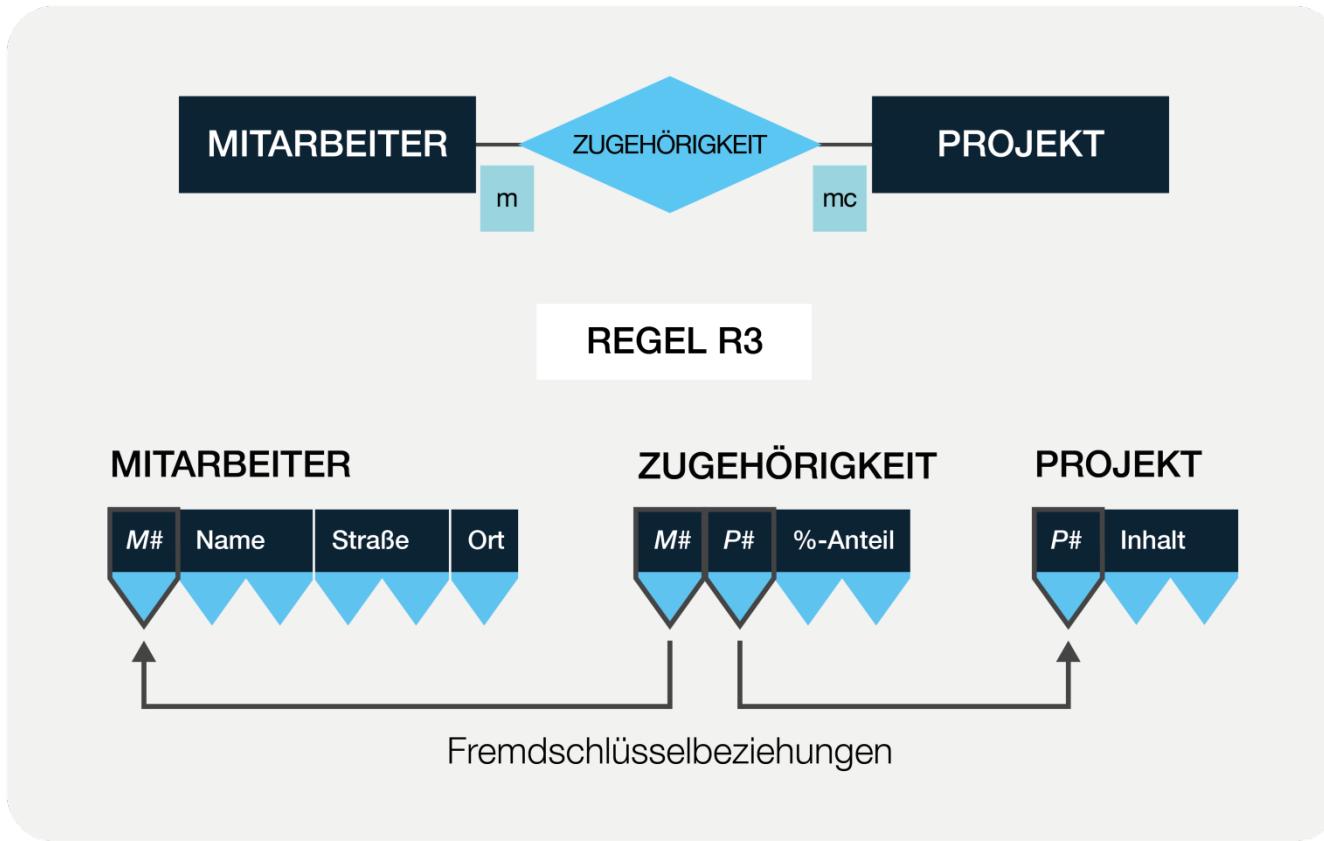
- 1 Entitätsmenge = 1 Tabelle
- 1 Attribut = 1 Spalte
- 1 Beziehungsmenge = 1 Tabelle (oder 1 Fremdschüssel falls eindeutig)

Primärschlüssel und Fremdschlüssel

- Ein **Schlüssel** ist ein Merkmal oder eine minimale Merkmalskombination, welche einen Datensatz eindeutig identifizieren.
- Der Schlüssel kann aus mehreren Attributen zusammengesetzt sein
- Es kann mehrere Schlüssel für eine Tabelle geben.
- Einer der Kandidatenschlüssel wird als Hauptschlüssel definiert und wird als **Primärschlüssel** (Primary Key, PK) bezeichnet.
- Eine Tabelle kann Zeilen in einer anderen Tabelle referenzieren. Ein solches referenzierendes Merkmal heisst **Fremdschlüssel** (Foreign Key, FK).

```
-- SQL Code:  
CREATE TABLE Abteilung (  
    Aid INTEGER PRIMARY KEY,  
    Bezeichnung VARCHAR(30) )
```

Abbildungsregel für komplex-komplexe Beziehungsmengen



-- SQL Code:

```
CREATE TABLE Zugehörigkeit (
    M# INTEGER REFERENCES Mitarbeiter(Mid),
    P# INTEGER REFERENCES Projekt(Pid),
    Prozentanteil DECIMAL(0,1),
    PRIMARY KEY (Mid, Pid) )
```

Abbildungsregel für einfach-komplexe Beziehungsmengen



REGEL R4

ABTEILUNG

A#	Bezeichnung

MITARBEITER

M#	Name	Ort	A#_Unterst.



Fremdschlüsselbeziehung

-- SQL Code:

```
CREATE TABLE Mitarbeiter (
    Mid INTEGER PRIMARY KEY,
    Name VARCHAR(30) NOT NULL,
    Ort VARCHAR(30),
    Aid_Unt INTEGER REFERENCES Abteilung(Aid) )
```

Abbildungsregel für einfach-einfache Beziehungsmengen



REGEL R5

ABTEILUNG

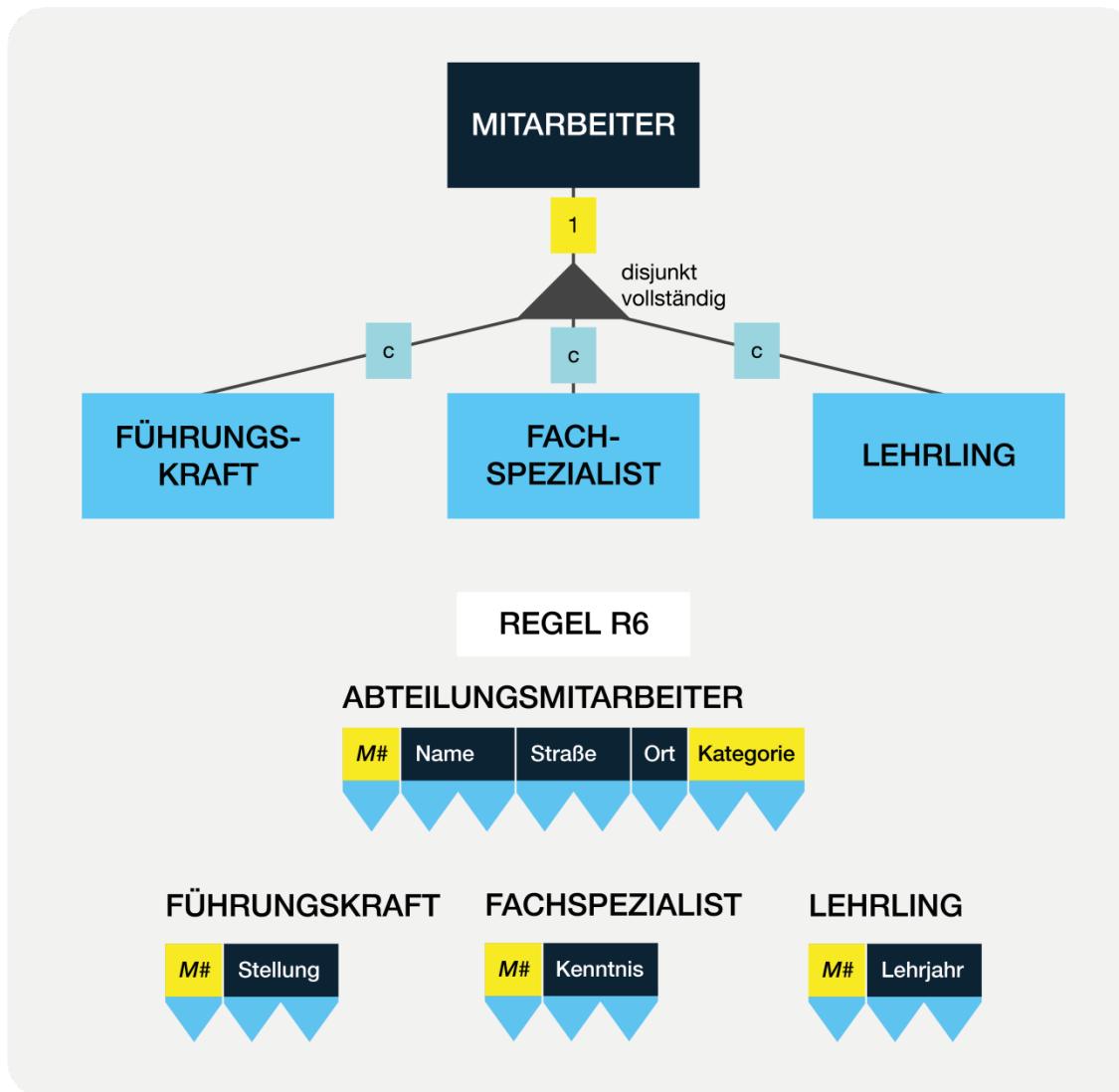
A#	Bezieh.	M#_Abt.leiter

MITARBEITER

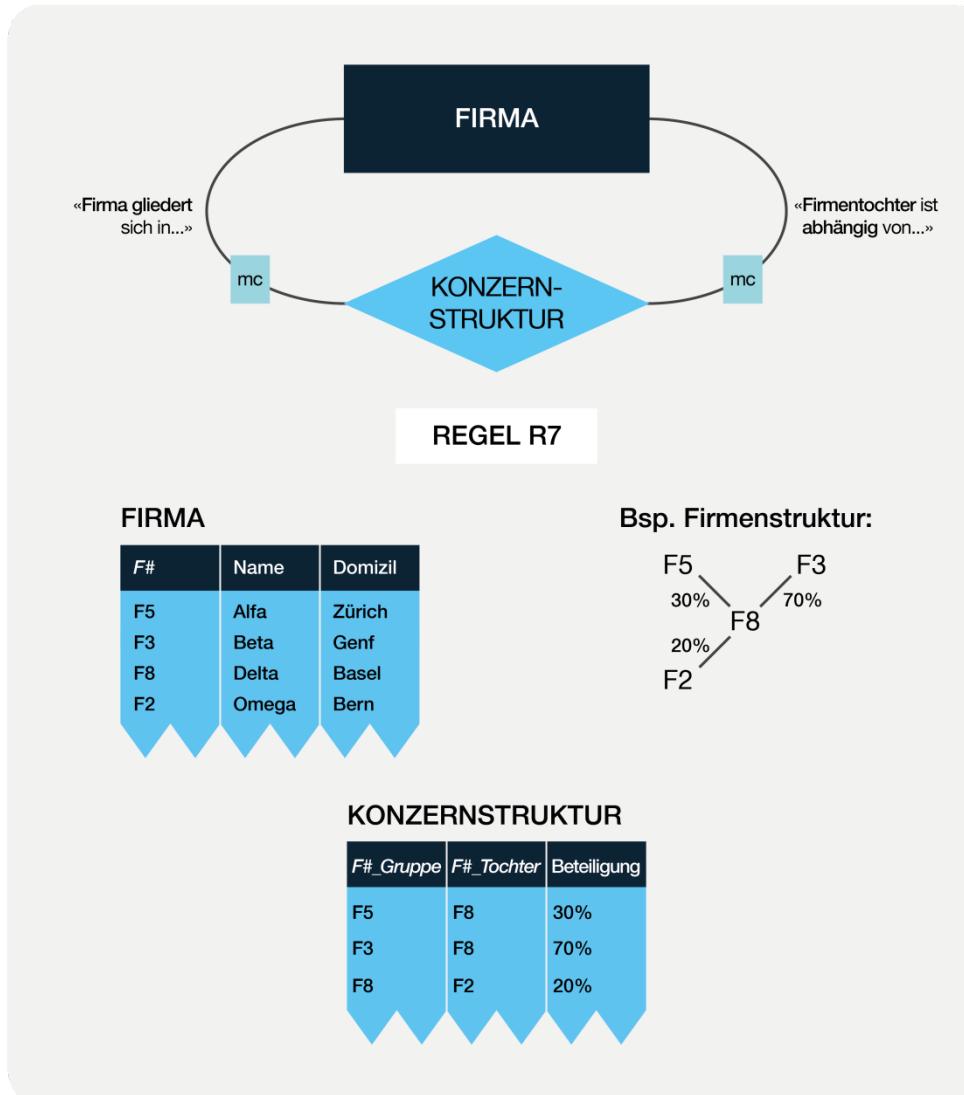
M#	Name	Ort

Fremdschlüsselbeziehung

Generalisation in Tabellenform



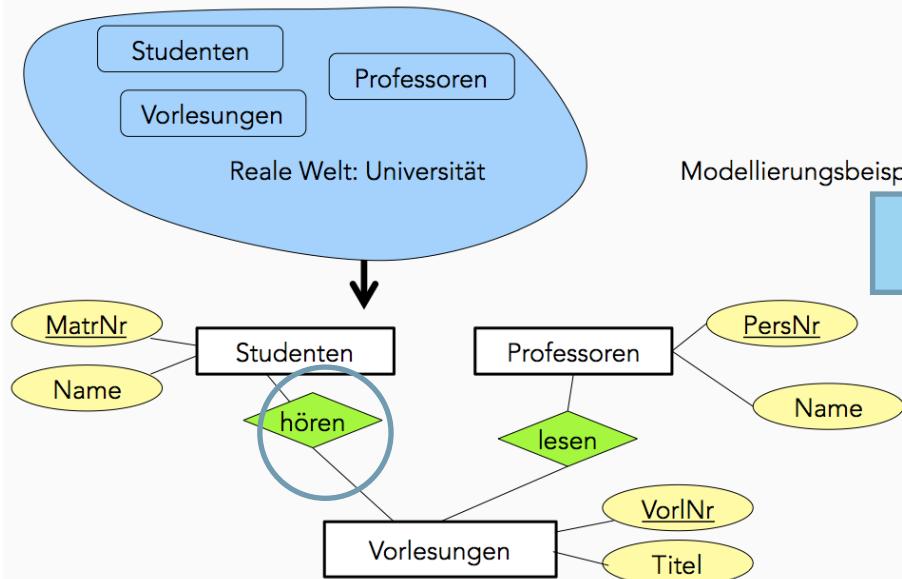
Netzwerkartige Firmenstruktur in Tabellenform



Relationship (Beziehungsmenge) vs. Relation (Tabelle) in RDBMS

Technisch (RDBMS)

Konzeptionell



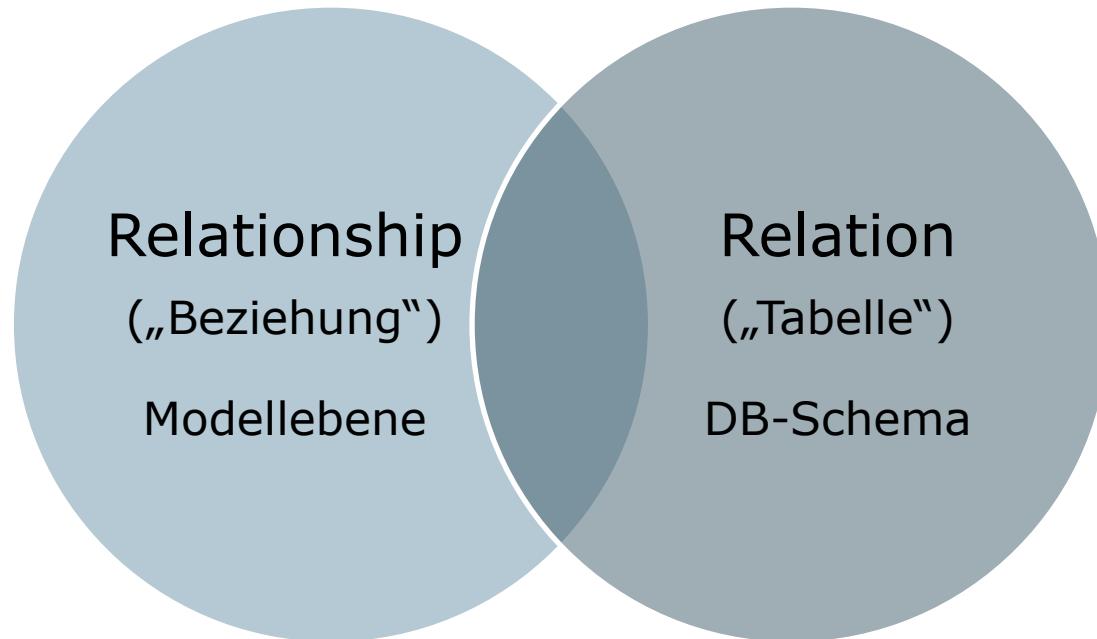
Modellierungsbeispiel

Abbildung

hören	
MatrNr	VorlNr
26120	5001
27550	5001
27550	4052
28106	5041
28106	5052
28106	5216
28106	5259
29120	5001
29120	5041
29120	5049
29555	5022
25403	5022

Eine Beziehung (engl. relationship) ist eine Assoziiierung zwischen Entitäten. Sie geben den Entitäten Bedeutung und Kontext.

Eine Tabelle oder Relation R ist eine *Teilmenge aus dem kartesischen Produkt von Wertebereichen*: $R \subseteq D_1 \times D_2 \times \dots \times D_n$ mit D_i als Wertebereich (engl. domain) des i -ten Attributs.



Interaktion

Bei der Abbildung konzeptioneller ER-Modelle in Tabellen werden einige, aber nicht alle Relationships als Relationen abgebildet. Fragen:

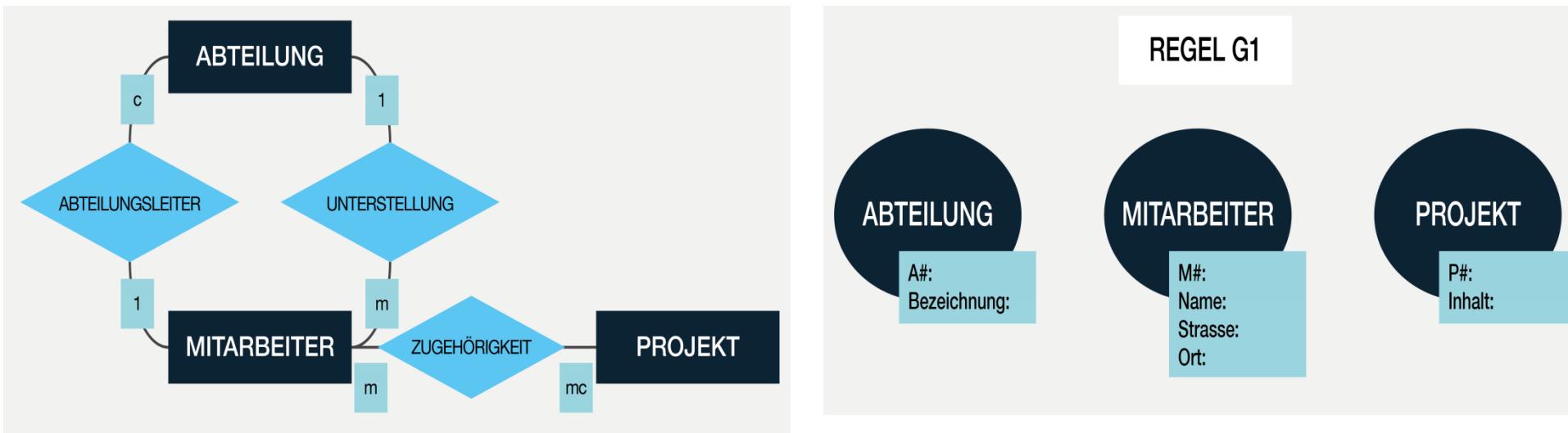
- * Welche Relationships werden als Relationen gespeichert?
- * Welche nicht?
- * Welche Relationen speichern keine Relationships?

VOM ER-MODELL ZUM DB-SCHEMA BEI NOSQL

Abbildungsregeln für Graphdatenbanken

- **Regel G1 (Entitätsmengen)**

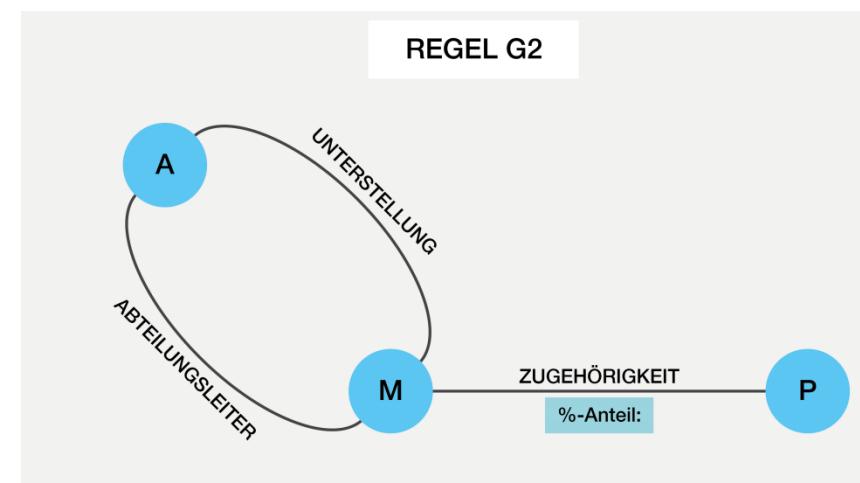
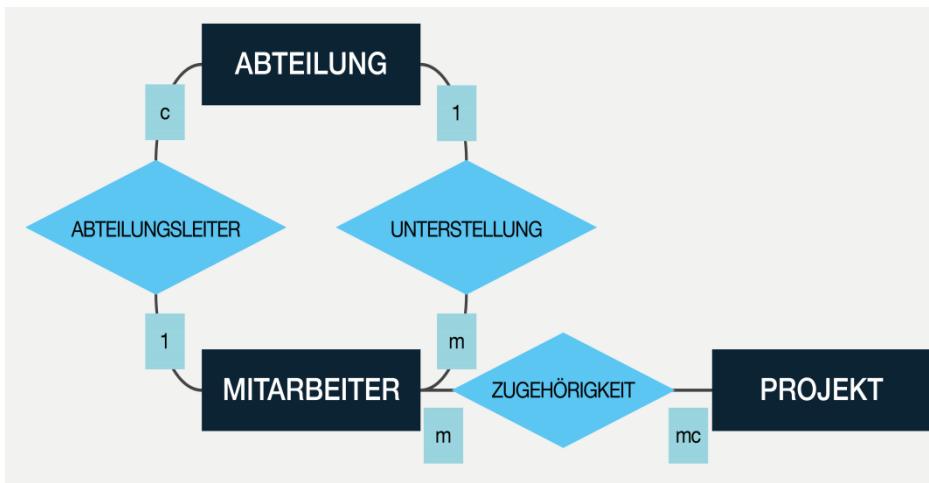
Jede Entitätsmenge muss als eigenständiger Knotentyp in der Graphdatenbank definiert werden. Die Merkmale der Entitäten werden als Eigenschaften der Knoten geführt.



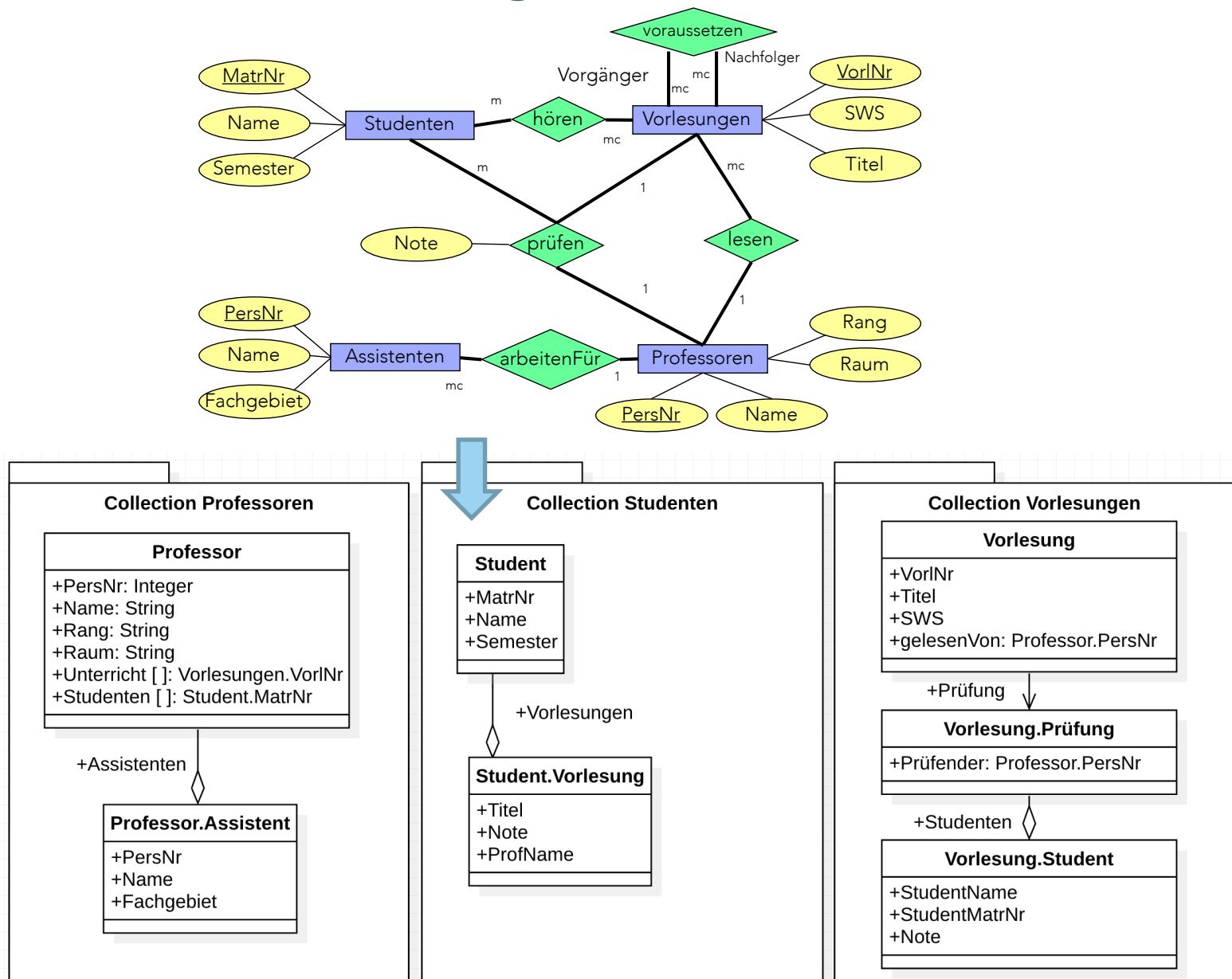
Abbildungsregeln für Graphdatenbanken

- **Regel G2 (Beziehungsmengen)**

Jede Beziehungsmenge kann als Kantentyp in der Graphdatenbank definiert werden. Eigenschaften der Beziehungsmengen werden den Kanten zugeordnet (attributierte Kanten).



Dokument-DB: Vom ER-Diagramm zum Dokumentmodell in UML



Von UML zu JSON-Strukturen für Dokumentdatenbanken



Variante für JSON Modell-Dokumentation oder Validierung: JSON Schema

<https://json-schema.org/learn/miscellaneous-examples.html>

{

```
{  
  "type": "object",  
  "properties": {  
    "MatrNr": { "type": "integer" },  
    "Name": { "type": "string" },  
    "Semester": { "type": "integer" },  
    "Vorlesungen": {  
      "type": "array",  
      "items": {  
        "type": "object",  
        "properties": {  
          "Titel": { "type": "string" },  
          "Note": { "type": "integer" },  
          "ProfName": { "type": "string", }  
        } } } } }
```

Instance

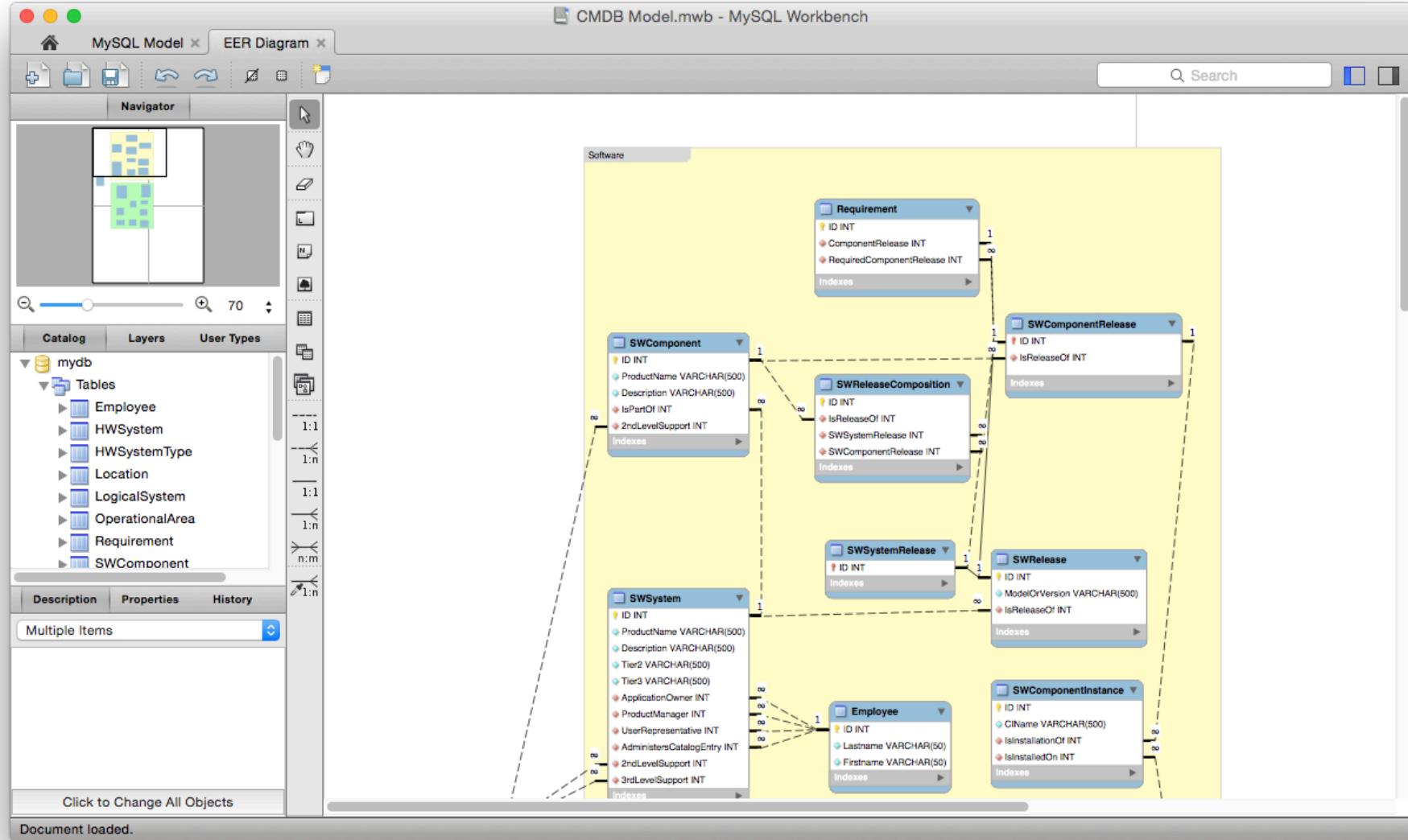
```
{  
  "MatrNr": 0,  
  "Name": "",  
  "Semester": 0,  
  "Vorlesungen": [  
    {  
      "Titel": "",  
      "Note": 0.0,  
      "ProfName": ""  
    } ]  
}
```

Validator: <https://www.jsonschemavalidator.net>

Schema Inference: <https://jsonschema.net>

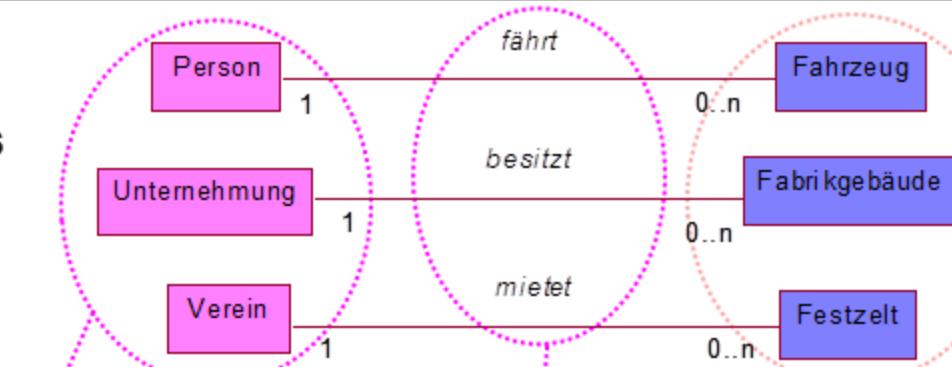
PRAXISBEISPIELE

Software-Unterstützung für die Datenmodellierung: MySQL Workbench



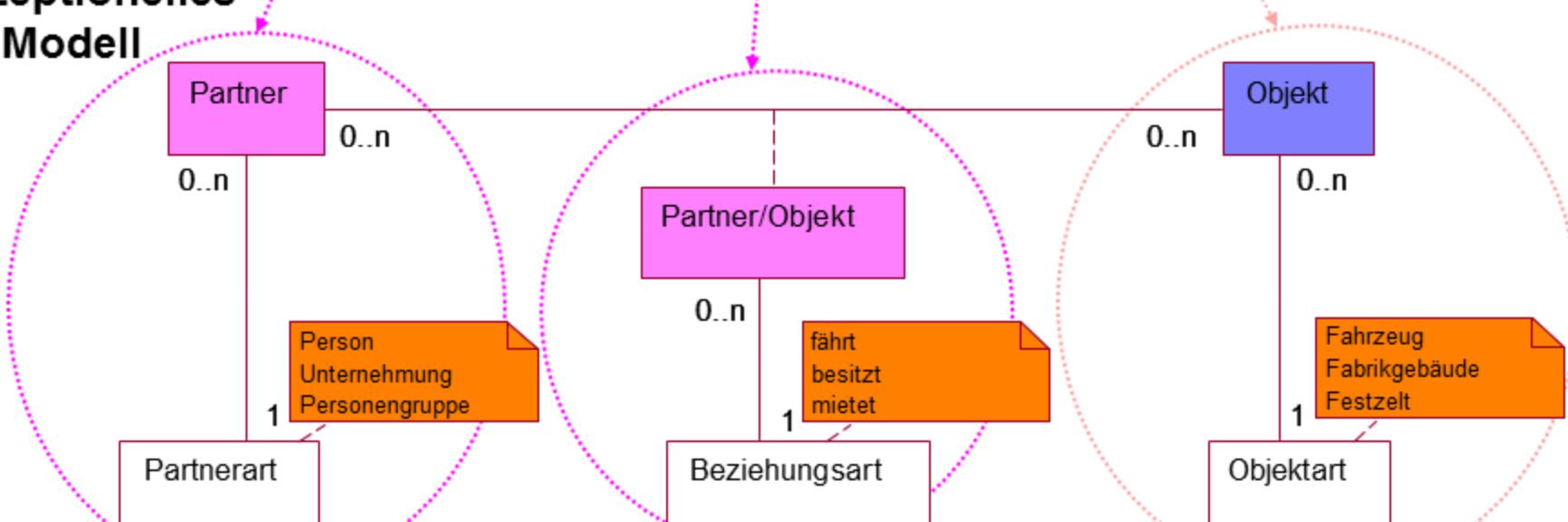
Vom SKM zum GKM

**spezialisiertes
konzeptionelles
Modell**



**generalisiertes
konzeptionelles
Modell**

generalisieren



Die Mobiliar

Versicherungen & Vorsorge

IBM Industry Models For Insurance - The Insurance Application Architecture (IAA)



Insurance Application Architecture (IAA)

What is the Insurance Application Architecture (IAA)?

The IAA is a set of information, process and integration models that represent leading practice systems development in the insurance industry. It is an architecture blueprint with detailed insurance business content that can be applied to initiatives on an enterprise level, or at a local level, to support insurance companies to define detailed specifications and cross-enterprise architectures for information systems. These models represent over 300 cumulative perspectives and viewpoints, incorporating input from many leading insurance organizations.

By providing a set of pre-defined business templates, IAA enables the scoping, specification and design of information solutions, which are:

- faster, through use of generic model specifications and designs
- cost effective, through reduced analysis costs and increased re-use of existing assets
- better, through increased quality and consistency
- lower risk, by building on good practice and by ensuring a strategic perspective

The IAA provides a common language for use across the organization. It includes a core set of models for insurance business requirements and core application function analysis, and design requirements. The models are designed to be readily customized and extended to cover the specific requirements of a insurance organization.

- business architecture and management
- enterprise application integration
- message-based architecture development
- business solution re-use
- information system modeling and requirements definition
- application systems package evaluation



IBM Component Business Model (CBM)

Insurance organizations are adopting a more compartmentalized approach to their businesses, resulting substantial changes in the effectiveness, efficiency and flexibility of their organizations, driving sustainable growth.

A compartmentalized approach:

- allows the organization to be more adaptive and to respond quickly to changing customer needs
- enables the organization to focus on achieving competitive differentiation
- optimizes interactions with partners, suppliers and customers
- allows the organization to identify and leverage best practice in different parts of the organization
- presents opportunity to shift the organization from a fixed to a flexible cost structure

The Component Business Model (CBM) is an organizing framework combining people, process, and technology perspectives that drives significant value creation and competitive differentiation within the organization. The CBM is a logical representation, or map, of a business that reveals its essential building blocks. A business component can be defined as a collection of the business activities it performs and its supporting people and systems requirements.

The Component Business Model (CBM) can be populated with IAA content thereby translating the CBM component to solutions. These three areas supported by the IAA are business intelligence, enterprise application integration and business process reengineering. IAA provides pre-packaged industry models that ensure that reuse rate of the high level function components is 80%.

Process & Integration

CBM Business Components, representing functional areas of a business, are underpinned by a number of IAA Business Solution Templates. Each IAA Business Solution Template can then be mapped to one or more IAA Foundation Models to create an Insurance Information Warehouse Model. The data model can then be used as the basis for the generation of databases (DW, or OLAP structures).

Master Data Management

IBM Master Data Management is SOA-based middleware designed to provide organizations the most flexible framework to support enterprise strategy and transactional and business services aligned with key business processes. IBM leverages technologies and components required for a successful enterprise MDM strategy: information integration, content management, business intelligence, and master data management. It can be used to support core business activities, such as sales and customer service – and master data solutions for specific industries.

WebSphere Customer Center (WCC) provides real-time, transactional customer data integration (CDI). Despite the significant time and resources already invested in CRM, many organizations still lack a true enterprise-wide view of their customer base. By creating a single customer view systems to an enterprise customer hub, WCC provides a unified view of the customer across multiple business and product lines. It delivers this single view and integrates with existing CRM and CPM applications, enabling multi-channel integration and consistent customer service.

Insurance Application Architecture (IAA)

IAA Foundation Models - for Communication and Standardization

Business Terms improving standardization

Clearly defined Business Terms improve standardization. IAA provides a catalogue of over 2500 terms grouped in domains, providing an easy entry point to the models. These business terms are also mapped to the Business Model.

Business Model describing business concepts

The IAA Business Model describes the business concepts relevant to the financial services industry and how these concepts relate to each other. There is a single business model in two respects. The Business Data Model is an entity model that maps the business objects to the data model. Using one or more representations for the Business Model is a question of personal preference since the focus at this level is on the business concepts. The main purpose of the Business Model is to serve as a communication mechanism between business and IT. It also provides a formalized view of the business to be used as a reference at any time in the life-cycle of a development project.

Business Activity Model organization functions

Functional requirements are expressed in the Business Activity Model as business activities. Elementary units of work that need to be performed as part of the business operations. It is possible to organize the activities differently, depending on the line of business or according to organizational principles.

IAA Information Models for Data Rationalization, Data Warehousing and Data Marts

Examples of the Business Scope of the Information Models

Risk/CRM	Claims	Policy	Business Performance	Sales/Order
Campaign lifetime analysis	Loss Efficiency	Underwriting	Business volume analysis	Corporated Financial Statements Analysis
Campaign communication analysis	Handling performance analysis	Policy event analysis	Forecasting/recovery parameter analysis	Corporated Statement Of Cash Flows
Campaign profitability analysis	Handling recovery analysis	Statistical analysis	Policy claim analysis	Corporated Statement Of Changes
Campaign sales analysis	Policy close off analysis	Surveillance analysis	Summarizer analysis	Corporated Balance Sheet Analysis
Campaigns driven by customer	Loss even analysis	Switching analysis	Underwriting analysis	Corporated Statement Of Income Analysis
CRM even analysis				
Gross-sell strategy analysis				
Group policy analysis				
Individual policy analysis				
Policyholder behaviour analysis				
Hourly volume analysis				
Intermediary communication analysis				
Customer retention analysis				
Customer exit/break analysis				
Customer risk analysis				

Business Solution Templates analyzing data for KPIs and reports

The Business Solution Templates (BSTs) provide a list of industry best-practice key performance indicators grouped by functional reporting and reporting requirements. These provide the basis for rapid customization and prototyping of customer reporting requirements. A range of OLAP reporting engines are available to BSs and provide a consistent reporting structure for every data mart produced.

Marts & Sample Applications ready-to-go reports

The IIA solution includes a number of data marts and sample applications which can be used to accelerate a data warehouse project. Each one is designed to focus on a particular business problem in contrast to the Enterprise Model which attempts to address all business problems. Available Samples include Campaign Management, Underwriting Profitability Analysis, Risk Pricing Analysis, Intermediary Performance Analytics and Financial Reporting.

Enterprise Model designing the central data warehouse

The Enterprise Model is a logical model representing the information of the data warehouse and is derived from the IAA Business Model. It forms the strategic blueprint for implementing or re-engineering a data warehouse on a project by project basis. The model allows warehouse interfaces to be scoped, analyzed and designed in such a way that each project builds on the results of previous projects. The model includes content to cover profitability, analytical CRM, financial reporting, and risk management.

DB2. Information Management Software



IAA Process & Integration Models for Business Process Modeling, Application Rationalization, Integration, Component Based Development, and SOA

Examples of the Business Scope of the Process & Integration Models

Channel management	Complaint plan establishment	Claim handling	Product portfolio management	Investment analysis
Channel agent	Complaint plan establishment	Account and mandate management	Product development	Investment monitoring and reporting
Internal agent	Internal agent	Policy management	Group scheme development	Investment operation
Intermediary agent	Intermediary agent	Agreement management	Product line planning	Investment strategy
Intermediary communication	Intermediary communication	Intermediary communication management	Product lifecycle management	
Intermediary contract management		Policy modification	Product performance management	
Intermediary performance management		Provider management	Provider management	

Enterprise Component Blueprint represent business functions as services

The Enterprise Component Blueprint addresses the definition of business functions and components required to support the business. This is achieved by defining the business functions and components in a consistent way. The ESB provides a central component for each business function. The ESB is a logical model that defines the functional behavior of the business functions. The ESB is a logical model that defines the functional behavior of the business functions.

Interface Design Model designing services & components

The Interface Design Model defines a standard set of interface definitions that promote the development of interoperable financial services software. It extends the applicability of the IAA offerings to the design phase of component-based development and integration projects. It also defines a set of highly reusable components for financial services software industry by applying the interface design methodology to the design of the IAA offerings. The IDM is a logical model that represents the IAA Object Model. The IDM takes the analysis level concepts into the design domain, refining them to form components and interfaces. The IDM addresses topics such as application portlet rationalization and application integration. The IDM can be further transformed, into detailed information sources, into native Java Bean Model (JBM), so that code can be generated using IBM Rational Software tools.

WebSphere. software

WebSphere. software

Application Components



Rational. software

Rational. software

Legacy / B2B Interactions

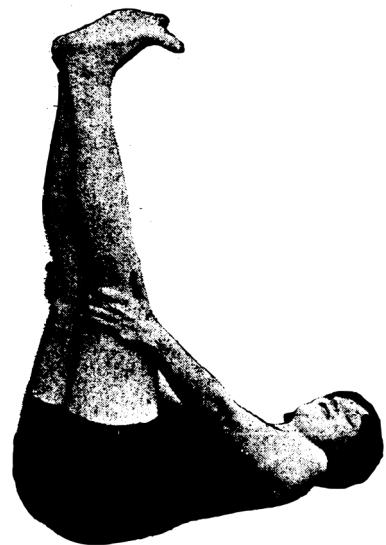


Lernziel: Datenbankmodellierung

1. Aufgrund einer beschriebenen Ausgangslage Entitätsmengen, Beziehungsmengen (inkl. Assoziation, Assoziationstyp und Grad und Merkmale (inkl. Identifikationsschlüssel) erkennen und diese in einem Entity-Relationship Modell darstellen
2. Schüssel und funktionale Abhängigkeit erkennen und für die Herleitung der Normalformen 1 2 anwenden
3. Transitive Abhängigkeit erkennen und für die Herleitung der Normalform 3 anwenden
4. Entitätsmengen, Beziehungsmengen, Attribute, Assoziationen und Assoziationstypen in ein relationales Datenbank Schema mit Tabellen umsetzen, inkl. Primär- und Fremdschlüssel







ÜBUNG: DATENBANK- MODELLIERUNG