Datenorganisation und Datenmodellierung

JProf. Dr. Gunnar Stevens

Human Computer Interaction
University of Siegen

gunnar.stevens@uni-siegen.de

Agenda

- Grundbegriffe
- Datenbanksysteme
  - Datenbank Aufbau
  - Datenbankmanagement Systeme (DBMS)
- Datenmodelle
  - Hierarchisches
  - Netzwerk
  - Relationales
- Datenmodellierung
  - Entity Relationship Modell

JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

GRUNDBEGRIFFE

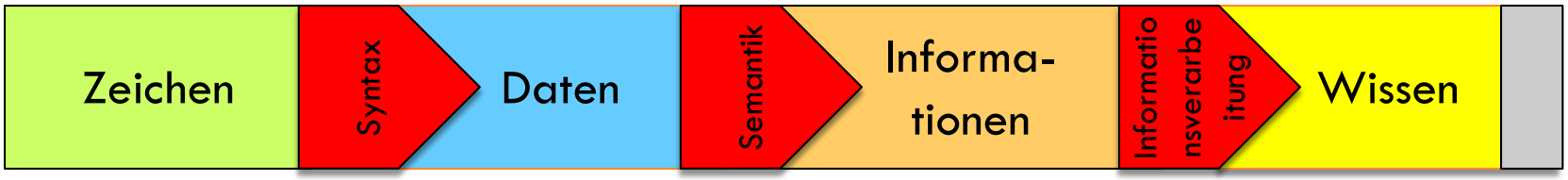
Definition Datenorganisation

- Datenorganisation ist eine Zusammenfassung von Verfahren zur
  - Strukturierung der Daten (logische Datenorganisation)
  - Datenablage auf Speichermedien und Bereithaltung für den Zugriff (physische Datenorganisation)

JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

Grundlegende Begriffe I

- Daten
  - Informationen, die weiterverarbeitet werden.
- Information
  - Eine Mitteilung (Nachricht), die beim Empfänger eine Unbestimmtheit beseitigt
- Wissen
  - Aus Daten gewonnene Information
  - Im jeweiligen Kontext interpretierte Information
  - Verarbeitung dieser Information

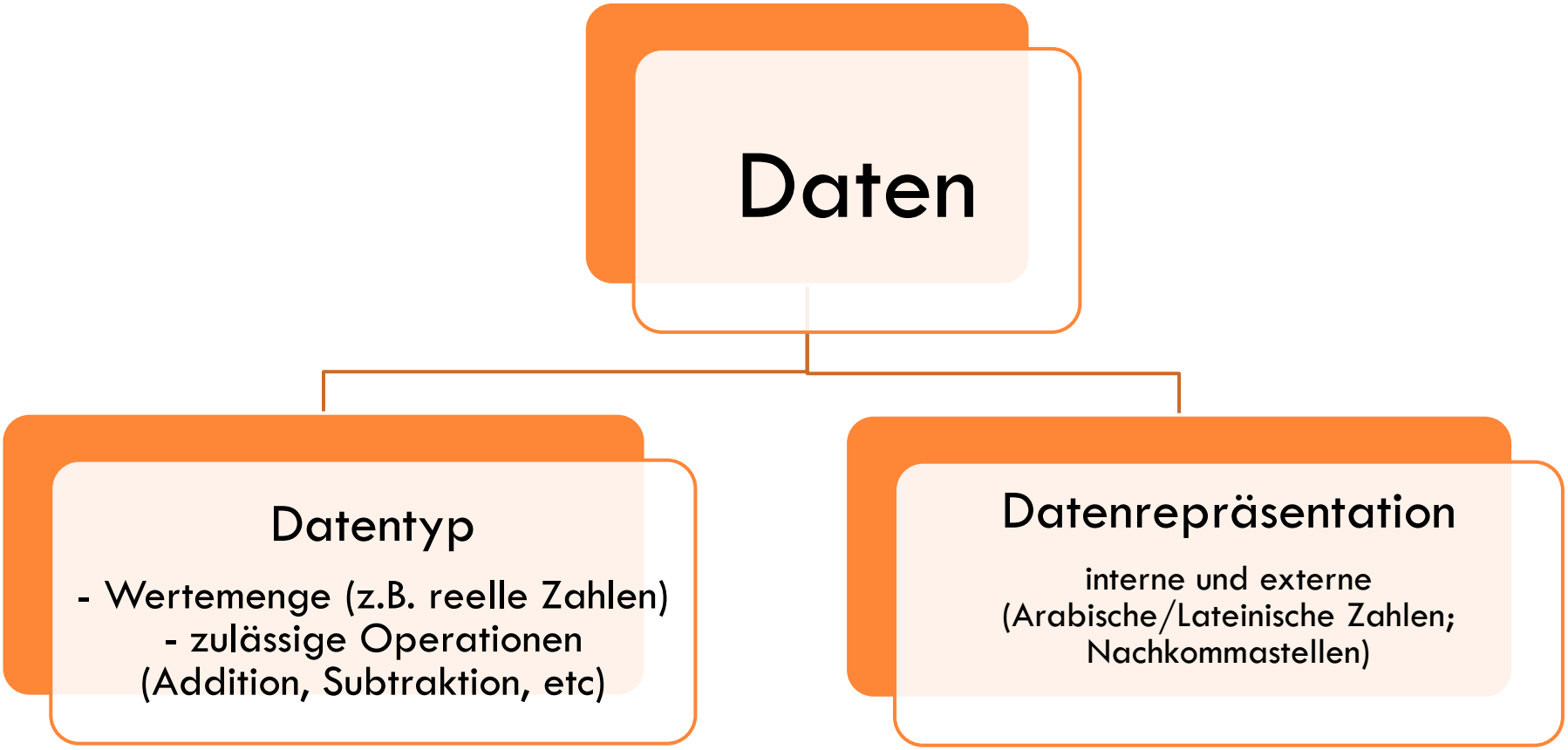


Beispiele

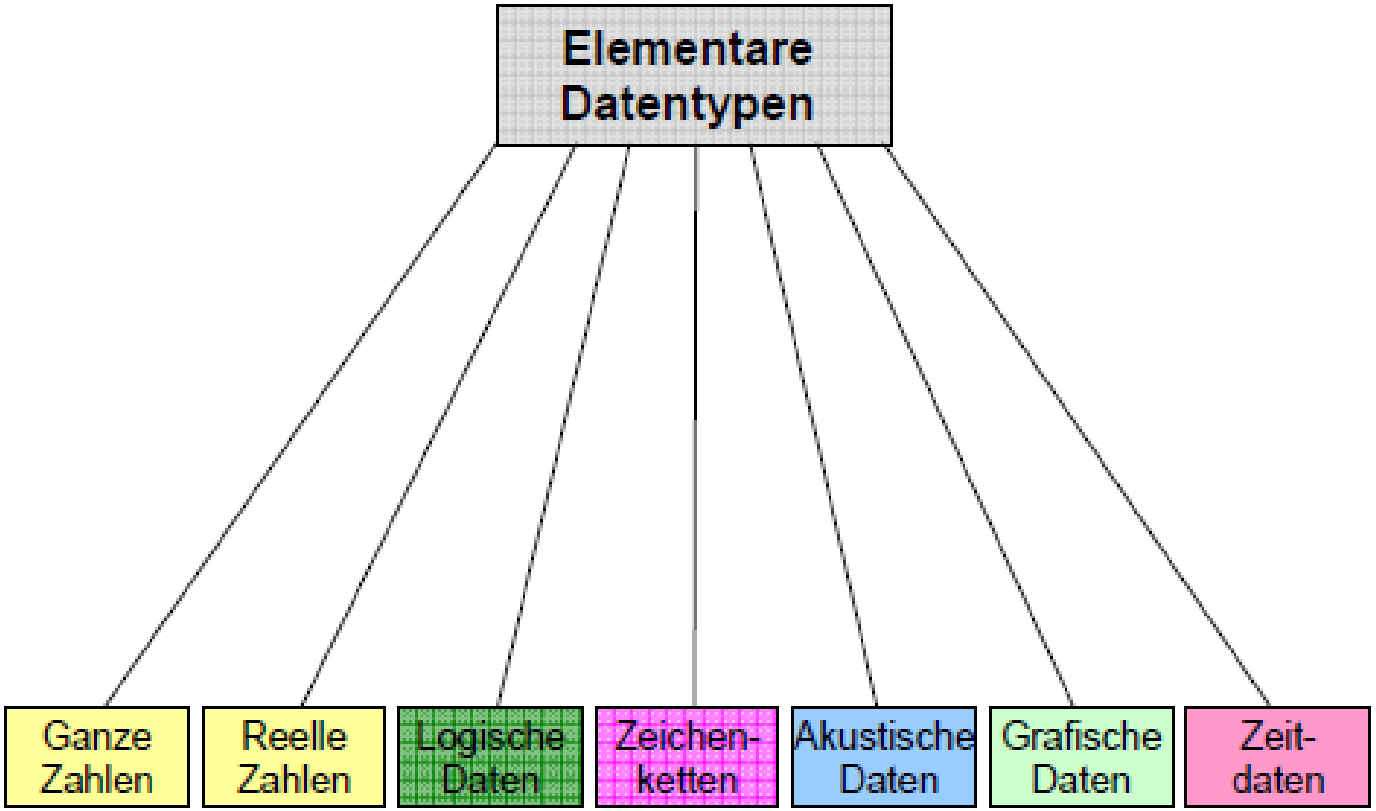
- Daten:
  - Aufträge, Kapazitäten, Liefertermine
- Information:
  - „Kapazität der Produktion reicht nicht aus, um die Aufträge zum zugesagten Termin abarbeiten zu können!“
- Wissen
  - „Kapazitäten lassen sich durch zusätzliche Schichten oder zusätzliche Maschinen erhöhen.“
  - „Fertigungstermine wichtiger Aufträge sind unbedingt einzuhalten.“
  - „Zusätzliche Schichten sind meist günstiger als die Anschaffung neuer Maschinen.“

JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

Daten und Datentypen



Elementare Datentypen



Einsatzformen und Verfahren

- Stammdaten:
  - Daten, die nur selten verändert werden
  - Personalstamm, Kundenstamm, Materialstamm
- Bestandsdaten:
  - Weisen Bestände aus
  - Lagerbestände, Kontostände
- Bewegungsdaten:
  - Daten mit mengen- oder wertmäßigen Zu- und Abgängen
  - Lagerentnahmen, Einzahlungen, Abbuchungen
- Metadaten:
  - Daten, die andere Daten beschreiben
  - Eigenschaften eines Datenfeldes, z.B. Fließtext oder feste Werte

Beispiel Krankenkasse KK

(087, Heinz, …

(021, Grippeimpfung, …

(087, 021, 997, offen)

(997, 087, 021, 80)

Daten

Stammdaten

Bestandsdaten

Bewegungs-daten

- KK-Kunden
(Kunden-Nr., Name, Adresse, etc.)
- Behandlungsarten
(Behandlungs-Nr., Adresse, etc.)

- Behandlungen
(Kunden-Nr., Behandlungs-Nr., Rechnungs-Nr, Bezahlt?, etc.)

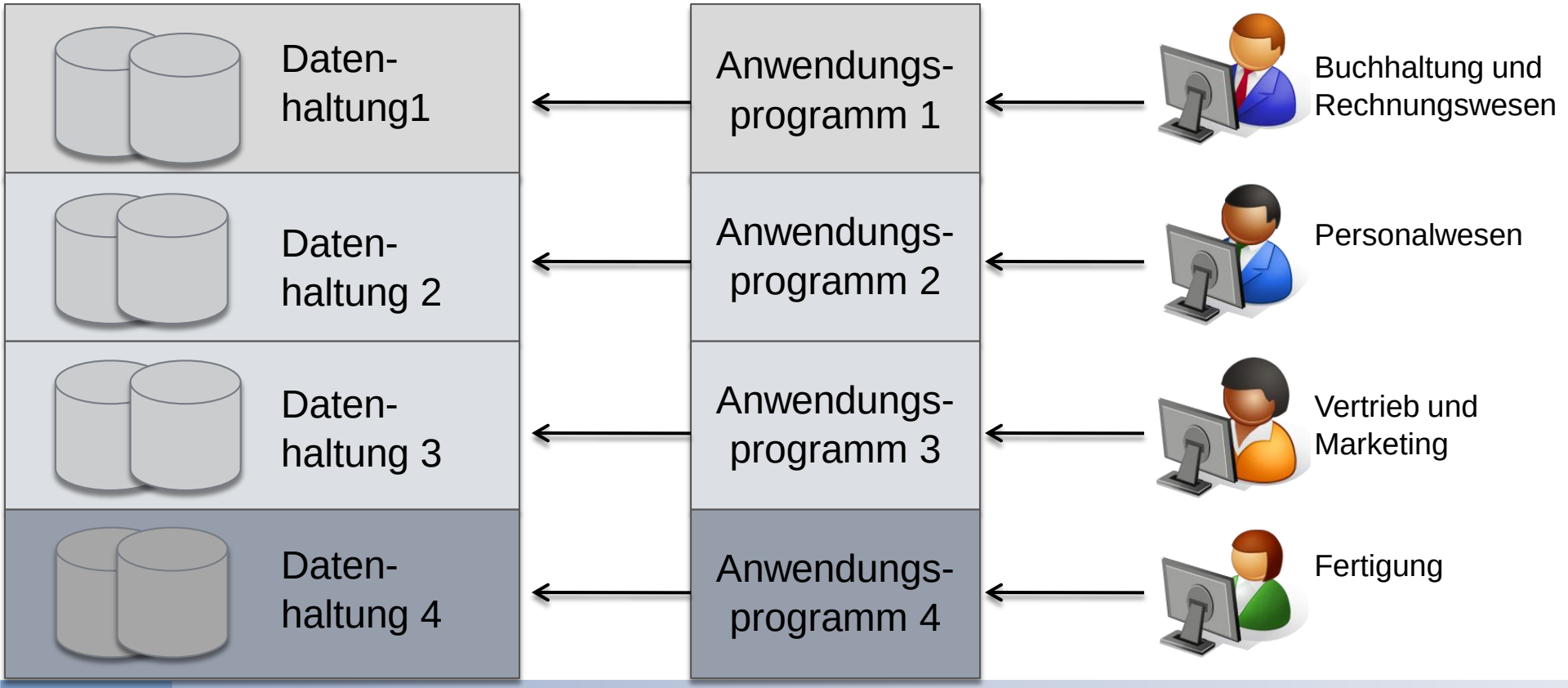
- Rechnungen
(Rechnung-Nr, Kunden-Nr., Behandlungs-Nr., Kosten, etc)

Meta-daten

PROBLEME UND ZIELE DER DATENORGANISATION

Traditionelle Datenverarbeitung

Separate Datenhaltung



Probleme separater Datenhaltung

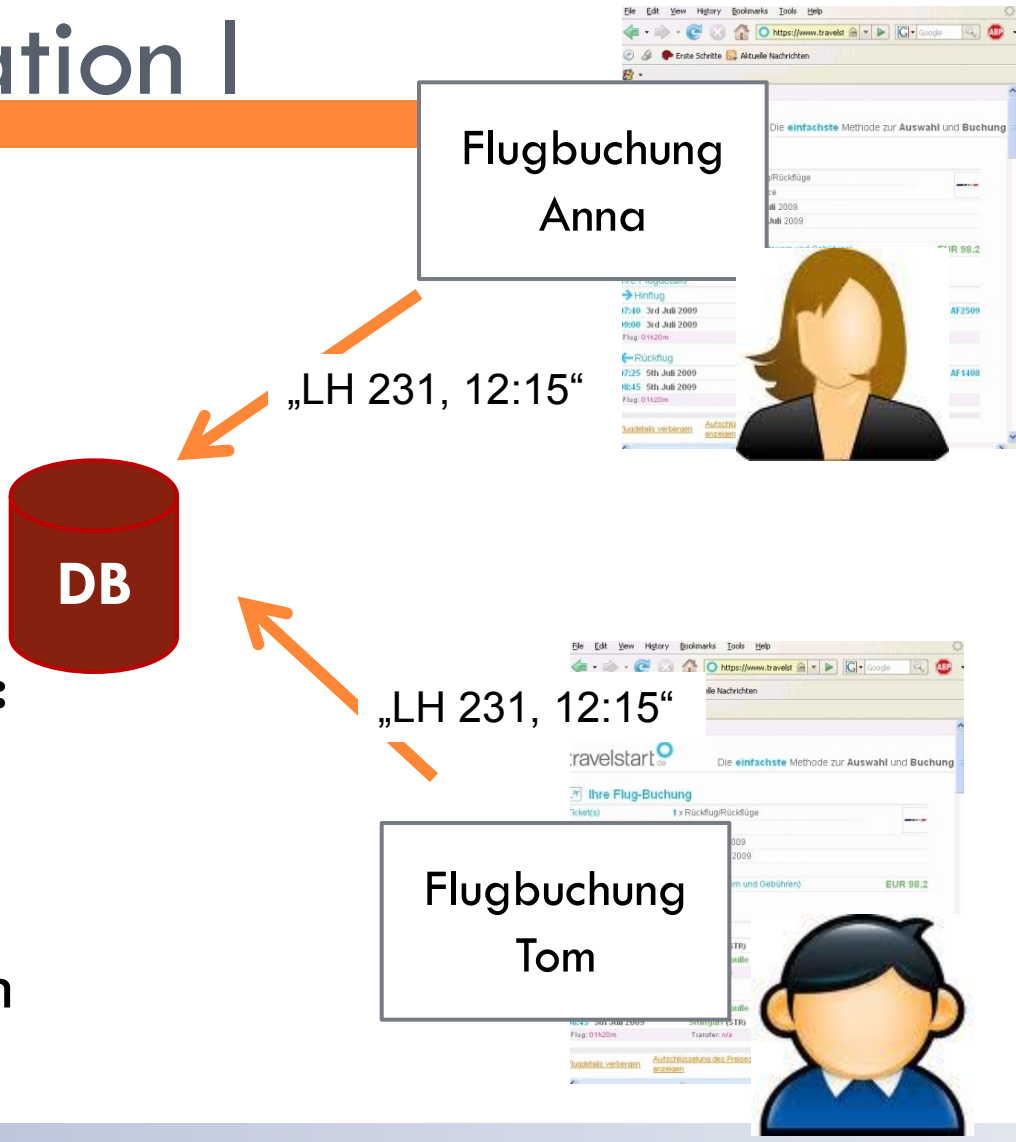
- **Datenredundanz**
  - Mehrfaches Vorkommen identischer Datenelemente in verschiedenen Tabellen/Verzeichnissen etc.
  - Redundanzen sind oftmals Ursache von inkonsistenten Daten, da mehrfach gespeicherte „gleiche“ Daten nicht überall gleich(zeitig) geändert werden
- **Dateninkonsistenz**
  - Die selben Attribute haben verschiedene Werte. Dateninkonsistenz kann zu falschen Informationen (und damit evtl. falschen Entscheidungen) führen
- **Abhängigkeit zwischen Anwendung und Daten**
  - Änderungen am Programmcode erfordern Änderungen an den zur Anwendung gehörenden Daten
  - Fehlender Datenaustausch und mangelnde Flexibilität
  - Mangelende Datensicherheit

Ziele der Datenorganisation

- Benutzerfreundlichkeit
  - Leichte Änderbarkeit/Aktualisierbarkeit
  - Schnelle Zugriffsmöglichkeiten (auf gespeicherte Daten)
- Flexibilität
  - Flexibilität der Auswertung und Verknüpfung der Daten
- Datenschutz und Datensicherheit
  - Schutz der Daten vor Verlust, Zerstörung, unbefugtem Zugriff
  - Verhinderung der unberechtigten Verwendung der Daten
- Datenintegrität
  - Korrektheit/Vollständigkeit der Daten
- Redundanzfreiheit
  - Keine mehrmalige Speicherung gleicher Daten

Probleme mangelnder Transaktionsisolation I

- Transaktion
  - ist eine Folge von Operationen, die als eine logische Einheit betrachtet werden.
- Beispiel „Flug buchen“:
  - Flugdaten eingeben
  - Flugdaten prüfen
  - Kundendaten eingeben
  - …



JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

Probleme mangelnder Transaktionsisolation II

- Lost Updates
  - Zwei Transaktionen modifizieren parallel denselben Datensatz und nach Ablauf dieser beiden Transaktionen wird nur die Änderung von einer übernommen.
- Dirty Read
  - Daten einer noch nicht abgeschlossenen, anderen Transaktion werden gelesen.
- Non-Repeatable Read
  - Wiederholte Lesevorgänge liefern unterschiedliche Ergebnisse.
- Phantom Read
  - Suchkriterien treffen während einer Transaktion auf unterschiedliche Datensätze zu, weil eine (während des Ablaufs dieser Transaktion laufende) andere Transaktion Datensätze hinzugefügt, entfernt oder verändert hat.

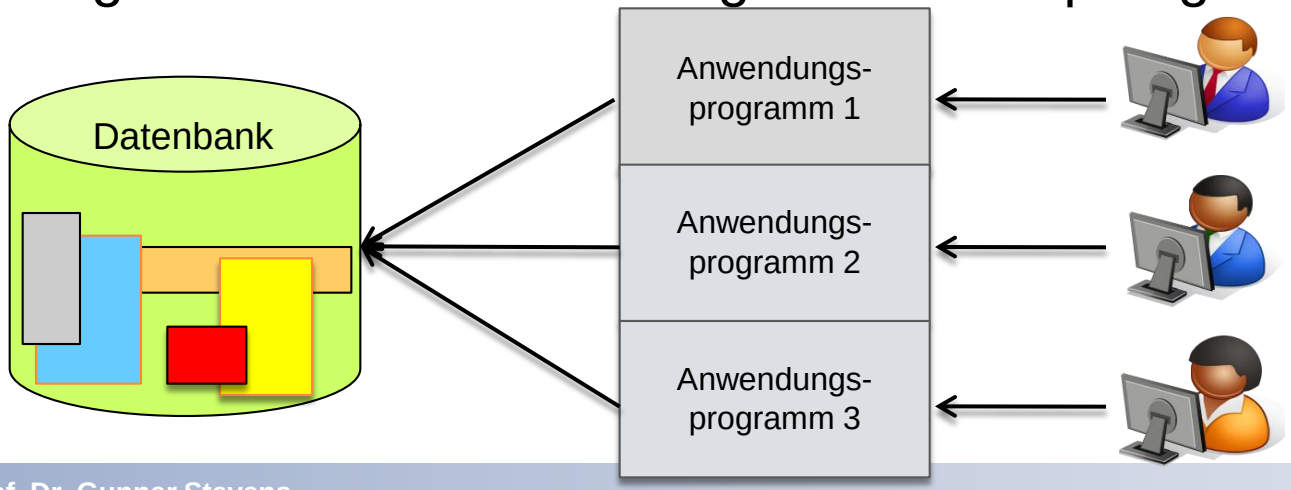
Ziele der Datenverwaltung

- Transaktionsisolation nach den ACID-Prinzip (**A**tomicity, **C**onsistency, **I**solation und **D**urability)
  - **Atomizität:** Eine Transaktion ist eine atomare Verarbeitungseinheit; sie wird entweder vollständig oder überhaupt nicht ausgeführt.
  - **Konsistenz:** Eine Transaktion ist konsistenzbewahrend, wenn ihre vollständige Ausführung die Datenbank von einem konsistenten Zustand in einen anderen überführt.
  - **Isolation:** Eine Transaktion soll so ausgeführt werden, als sei sie isoliert von anderen Transaktionen.
  - **Dauerhaftigkeit:** Die von einer bestätigten Transaktion in die Datenbank geschriebenen Änderungen müssen in der Datenbank fortbestehen.

VON DATEI-SYSTEMEN ZU DATENBANK-SYSTEMEN

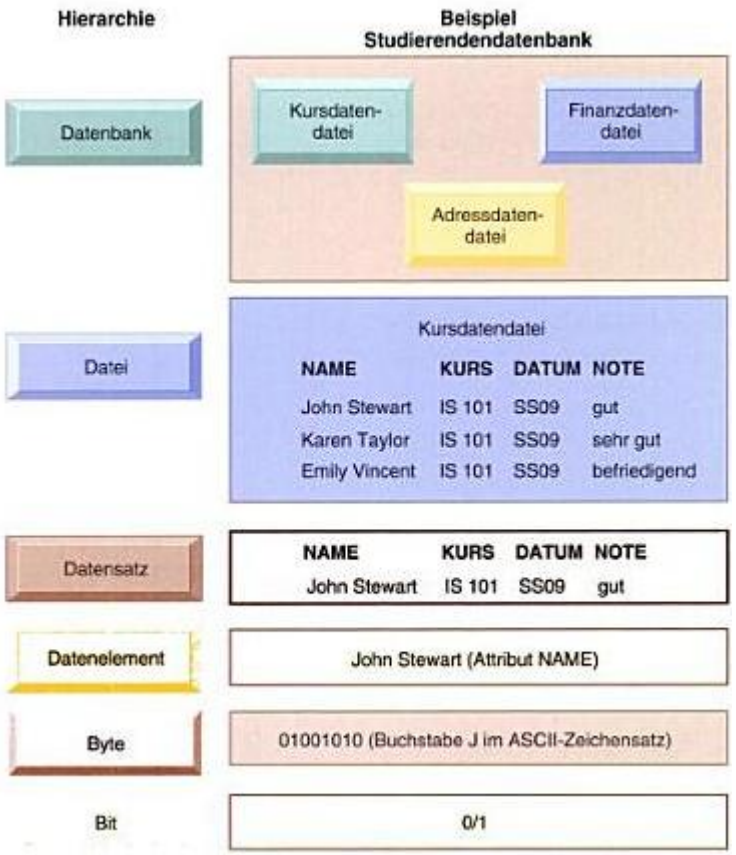
Datenbank Eigenschaften

- Möglichst redundanzfreie Sammlung von Daten
- Ermöglicht Zugriff von mehreren Benutzer oder Anwendungen
  - gleichzeitiger Zugriff (ACID Prinzip)
  - auf effiziente Weise
- Ermöglicht flexible Auswertung und Verknüpfung der Daten

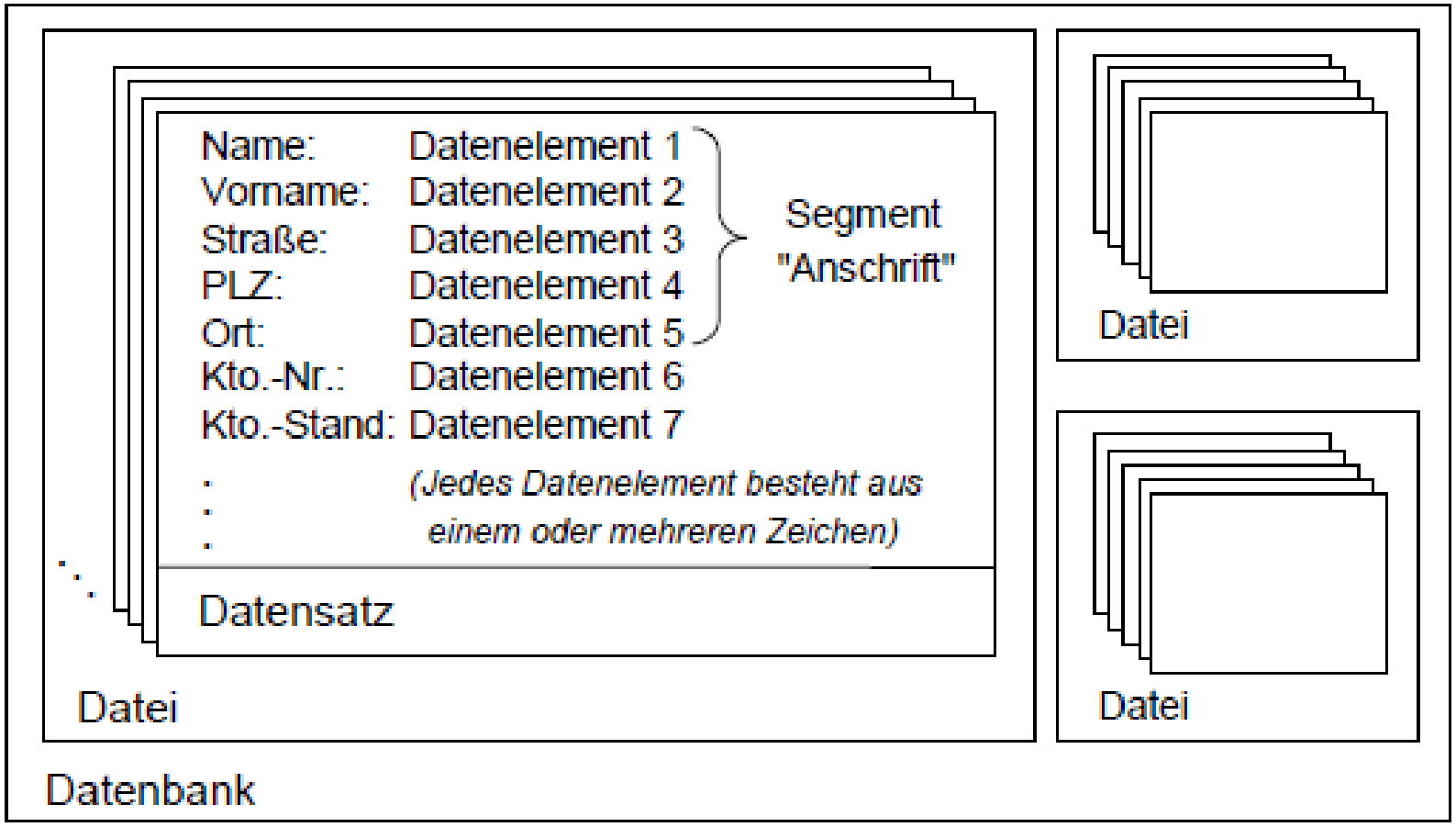


Hierarchischer Aufbau der Datenorganisation

- **Datenbank**
  - Sammlung logisch zusammengehörender Dateien (und deren Verknüpfungen)
- **Datei**
  - Zusammenfassung logisch zusammengehörender, gleichartiger Datensätze
- **Datensatz**
  - Inhaltlich zusammenhängende Datenelemente werden zu einem Datensatz zusammengefasst
- **Datenelement**
  - Die kleinste gespeicherte logische Dateneinheit



Hierarchischer Aufbau der Datenorganisation



SPARC/ANSI 3-Ebenen Modell

- 1972 entwickelt zur Standardisierung von Datenbankarchitekturen vom **S**tandard **Pla**ning **R**equirement **C**ommittee
- Umsetzung des „Seperation of Concerns“ Prinzips
  - Hohes Maß an Datenunabhängigkeit
  - Änderungen auf einer Ebene ohne andere Ebenen zu beeinträchtigen

1. Externe Ebene
   - Anwendungspezifische Repräsentation
2. Konzeptuelle Ebene
   - Logische Datenorganisation
3. Interne Ebene
   - Physische Datenorganisation

JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

SPARC/ANSI 3-Ebenen Modell

Benutzerebene	•Erzeugt Sichten (Views), um Benutzern die Daten darzustellen •Wie kann ich mit welchen Daten den Benutzer bei seiner Arbeit unterstützen?
konzeptionelle Ebene	•Welche Daten werden gespeichert? •In welcher Beziehung stehen sie zueinander? •Wer darf auf welche Daten zugreifen? •Wie wir die Datenintegrität hergestellt und geschützt? •Wie werden Redundanzen verhindert? •Abbildung der Wirklichkeit in Tabellen
physikalische Ebene	•Zugriffsverfahren •Speicherstruktur

SPARC/ANSI 3-Ebenen Modell

Externe Schicht

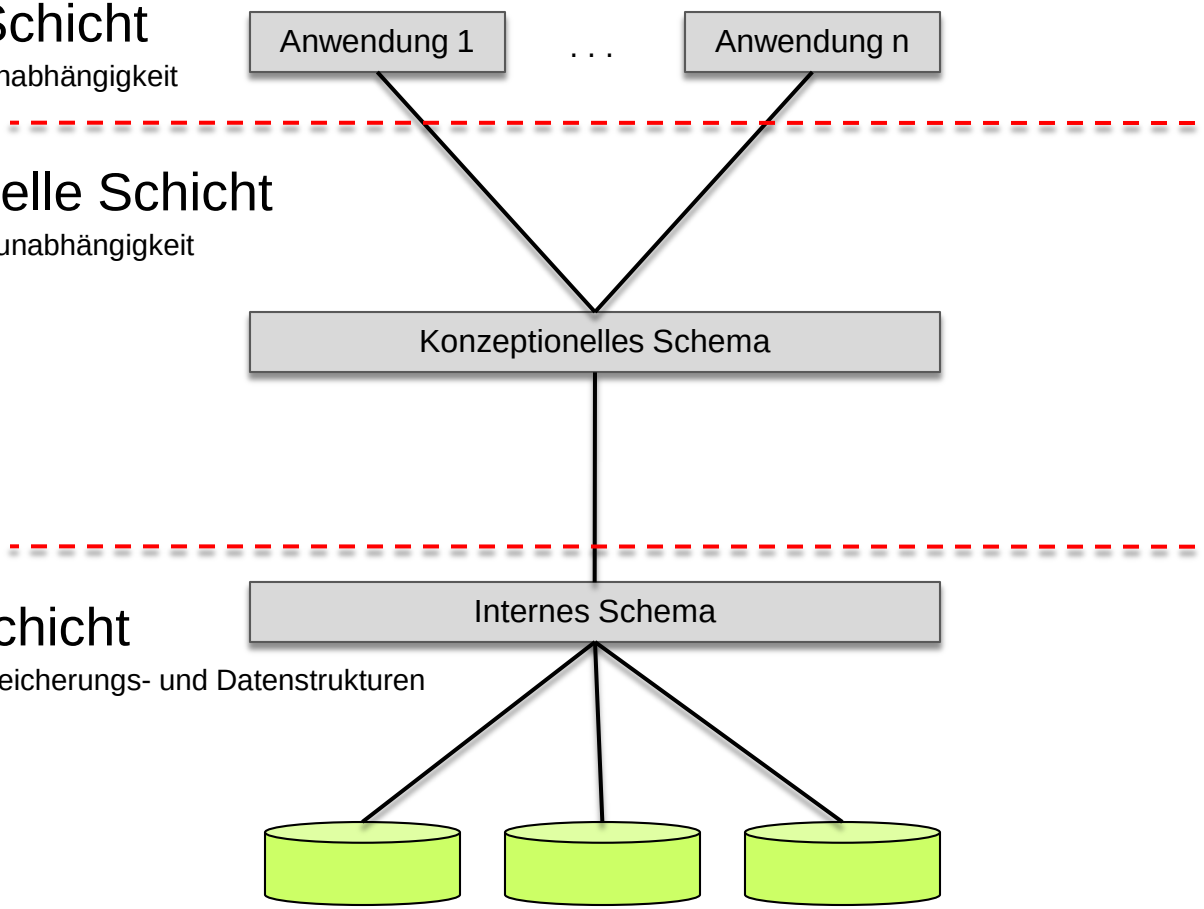
Logische Datenunabhängigkeit

Konzeptuelle Schicht

Physische Datenunabhängigkeit

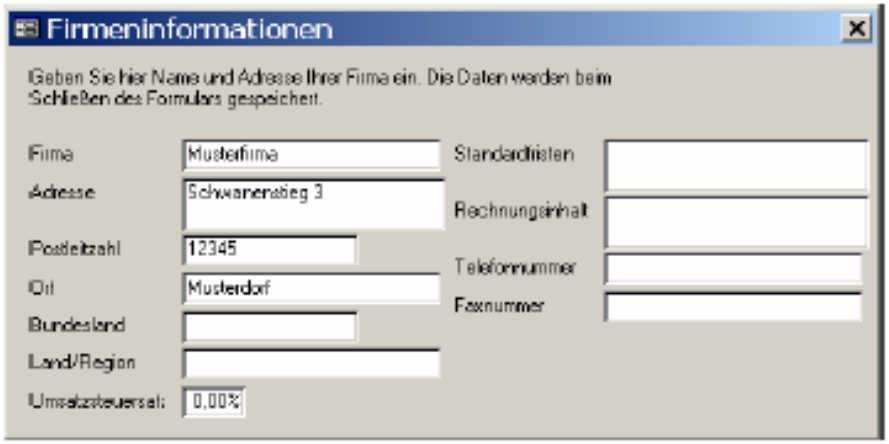
Interne Schicht

Abbildung auf Speicherungs- und Datenstrukturen



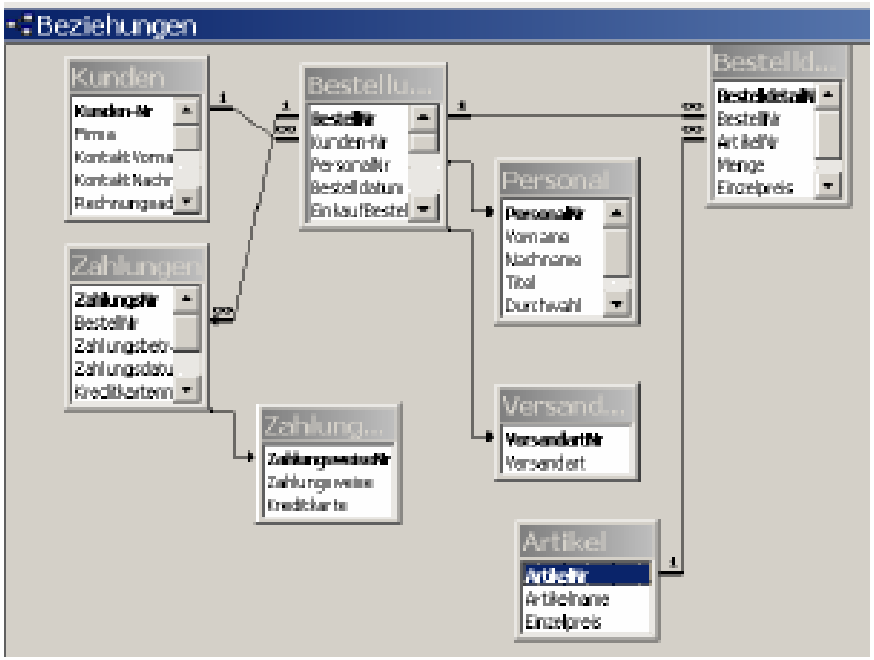
Externe Sicht

- Beschreibt die Sicht eines oder mehrerer Benutzer auf die Daten
- Für den Benutzer nicht relevante Daten werden vor ihm verborgen



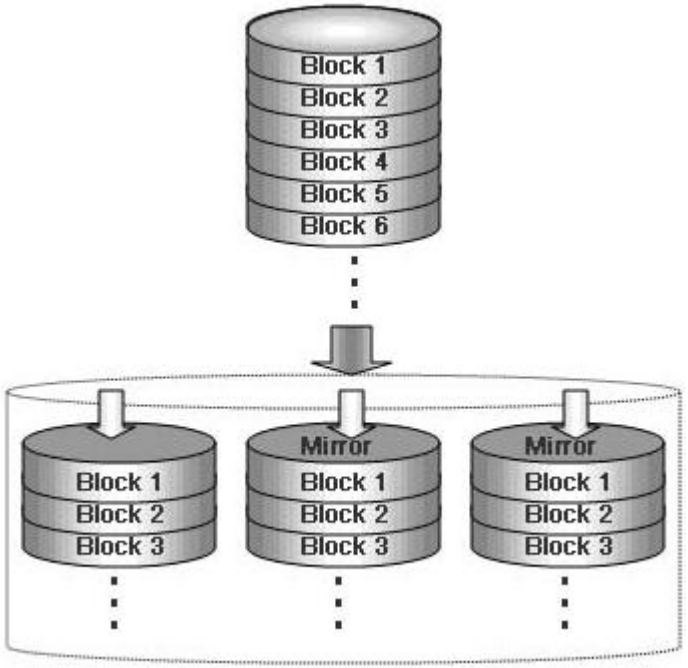
Logische / Konzeptuelle Sicht

- Das konzeptuelle Schema legt die Strukturen der konzeptuellen Sicht der gesamten Datenbank fest
- Berücksichtigt werden Entitäten, Datentypen, Beziehungen und Integritätsbedingungen
- Physikalische Speicherstrukturen werden verborgen



Physische / Interne Sicht

- Beschreibt die physikalischen Speicherstrukturen der Datenbank
- Fokus auf effiziente Abspeicherung der Datenbank-Objekte auf dem Speichermedium
  - Physische Datenhaltung (Cluster)
  - Logische Datenhaltung (Dateisystem)



DATENBANKMANAGEMENT SYSTEME (DBMS)

Datenbankmanagementsysteme (DBMS)

- Sammlung von Programmen zum Erstellen und Verwalten einer Datenbank, die es mehreren Anwendungen gleichzeitig ermöglicht auf die Daten in effizienter Weise zuzugreifen

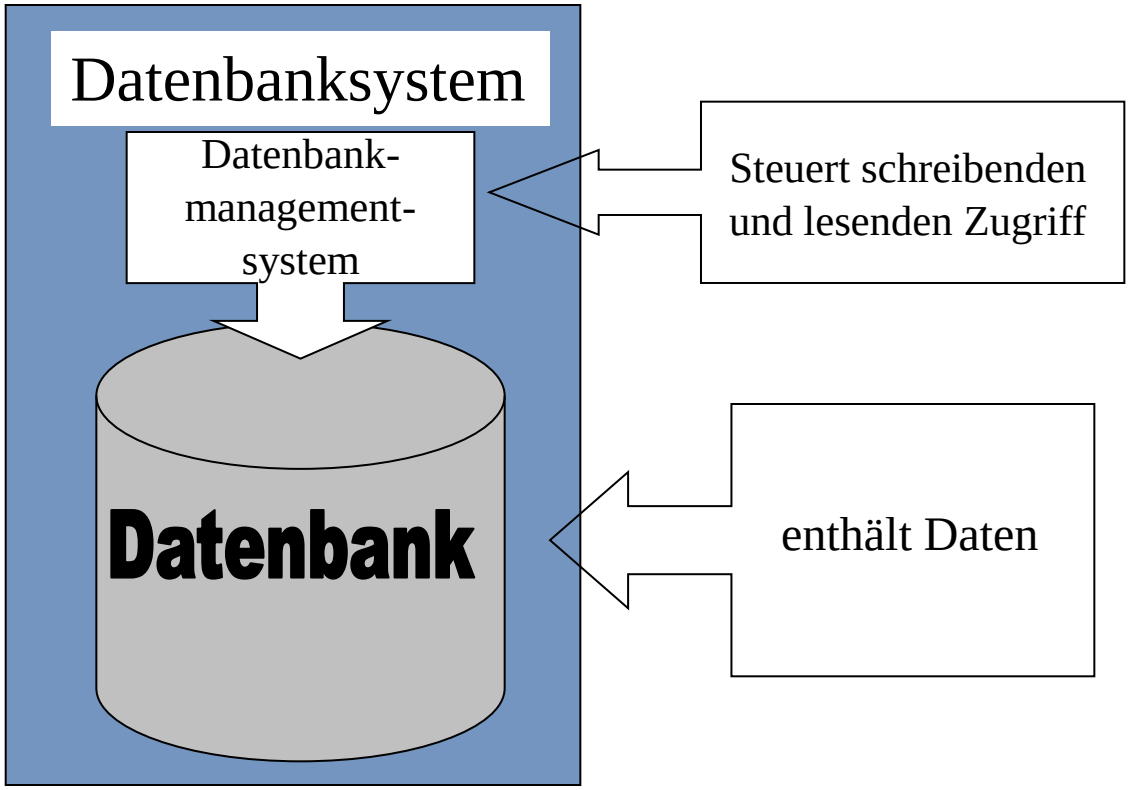
- **Vorteile**
  - Ermöglicht die Speicherung, Extrahierung und Manipulierung von Daten ohne eigene Dateien erstellen zu müssen
  - Löst einige Probleme traditioneller Datenverarbeitung, z.B. Einhaltung des ACID-Prinzips, Redundanzfreiheit, etc.
  - Reduziert Komplexität der Programmierung/Modellierung der Datenverwaltung
    - Trennt die physische und logische Datenstruktur
    - Trennt Programmcode und Daten

Aufgaben des DBMS

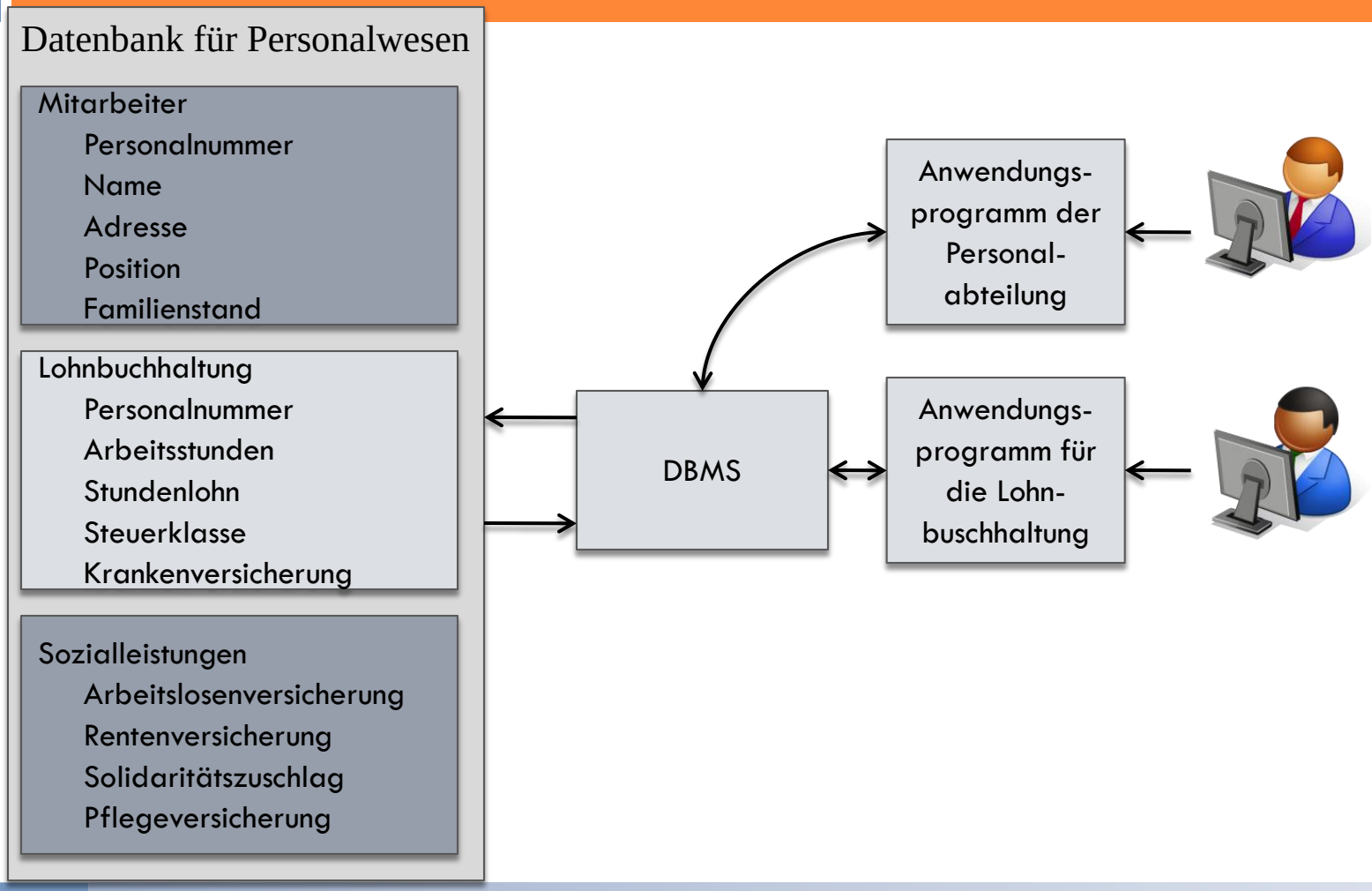
- Schnittstelle zwischen Benutzer und Datenbank und alleiniger physikalischer Zugriff auf den Datenbereich
  - Benutzerverwaltung
  - Transaktionsverwaltung
- Verbindung der logischen und physikalischen Ebene
  - Umsetzung des Datenmodells
- Datenbankmanagementsystem bietet
  - Abfragesprache, um Daten zu lesen/schreiben.
  - Spezifikationssprache, um Datenmodell zu erstellen

JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

Aufbau Datenbankumgebung

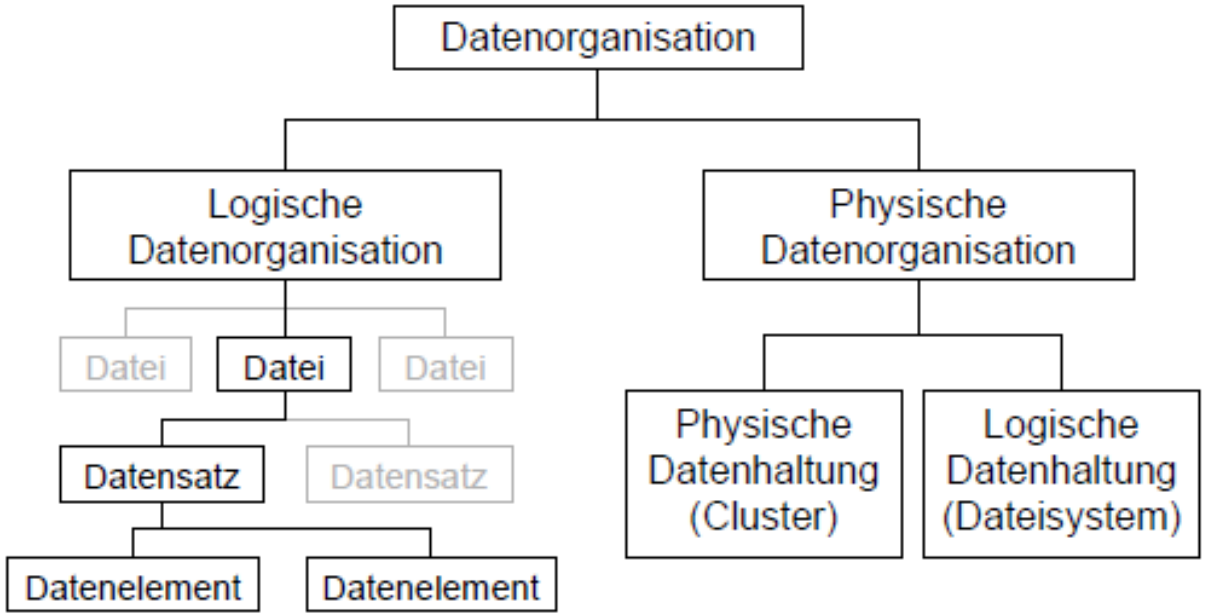


Beispiel

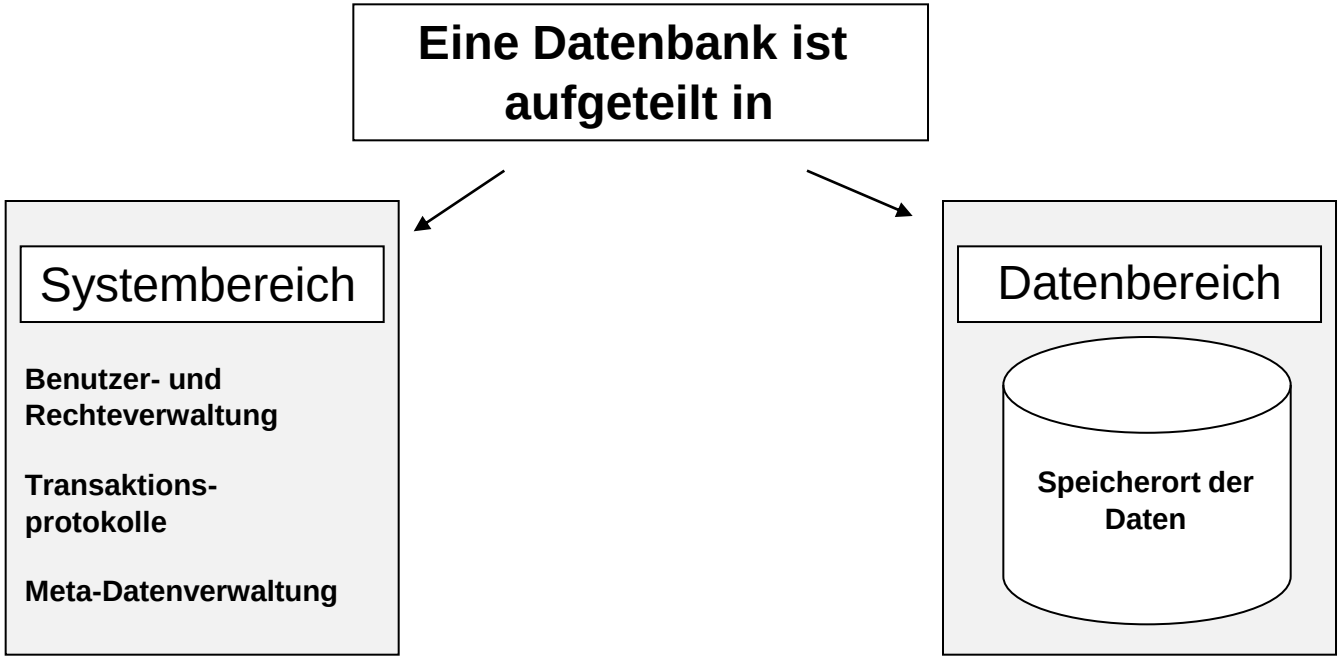


JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

Aufbau Datenorganisation I



Aufbau Datenorganisation I



JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

DBMS Sprachen

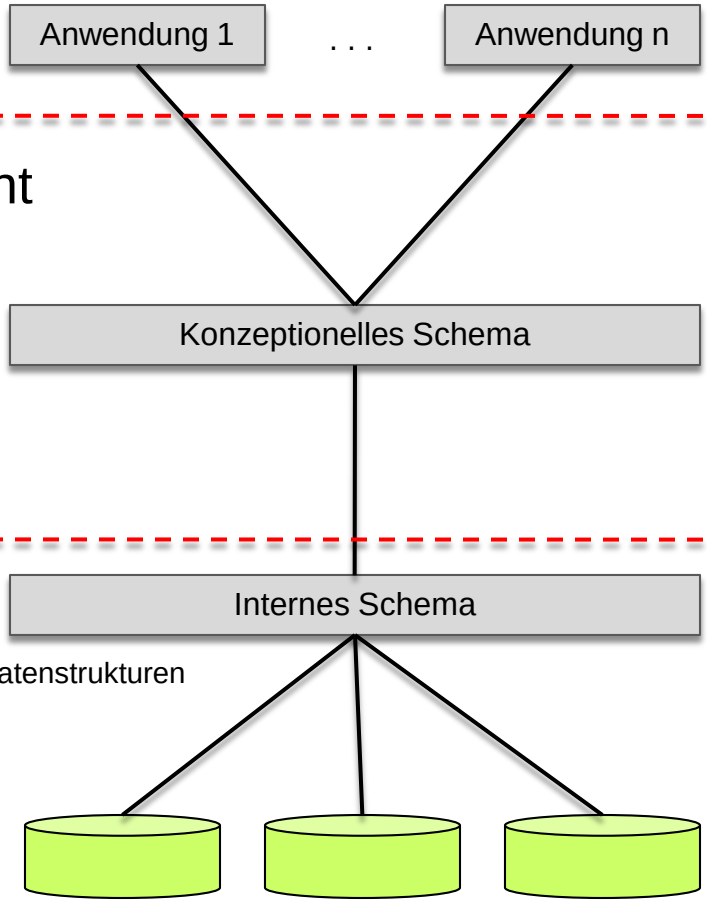
- Data Query Language (DQL)
  - Sprache zur Formulierung von Abfragen
- Data Manipulation Language (DML)
  - Sprache zur Bearbeitung des Datenbestandes (Ändern, Löschen, …)
- Data Definition/Description Language (DDL)
  - Sprache zur Beschreibung der logischen Datenstrukturen einer Datenbank
- Data Storage Description Language (DSDL)
  - Sprache zur Beschreibung der physischen Organisation der Daten innerhalb des Datenbanksystems

JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

DBMS Sprachen

Externe Schicht

Logische Datenunabhängigkeit



DML,DQL

Konzeptuelle Schicht

Physische Datenunabhängigkeit

DDL

Interne Schicht

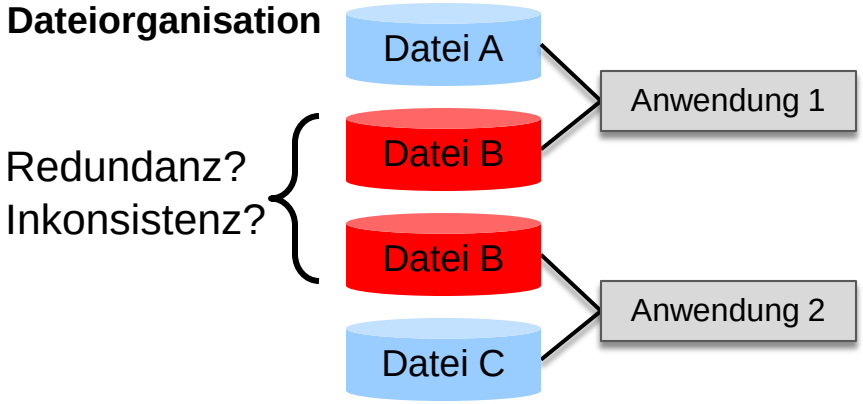
Abbildung auf Speicherungs- und Datenstrukturen

DSDL

JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

Zusammenfassender Vergleich Datei- vs. Datenbankorganisation

Dateiorganisation



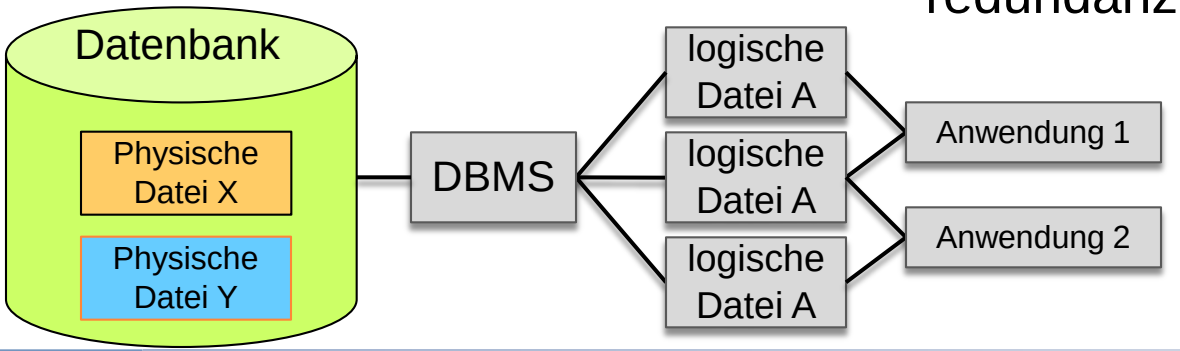
Dateiorganisation:

- Wiederholte Speicherung gleicher Daten. Datei B ist redundant
- erhöhter Speicherplatzbedarf
- Konsistenzprobleme

Datenbankorganisation :

- Physische Daten sind redundanzfrei und konsistent
- Die Daten sind unabhängig von den einzelnen Programmen, die auf sie zugreifen

Datenbankorganisation

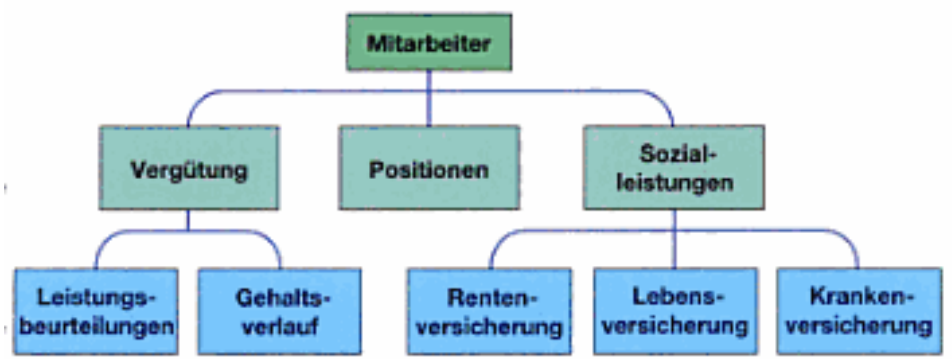


JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

DATENMODELLE & DATENMODELLIERUNG

Hierarchisches Datenmodell

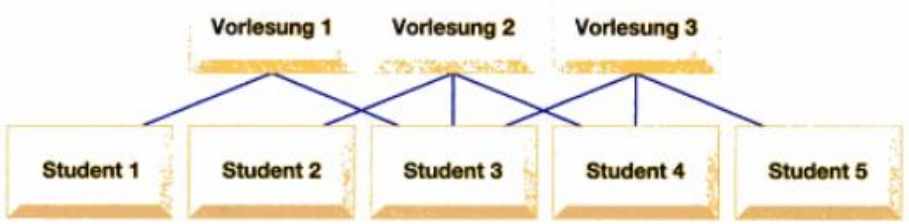
- Das älteste logische Datenmodell
- Datensätze werden durch *Eltern-Kind-Beziehungen* verknüpft (liefert Baumstruktur)
- Hierarchische Datenspeicherung erlebt Renaissance mit XML



Quelle: Laudon et al. 2010, 299

Netzwerk Datenmodell

- Älteres Datenmodell
- Konzepte sehr nah an Implementierung
- Modell fordert keine strenge Hierarchie sondern kann auch m:n-Beziehungen abbilden, d. h. ein Datensatz kann mehrere Vorgänger haben. Auch können mehrere Datensätze an höchster Stelle stehen.



Quelle: Laudon et al. 2010, 299

Relationale Datenmodell I

Tabelle = Relation

Spalten=Colums

KNr	Name	Vorname	PLZ	Ort	Strasse
1	Hugenbubel	Herbärt	12345	Steinhausen	Tomatenweg 10
2	Spitzweg	Carl	81526	Himmelhausen	Sternenweg 18
3	Cron	Maria	36645	Blaupausen	Kopierstr. 18

Rows(Tupel)

Sammlung aller Attribute eines Datensatzes

Sammlung gleicher Attribute

Feld, alle Felder einer Zeile bilden einen Datensatz, der alle Attribute eines Objektes (Entität) umfasst. (vgl. Tupel)

Die Spalte KNr. eignet sich dazu, als Primärschlüssel definiert zu werden, weil alle Werte eineindeutig sind, da sie nur einmal vorkommen.

Relationales Datenbankmodell II

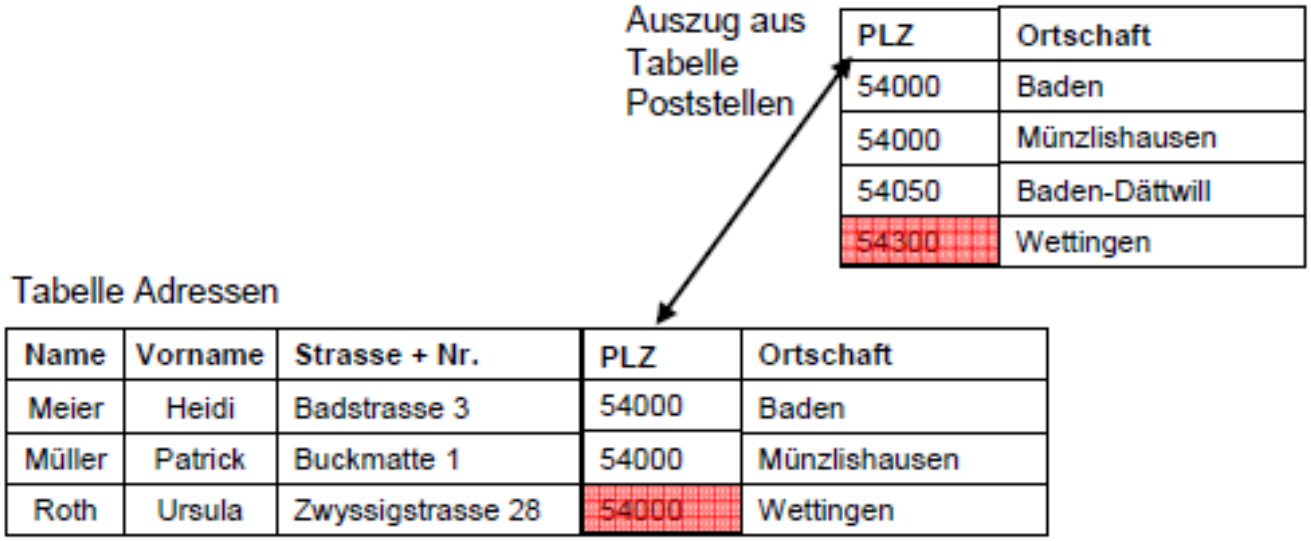
- Organisation in Tabellen (Relations), die voneinander weitgehend unabhängig sind.
- Eine Tabelle ist ein Konstrukt aus Zeilen (Reihen, rows, auch Tupel genannt) und Spalten (columns)
- Eine Zeile ist eine Sammlung von Attributen eines Datensatzes (Objekt, auch Entität), die Spalten stellen Attribute gleicher Art (z.B. Name, Vorname) dar.
- Tabellen können dadurch miteinander verknüpft werden, dass sie (mindestens) ein gleiches Attribut beinhalten.
- Es können beliebig viele Sichtweisen (views) erzeugt werden, die eine Zusammenstellung verschiedener Attribute und verschiedener Tabellen beinhaltet.
- Der Bereich der Werte, die ein Attribut (d.h. ein Eintrag in einer bestimmten Spalte) annehmen kann, nennt man Wertebereich (domain) des Attributes. Dieser Wertebereich wird durch *Einschränkungen (constraints)* näher bestimmt.

Relationales Datenbankmodell III

- Enthält ein Datensatz ein Attribut, welches ein Objekt eindeutig charakterisiert, z.B. Kundenummer) kann dieses Objekt als Primärschlüssel definiert werden.
- Wird der Primärschlüssel in einer anderen Tabelle ebenfalls als Attribut verwendet, so spricht man von einem Fremdschlüssel.
- Mit Hilfe dieser Schlüssel können die Tabellen in Beziehungen zueinander gesetzt werden.

JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

Beispiel



- Daten strukturiert?
- Daten konsistent?
- Daten redundant?

DATENMODELLIERUNG - BEISPIEL

Beispiel Datenbankdesign

Entwickelt werden soll die Datenbank eines kleinen Online-Computershops, in dem Kunden Waren in einen Warenkorb legen und später bestellen können.

- Dazu ist es notwendig, aufgrund der zur Erledigung des Geschäftsvorfalls benötigten Informationen ein formales Datenmodell zu spezifizieren.
- Die Modellierung der Datenstruktur gibt es verschiedene Methoden,
  - z.B. die Entity-Relationship-Modellierung.

Planung einer Datenbank

Die Planung einer Datenbank gliedert sich unter Anwendung des ER-Modells grob in fünf Teilschritte

- Entwicklung des semantischen Modells
- Definition der Objekte
- Zuordnung der Attribute
- Analyse der Entitätsbeziehungen
- Erstellung eines ER-Diagramms

Anmerkung: Das sehr einfach gehaltene Beispiel des Onlineshops soll Ihnen beim Verstehen dieser Schritte helfen.

Entwicklung des semantischen Modells

Im Rahmen der Entwicklung des semantischen Modells soll festgestellt werden, welche Anforderungen an die Datenbank gestellt werden.
In Beispiel „Webshop“ wäre dies z.B. :

Die Kunden unserer Seite wählen Produkte aus und legen sie in den Warenkorb.
Die Produkte des Warenkorbes können in eine Bestellung übergehen.

Definition der Objekte

Ein Objekt bzw. Entität (letztlich eine Tabelle) beschreibt ein abgrenzbares Phänomen, des Weltausschnittes, der modelliert werden soll.
In Beispiel „Webshop“ sind folgende Objekte denkbar:

- Kunde
- Produkt
- Warenkorb
- Bestellung

Zuordnung der Attribute

Jedes Objekt weist bestimmte Eigenschaften auf, mit denen es charakterisiert werden kann. Um optimale Funktionstüchtigkeit zu gewährleisten, müssen alle notwendigen Attribute erfasst werden.

Um eine Eindeutigkeit der Datensätze herzustellen, werden regelmäßig durchlaufende Nummern pro Datensatz vergeben. Diese Nummern werden zu Primärschlüsseln verarbeitet.

KUNDE	KundenNr, Name, Adresse
PRODUKT	ProduktNr, Produktname, Beschreibung, Preis
WARENKORB	KorbNr, Menge, Datum/Zeit
BESTELLUNG	BestellNr, KundenNr, Menge, Gesamtpreis, Datum

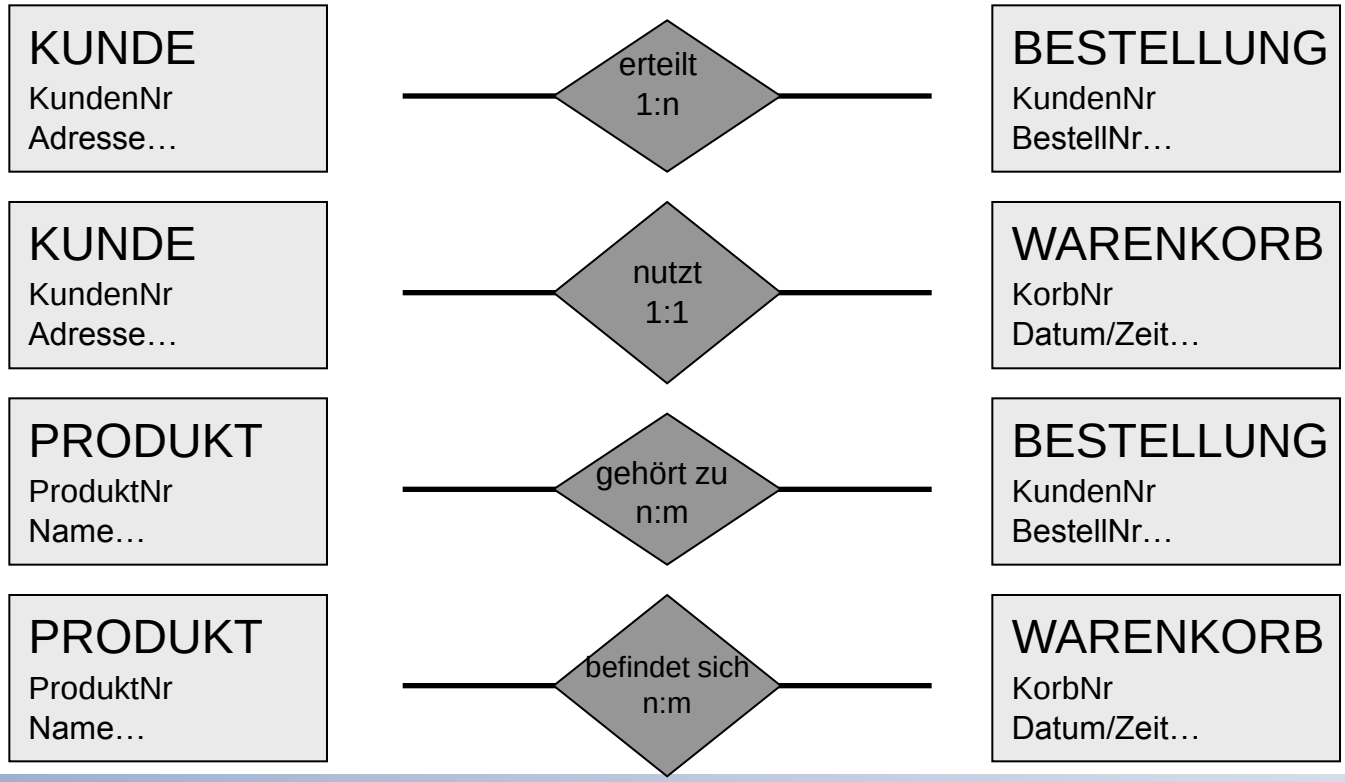
Analyse der Entitätsbeziehungen I

Beziehungen zwischen Entitäten sind Kernbestandteil des ER-Modells. So kann z.B. die Kundennummer die Entitäten BESTELLUNG und KUNDE miteinander verknüpfen.

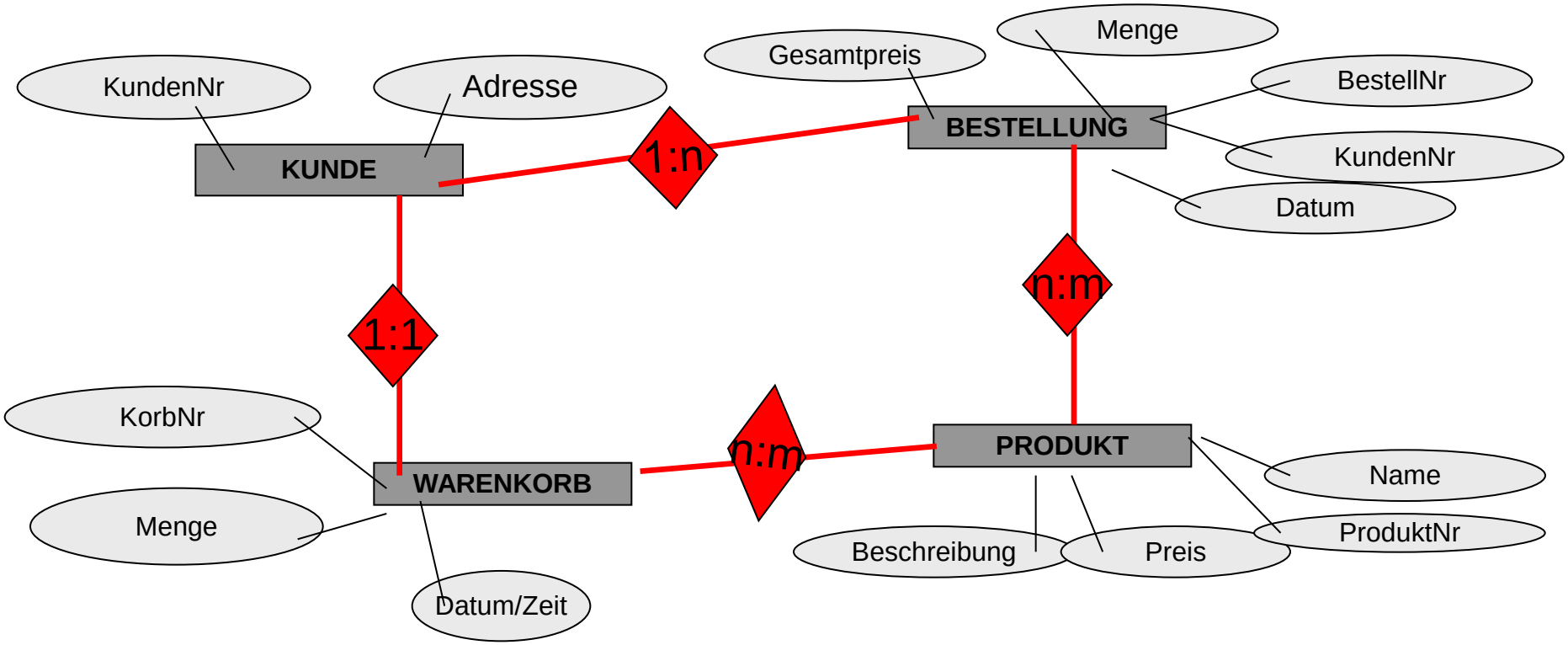
(Sie ist in der Tabelle Kunde Primär- und in der Tabelle Bestellung Fremdschlüssel).

Analyse der Entitätsbeziehungen II

Bei den gegebenen Tabellen KUNDE, PRODUKT, WARENKORB und BESTELLUNG ergeben sich folgende Assoziationen:



Erstellung eines Entitätsdiagramms



Anmerkung: Auf die Darstellung der n:m Beziehungen wird in diesem Beispiel verzichtet.

ENTITY-RELATIONSHIP-MODELL (ERM)

Entity-Relationship-Modell (ERM)

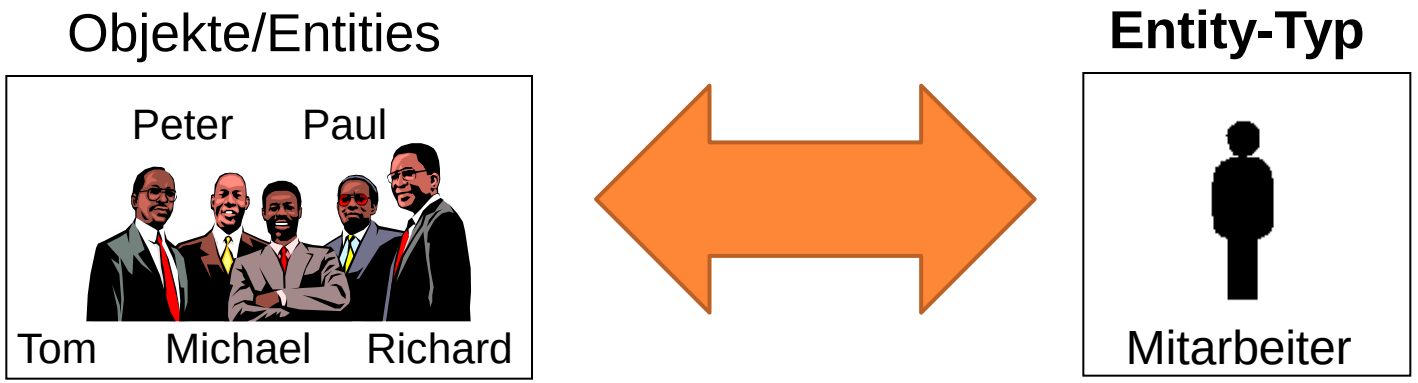
- Modellierungsansatz von Chen 1976
- Ziel: graphische Modellierung **statischer** Datenstrukturen der gegebenen Domaine (konzeptuelles, high-level Datenmodell)
  - Modelliert werden Gegenstände/Sachverhalte (**Entities**) sowie deren Beziehungen (**Relationships**
  - Keine Betrachtung des dynamischen Verhaltens (z.B. Geschäftsprozesse), sondern nur des statischen Aspekts des zu modellierenden Ausschnitts

Beschreibungsformen für Daten – Entities

- Entities sind grundsätzlich individuelle Exemplare
- Beispiele für reale, physisch existierende Objekte:
  - Der PKW der Marke X mit dem amtlichen Kennzeichen SI-XY 1
  - Der in der Personalabteilung der Firma Y beschäftigte Angestellte Klaus Meier
- Beispiel für ein materiell nicht existierendes Objekt:
  - Die telefonische Bestellung des Kunden K. zur Lieferung von 50 Stück des Artikels 4711
  - Zur Bestellung gehören zwar physisch existierende Objekte wie Kunde und Artikel, aber die Bestellung selbst ist immateriell

Entities und Entitytypen

- Gleichartige Entities (Objekte) werden zu einer Entitymenge (Objektmenge) zusammengefasst und durch einen **Entity-Typen** beschrieben.
- Beispiele:
  - alle in einem Unternehmen beschäftigten Personen werden zur Menge Mitarbeiter zusammengefasst
  - alle an einer Universität eingeschriebenen Personen werden zur Menge Studenten zusammengefasst



Entities und Entitytypen

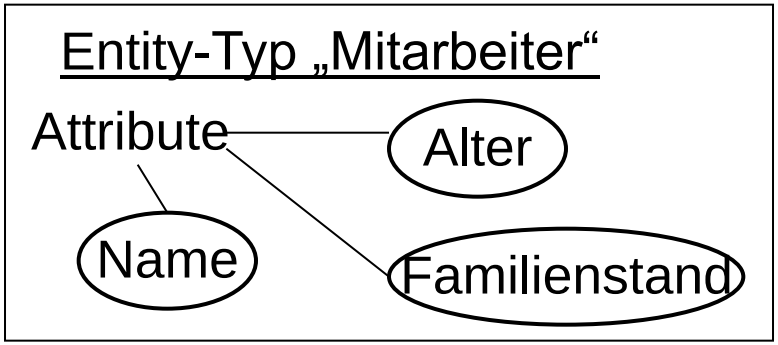
- Graphische Notation (Chen-Notation):

Entitätstypen werden durch Rechtecke dargestellt

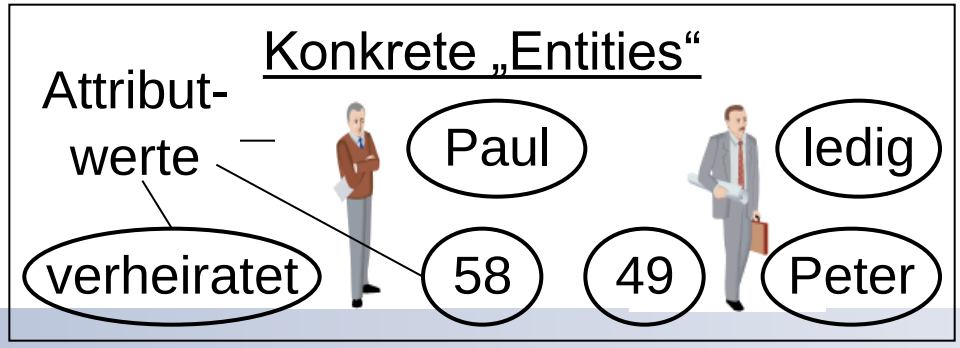
- Beispiel:

Beschreibungsformen für Daten – Attribute

- Daten beziehen sich auf die Eigenschaften oder Attribute von Entities, sind also Informationen über Entities
- Beispiel: Der Entitytyp Mitarbeiter hat z.B. die Eigenschaften Name, Alter, Familienstand.

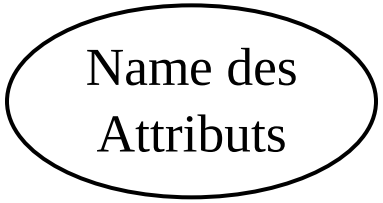


- Beispiel:
  Der Mitarbeiter Paul
  ist 58 Jahre alt
  und verheiratet

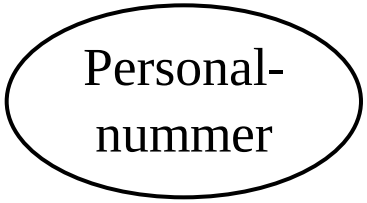


Attribute

- Ein Attribut beschreibt eine Eigenschaft von Entities eines Entitytyps bzw. von Beziehungen eines Beziehungstyps
- Beispiel: Die Personalnummer des Mitarbeiters „Meier“ ist 1234. Personalnummer ist Attribut des Entitytypen Mitarbeiter und hat den Wert 1234 für den Mitarbeiter „Meier“
- Graphische Notation:

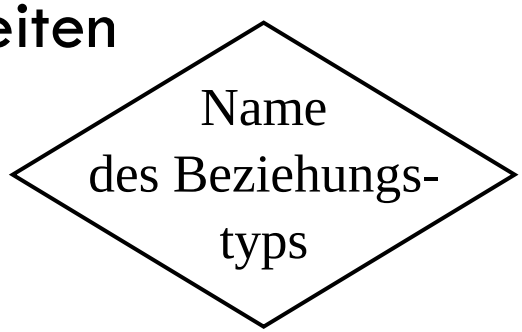


- Beispiel:

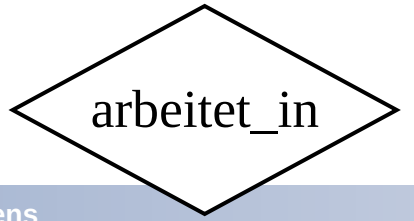


Beziehungen und Beziehungstypen

- Beziehungen (Relationships) stellen wechselseitige Assoziationen, d.h. Wechselwirkungen und Abhängigkeiten, zwischen den Entities dar
- Gleichartige Beziehungen werden zu Beziehungstypen zusammengefasst
- Beispiel: Assoziation „arbeitet in“ bezeichnet, dass Mitarbeiter in Projekten arbeiten
- Graphische Notation:

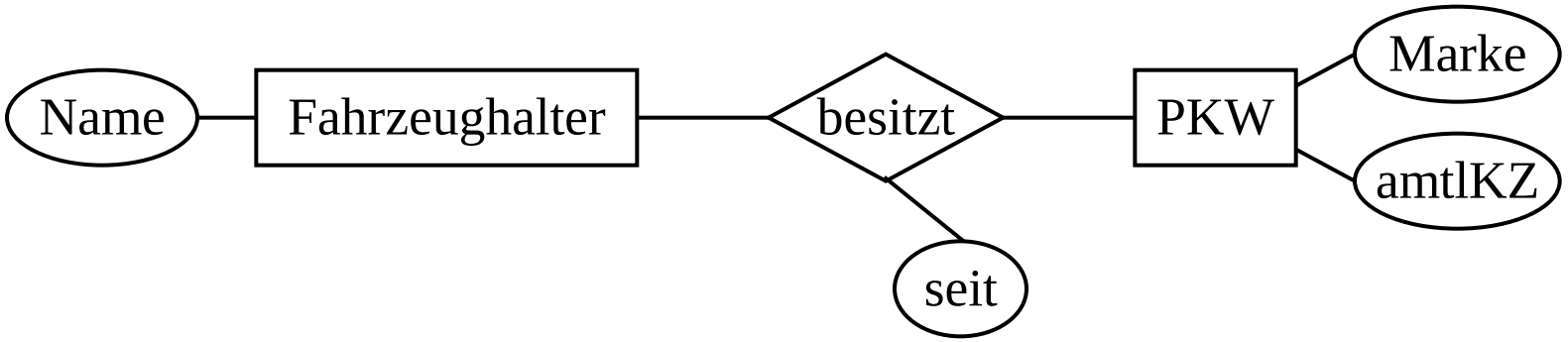


- Beispiel:



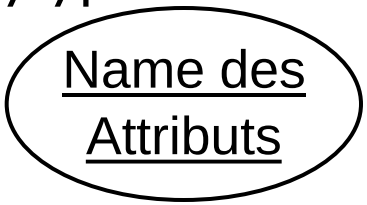
Beziehungen und Beziehungstypen

- Beziehungstypen können wie Objekte/Entities Attribute besitzen
- Beispiel:
  „Paul besitzt seit 01.01.2000 einen PKW der Marke X mit dem amtlichen Kennzeichen SI-XY 1. Ein PKW der Marke Y mit dem amtlichen Kennzeichen SI-YX 11 ist seit 01.06.2000 im Eigentum von Peter“

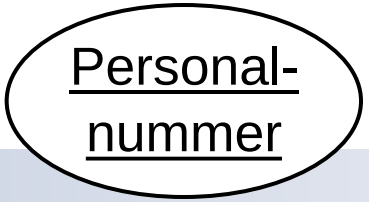


Primärschlüssel

- Attribute werden in **beschreibende** und **identifizierende Attribute** unterteilt
  - Ein identifizierendes Attribut (oder eine Kombination von mehreren Attributen) eines Entitätstyps kennzeichnet alle Entitäten eindeutig
  - Der Primärschlüssel ist eine minimale Menge von Attributen, die einen Entity in einem Entitytypen eindeutig identifizieren
- Graphische Notation:

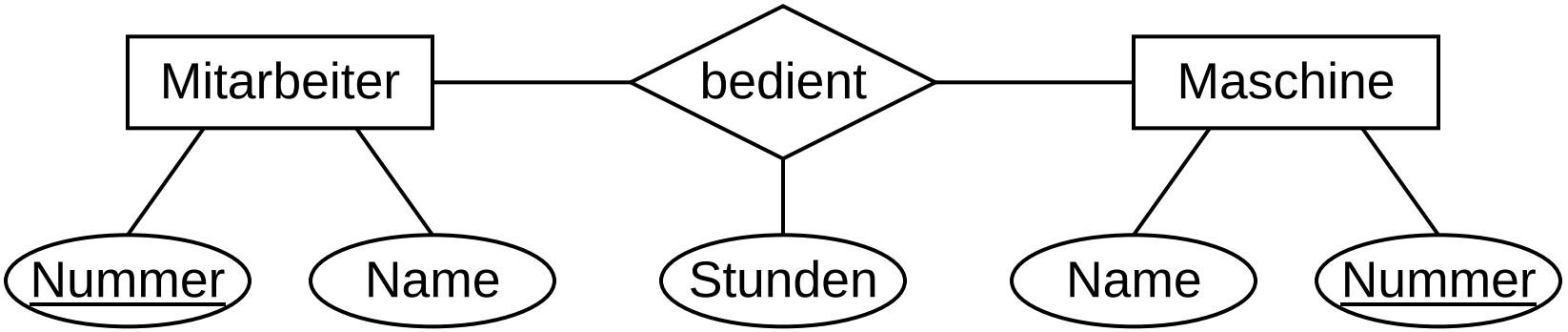


- Beispiel: „Personalnummer“ 1234 des „Mitarbeiters“ Meier identifiziert diesen



Beispiel

„Es gibt mehrere Mitarbeiter, die mit Namen und Nummer näher beschrieben werden sowie mehrere Maschinen, die ebenfalls Nummer und Name besitzen. Mitarbeiter können Maschinen bedienen, wobei die Dauer der Bedienung näher durch die Angabe der Stunden beschrieben wird“



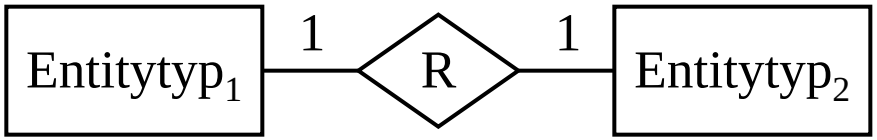
Kardinalitäten von Beziehungstypen

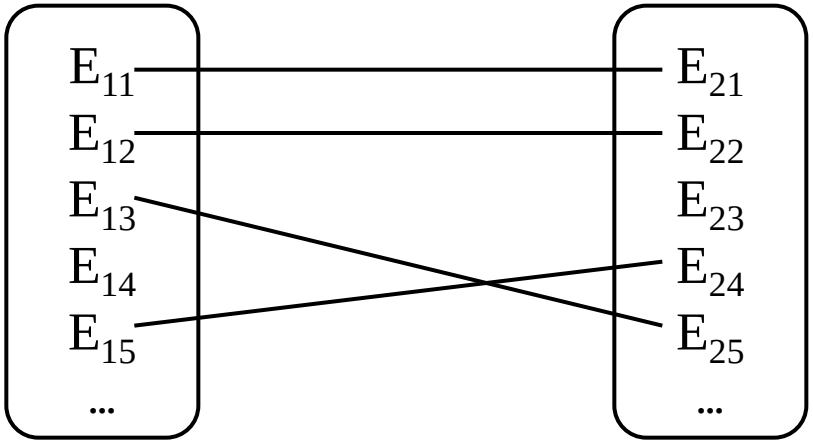
- Die Beziehung zwischen Entity-Typen kann durch die **Kardinalität**, die Anzahl der Beziehungen, die jedes einzelne Entity der Menge *höchstens* eingehen kann, näher beschrieben werden

- Typen binärer Beziehungen zwischen zwei Entity-Typen:
  - 1:1-Beziehungen,
  - 1:N-Beziehungen,
  - N:1-Beziehungen und
  - N:M-Beziehungen.

1:1-Beziehungstyp

- Jedes Entity E_{1x} kann mit maximal einem Entity E_{2x} die benannte Beziehung eingehen und umgekehrt

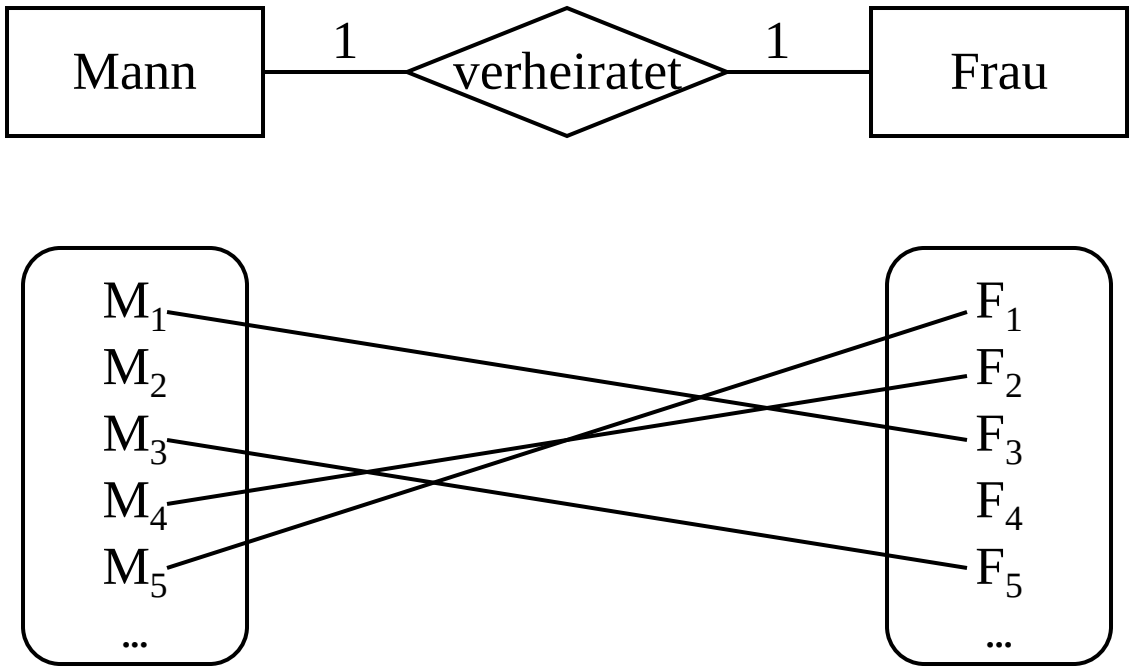




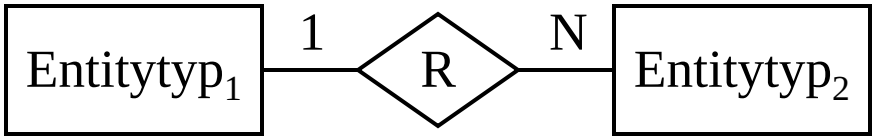
- Einem Entity des ersten Entitytypen wird höchstens ein Element des zweiten Entitytypen zugeordnet und umgekehrt
- Jede Instanz von $Entitytyp_1$ ist mit **höchstens einem** (0 oder 1) Element aus $Entitytyp_2$ assoziiert, eine Instanz von $Entitytyp_2$ mit **höchstens einem** (0 oder 1) von $Entitytyp_1$

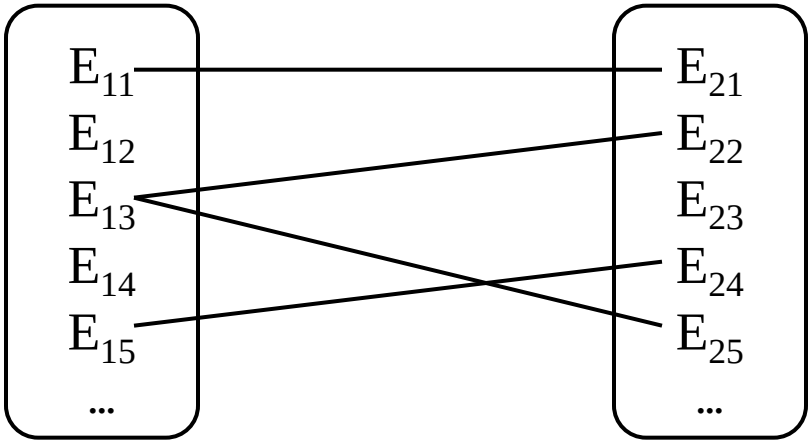
JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

Beispiel: 1:1-Beziehungstyp



1:N-Beziehungstyp

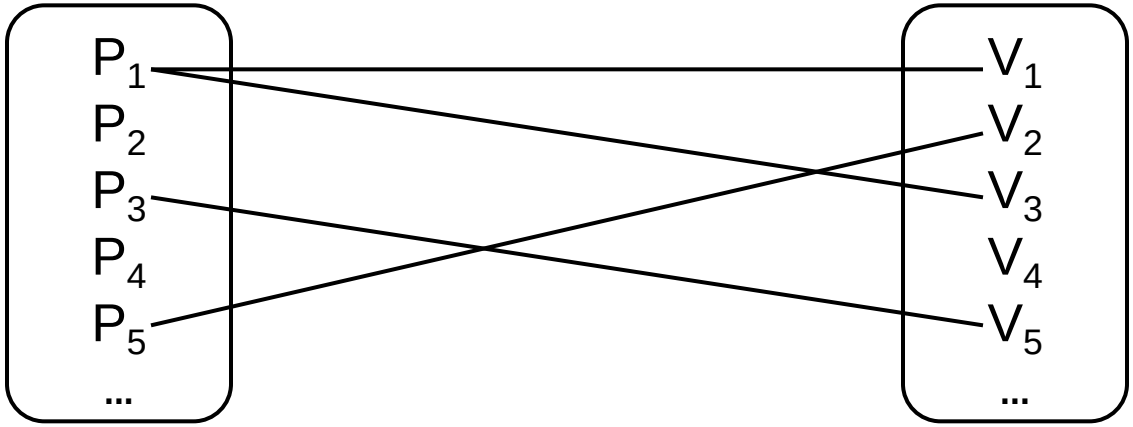




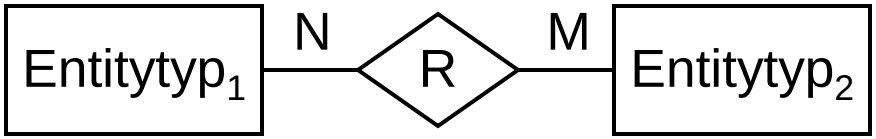
- Einem Entity des ersten Entitytypen werden
  - höchstens N (N=0,1,2,...), d.h keine, ein oder mehrere Entities des zweiten Entitytypen zugeordnet;
  - jedem Entity des zweiten Entitytypen wird höchstens ein (d.h. kein oder ein) Entity des ersten Entitytypen zugeordnet.
- Einer Instanz aus E_1 können N Elemente aus E_2 zugeordnet sein; einer Instanz aus E_2 kann höchstens ein Element aus E_1 zugeordnet werden.

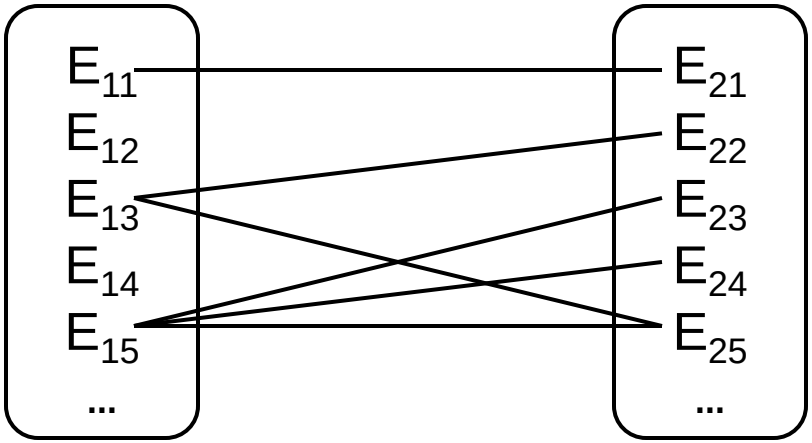
Beispiel: 1:N-Beziehungstyp

„Professoren lesen Vorlesungen. Jeder Professor kann mehrere Vorlesungen lesen, jede Vorlesung wird von einem Professor gelesen.“



N:M-Beziehungstyp

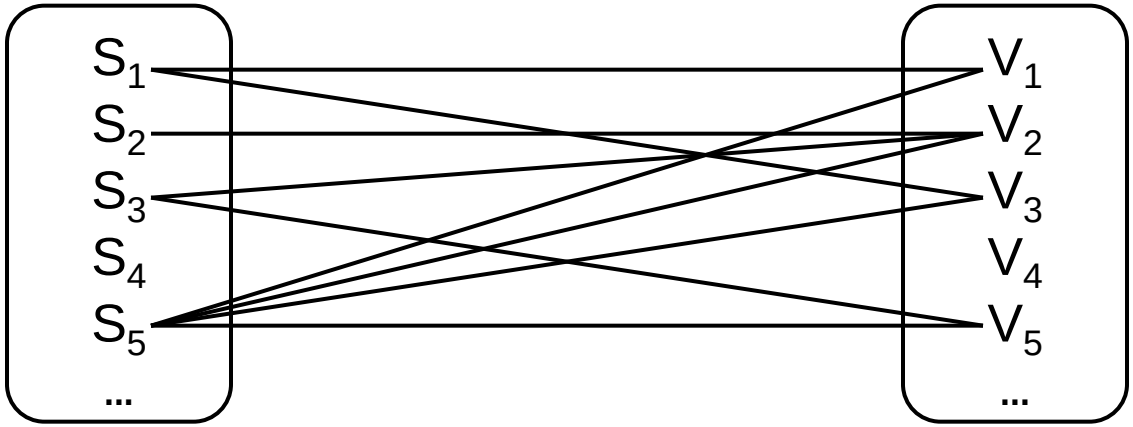




- Jedem Entity des zweiten Entitytypen werden höchstens N (N=0,1,2,...), d.h. keine, eine oder mehrere Entities des ersten Entitytypen zugeordnet
- umgekehrt werden jedem Entity des ersten Entitytypen höchstens M (M=0,1,2,...), d.h. keine, eine oder mehrere Entities des zweiten Entitytypen zugeordnet

Beispiel: N:M-Beziehungstyp

„Jeder Student kann mehrere Vorlesungen hören. Die Vorlesungen werden von mehreren Studenten gehört.“



Beispiel

Ein Mitarbeiter hat einen Namen und einen Wohnort. Ein Mitarbeiter arbeitet in einer Abteilung. Ein Mitarbeiter arbeitet an mehreren Projekten. Die Zeit, die ein Mitarbeiter für ein Projekt investiert, ist festgeschrieben. Jede Abteilung hat einen Namen. Jedes Projekt hat einen Namen. In einer Abteilung sind mehrere Mitarbeiter beschäftigt. An einem Projekt arbeiten mehrere Mitarbeiter.

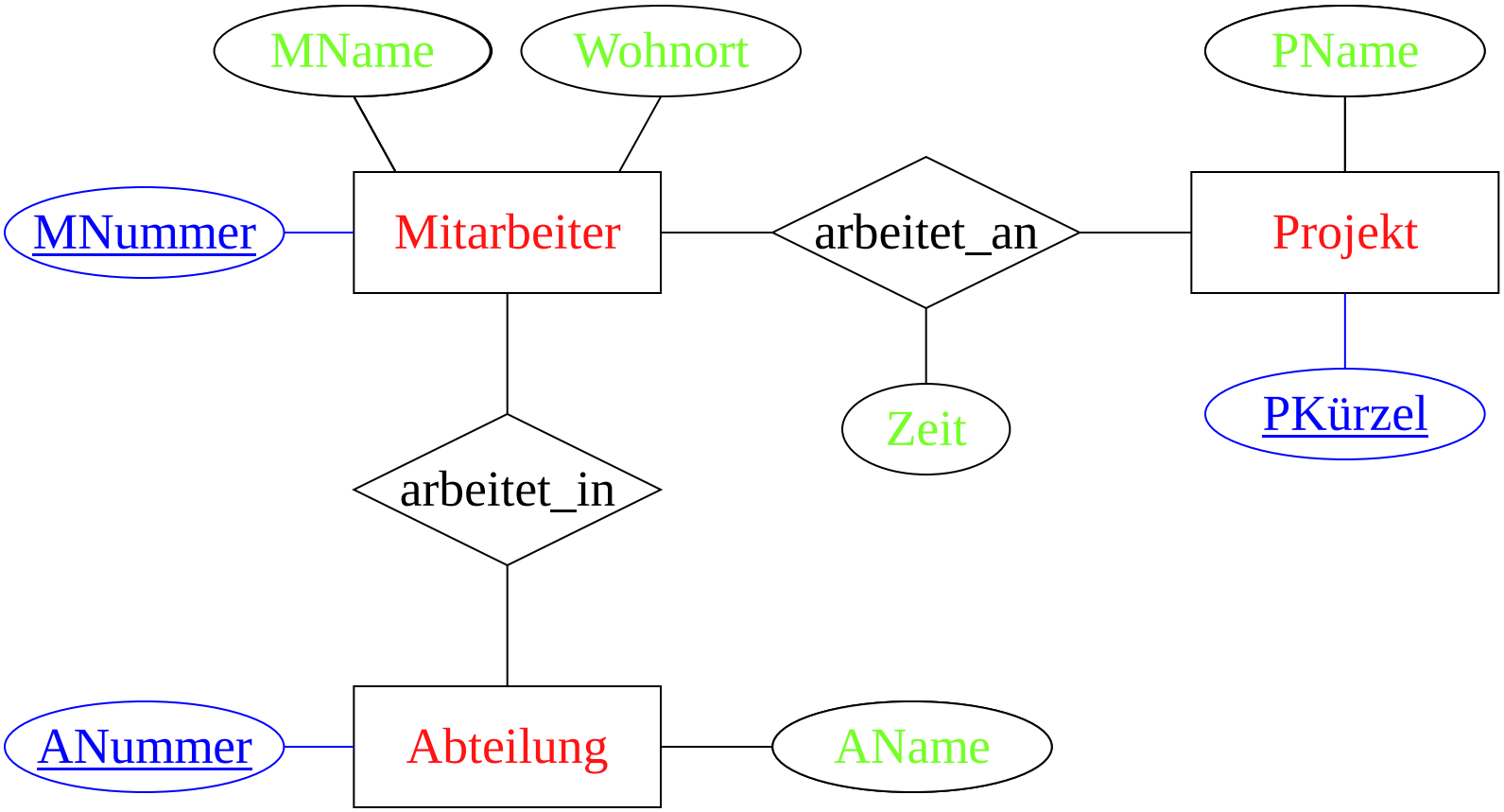
JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

Beispiel

Ein Mitarbeiter hat einen Namen und einen Wohnort. Ein Mitarbeiter **arbeitet in** *einer* Abteilung. Ein Mitarbeiter **arbeitet an** mehreren Projekten. Die Zeit, die ein Mitarbeiter für ein Projekt investiert, ist festgeschrieben. Jede Abteilung hat einen Namen. Jedes Projekt hat einen Namen. In einer Abteilung sind *mehrere* Mitarbeiter beschäftigt. An einem Projekt arbeiten *mehrere* Mitarbeiter.

JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

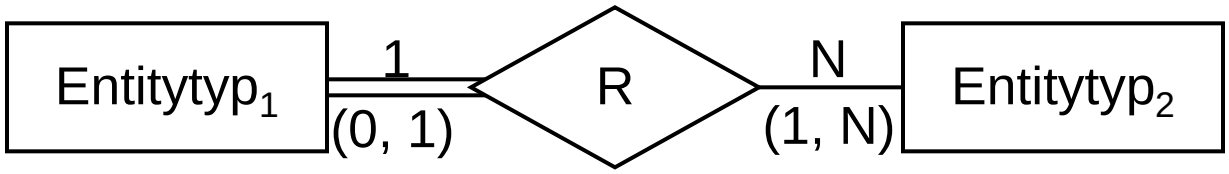
ER-Diagramm zur „Beispielminiwelt“



Minimalkardinalitäten in Chen-Notation

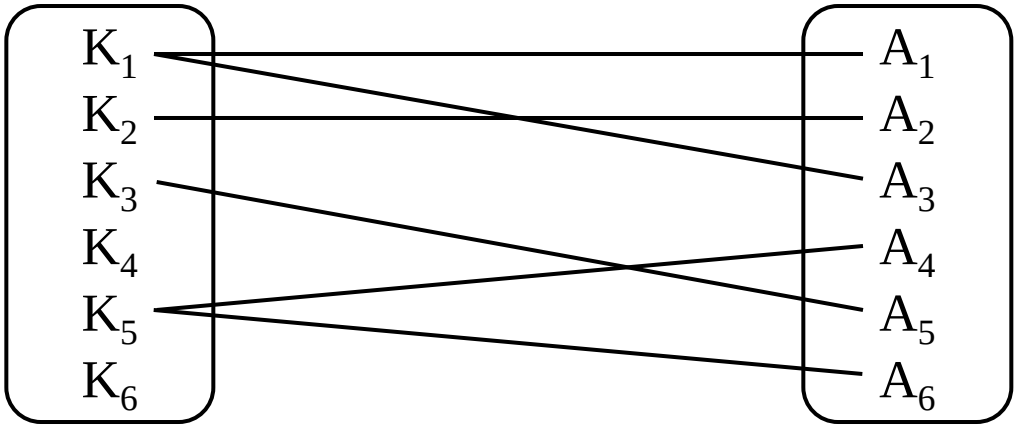
- Bisherige Notation gibt nur die Maximalkardinalitäten an
- **Minimalkardinalitäten** geben an, ob ein Element eines Entity-Typen eine Beziehung zu mindestens einem anderen Entity-Typen eingehen muss (obligatorisch) oder nicht (optional)

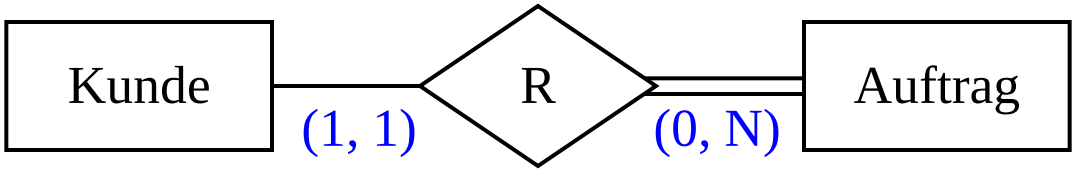
- Beispiel: Jede Instanz aus $Entitytyp_1$ ist mit mindestens einem und höchstens N Elementen aus $Entitytyp_2$ assoziiert. Eine Instanz von $Entitytyp_2$ mit höchstens einem Element von $Entitytyp_1$



Beispiel: „Kundenauftrag“

- Jeder Kunde kann mehrere Aufträge erteilen
- Jeder Auftrag wird von **genau einem** Kunden erteilt





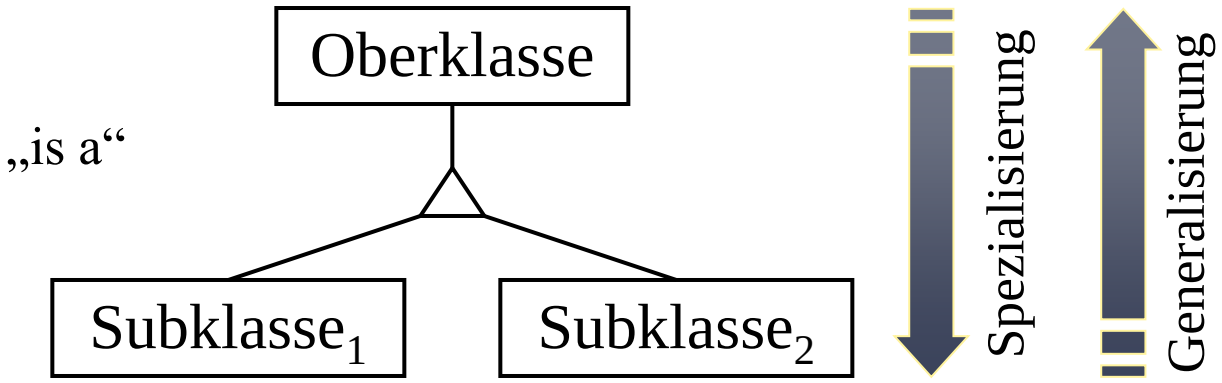
Beispiele

- Beispiel „Prüfung“: Eine Klausur umfasst mehrere Vorlesungen, wobei mindestens eine Vorlesung geprüft wird. Eine Vorlesung wird in maximal einer Klausur abgeprüft.

- Beispiel „Projektarbeit“: An einem Projekt arbeitet mindestens ein Mitarbeiter. Jeder Mitarbeiter kann an mehreren Projekten beteiligt sein.

Generalisierung/Spezialisierung („is a“)

- Haben verschiedene Entitätstypen gleiche Attribute, so können diese in einer Oberklasse zusammengefasst werden (Generalisierung)
- Die ursprünglichen Entitätstypen werden Subklassen genannt, die durch Spezialisierung aus der Oberklasse abgeleitet werden



- Von der Oberklasse erben die Subklassen:
  - die Attribute und
  - die Beziehungen

Generalisierung/Spezialisierung

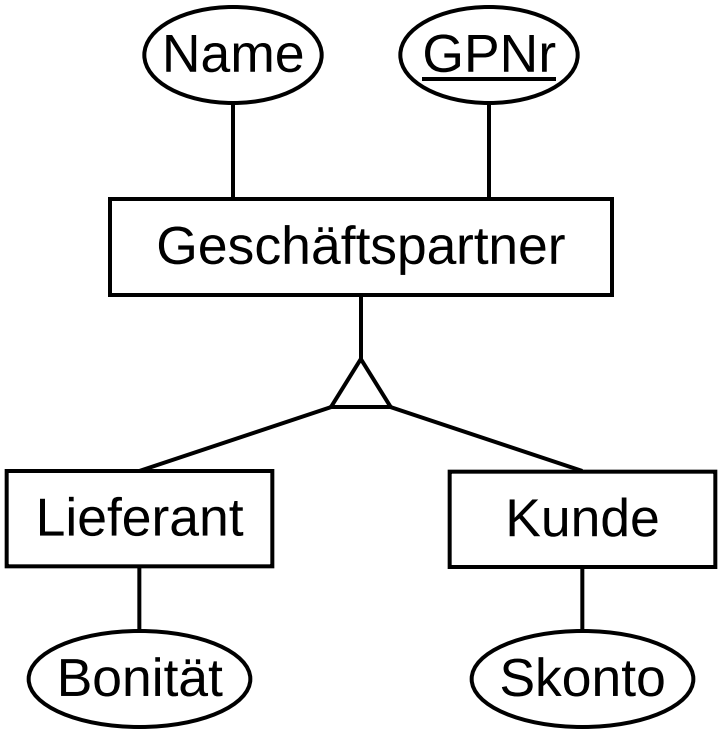
- Arten
  - Disjunkt (vs. nicht disjunkt bzw. überlappend)
    - Jede Entität der Oberklasse *gehört höchstens* einer Subklassen an
  - Total (vs. partial bzw. partiell)
    - Jede Entität der Oberklasse *gehört mindestens* einer Subklasse an

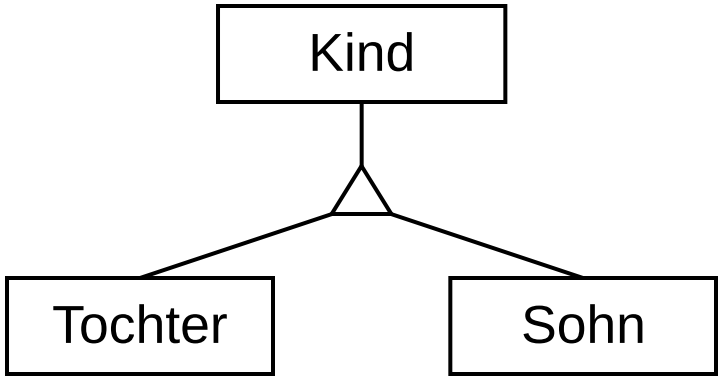
Beispiele

- Total, disjunkt
  - Oberklasse: Person
  - Subklassen: Mann, Frau
- Partiell, nicht disjunkt
  - Oberklasse: Person
  - Subklassen: Mann, Angestellter
- Partiell, disjunkt
  - Oberklasse: Fahrzeug
  - Subklassen: Auto, Fahrrad
- Total, nicht disjunkt:
  - Oberklasse: Hochschulangehöriger
  - Subklassen: Studierender, Angestellter

Beispiele

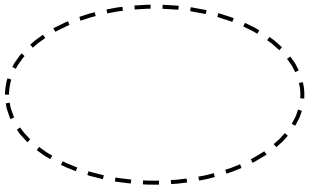
- Disjunkt?
- Total?



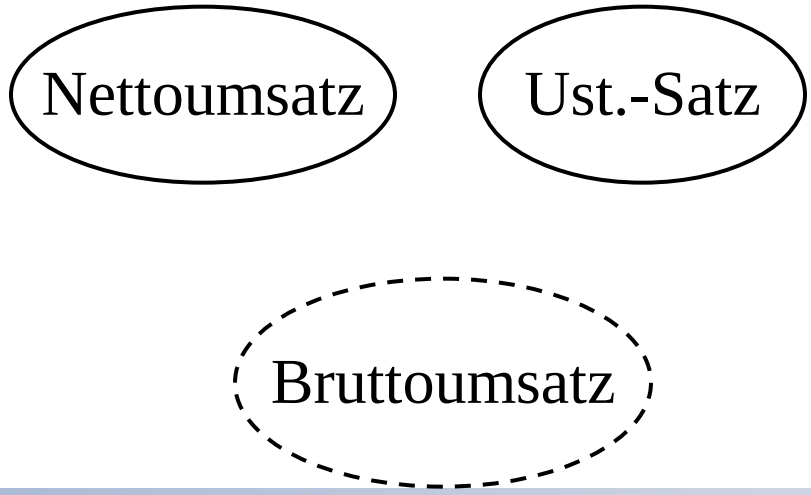


Abgeleitetes Attribut

- Abgeleitete Attribute lassen sich aus anderen Attributen des Typs berechnen
- Graphische Notation:

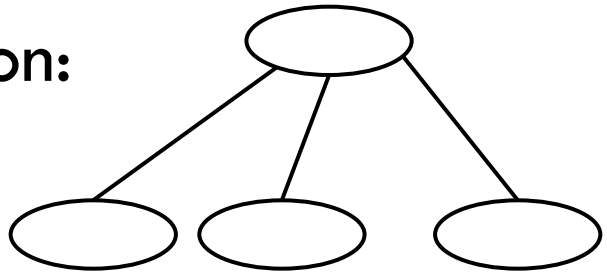


- Beispiel:

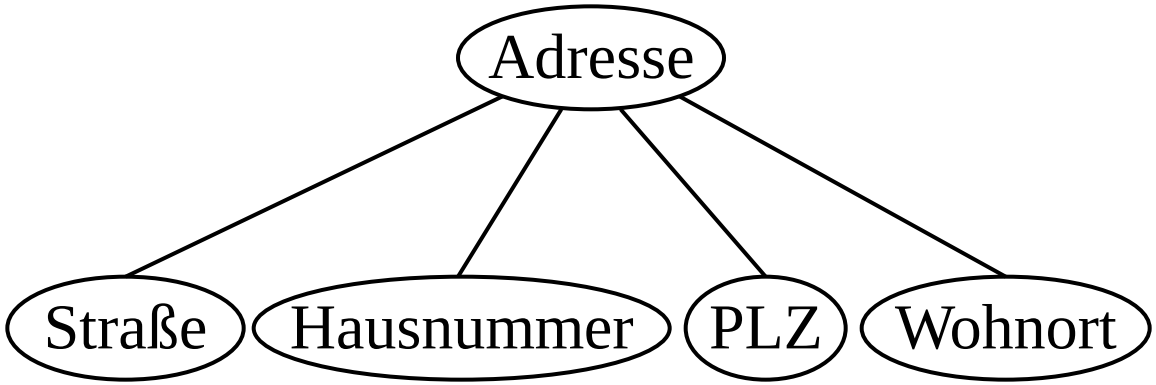


Zusammengesetztes Attribut

- Attribute, die aus mehreren Attributen bestehen – also nicht atomar sind – heißen **zusammengesetzt**
- Graphische Notation:

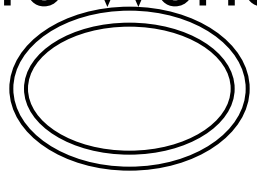


- Beispiel:



Mehrwertiges Attribut

- Mehrwertige Attribute (evtl. nicht atomar!) erlauben, dass ein Attribut mehrere Werte annehmen kann
- Graphische Notation:

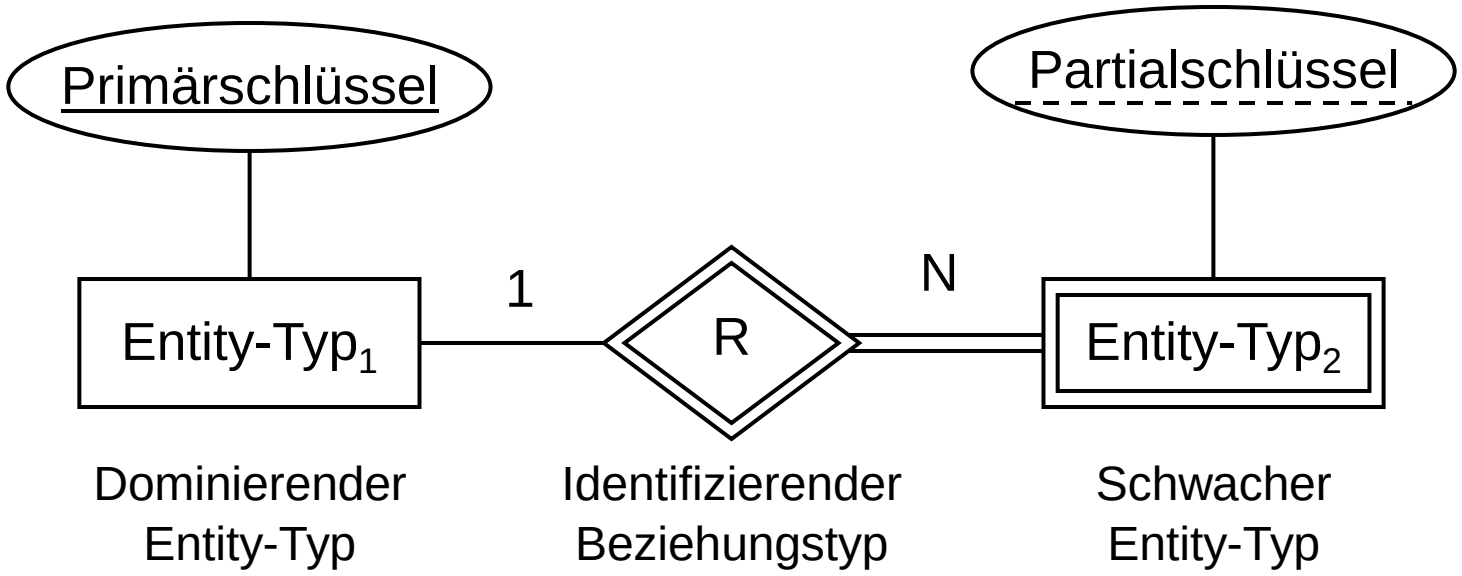


- Beispiel:

Weiteres Beispiel: Vornamen, z.B. Karl-Theodor Maria Nikolaus Johann Jacob Philipp Franz Joseph Sylvester zu Guttenberg.

Schwacher (existenzabhängiger) Entity-Typ

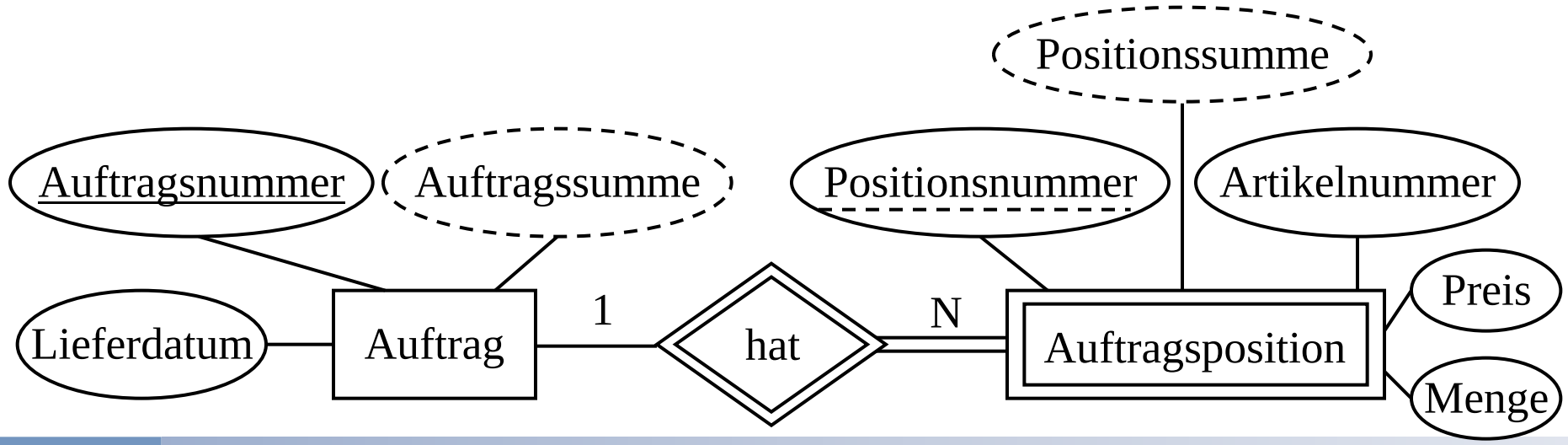
- Die Existenz von Entities eines schwachen Entitytyps („weak entity“) hängt von der Existenz eines Entities des dominierenden Typen ab
- Abhängigkeit wird durch identifizierende Beziehung modelliert



Beispiel: Schwacher Entity-Typ

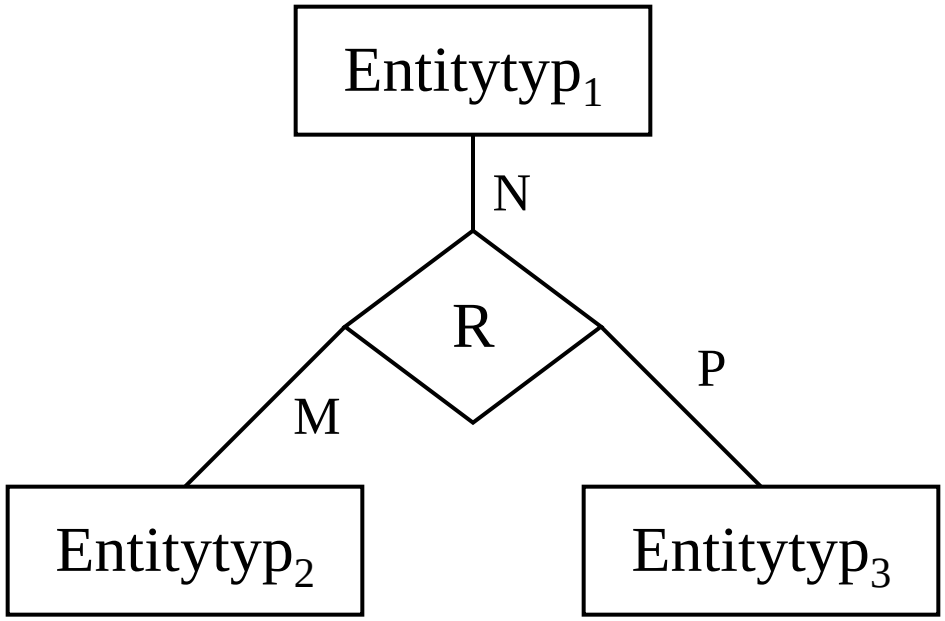
Auftragsnummer: xyz **Lieferdatum:** TT.MM.JJJJ

Position	*Artikel*	*Bezeichnung*	*Preis*	*Menge*	*Positionssumme*
1	4711	Tapete	6,50 €	6	39,00 €
2	874	Kleister	4,00 €	2	8,00 €
3	...	...	...	...	...
			Auftragssumme:		**...**

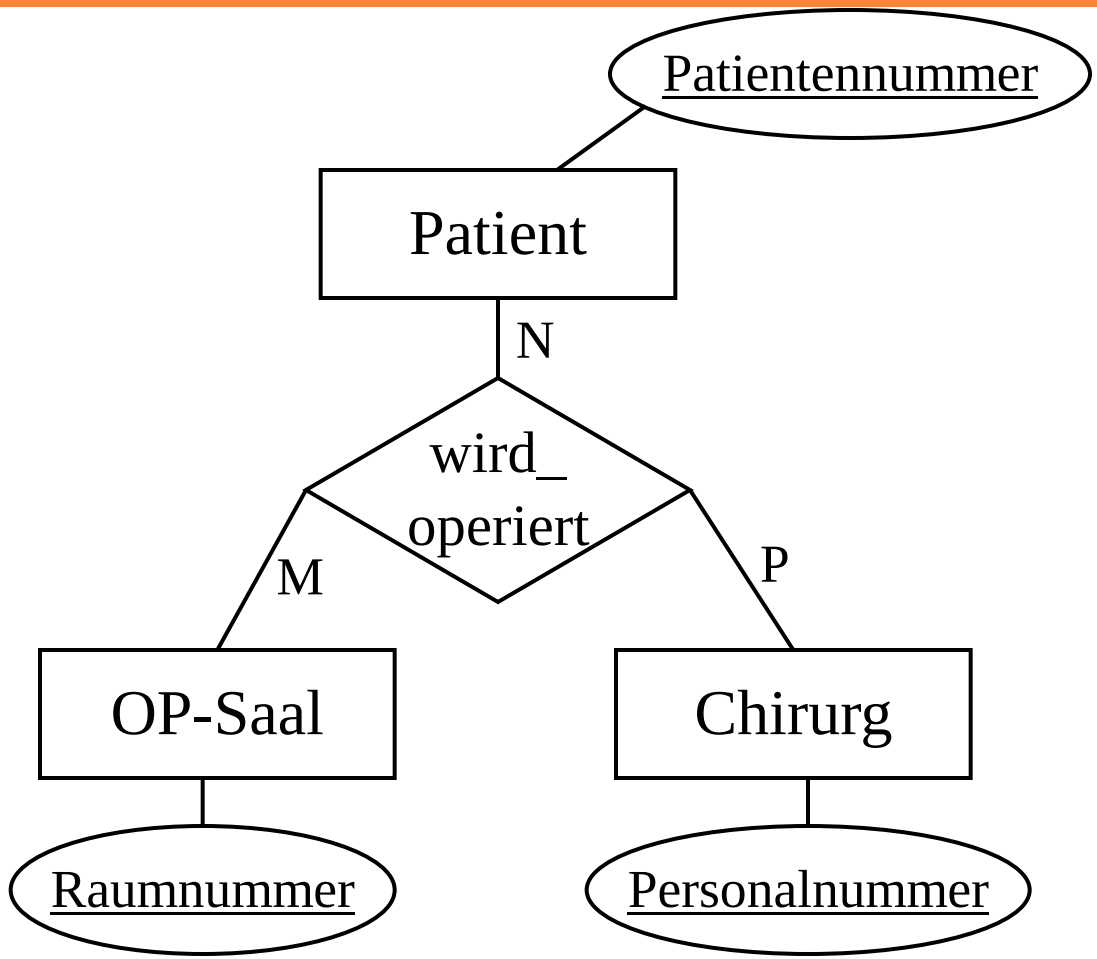


Beziehungstyp vom Grad 3

- Zur Modellierung bestimmter Sachverhalte können binäre Beziehungen evtl. nicht ausreichen
- Beziehungen zwischen drei Entitytypen können explizit modelliert werden



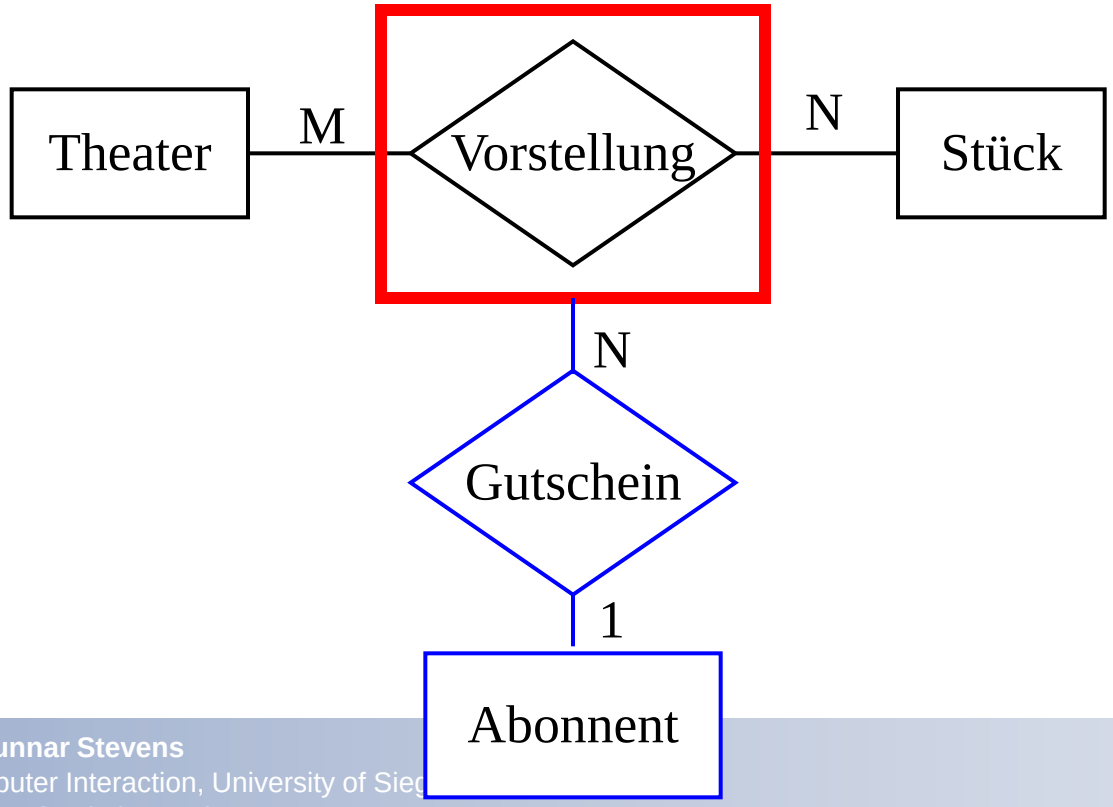
Beispiel: Beziehungstyp vom Grad 3



JProf. Dr. Gunnar Stevens
Human Computer Interaction, University of Siegen
gunnar.stevens@uni-siegen.de

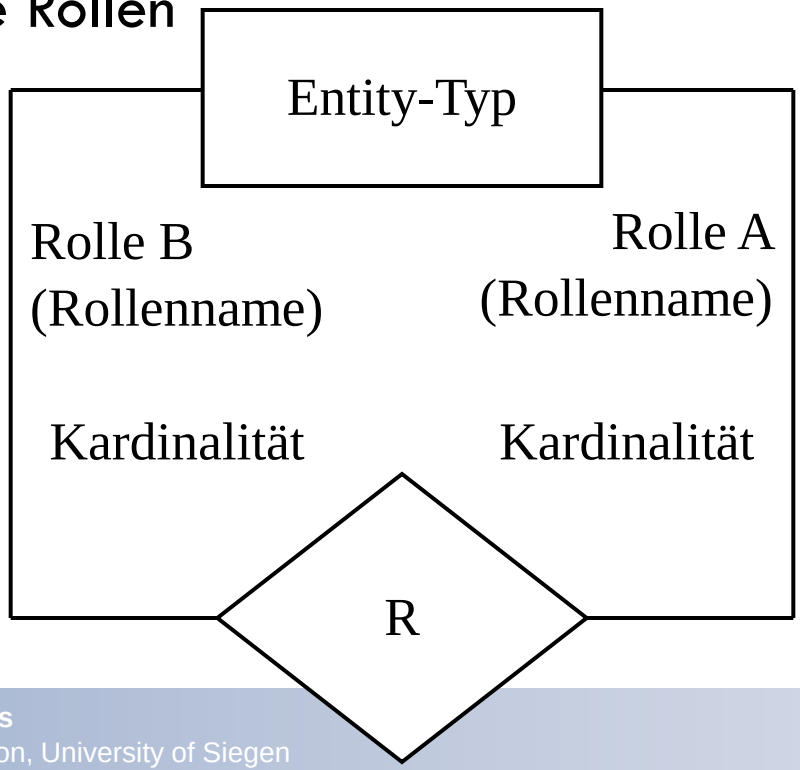
Uminterpretierter Beziehungstyp

- Es dürfen nur Beziehungen zwischen Entitäten, nicht zwischen Beziehungen modelliert werden
- Lösung: Beziehung als Entität modellieren



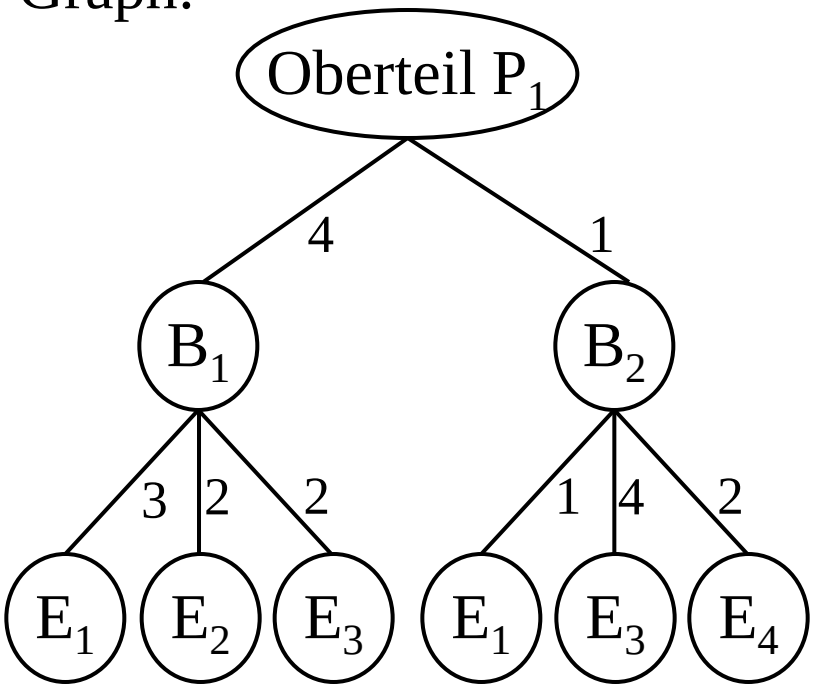
Rekursive Beziehung/Assoziation

- Derselbe Entity-Typ nimmt mehr als einmal an einem Beziehungstypen teil
- Verdeutlichung der Semantik der Beziehung durch unterschiedliche Rollen

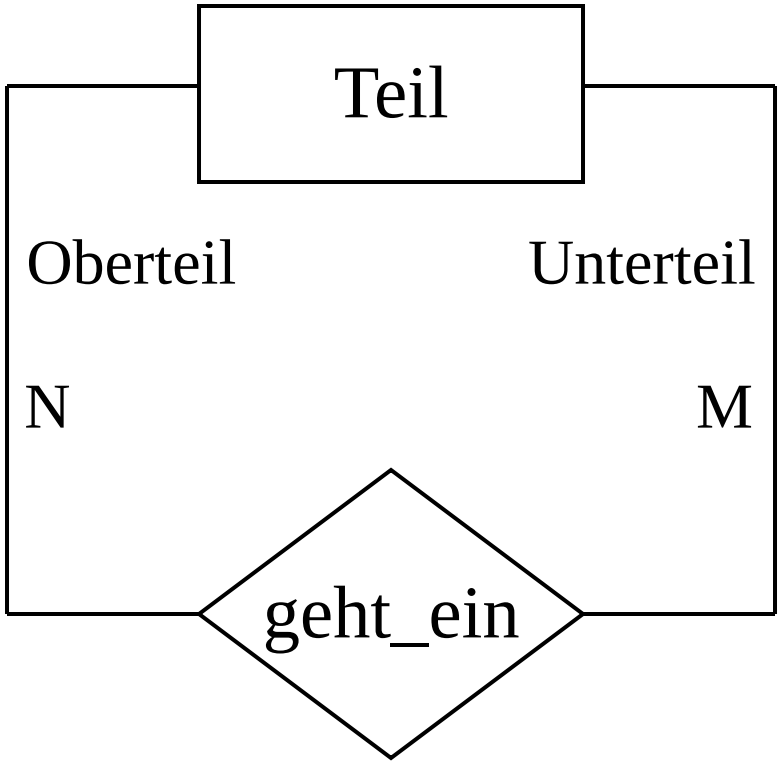


Beispiel: Rekursive Beziehung I

Graph:



ERM-Darstellung:

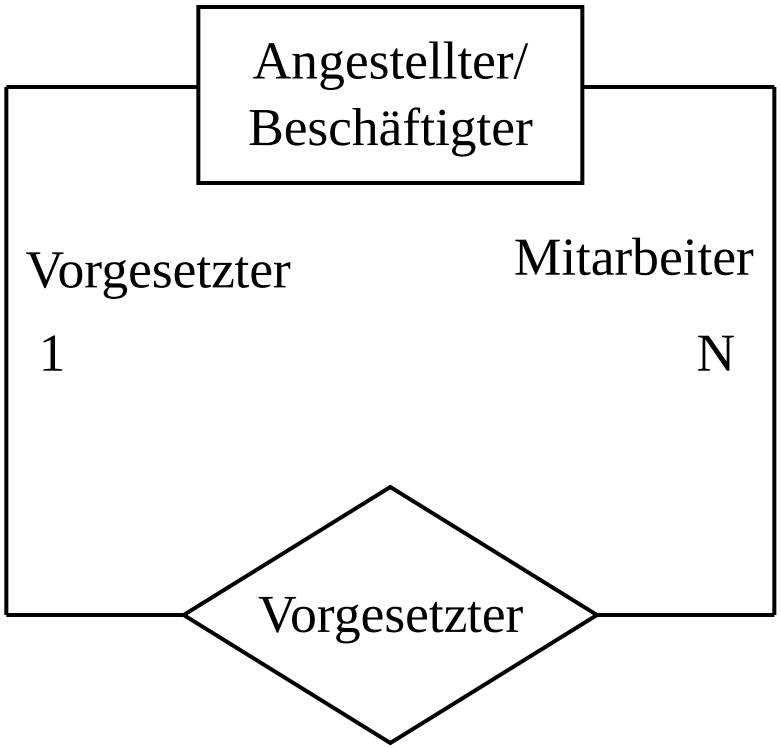


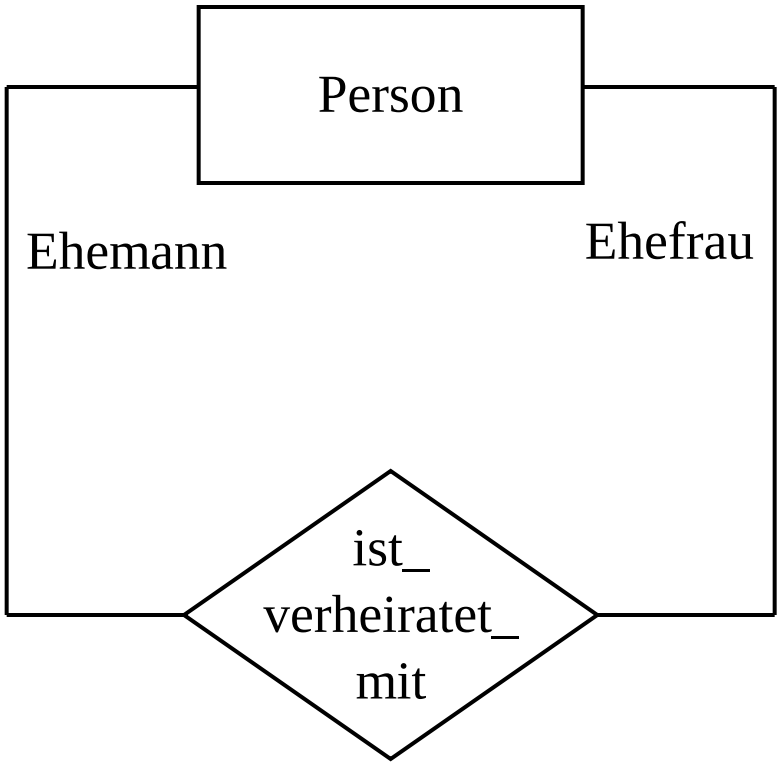
E_i = Einzelteil i, i=1, ..., n

B_j = Baugruppe j, j=1, ..., m

P_k = Enderzeugnis k, k=1, ..., p

Beispiele: Rekursive Beziehung II





NÄCHSTES MAL:

- UMSETZUNG DER PLANUNG
- ABFRAGE DER DATEN