Kapitel 10: Datenspeicherung

10. Datenspeicherung

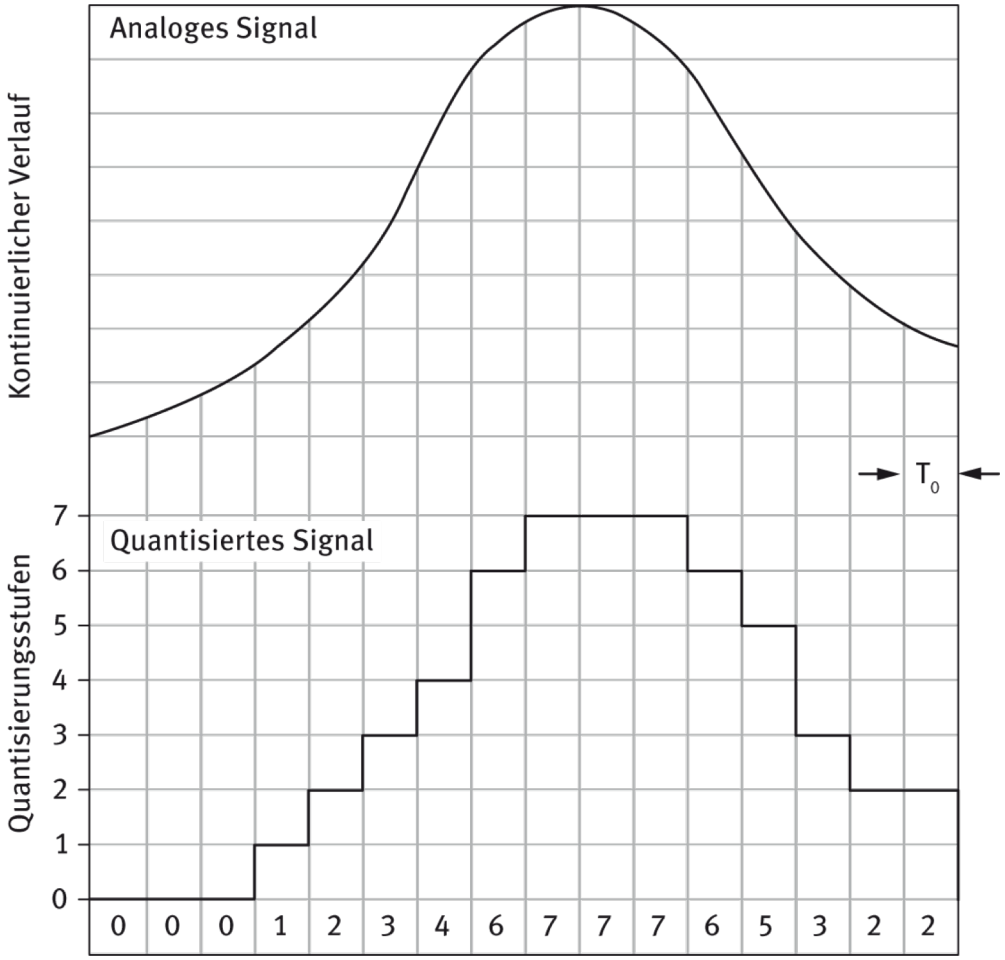
1. Information und Daten
2. Datenstrukturen
3. Datenbanken
4. Dokumentzentrierte Datenorganisation
5. Skalierbare Datenspeicherung

10.1 Information und Daten

Information und Daten

- **Daten** (engl.: data) stellen **Information** (das heißt Angaben über Sachverhalte und Vorgänge; engl.: information) aufgrund bekannter oder unterstellter Abmachungen in einer maschinell verarbeitbaren Form dar.
- **Digitale Daten** (engl.: digital data) werden durch Zeichen repräsentiert. Ein *Zeichen* (Synonym: Symbol; engl.: character, symbol) ist ein Element aus einer zur Darstellung von Information vereinbarten endlichen Menge von verschiedenen Elementen, dem sogenannten **Zeichenvorrat** (engl.: character set; Synonym: Alphabet, engl.: alphabet).
- **Analoge Daten** (engl.: analog data) werden durch kontinuierliche Funktionen repräsentiert. Die analoge Darstellung erfolgt durch eine physikalische Größe, die sich entsprechend den abzubildenden Sachverhalten oder Vorgängen stufenlos ändert.

Umwandlung von einem analogen Signal in digitalisierte Werte

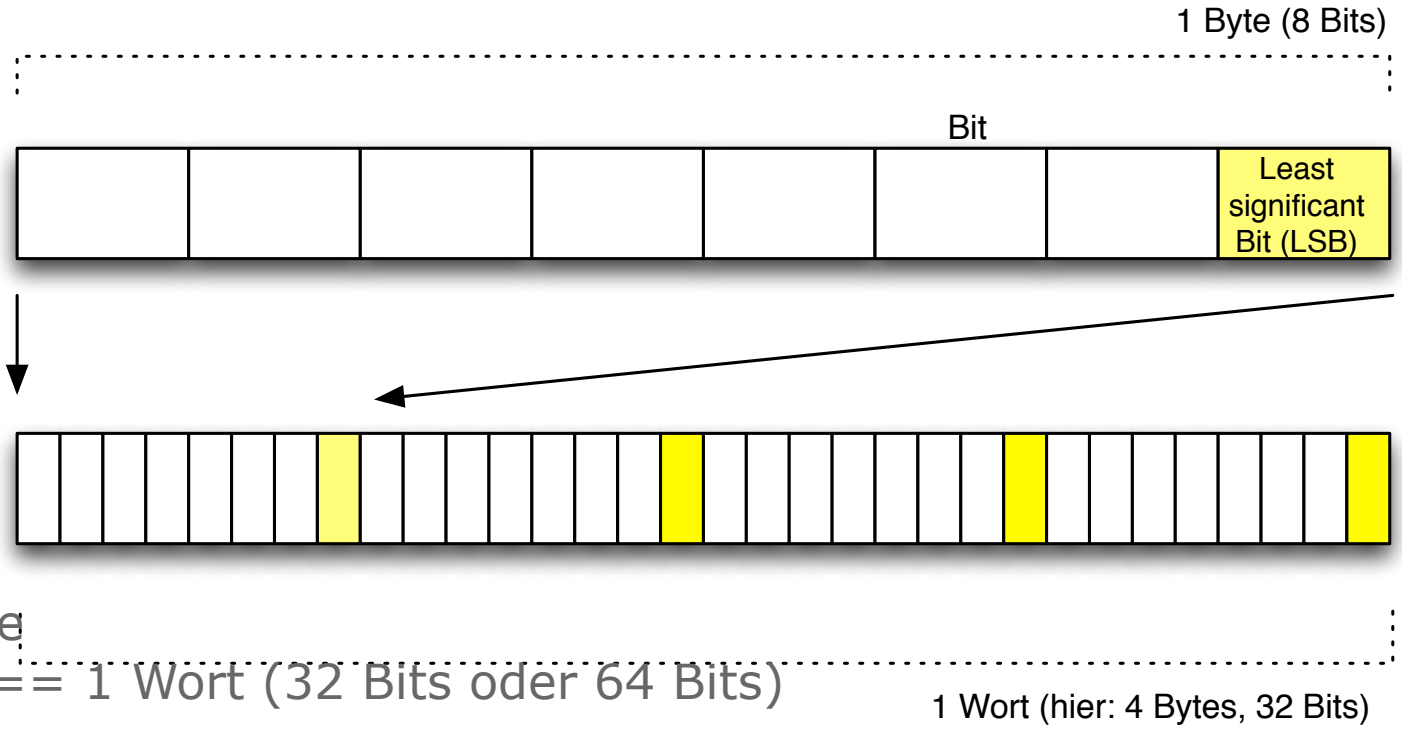


Bits und Bytes

- Ein **Binärzeichen** oder **Bit** (Synonyme; engl.: binary digit; bit) ist ein Zeichen aus einem Zeichenvorrat von zwei Zeichen. Zur Darstellung der Bits können beliebige Zeichen benutzt werden; wir verwenden die Zeichen 0 (binäre Null) und 1 (binäre Eins).
- Auf den heute üblichen Rechnersystemen ist ein **Byte** (engl.: byte) die kleinste adressierbare Einheit und entspricht einer Folge von 8 Bits.

Bits und Bytes

Einheiten:
- Bit (0,1)
- Byte
- Wort



8 Bits == 1 Byte
4 oder 8 Bytes == 1 Wort (32 Bits oder 64 Bits)

Typische Größenangaben

- Speicher: KB (Kilo Byte), MB, GB, TB, PB
- Datenübertragung: Kbit (Kilo Bit), Mb, Gb,...

Stellenwertsystem

- Ein Zahlensystem, bei dem der Wert einer Ziffer innerhalb einer Ziffernfolge von ihrer Stellung (ihrer Position) abhängt, heißt **Stellenwertsystem**. Bei Stellenwertsystemen nimmt der Wert einer Ziffer von Ziffernposition zu Ziffernposition jeweils um einen Faktor zu, welcher der *Basis* des Zahlensystems entspricht.
- Das **dezimale Zahlensystem** (engl.: decimal number system) ist ein Stellenwertsystem mit der Basis 10 und umfasst somit einen Ziffernvorrat von zehn Ziffern (0, 1, ..., 9).
- Das **Dualsystem** (engl.: binary number system) ist ein Stellenwertsystem mit der Basis 2 und verwendet entsprechend zur Darstellung von Werten zwei verschiedene Ziffern.
- Das **Oktalsystem** (engl.: octal number system) ist ein Stellenwertsystem mit der Basis 8, das **Hexadezimalsystem** (engl.: hexadecimal number system) ist ein Stellenwertsystem mit der Basis 16.

Exkurs: Stellenwertsysteme

- Unterscheidung Ziffernwert und Stellenwert
  - Dezimalzahl „675“:
    $675 = 6 \times 10^2 + 7 \times 10^1 + 5 \times 10^0$
    - Ziffernwert: 0..9,
    - Stellenwert: Anzahl der Ziffernwerte hoch Position
    - Dezimalsystem: Stellenwertsystem zur Basis 10
  - Allgemein:

$$W = \sum_{i-0}^{n-1} b_i \times B^i$$

  - b_i Ziffernwert, B^i Stellenwert der i-ten Ziffer
- Beispiele für Stellenwertsysteme: Dezimalsystem, Dualsystem, Oktalsystem, Hexadezimalsystem

Beispiele

- Zahlenwert zur Basis 10 (Dezimalzahl)

$$W = \sum_{i-0}^{n-1} b_i \times B^i$$

$675 = 6 \times 10^2 + 7 \times 10^1 + 5 \times 10^0 = 600 + 70 + 5$

- Zahlenwert zur Basis 2 (Dualzahl)

$1100_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 = 8 + 4 + 0 + 0 = 12$

- Maximaler Wert, der in einem Byte (8 Bits) dargestellt werden kann:

$11111111_2 = 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255 = 2^8\text{-}1$

- Dezimalzahl 65 als binärer Wert:

$01000001_2 = 0\times128 + 1\times64 + 0\times32 + 0\times16 + 0\times8 + 0\times4 + 0\times2 + 1\times1 = 65$

Codierung von ganzen Zahlen

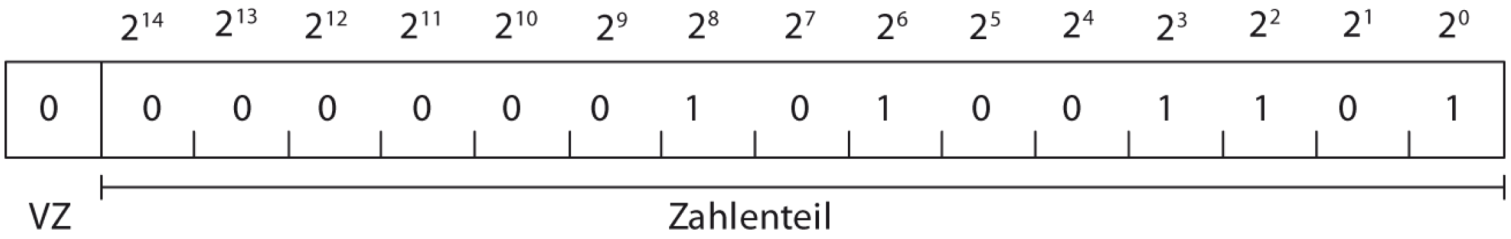
- Die **Repräsentationsgröße** (engl.: representation size) legt die vorgesehene Datenmenge (in Bits oder Bytes) fest, die für ein Datenelement vorgesehen ist. Sie bestimmt somit auch, wie viele unterschiedliche Werte in dem Datenelement dargestellt werden können.
- Positive ganze Zahlen:

Repräsentationsgröße	Minimaler Wert	Maximaler Wert
8 Bit (1 Byte)	0	$255 = 2^8-1$
16 Bit (2 Byte)	0	$65.535 = 2^{16}-1$
32 Bit (4 Byte)	0	$4.294.967.295 = 2^{32}-1$
64 Bit (8 Byte)	0	$18.446.744.073.709.551.615 = 2^{64}-1$

- Ganze Zahlen mit Vorzeichen:

Repräsentationsgröße	Minimaler Wert	Maximaler Wert
8 Bit (1 Byte)	–128	127
16 Bit (2 Byte)	–32.768	32.767
32 Bit (4 Byte)	–2.147.483.648	2.147.483.647
64 Bit (8 Byte)	–9.223.372.036.854.775.808	9.223.372.036.854.775.807

Darstellung des Wertes 333 als Dualzahl mit Vorzeichen



Codierung von Kommazahlen

- Die **Festkommadarstellung** (engl.: fixed point representation) ist eine Form der ziffernweisen Codierung, bei der an einer (gedachten) Stelle das Komma eingefügt wird. Die Position des Kommas wird getrennt gespeichert.
- Die **Gleitkommadarstellung** (engl.: floating point representation) ist eine Form der Codierung von Kommazahlen, bei der jede Kommazahl durch den Zahlenwert (*Mantisse*) und eine Größenordnung (*Exponent*) dargestellt wird. Der Zahlenwert ergibt sich aus der Formel:
  $\text{Wert} = \text{Mantisse} \times \text{Basis}^{\text{Exponent}}$

Codierung von Texten

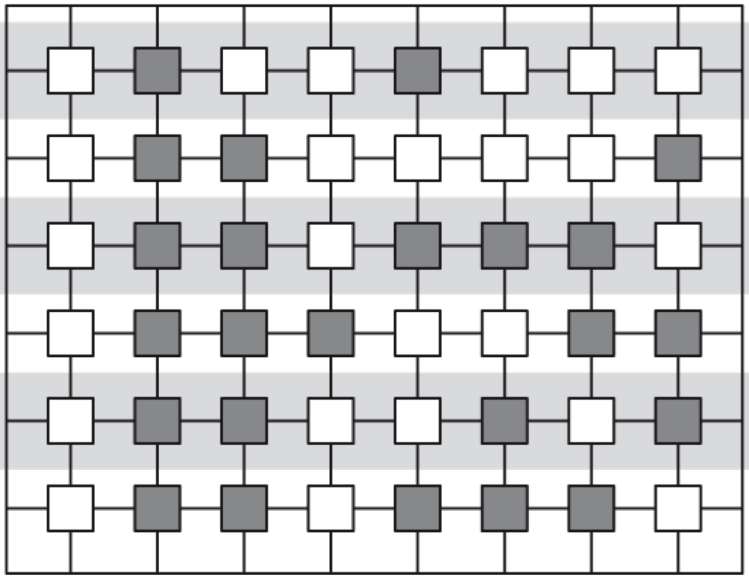
- Eine **Zeichencodierung** (engl.: character encoding) bestimmt die Darstellung (Codierung) von 3 Zeichen (beispielsweise Buchstaben, Ziffern oder Steuerzeichen) auf einem Rechner. Ein **Zeichensatz** (engl.: character set) bestimmt eine mögliche Zeichencodierung. Durch einen Zeichensatz wird jedem Zeichen ein Zahlenwert zugewiesen, der die Position des Zeichens innerhalb des Zeichensatzes bestimmt. Dieser Zahlenwert wird auch meistens bei der Sortierung von Texten, die durch den Zeichensatz dargestellt werden, herangezogen.
- **ASCII** (Abkürzung von engl.: American standard code for information interchange) ist ein genormter, relativ alter Zeichensatz für Schrift- und Steuerzeichen mit einer Repräsentationsgröße von 7 Bit. Der ASCII-Zeichensatz umfasst $2^7 = 128$ Zeichen (die Werte 0 bis 127). Der 7-Bit-ASCII-Code reicht aus, um beliebige englischsprachige Texte darstellen zu können. Umlaute sind in diesem Zeichensatz nicht enthalten.

7-Bit-ASCII-Code

Linkes Halbbyte \ Rechtes Halbbyte	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000																
0001																
0010		!	“	#	$	%	&	‘	(	)	*	+	,	_	.	/
0011	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0101	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
0110	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0111	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	–

- Ein **Schriftzeichen** (Zeichen, engl.: character) ist ein Buchstabe, eine Ziffer, ein Sonderzeichen oder das Leerzeichen. Unter *Sonderzeichen* versteht man allgemein gebräuchliche Zeichen, die nicht den Buchstaben oder Ziffern zuzuordnen sind, wie zum Beispiel + – * / = . , ; : ? ! $ ( ) [ ] { } % & usw.

Rechnerinterne Zeichendarstellung (ASCII-Code)

Bitkombinationen		Speicherung	Buchstabe
Linkes Halbbyte	Rechtes Halbbyte		
0100	1000		H
0110	0001		a
0110	1110		n
0111	0011		s
0110	0101		e
0110	1110		n

Unicode

- **Unicode** ist ein international genormter Zeichensatz, der mit dem Ziel entwickelt wird, eine einheitliche Codierung für jedes Textdokument aller Sprachen und Kulturen der Erde bereitzustellen. Unicode wird gemeinsam von dem Unicode-Konsortium und dem ISO/IEC-Standardkomitee 10646 definiert. Neben den Zeichen der westlichen und slawischen Sprachen werden durch Unicode unter anderem Zeichen für Arabisch und Hebräisch, Griechisch, Kyrillisch und Armenisch, Indisch, Einheitszeichen aus dem Chinesischen, Koreanischen und Japanischen, mathematische, technische und grafische Symbole sowie spezielle Zeichen für Anwendungen definiert.
- **UTF-8** (Abkürzung von engl.: Unicode transformation format) ist eine byteorientierte Codierung von Unicode, die unter anderem aus Kompatibilitätsgründen zu existierenden Systemen mit 8-Bit-Zeichen entwickelt wurde. Bei der UTF-8-Codierung besitzen unterschiedliche Zeichen unterschiedliche Repräsentationsgrößen. Ein Unicode-Zeichen entspricht einer Sequenz von ein bis vier Bytes in der UTF-8-Codierung.

10.2 Datenstrukturen

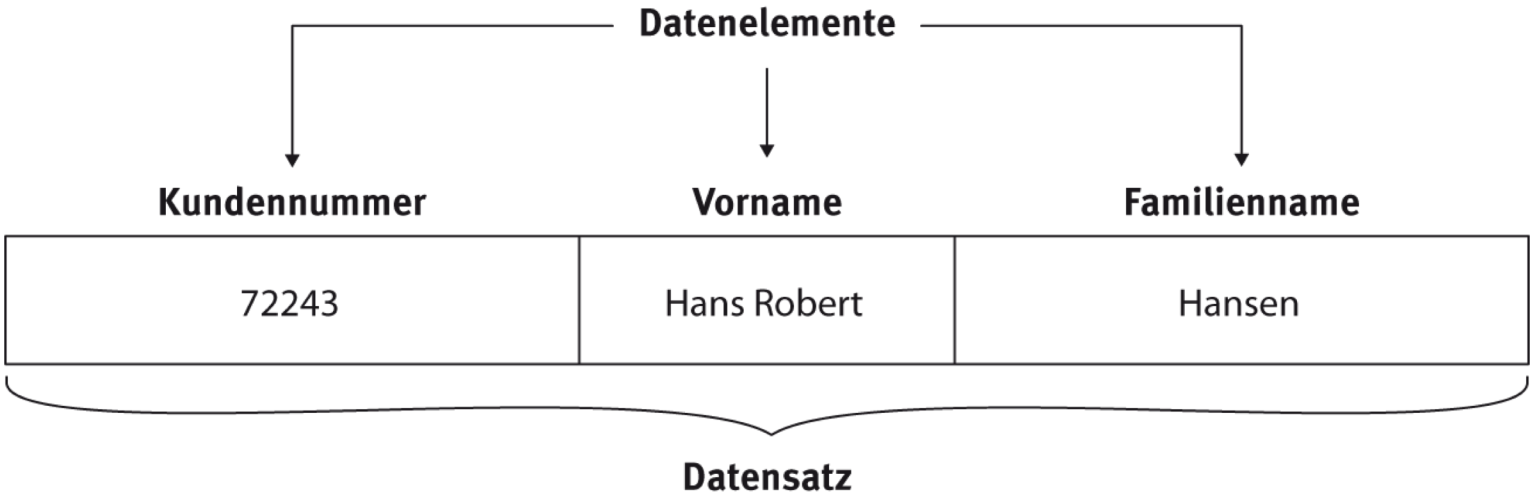
Datenelemente

- Ein **Datenelement** (engl.: data element) ist ein Speicherbereich, der einen *Namen* (Bezeichner, engl.: identifier), einen *Inhalt* (Wert, engl.: value) und einen *Datentyp* besitzt. Der Name dient zur Identifikation eines Datenelements. Der **Datentyp** (oder kurz Typ, engl.: data type) bestimmt, welche Operationen mit Werten dieses Typs durchgeführt werden können und wie die Werte im Rechner repräsentiert werden.
- Datenelemente, deren Werte durch Operationen verändert werden können, bezeichnet man als **Variablen** (engl.: variable). Sind die Werte von Datenelementen unveränderlich, so spricht man von **Konstanten** (engl.: constant).

Einfache Datenstrukturen

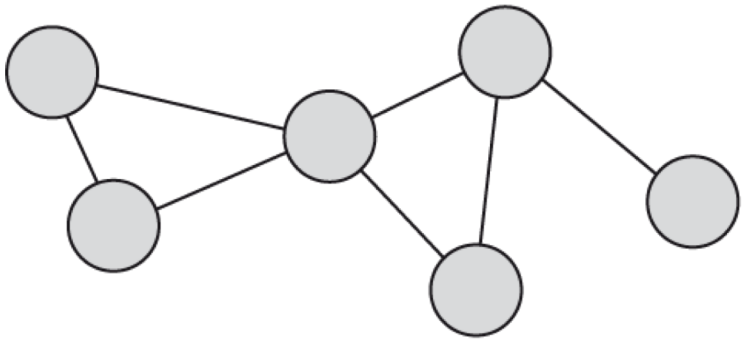
- Eine **Aggregation** (engl.: aggregation) drückt aus, dass eine bestimmte konzeptionelle Einheit *Bestandteil* (engl.: part of) einer anderen ist.
- Ein **Datensatz** (engl.: record) ist die Aggregation von *unterschiedlichen* Datenelementen. Die Aggregation von *gleichartigen* Datenelementen einer bestimmten Menge heißt **Array** (vielfach auch Feld, engl.: array).
- Unter einem **Zeiger** (engl.: pointer) versteht man einen Verweis (eine Referenz) auf eine Speicheradresse. Diese Speicheradresse entspricht dem Ort, an dem der Wert des referenzierten Datenelements gespeichert ist.
- Die **Datenstruktur** (engl.: data structure) bezeichnet die Summe aller elementaren und komplexen Datenelemente inklusive ihrer Referenzen.

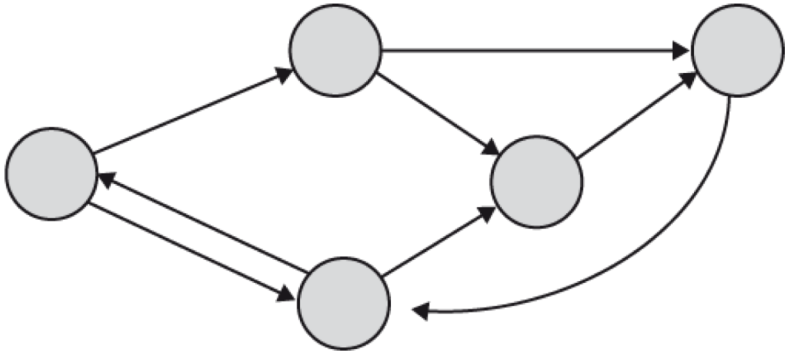
Datensatz


Datenelemente
Kundennummer
Vorname
Familienname
72243
Hans Robert
Hansen
Datensatz

Graphenbasierte Datenstrukturen

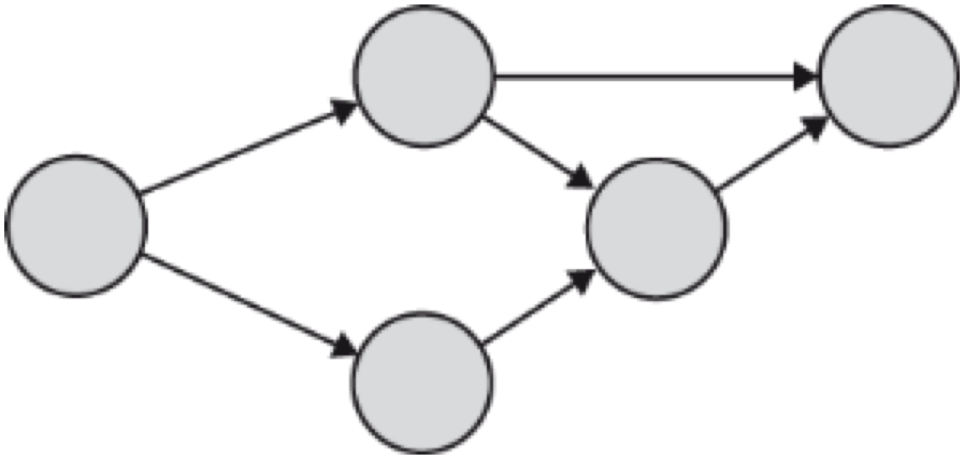
- Ein **Graph** (engl.: graph structure) ist eine Datenstruktur, die aus **Knoten** (engl.: node) und **Kanten** (engl.: edge) aufgebaut ist, wobei die Knoten durch Kanten verbunden sind. Man spricht von einem **gerichteten Graphen** (engl.: directed graph), wenn die Kanten nur in einer Richtung durchlaufen werden können, andernfalls ist der Graph **ungerichtet** (engl.: undirected graph).





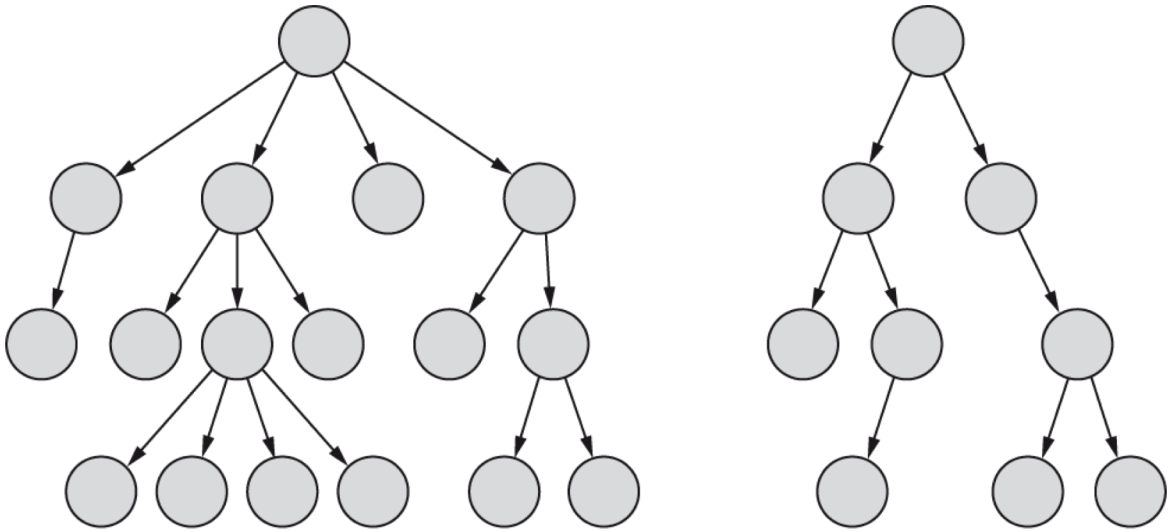
Gerichteter azyklischer Graph

- Ein **gerichteter azyklischer Graph** (engl.: directed acyclic graph, häufige Abkürzung: DAG) ist ein gerichteter Graph, in dem keine Zyklen erlaubt sind.
- Unter einem **Pfad** (engl.: path) versteht man einen Weg, der ausgehend von einem Knoten über eine oder mehrere Kanten zu einem Zielknoten führt.



Baum

- Unter einem **Baum** (engl.: tree) versteht man einen gerichteten azyklischen Graphen mit einem Wurzelknoten, bei dem jeder Knoten maximal einen Vorgängerknoten besitzt.
- Die **Ordnung (Grad)** bestimmt die maximale Anzahl der unmittelbaren Nachfolger eines Knotens eines Graphen. Ein *Baum der Ordnung zwei* heißt **Binärbaum** (oder *binärer Baum*, engl.: binary tree).

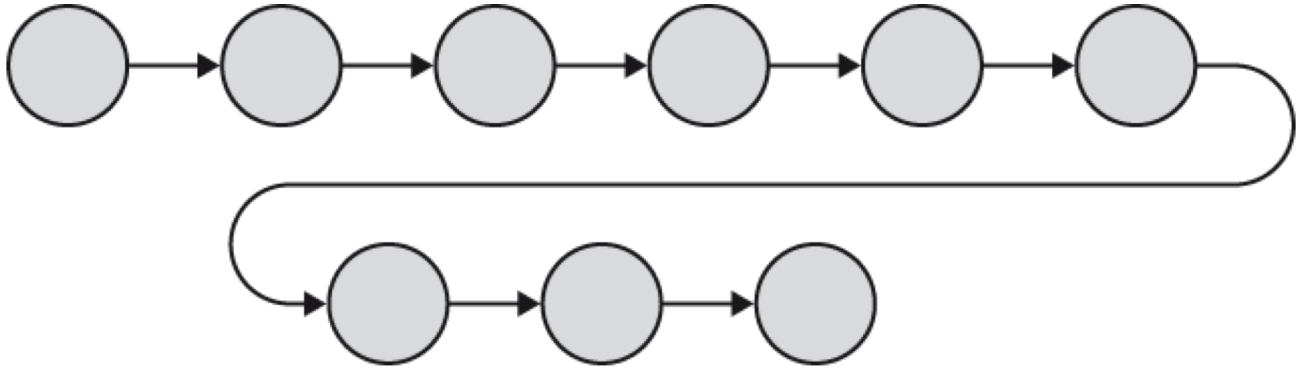


a) Baumstruktur der Ordnung 4

b) Binärbaum

Liste

- Unter einer **linearen Liste** (engl.: linear list) versteht man einen Graphen der Ordnung eins. Jedes Listenelement besitzt maximal einen unmittelbaren Vorgängerknoten und maximal einen unmittelbaren Nachfolgerknoten. Das Äquivalent zur Wurzel des Baums heißt **Anker** (engl.: anchor) der Liste.



Dateien

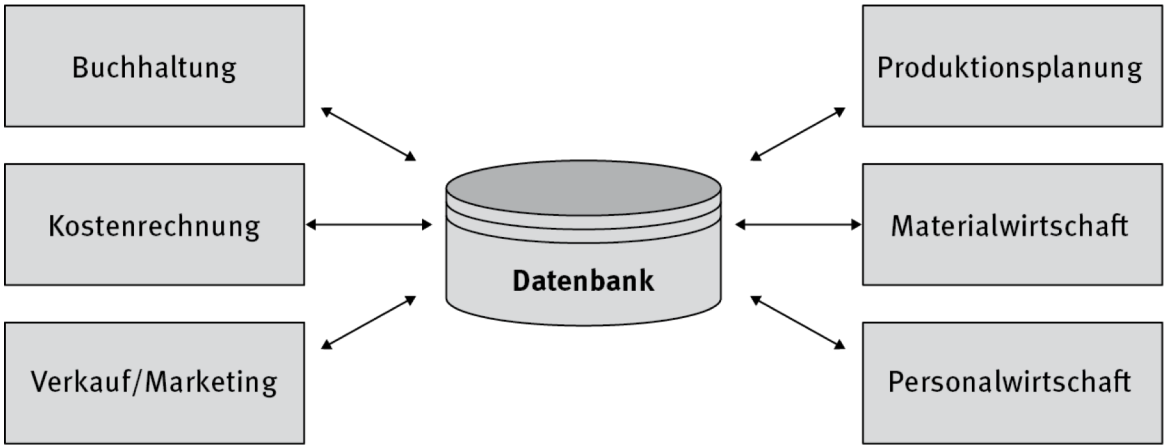
- Eine **Datei** (engl.: file) ist eine Sammlung von zusammengehörigen Daten, die primär zur dauerhaften (persistenten) Speicherung von Daten auf einem Speichermedium dient. Dateien besitzen einen Namen und werden über das Dateisystem des Betriebssystems verwaltet. Die wichtigsten Operationen, die mit Dateien durchgeführt werden, sind das Öffnen, Schließen, Kopieren, Umbenennen und Löschen von Dateien, sowie Lese-, Schreib- und Suchoperationen innerhalb der Datei. Die Art und Weise der Speicherung und Codierung der Dateiinhalte wird durch das **Dateiformat** (engl.: file format) bestimmt.
- Bei einer **sequenziellen Speicherform** (eng.: sequential storage) ist ausschließlich ein systematisches Durcharbeiten der Datenelemente innerhalb einer Datei von Beginn an möglich. Bei einer **direkt adressierbaren Speicherform** (engl.: direct access storage) kann bei Kenntnis der Adresse direkt auf ein Datenelement zugegriffen werden.

10.3 Datenbanken

Datenbank

- **Datenbank:** zentral verwalteter Datenbestand, der über anwendungsunabhängige Zugriffsverfahren nutzbar gemacht wird (Ziele: Entkopplung von Anwendungsprogrammen von Aspekten der Datenspeicherung, gemeinsam nutzbare Daten)

- **Datenbankverwaltungssystem** (DBMS): verwaltet Datenbank, regelt Zugriffschutz, ermöglicht gleichzeitige Zugriffe von mehreren Anwendungsprogrammen, verwendet eigene Sprache für Definition und Abfrage von Daten.



Transaktionsorientierte Datenbanken

- Datenbankoperationen, die durch mehrere Einzelschritte (Aktionen) durchgeführt werden, werden als **Transaktionen** (engl.: transaction) bezeichnet.
- *ACID-Eigenschaften* (engl.: ACID properties) definieren das gewünschte Verhalten von Datenbanksystemen:
  - *Atomarität (engl.: atomicity)*
  - *Konsistenz (engl.: consistency)*
  - *Isolation (engl.: isolation)*
  - *Dauerhaftigkeit (engl.: durability)*

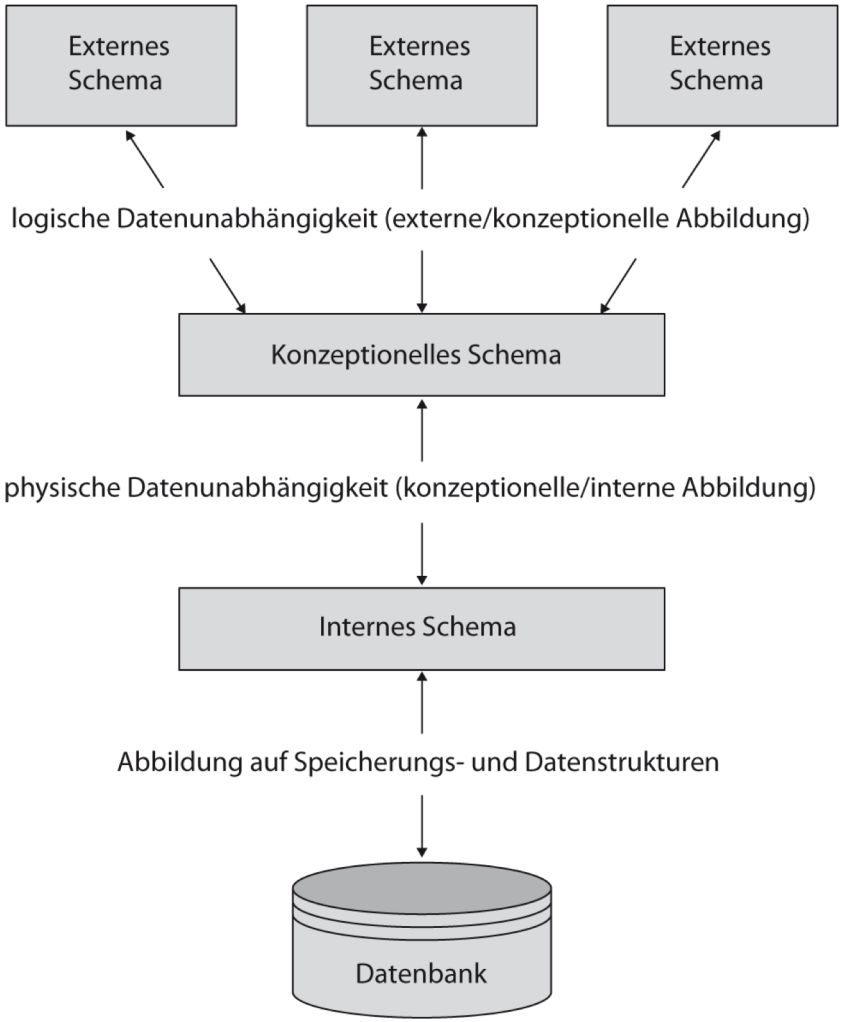
- Unter einer **In-Memory-Datenbank** (engl.: in-memory database, abgekürzt: IMDB) versteht man eine Datenbank, die nur den (flüchtigen) Arbeitsspeicher für die Datenhaltung nutzt. Durch den Entfall des Zugriffs auf externe Speicher sind die Zugriffs- und Verarbeitungszeiten sehr kurz, allerdings können bei Systemabstürzen Daten verloren gehen.

Wichtige Typen von Datenbanken

- **Transaktionsorientierte Datenbank:** für Transaktionen, wichtigste Form, Standardannahme.
- **Historische Datenbank:** erlaubt Abfragen über Zustände und Ereignisse zu einem früheren Zeitpunkt (wer hat wann welche Werte eingefügt).
- **Temporale Datenbank:** ermöglicht beliebige Abfragen zu beliebigen Zeitpunkten (in der Vergangenheit und mglw. in der Zukunft, wenn entsprechende Information vorliegt.

- **Blockchain:** spezielle Form einer historischen Datenbank, bei der die Unveränderbarkeit der Daten durch kryptografische Prüfsummen zugesichert wird.

ANSI-SPARC-Dreischichtenmodell



ANSI-SPARC-Dreischichtenmodell

- Die **externen Schemata** (engl.: external schema) beschreiben jene Ausschnitte des konzeptionellen Schemas, die für einzelne Anwendungen relevant sind. Ein externes Schema ist eine abgegrenzte, anwendungs- und benutzerspezifische Sicht auf eine Datenbank, die jeweils genau an die spezifischen Bedürfnisse angepasst ist.
- Das **konzeptionelle Schema** (engl.: conceptual schema) ist das Ergebnis der Abbildung eines konzeptionellen Datenmodells in ein konkretes Datenmodell, das in einem bestimmten Datenbanksystem implementiert werden kann.
- Die **interne Schicht** (engl.: internal level) eines Datenbanksystems bestimmt die physische Datenorganisation (= physische Anordnung der Daten auf den peripheren Speichern) und legt die Zugriffspfade für die Daten fest. Die Zielsetzung ist hierbei eine minimale Zugriffszeit bei möglichst optimaler Speicherplatzausnutzung. Häufig auftretende Operationen (Abfragen oder Änderungen) in der Datenbank sollen besonders schnell durchgeführt werden können.

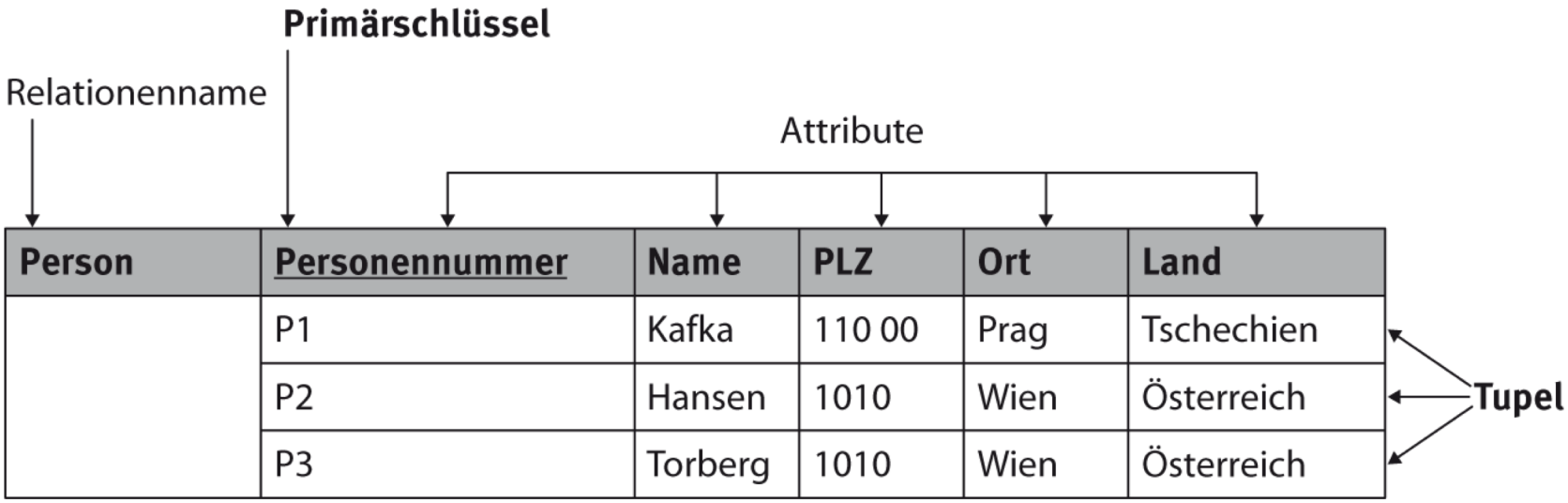
Relationales Datenmodell

- **Datenmodell:** wie werden Daten gespeichert (Strukturierung, welche Datenelemente, ...)
- **Relationales Datenmodell:** Daten werden durch eine Menge von **Relationen** definiert und in entsprechenden **Tabellen** gespeichert
  - Eine **Relation** wird durch den Namen der Relation, Attribute und Einschränkungen und Abhängigkeiten definiert

    `PERSON(Personennummer, Name, PLZ, Ort, Land)`

  - **Tabellen**
    - speichern Daten
    - Eine Ausprägung entspricht einer Zeile (ein Tupel) in der Tabelle
    - Merkmale (Attribute) der Ausprägung entsprechen den Spalten

Darstellung der Tabelle „Person“



Person	Personennummer	Name	PLZ	Ort	Land
	P1	Kafka	110 00	Prag	Tschechien
	P2	Hansen	1010	Wien	Österreich
	P3	Torberg	1010	Wien	Österreich

Regeln zur Definition von Tabellen

- Eine **funktionale Abhängigkeit** (engl.: functional dependency) zwischen den Attributmengen *X* 3 und *Y* besteht dann, wenn für jede Ausprägung von *X* eine eindeutige Ausprägung von *Y* existiert. Man sagt „*X bestimmt Y*“ oder „*Y hängt funktional von X ab*“.
- Eine **Inklusionsabhängigkeit** (engl.: inclusion dependency) definiert, dass sämtliche Ausprägungen eines Attributs (einer Attributmenge) in den Ausprägungen eines anderen Attributs (einer anderen Attributmenge) enthalten sein müssen.
- Bei der **Normalisierung** (engl.: normalization) eines relationalen Schemas werden die Attribute derart auf Relationen verteilt, dass beim Einfügen, Löschen oder Ändern von Datensätzen keine Inkonsistenzen auftreten.

Darstellung der Tabellen „Produkt“ und „Preis“ mit Fremdschlüssel „Preisgruppe“

Produkt

Produkt-Nr.	Bezeichnung	Preisgruppe
Prod1	Notizblock A4 kariert	G3
Prod2	Notizblock A5 liniert	G2
Prod3	Notizblock A4 liniert	G3
Prod4	Notizblock A6 glatt	G1
Prod5	Kopierpapier 500 Blatt	G7
Prod6	Notizblock A4 glatt	G3

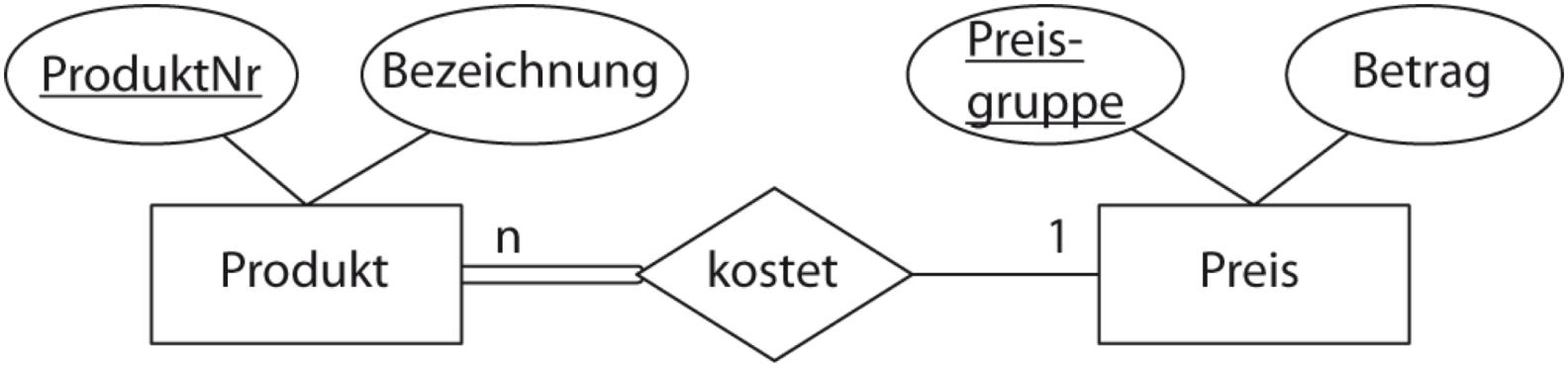
Preis

Preisgruppe	Betrag
G1	0,50
G2	1,50
G3	3,00
G4	3,25
G5	5,00
G6	5,50

Ableitung von relationen Schemata aus ER-Diagrammen

1. *Entitätstypen* (als Rechtecke visualisiert) werden zu Tabellen. Der Name der Tabelle entspricht dem Namen des Objekttyps.
2. *Attribute* jedes Entitätstyps (als Ovale visualisiert) werden den entsprechenden Tabellen zugeordnet. Sie repräsentieren die Spalten einer Tabelle. Die identifizierenden Attribute werden zu Primärschlüsseln.
3. *Mehrwertige Attribute* werden auf eigene Tabellen abgebildet. Dabei werden der Primärschlüssel des Objekttyps und das mehrwertige Attribut selbst zu den Spalten dieser neuen Tabelle.
4. *Um 1:1-Beziehungen* (Beziehungen werden generell als Raute visualisiert) abzubilden, muss der Primärschlüssel eines der beteiligten Entitätstypen als Fremdschlüssel in die Tabelle des anderen Entitätstyps aufgenommen werden. Über den Fremdschlüssel lassen sich Datensätze dieser beiden Tabellen in Beziehung setzen.
5. *Bei 1:n-Beziehungen* wird der Primärschlüssel des Entitätstyps auf der durch „1" gekennzeichneten Seite als Fremdschlüssel in die Tabelle des anderen Entitätstypen (auf der mit „n" gekennzeichneten Seite) aufgenommen. Zusätzlich werden Attribute, die der Beziehung zwischen den beiden Entitätstypen direkt zugeordnet sind, ebenfalls in die Tabelle des Entitätstypen auf der „n-Seite" der Beziehung aufgenommen.
6. *Bei n:m-Beziehungen* zwischen Entitätstypen wird jeweils eine eigene Tabelle gebildet. Der Tabellenname entspricht hierbei dem Beziehungsnamen. Die Attribute dieser Tabelle sind die Primärschlüssel der an der Beziehung beteiligten Entitätstypen, sowie, falls vorhanden, die der Beziehung direkt zugeordneten Attribute.

Ausschnitt eines ER-Diagramms und relationales Schema



PRODUKT(ProduktNr, Bezeichnung, Preisgruppe)
PREIS(Preisgruppe, Betrag)

Relationale Operationen

- Das relationale Datenmodell definiert allgemeingültige (anwendungsunabhängige) Operationen, die auf Tabellen operieren. Auch die Ergebnisse dieser relationalen Operationen sind wiederum Tabellen, sodass auf uniforme Weise die Operationen beliebig kombiniert werden können. Diese Operationen dienen als Basis jeder *Abfragesprache für relationale Datenbanken*. Die wichtigsten relationalen *Operationen* sind:
  - *Selektion (engl.: selection):* Auswahl einer Untermenge aller Tupel einer Tabelle,
  - *Projektion (engl.: projection):* Auswahl einer Untermenge der Attribute einer Relation und
  - *Verbund (engl.: join):* Verknüpfung von Tabellen anhand selektierter Attribute.

Beispiel für eine Selektion

Ausgangstabelle

Personennummer	Name	PLZ	Ort	Land
P1	Kafka	110 00	Prag	Tschechien
P2	Hansen	1010	Wien	Österreich
P3	Adams	CB30EQ	Cambridge	England
P4	Clemens	06876	Redding	USA
P5	Torberg	1010	Wien	Österreich
P6	Musil	9020	Klagenfurt	Österreich
P7	Neumann	1010	Wien	Österreich
P8	Schnitzler	1010	Wien	Österreich
P9				
P10				
P11				

Personennummer	Name	PLZ	Ort	Land
P2	Hansen	1010	Wien	Österreich
P7	Torberg	1010	Wien	Österreich
P9	Neumann	1010	Wien	Österreich
P11	Schnitzler	1010	Wien	Österreich

Ergebnistabelle

Beispiel für eine Projektion

Ausgangstabelle

Personennummer	Name	PLZ	Ort	Land
P1	Kafka	110 00	Prag	Tschechien
P2	Hansen	1010	Wien	Österreich
P3	Schiller	99423	Weimar	Deutschland
P4	Rosegger	8670	Krieglach	Österreich
P5	Adams	CB3 0EQ	Cambridge	England
P6	Clemens	06876	Redding	USA
P7	Torberg	1010	Wien	Österreich
P8	Musil	9020	Klagenfurt	Österreich
P9	Neumann	1010	Wien	Österreich
P10	Canetti	8001	Zürich	Schweiz
P11	Schnitzler	1010	Wien	Österreich

Name	PLZ	Ort
Kafka	110 00	Prag
Hansen	1010	Wien
Schiller	99423	Weimar
Rosegger	8670	Krieglach
Adams	CB3 0EQ	Cambridge
Clemens	06876	Redding
Torberg	1010	Wien
Musil	9020	Klagenfurt
Neumann	1010	Wien
Canetti	8001	Zürich
Schnitzler	1010	Wien

Ergebnistabelle

Beispiel für einen Verbund von zwei Tabellen

Produkt

ProduktNr	Bezeichnung	Preisgruppe
Prod1	Notizblock A4 kariert	G3
Prod2	Notizblock A5 liniert	G2
Prod3	Notizblock A4 liniert	G3
Prod4	Notizblock A6 glatt	G1
Prod5	Kopierpapier 500 Blatt	G7
Prod6	Notizblock A4 glatt	G3

Preis

Preisgruppe	Betrag
G1	0,50
G2	1,50
G3	3,00
G4	3,25
G5	5,00
G6	5,50

SQL

- Die **Structured Query Language** (abgekürzt: *SQL*) ist eine Definitions- und Abfragesprache für relationale Datenbanksysteme und stellt den *Marktstandard* für Datenbanksprachen dar. SQL ist eine sogenannte *relationen-algebraische Sprache,* die mächtige Ausdrucksmittel zur Abfrage und Verknüpfung von Tabellen zur Verfügung stellt. SQL ist *mengenorientiert* und *deskriptiv*.
- Datendefinition:

```
CREATE TABLE Produkt (
   ProduktNr INTEGER NOT NULL UNIQUE,
   Bezeichnung CHAR (250) NOT NULL,
   Preisgruppe CHAR (10) NOT NULL,
   PRIMARY KEY(ProduktNr),
   FOREIGN KEY(Preisgruppe) REFERENCES Preis(Preisgruppe)
)
```

- SQL-Anfrage:

```
SELECT ProduktNr, Bezeichnung, Preis
FROM Produkt, Preis
WHERE Produkt.Preisgruppe = Preis.Preisgruppe AND Betrag <= 3.0
```

Nicht relationale Datenmodelle

- Auf dem **hierarchischen Datenmodell** (engl.: hierarchical data model) basierende Datenbanksysteme repräsentieren Anwendungsdaten durch Baumstrukturen.
- Datenbanksysteme nach dem **Netzwerkdatenmodell** (engl.: network data model) stellen die Informationsstruktur durch Graphstrukturen dar.

10.4 Dokumentzentriete Datenorganisation

Dokumentzentrierte Datenorganisation

- Das **Datenaustauschformat** (engl.: data interchange format) dient als Schnittstelle zum Import und Export von Daten aus einem Informationssystem.
- Ein **strukturiertes Dokument** (engl.: structured document) ermöglicht die gemeinsame Darstellung von semantisch zusammengehörenden Daten in einer gemeinsamen Datenstruktur. Diese Datenstruktur unterstützt die Speicherung, Übertragung und (teil-)automatisierte Weiterverarbeitung beliebiger Dateninhalte. Man spricht hierbei von einem Dokument, da diese Datenstrukturen nicht nur für den Rechner verarbeitbar sind, sondern auch von Menschen gelesen werden können.

JSON

- **JSON** (Abkürzung von engl.: JavaScript object notation) ist ein offener Standard für die Beschreibung baumstrukturierter Daten. Ein JSON-Objekt kann weitere strukturierte oder nicht strukturierte Datenelemente enthalten (Aggregation), wobei jedes Attribut einen oder mehrere Werte enthalten kann (mehrwertige Attribute).

```
{
   "Titel": "Wirtschaftsinformatik",
   "Autor": [
            {"Vorname": "Hans Robert", "Familienname": "Hansen"},
            {"Vorname": "Jan", "Familienname": "Mendling"},
            {"Vorname": "Gustaf", "Familienname": "Neumann"}],
   "Verlag": "De Gruyter",
   "Schlagwörter": ["Wirtschaftsinformatik", "Einführung"]
}
```

Extensible Markup Language (XML)

- Eine **Auszeichnungssprache** (*Markup-Sprache*, engl.: markup language) ist eine Sprache, die Regeln zur Auszeichnung von Textelementen bereitstellt. Beliebigen Textelementen können Eigenschaften zugewiesen werden, wodurch die Bedeutung (Semantik) dieser Textelemente ausgedrückt werden kann. Zusätzlich ermöglicht dies die maschinelle Weiterverarbeitbarkeit. Die Textelemente werden durch eine **Startmarkierung** (engl.: start tag) eingeleitet, welche die Art der enthaltenen Daten bestimmt, und durch eine **Endemarkierung** (engl.: end tag) abgeschlossen.
- **XML** (Abkürzung von engl.: extensible markup language) ist eine Metasprache für die Definition von anwendungsspezifischen Auszeichnungssprachen. Mittels XML können Auszeichnungssprachen für beliebige Anwendungsbereiche maßgeschneidert werden, um anwendungsspezifische Datenstrukturen zu beschreiben. Mittels XML werden Dokumenttypen definiert, die eine Klasse von *XML-Dokumenten* (beispielsweise Bestellungen) in ihrem Aufbau und ihren Inhalten festlegen. XML-Dokumente sind auch ohne anwendungsspezifische Werkzeuge für den Menschen intuitiv verständlich und sind zudem relativ einfach maschinell weiter verarbeitbar.

Beispiel eines XML-Dokuments für eine Warenbestellung

```
<?xml version="1.0" ?>
<Bestellung Datum="30.06.2018">
    <Lieferadresse>
        <Name>Peter Schmidt</Name>
        <Straße>Schleicher-Allee 11</Straße>
        <Stadt>Wien</Stadt>
        <Postleitzahl>1090</Postleitzahl>
        <Land>Österreich</Land>
    </Lieferadresse>
    <Rechnungsadresse>
        <Name>Andrea Müller</Name>
        <Straße>Getreidestraße 77</Straße>
        <Stadt>Wien</Stadt>
        <Postleitzahl>1200</Postleitzahl>
        <Land>Österreich</Land>
    </Rechnungsadresse>
    <Artikel>
        <Buch Kategorie="Belletristik" Verlag="Klett-Cotta">
            <Titel>Der Herr der Ringe - Die Gefährten (Teil 1)</Titel>
            <Autoren>J.R.R. Tolkien</Autoren>
            <Preis Währung="Euro">24.90</Preis>
            <ISBN-Nummer>3-608-95536-4</ISBN-Nummer>
            <Bestellmenge>1</Bestellmenge>
        </Buch>
        <Buch Kategorie="Belletristik" Verlag="Klett-Cotta">
            <Titel>Der Herr der Ringe - Die zwei Türme (Teil 2)</Titel>
            <Autoren>J.R.R. Tolkien</Autoren>
            <Preis Währung="Euro">24.90</Preis>
            <ISBN-Nummer>3-608-95537-2</ISBN-Nummer>
            <Bestellmenge>1</Bestellmenge>
        </Buch>
    </Artikel>
</Bestellung>
```

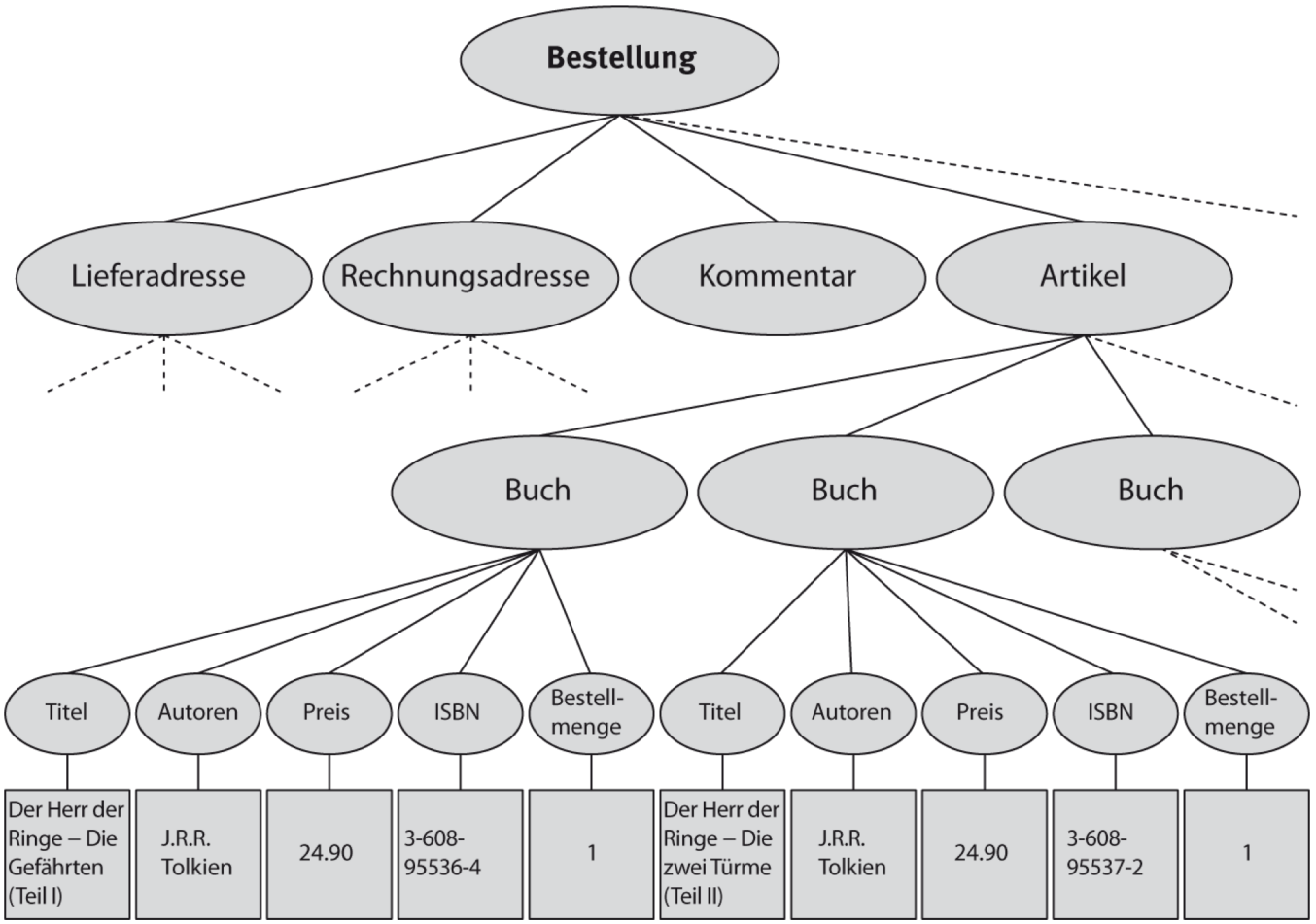
Wohlgeformete und gültige XML-Dokumente

- **Wohlgeformte XML-Dokumente** (engl.: well-formed XML document) entsprechen in ihrem Aufbau den syntaktischen Regeln von XML.
- **Gültige XML-Dokumente** (engl.: valid XML document) sind *wohlgeformte XML-Dokumente*, die zusätzlich auch dem Schema der *Dokumenttypdefinition* genügen müssen, durch welche die Struktur der XML-Dokumente festgelegt wird.

Definition von Dokumenttypen in XML

- Der Standard **XML-Schema** oder **XSD** (Abkürzung von engl.: XML schema definition) ist eine Empfehlung des W3C und ermöglicht die Definition von XML-Dokumenttypen. XML-Schema erlaubt im Gegensatz zur DTD auch Werteinschränkungen von nicht strukturierten Elementen.
- Der XML-Schema-Standard definiert für Elementinhalte folgende mögliche Werteinschränkungen:
  - *Primitive Datentypen (engl.: primitive data type):* Vordefinierte elementare Datentypen wie beispielsweise Dezimalzahlen, Gleitkommazahlen, Zeichenketten, Zeitpunkte oder Zeitdauern.
  - *Abgeleitete Datentypen (engl.: derived data type):* Aus den primitiven Datentypen können mittels Nebenbedingungen abgeleitete Datentypen definiert werden (beispielsweise positive Zahlen oder Werte zwischen 100 und 200).
  - *Zusammengesetzte Datentypen (engl.: complex data type):* Diese Datentypen können sich aus mehreren Elementen zusammensetzen (beispielsweise eine Adresse mit Postleitzahl und Straße). Auch ein Dokumenttyp ist letztendlich ein zusammengesetzter Typ.

Baumstruktur des XML-Dokuments



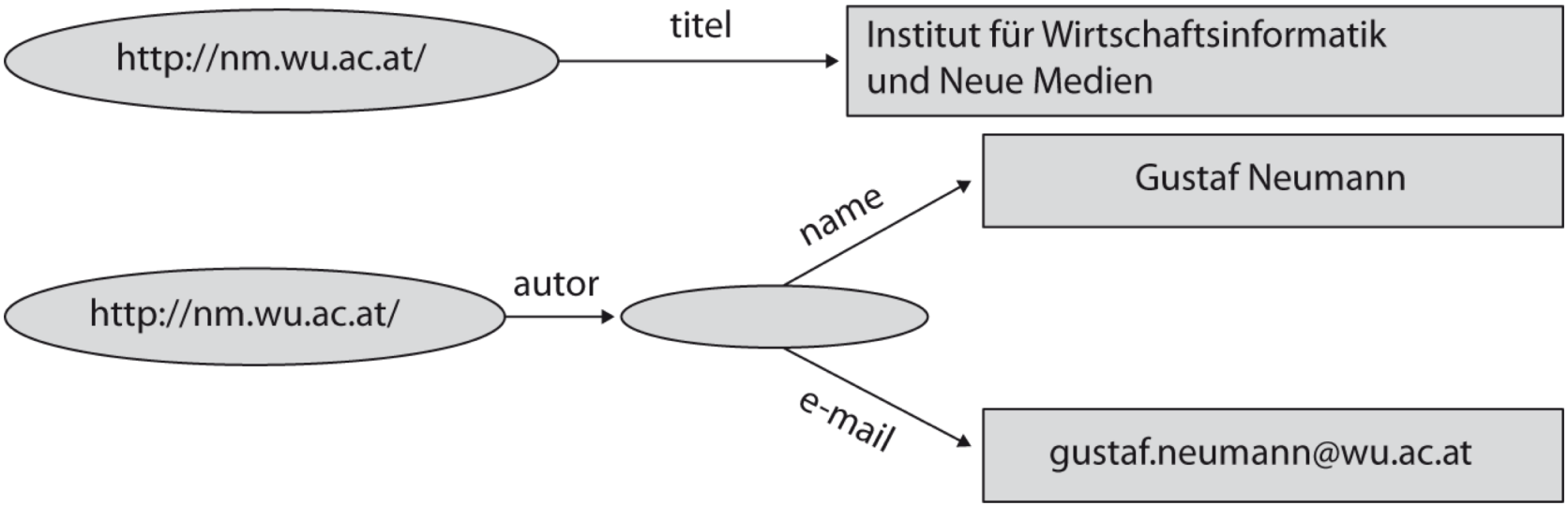
XML-Abfragesprachen

- **DOM** (Abkürzung von engl.: document object model) definiert eine standardisierte Programmierschnittstelle, über die programmiersprachenunabhängig XML-Strukturen verarbeitet werden können. Bei der Verarbeitung mittels DOM wird das gesamte XML-Dokument als eine Objektstruktur mit eingeschachtelten Unterobjekten betrachtet. Über die DOM-Schnittstelle werden Methoden zum Durchwandern, Auslesen, Einfügen und Löschen von Knoten der Dokumentstruktur bereitgestellt.
- **XPath** (Abkürzung von engl.: XML path language) ist eine einfache Abfragesprache für XML-Dokumente, über die Elemente oder Attribute mit angeführten Eigenschaften aus XML-Dokumenten extrahiert werden können. Die Bezeichnung *XPath* ist darauf zurückzuführen, dass Abfragen in der Form eines (Teil-)Pfads von der Wurzel zu den gesuchten Elementen beschrieben werden. Für jeden Teilausdruck entlang des Pfads können *Einschränkungen* (engl.: constraint) angegeben werden, die die Lösungsmenge einschränken. XPath liegt derzeit in Version 3.0 vor und wird vom W3C standardisiert.
- Die Abfragesprache **XQuery** (Abkürzung von engl.: XML query language) ist eine weitere, mächtige Abfragesprache für XML-Dokumente, die vom W3C standardisiert wurde. XQuery basiert auf XPath und erlaubt die Verknüpfung mehrerer XML-Dokumente in einer Anfrage.
- **XSLT** (Abkürzung von engl.: extensible style sheet language transformation) ist eine regelbasierte Sprache zur Transformation (Umwandlung) von XML-Dokumenten, die vom W3C standardisiert wird.

Semantisches Web und RDF

- Das Ziel des **semantischen Webs** (engl.: semantic web) ist es, durch Datenformate und Protokolle die Inhalte von Ressourcen, die über das Web gefunden werden können, besser zugänglich zu machen. Während man bei der textuellen Suche nur nach Wörtern suchen kann, die in den Dokumenten vorkommen, kann man über die Methoden des semantischen Webs auch allgemeine, inhaltliche Zusammenhänge ausdrücken. Deshalb wird das semantische Web auch zunehmend als **Datenweb** (engl.: web of data) bezeichnet.
- **Metadaten** (engl.: meta data) sind „Daten über Daten". Sie beschreiben Dateninhalte anhand eines bestimmten, kontrollierten *Vokabulars*. Das Vokabular definiert, welche Attribute für welche Dateninhalte vergeben werden können.
- Das **Resource Description Framework** (abgekürzt: RDF) definiert ein allgemeines Metadatenformat für Information, die im Internet verfügbar ist. Jedes Dokument (oder Dienst oder digitales Gut), das im Internet mittels Webadresse (URI, siehe Kapitel 12) adressierbar ist, wird in RDF als *Ressource* (engl.: resource) bezeichnet. Mittels RDF können *Aussagen* (engl.: RDF statement) über Ressourcen durch ein erweiterbares *Vokabular* (RDF-Schema) formuliert werden. RDF ist eine Empfehlung des W3C.
- Unter **Linked Open Data** (abgekürzt: LOD) versteht man frei zugängliche und frei verwendbare Daten, die über das Internet bezogen werden können, und die mittels RDF beschrieben sind.

Beispiel eines RDF-Modells



Allgemeiner Aufbau von RDF-Graphen und RDF-Schema

- Ein **RDF-Schema** (engl.: RDF schema) definiert ein *kontrolliertes Vokabular*, das Bezeichnungen für *gültige Eigenschaften* (wie beispielsweise Titel, Autor, Farbe) und auch deren *Wertebereich* (engl.: range) definiert. Das RDF-Schema legt somit fest, welche Wertausprägungen gültig sind, und welchen *Ressourcen* (engl.: domain) diese Eigenschaften zugewiesen werden dürfen (beispielsweise Webseiten, Büchern, Personen). Für gleichartige Ressourcen können *Klassen* gebildet werden.

Skalierbare Datenspeicherung und Big Data

- **Big Data:** Datenkollektionen, deren Größe die Fähigkeiten einzelner Rechnersysteme überschreiten, um diese zu speichern, zu durchsuchen, zu analysieren und zu verwalten (offene Definition)
- **Verteiltes Datenbanksystem:** ein Datenbanksystem, das für die Speicherung, Abfrage und Verwaltung eines Datenbestands mehrere getrennte Rechner verwendet, die über ein Kommunikationsnetzwerk (beispielsweise das Internet) miteinander verbunden sind.
- **Problem:** Was passiert, wenn Daten über mehrere Rechner konsistent gehalten werden müssen (ACID-Prinzip), aber einzelne der Rechner sind nicht erreichbar (Neustart, Netzwerkproblem, usw.)

Speicherung in einer Schlüssel/Wert-Datenbank

Schlüssel	Wert
wi1-titel	“Wirtschaftsinformatik”
wi1-schlagwörter	["Wirtschaftsinformatik", "Einführung"]
wi1-autoren	[“hrh”, “jm”,“gn”]
hrh	“Vorname” → “Hans Robert”,“Nachname” → “Hansen”

Zeilen- und spaltenorientierte Speicherung

Produkt-Nr.	Preis	Menge
p1	10,3	7
p2	10,1	4
p1	10,4	3
p2	10,2	10

Tabelle

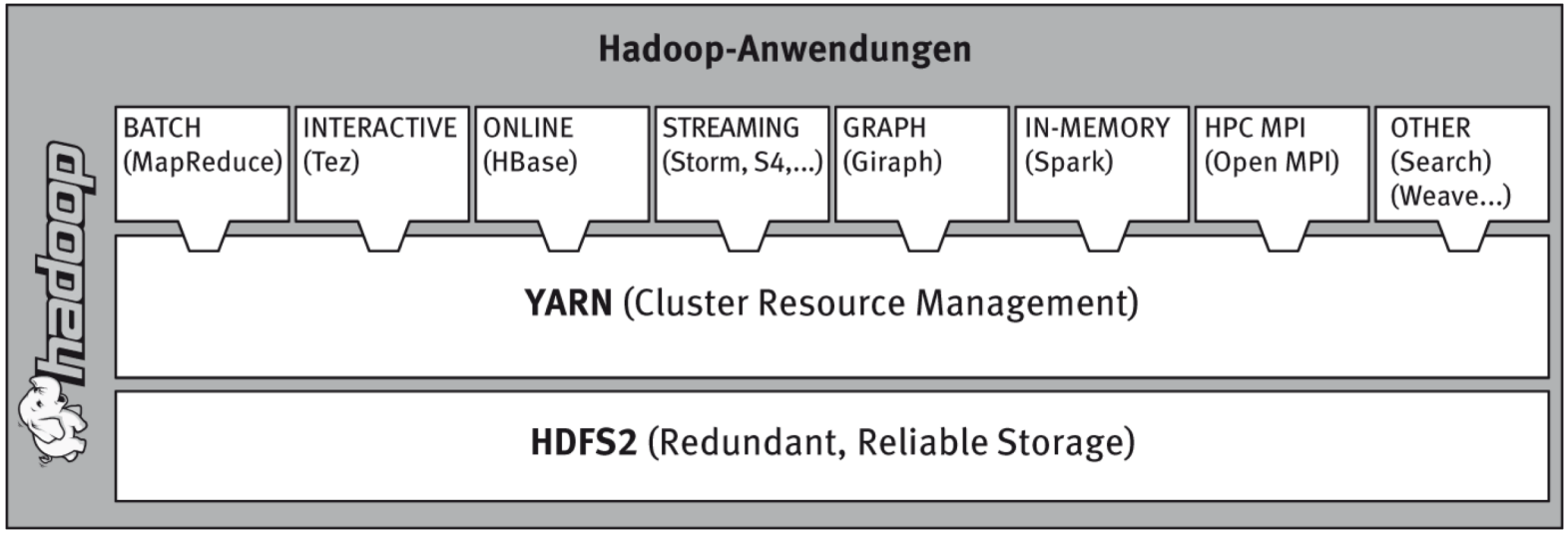
Zeilenorientierte Speicherung

Produkt-Nr.	Preis	Menge
p1	10,3	7
p2	10,1	4
p1	10,4	3
p2	10,2	10

Spaltenorientierte Speicherung

Produkt-Nr.	Preis	Menge
p1	10,3	7
p2	10,1	4
p1	10,4	3
p2	10,2	10

Apache Hadoop und aufbauenden Komponenten

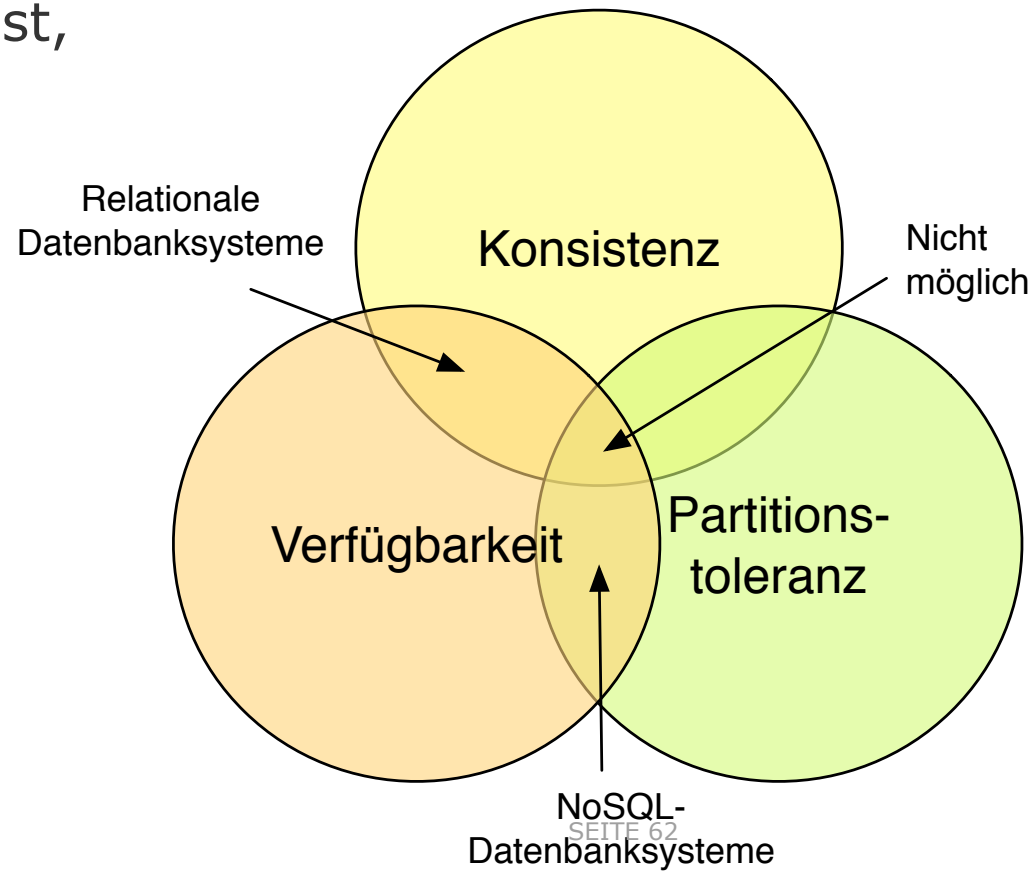


Skalierbare verteilte Datenbanken

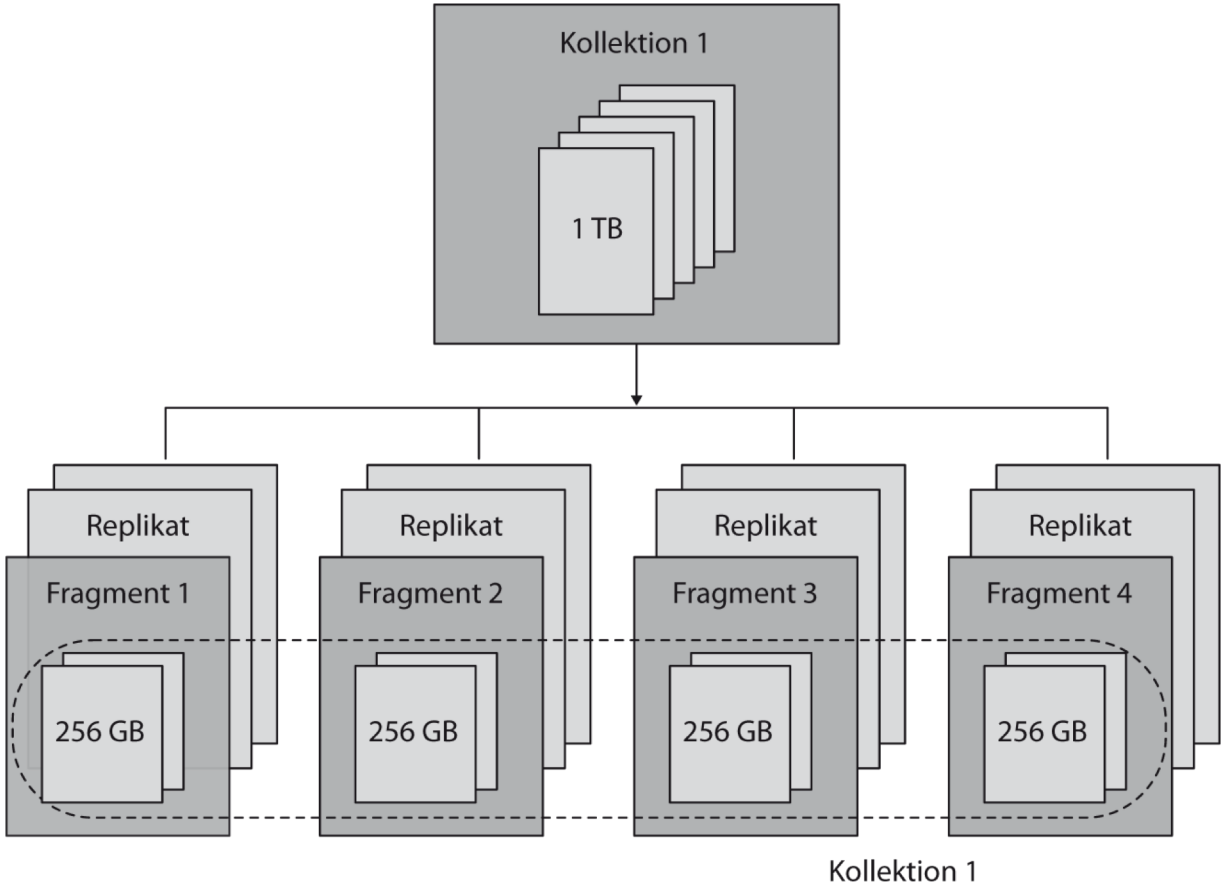
- Unter einem **verteilten Datenbanksystem** (engl.: distributed database system) versteht man ein Datenbanksystem, das für die Speicherung, Abfrage und Verwaltung eines Datenbestands mehrere getrennte Rechner verwendet, die über ein Kommunikationsnetz (beispielsweise das Internet) miteinander verbunden sind.
- Ein **horizontal skalierbares Datenbanksystem** (engl.: horizontally scalable database system) ist ein Datenbanksystem, dessen Leistung durch die Hinzunahme weiterer Rechner gesteigert werden kann.
- Unter der **Datenbankreplikation** (engl.: database replication) versteht man das Anlegen und Verwalten von *Duplikaten* einer Datenbank, meist nach einem Master-Slave-Prinzip (Hauptkopie und Replikat).
- Unter der **Datenbankfragmentierung** (engl.: database sharding) versteht man die *Aufteilung der Speicherung* eines Datenbestands auf mehrere Rechner.

CAP-Theorem

Das CAP-Theorem (Abkürzung von engl.: consistency, availability, partition tolerance) besagt, dass es in einem verteilten System unmöglich ist, gleichzeitig die Eigenschaften der Konsistenz, der Verfügbarkeit und der Partitionstoleranz zu garantieren.



Speicherung einer Kollektion mit mehreren Fragmenten und Replikaten auf mehreren Rechnern



Die wichtigsten Punkte

1. Alle Daten, die von einem Rechner verarbeitet werden, werden rechnerintern immer als Folge von Bits gespeichert. Für die Codierung der Daten gibt es je nach Verwendungszweck (beispielsweise Zahlenwerte oder Texte) unterschiedliche Ansätze, welche die Darstellungs- und Verwendungsmöglichkeiten der Daten beeinflussen.
2. Elementare Datenelemente werden zu Datenstrukturen zusammengefasst, durch die Zusammenhänge zwischen Datenelementen beschrieben werden. Datenstrukturen können beispielsweise ausdrücken, wie eine Datenstruktur aus anderen Datenstrukturen aufgebaut wird. Die Graphentheorie bietet die Grundlage für die Klassifikation von komplexen Datenstrukturen, wobei für unterschiedliche Typen von Graphen auch unterschiedlich effiziente Verfahren zur Verarbeitung der Graphen existieren.
3. Komplexe Datenbestände werden in betrieblichen Informationssystemen in der Regel in Datenbanken gespeichert, die von Anwendungsprogrammen genutzt werden. Datenbankverwaltungssysteme ermöglichen die Definition von Datenstrukturen, regeln die Konsistenzbestimmungen, den mehr oder minder zeitgleichen Zugriff von mehreren Programmen auf diese Daten und unterstützen die Rechteverwaltung. Datenbankverwaltungssysteme bieten Abfragesprachen für den Datenzugriff an. Die heute mit Abstand am weitesten verbreiteten Datenbanksysteme verwenden das relationale Datenmodell, das streng tabellarische Daten vorschreibt.
4. Das relationale Datenmodell verlangt oft, dass semantisch zusammengehörige Daten in unterschiedlichen Datenstrukturen gespeichert werden. Deshalb eignen sich relationale Strukturen nur beschränkt für den Datenaustausch. Durch dokumentzentrierte Formen der Datenorganisation können zusammengehörige Daten in strukturierten Dokumenten gehalten und mit Dritten ausgetauscht werden. Ein wichtiger Standard hierfür ist XML, das erlaubt, je nach Verwendungszweck unterschiedliche Typen von strukturierten Dokumenten zu definieren. Rund um XML ist eine Vielzahl von weiteren Standards entstanden, die beispielsweise die Abfrage oder Verarbeitung von XML-Daten vereinfachen. Der Standard RDF ermöglicht die semantische Verarbeitung von beliebigen (auch bereits existierenden) Daten, über die mithilfe eines kontrollierten Vokabulars Aussagen getroffen werden können. RDF hat eine große Bedeutung beispielsweise bei der Definition von frei verfügbaren Daten (engl.: public linked open data), die von zahlreichen Institutionen angeboten werden.
5. Die Speicherung von sehr großen Datenmengen übersteigt die Rechen- und Speicherleistungen auch der heute größten verfügbaren Rechner. Für die Verwaltung von Datenmengen, wie sie beispielsweise bei den führenden Internet-Anbietern wie Google oder Facebook anfallen, wurden skalierbare, verteilte Datenbanksysteme mit alternativen Datenmodellen entwickelt, die sich besser als das relationale Modell mit seinen sehr hohen Konsistenzanforderungen eignen.

Online-Materialien

Übungs- und Lehrmaterialien zu diesem Kapitel finden Sie im Web über den abgebildeten QR-Code. Richten Sie Ihre Smartphone- oder Tablet-Kamera auf das nebenstehende Bild, um zu den Inhalten zu gelangen.