

Data Engineer

DB-Design Teil 3 **UML**

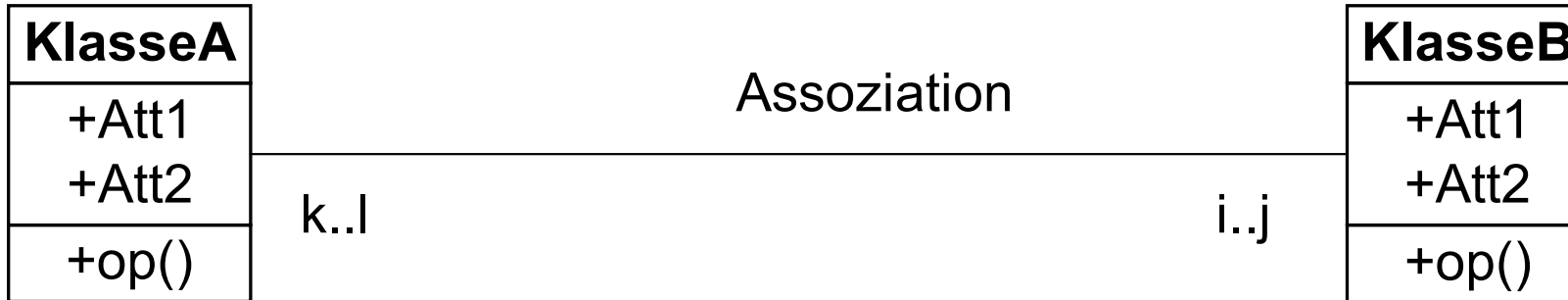
Datenbanken: **DBMS Architektur von DBMS**

..

Datenmodellierung mit UML

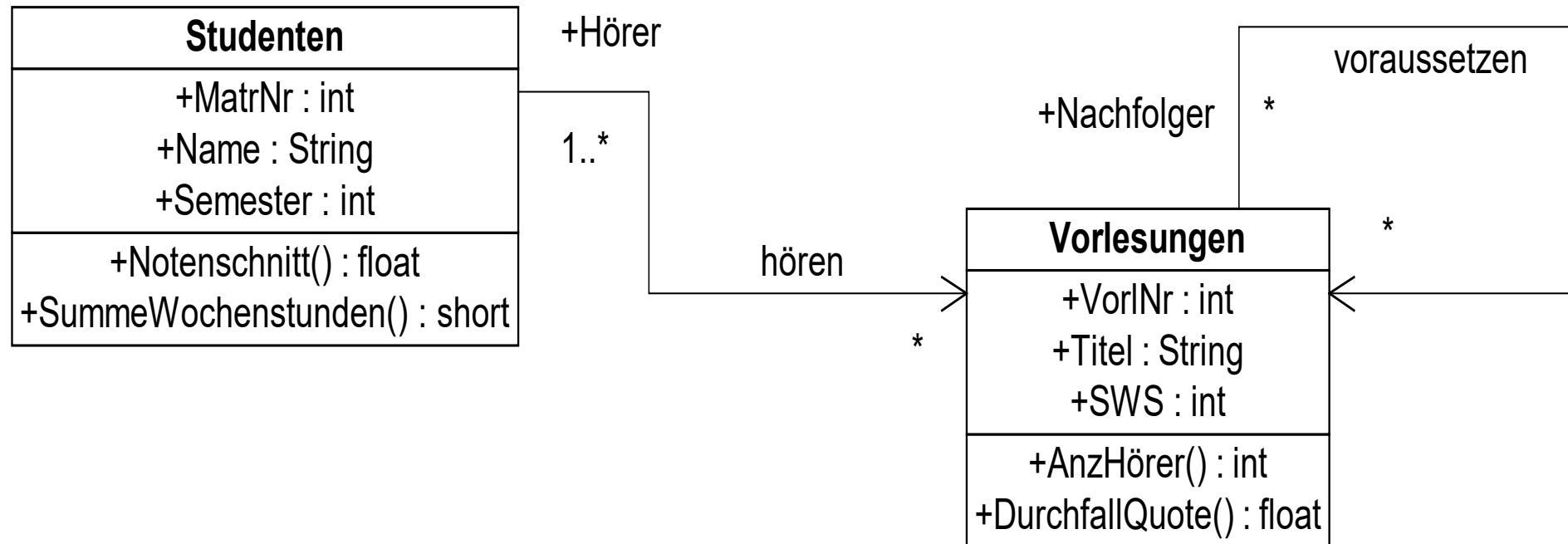
- Unified Modelling Language UML
- De-facto Standard für den objekt-orientierten Software-Entwurf
- Zentrales Konstrukt ist die Klasse (class), mit der gleichartige Objekte hinsichtlich
 - Struktur (~Attribute)
 - Verhalten (~Operationen/Methoden)modelliert werden
- Assoziationen zwischen Klassen entsprechen Beziehungstypen
- Generalisierungshierarchien
- Aggregation

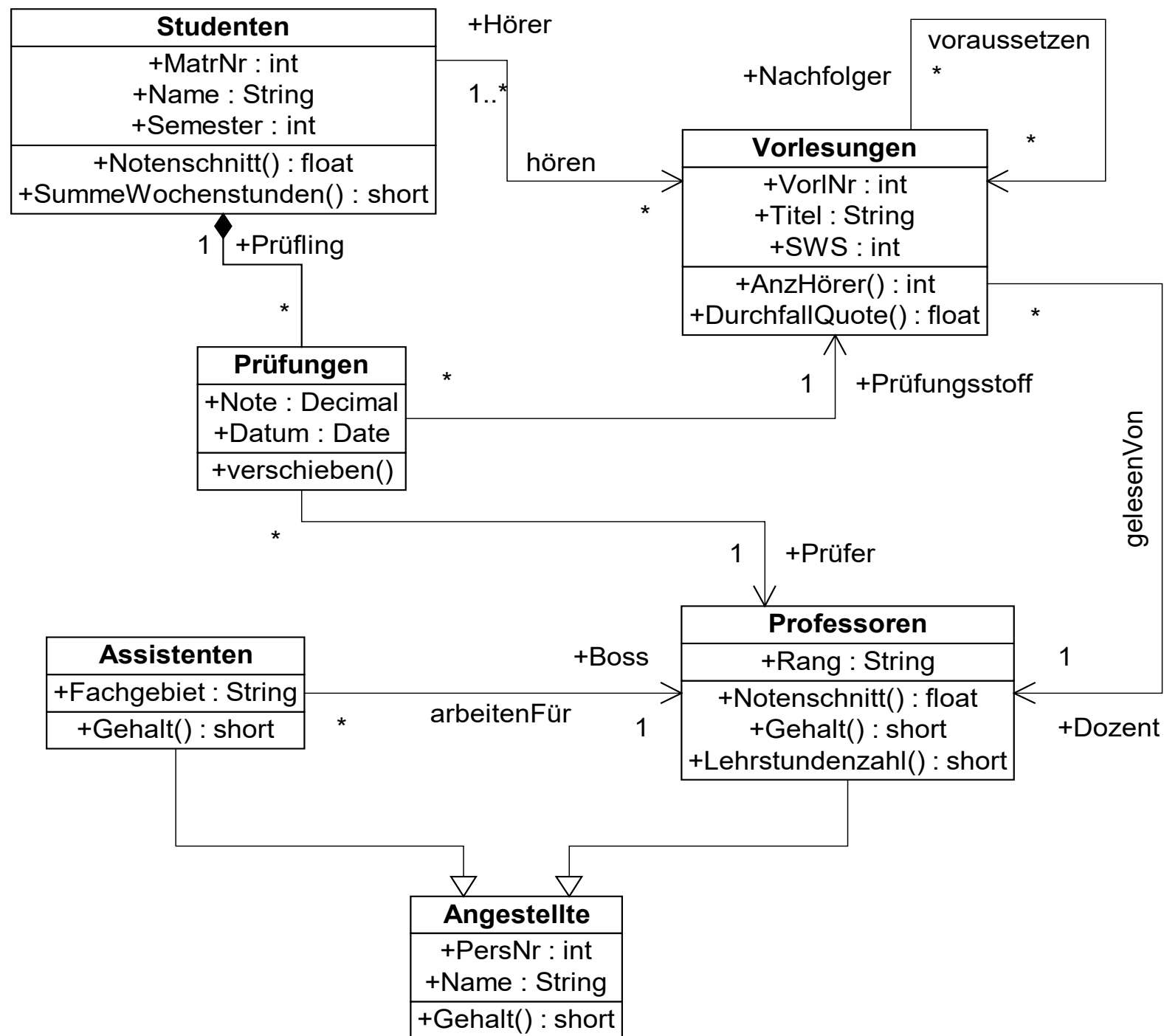
Multiplizität



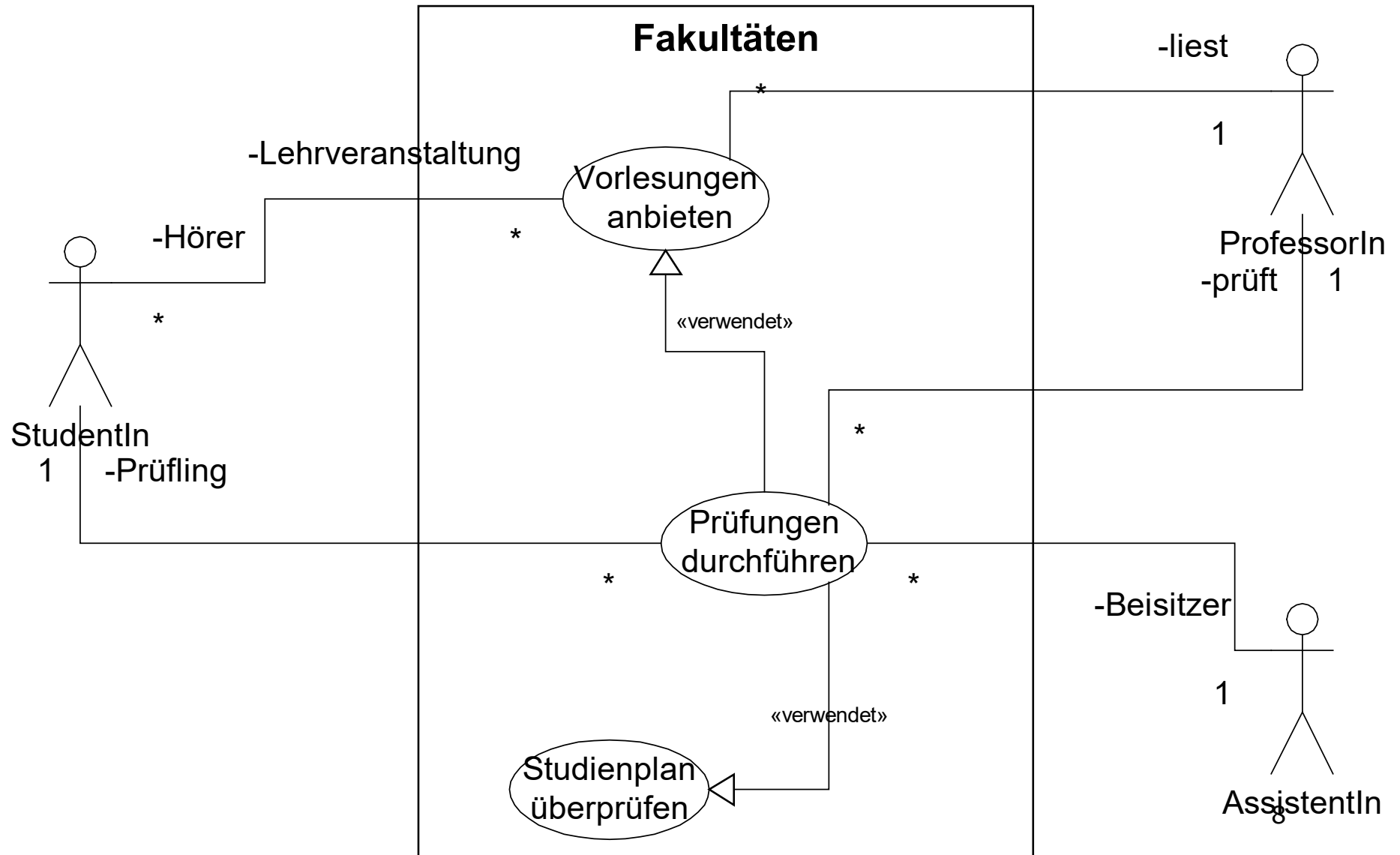
- Jedes Element von KlasseA steht mit mindestens i Elementen der KlasseB in Beziehung
- ... und mit maximal j vielen KlasseB-Elementen
- Analoges gilt für das Intervall k..l
- Multiplizitätsangabe ist analog zur Funktionalitätsangabe im ER-Modell
 - **Nicht** zur (min,max)-Angabe: **Vorsicht!**

Klassen und Assoziationen



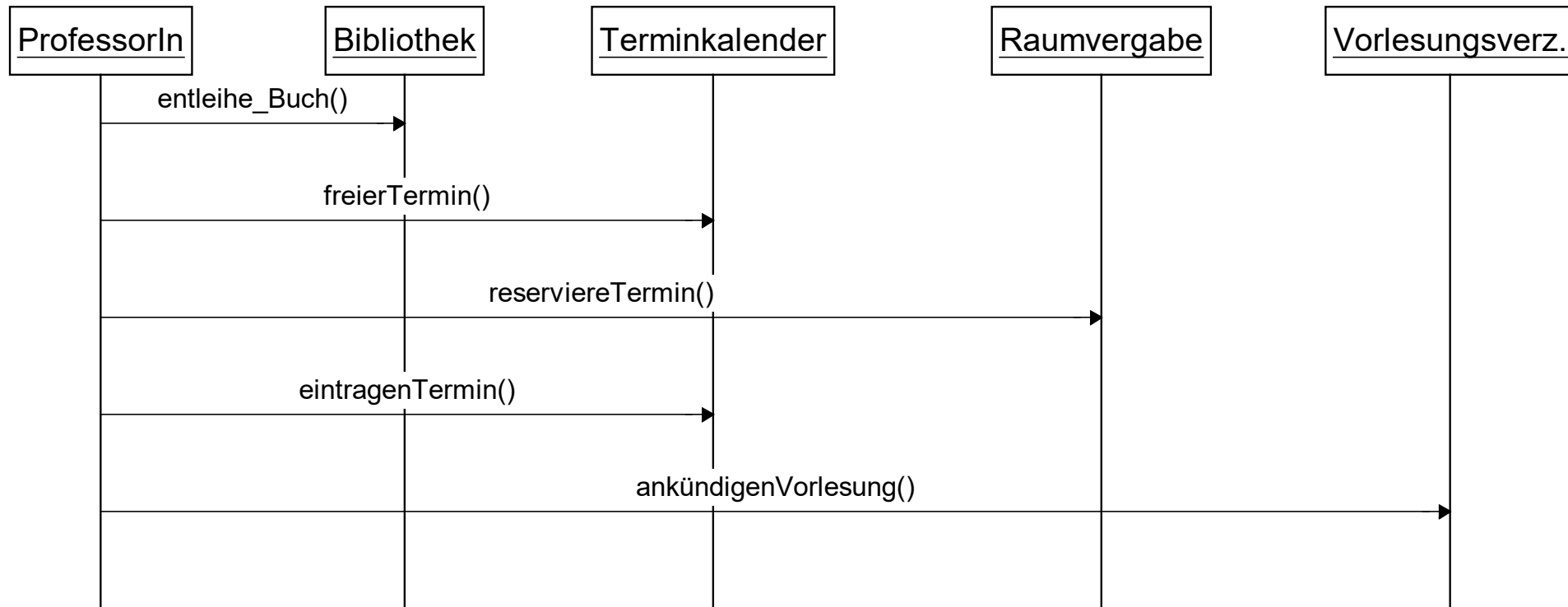


Anwendungsfälle (use cases)

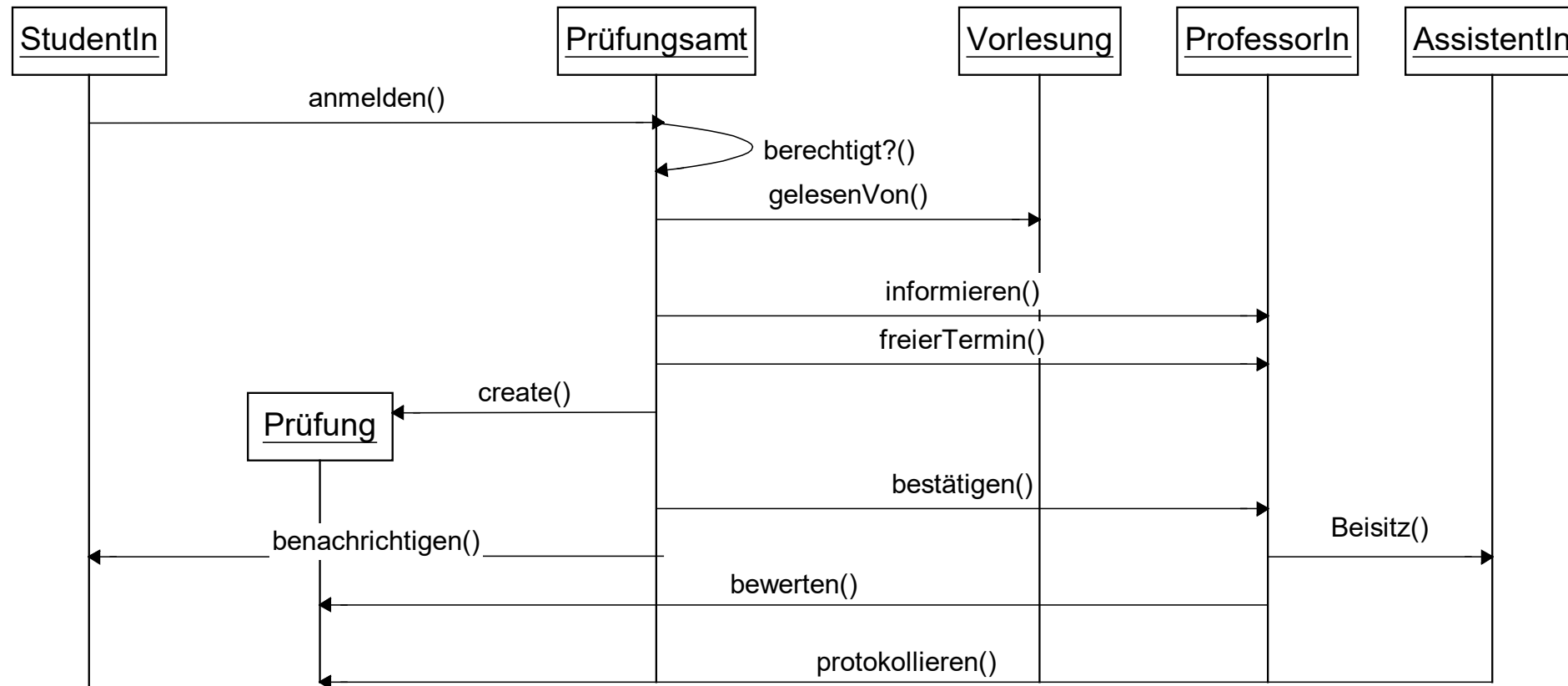


Interaktions-Diagramm:

Modellierung komplexer Anwendungen



Interaktions-Diagramm: *Prüfungsdurchführung*



Datenbanken

Definition

Ein **Datenbanksystem** ist ein computergestütztes System, bestehend aus

- **Datenbasis** zur Beschreibung eines Ausschnitts der Realwelt
- **Programme** zum geregelten Zugriff auf die Datenbasis.

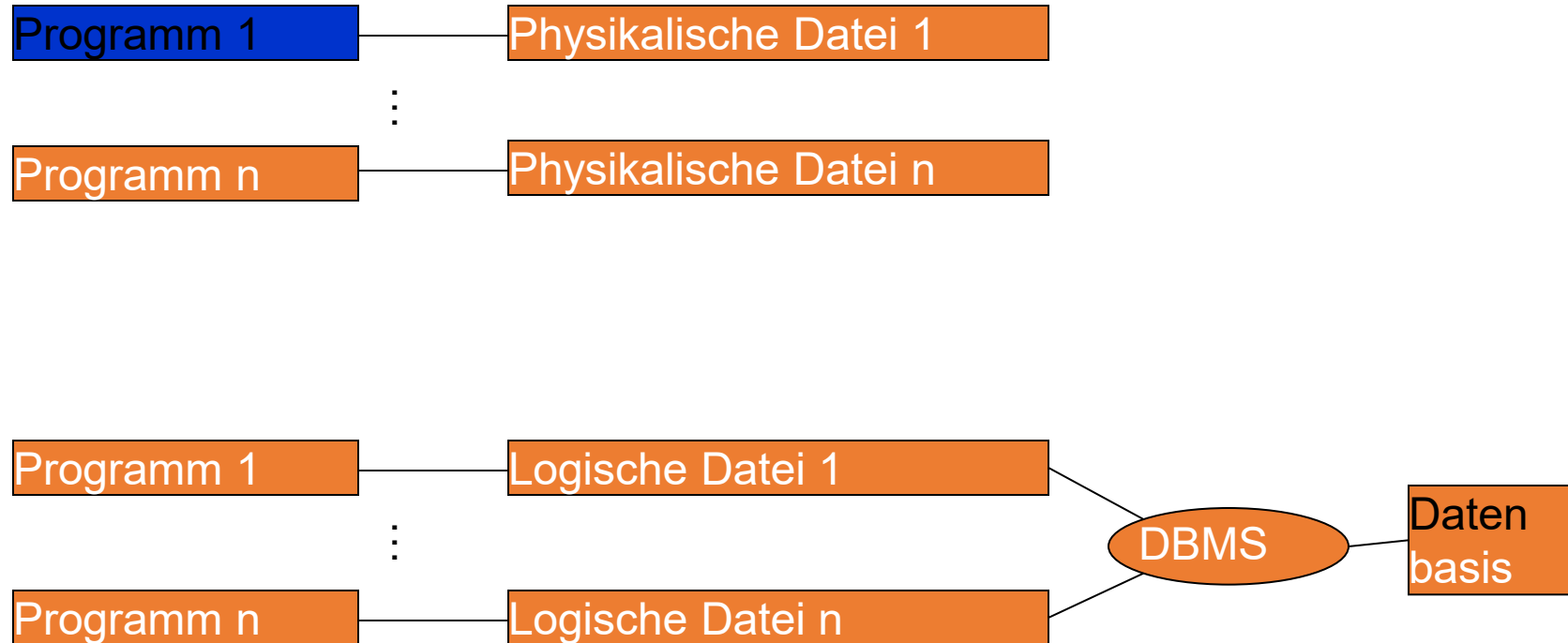
(auch genannt: *Datenbankverwaltungssystem*,
DBMS = data base management system)

Separate Abspeicherung
von miteinander in Beziehung stehenden Daten

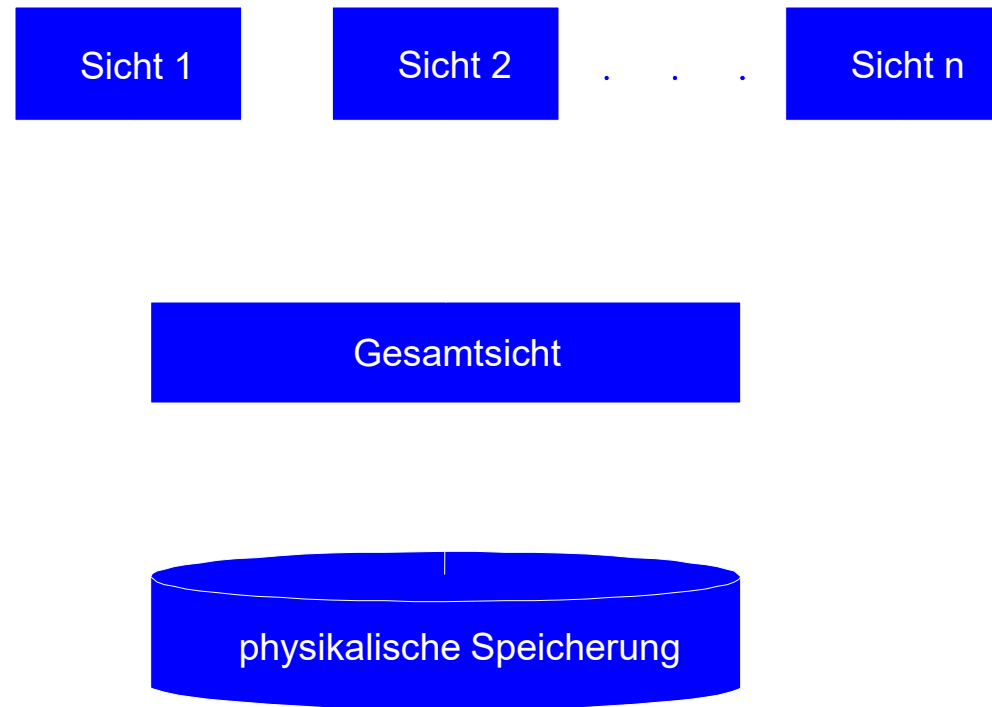


- Redundanz
- Inkonsistenz
- Integritätsverletzung
- Verknüpfungseinschränkung
- Mehrbenutzerprobleme
- Verlust von Daten
- Sicherheitsprobleme
- Hohe Entwicklungskosten

Isolierte Dateien versus zentrale Datenbasis



Datenabstraktion



Schema versus Ausprägung

Datenbankschema = Struktur der abspeicherbaren Daten

Datenbankausprägung = momentan gültiger Zustand der Datenbasis

Transformationsregeln

Transformationsregeln für Verbindungen zwischen den Ebenen

Bundesbahn:

konzeptuelles Schema = *Kursbuch*

externes Schema = *Städteverbindungen Osnabrück*

internes Schema = Abbildung auf Dateisystem

Personaldatei:

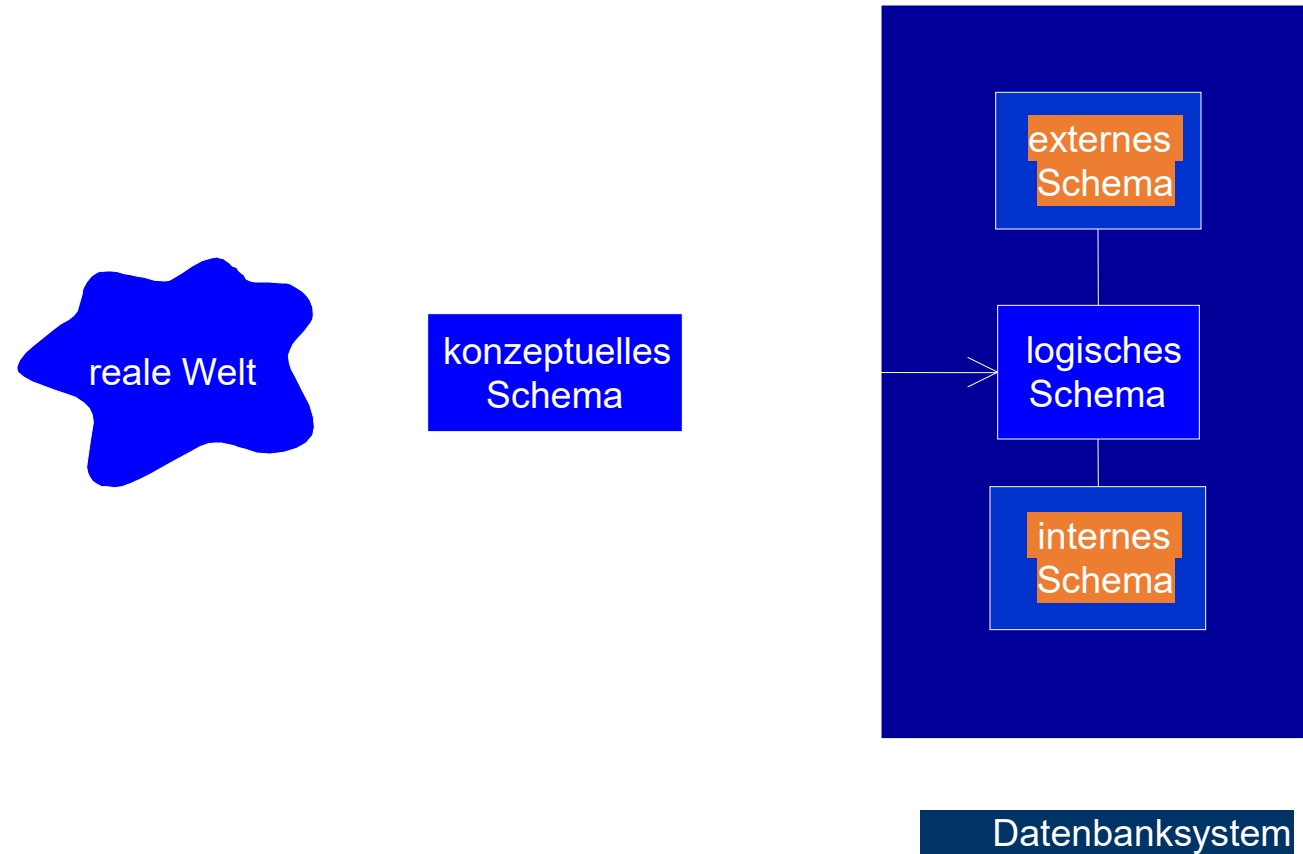
konzeptuelle Ebene = *Angestellte* mit ihren Namen,
Wohnorten und Geburtsdaten

externes Schema = *Geburtstagsliste* mit
Name, Datum, Alter

internes Schema = Abbildung auf Dateisystem

Datenunabhängigkeit

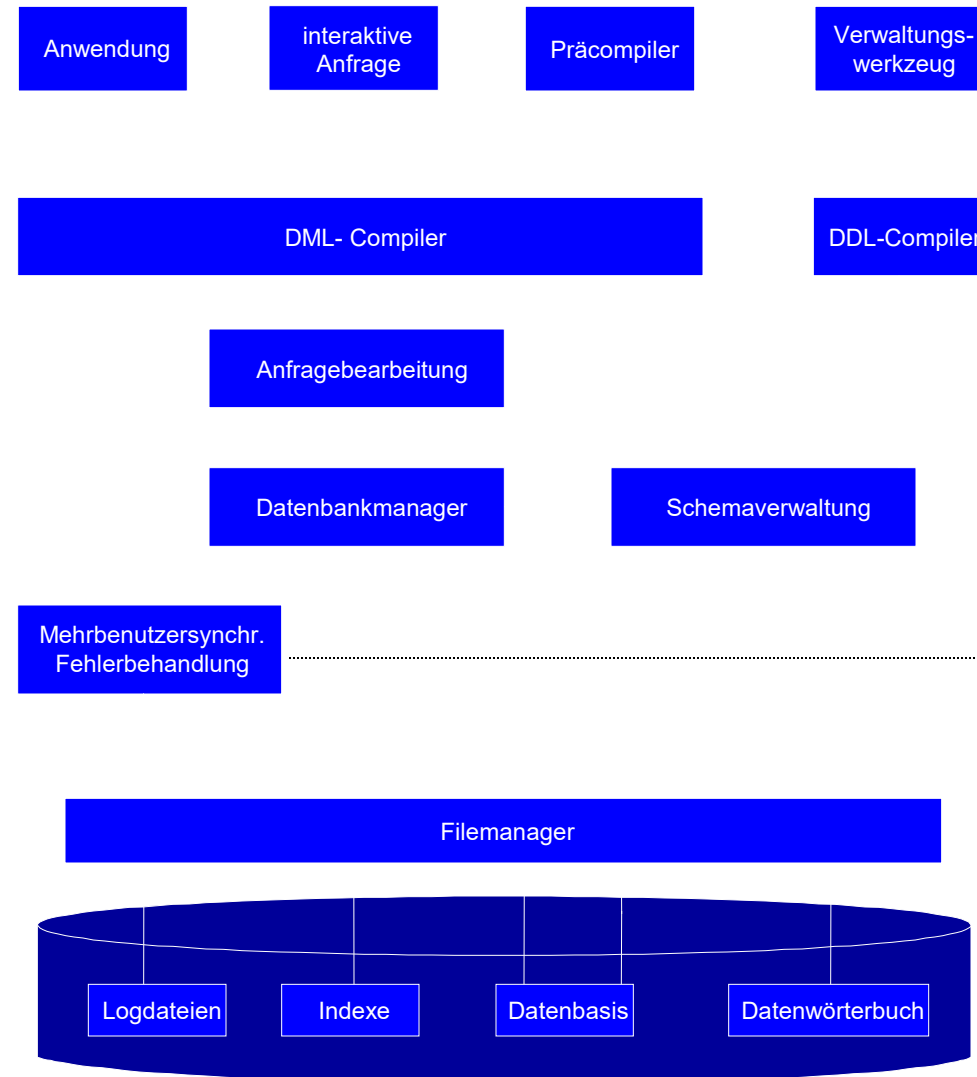
- **Physische Datenunabhängigkeit:**
keine Änderung des externen Schemas
bei Änderung des internen Schemas
- **Logische Datenunabhängigkeit:**
keine Änderung des externen Schemas
bei Änderungen des konzeptuellen Schemas



Logisches Schema

- Das hierarchische Modell (z. B. IMS von IBM)
- Das Netzwerkmodell (z. B. UDS von Siemens)
- Das relationale Modell (z. B. Access von Microsoft)
- Das objektorientierte Modell (z. B. O₂ von O₂ Technology)

Architektur eines DBMS



Datenbankmanagementsysteme Einführung + Grundlagen

Grundbegriffe

- Filezugriff vs. DBMS
 - Multinutzerbetrieb,
- referentielle Integrität, Redundanz und Inkonsistenz
- Primärschlüssel, Fremdschlüssel
- Indizierung
- Import und Export; Backups und Rollback
- Benutzerverwaltung

Definitionen – Datenbank

- (Verteiltes) Computersystem das Nutzdaten (und Metadaten) enthält
- Nutzdaten
 - Daten die von Benutzern angelegt und abgerufen werden
 - Dienen zum Informationsgewinn
- Metadaten
 - Daten über Daten
 - Strukturieren die Nutzdaten

Definitionen – Datenbanksystem

- System zur elektronischen Datenverwaltung, welches Daten dauerhaft, effizient, und widerspruchsfrei speichert
- Kann Daten in Teilmengen in bedarfsgerechten Darstellungsformen für Benutzer und Anwendungsprogramme bereitstellen
- Besteht aus zwei Teilen: Verwaltungssoftware und Datenbank
 - Verwaltungssoftware: Datenbankmanagementsystem (DBMS)
 - Datenbank: Der eigentliche Ort an dem Daten gespeichert werden (DB)

Definitionen – DBMS

- Organisiert intern die Speicherung der Daten nach einem vorgegebenen Datenmodell (zB relational)
- Stellt als Schnittstelle eine Datenbanksprache zur Formulierung von Abfragen zur Verfügung – behandelt auch alle Lese- und Schreibzugriffe
- Abfragen Dienen nicht nur zum einholen von Informationen, sondern auch zum ändern von Daten oder der Datenstruktur.

Definitionen – DBMS

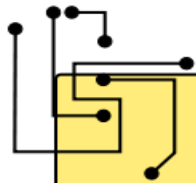
- Referentielle Integrität
 - Ein neuer Datensatz mit einem Fremdschlüssel kann nur dann in einer Tabelle eingefügt werden, wenn in der referenzierten Tabelle ein Datensatz mit entsprechendem Wert im Primärschlüssel oder einem eindeutigen Alternativschlüssel existiert.
 - Eine Datensatzlöschung oder Änderung des Schlüssels in einem Primär-Datensatz ist nur möglich, wenn zu diesem Datensatz keine abhängigen Datensätze in Beziehung stehen.
- Primärschlüssel
- Fremdschlüssel
- Alternativschlüssel

DBMS – Schichtenmodell



externe Ebene

Benutzeroberflächen, Datensichten,
API und Schnittstellen



konzeptionelle Ebene

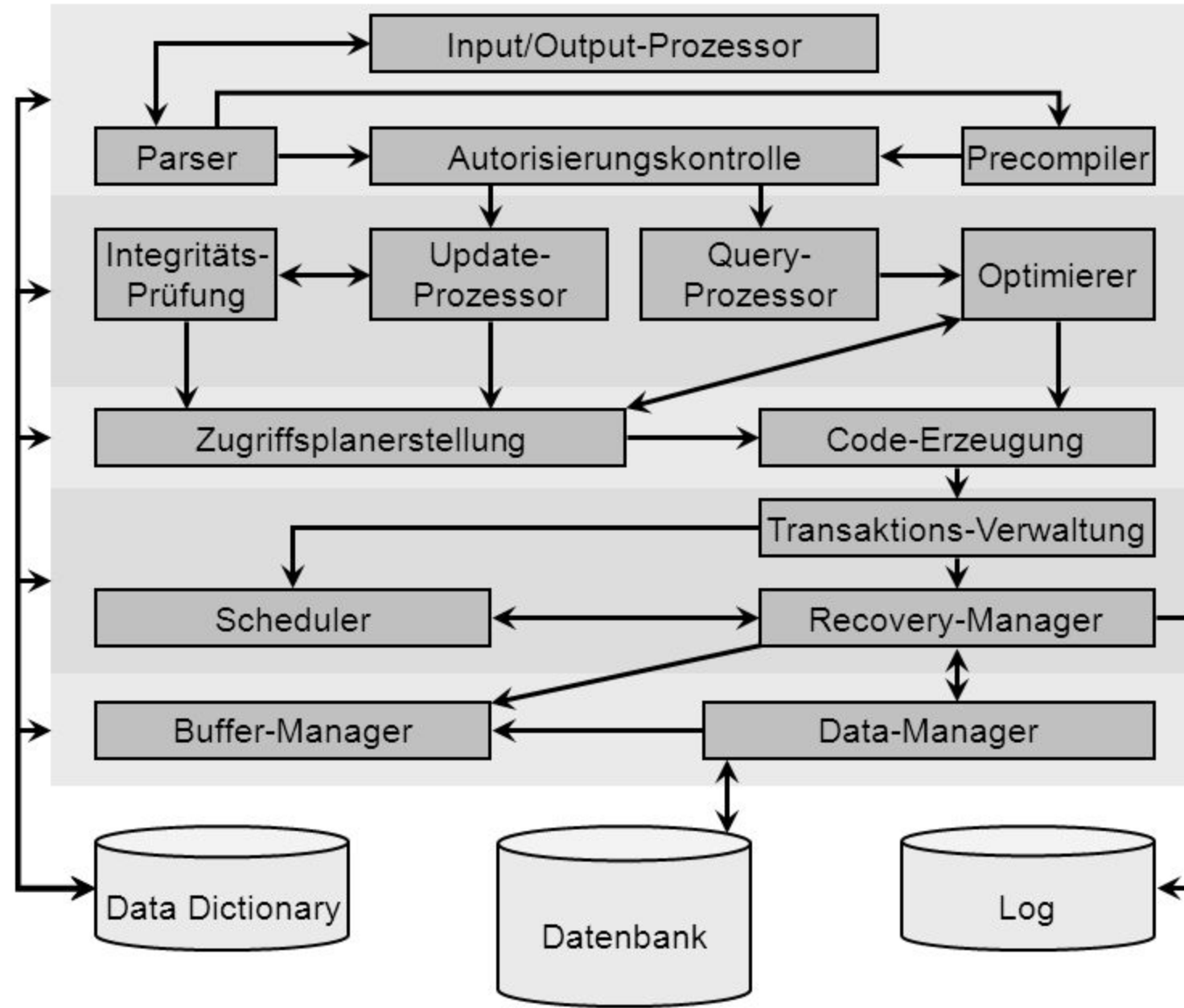
Beziehungen, Daten



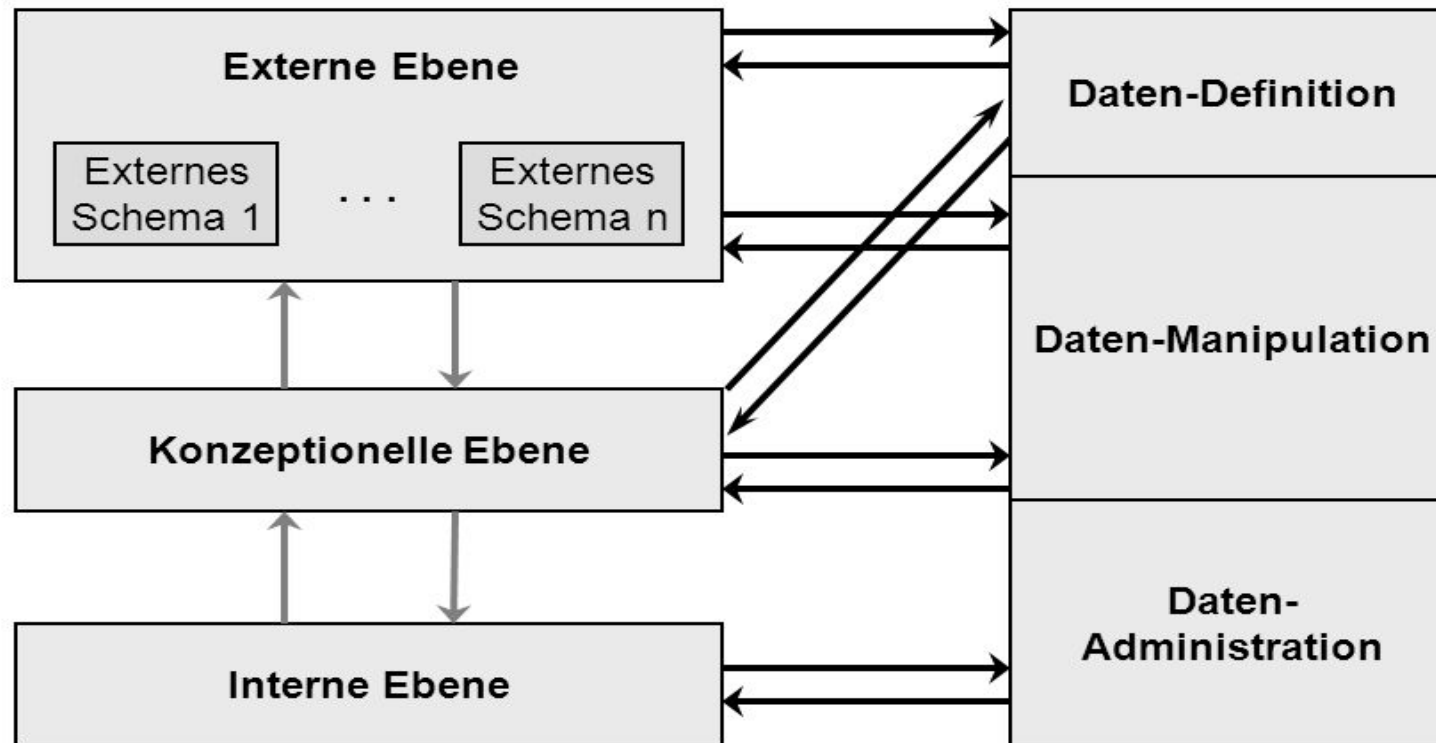
interne Ebene

Art und Form der Speicherung

2.5 Komponenten eines DBMS

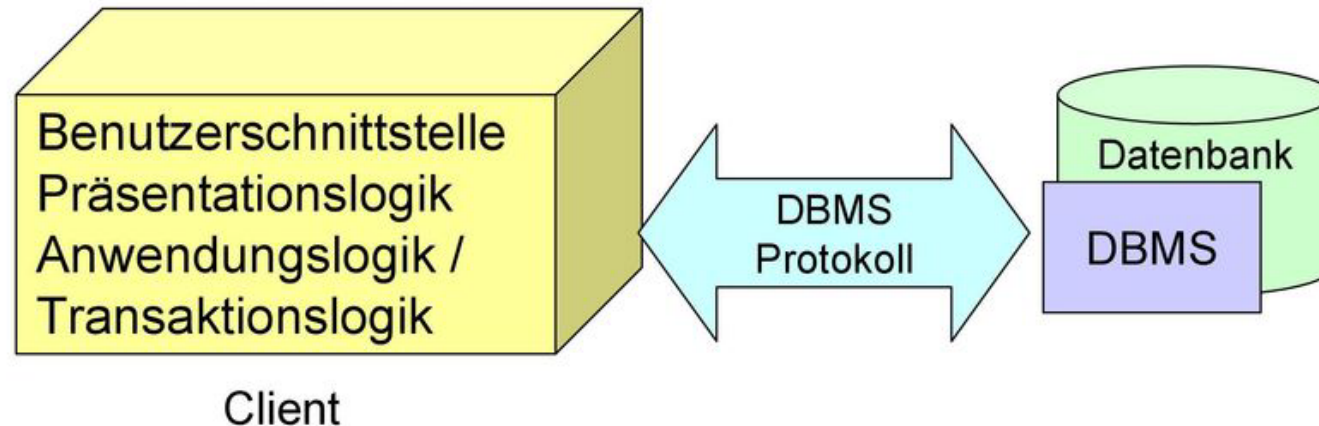


2.1 3-Ebenen-Datenbank-Architektur



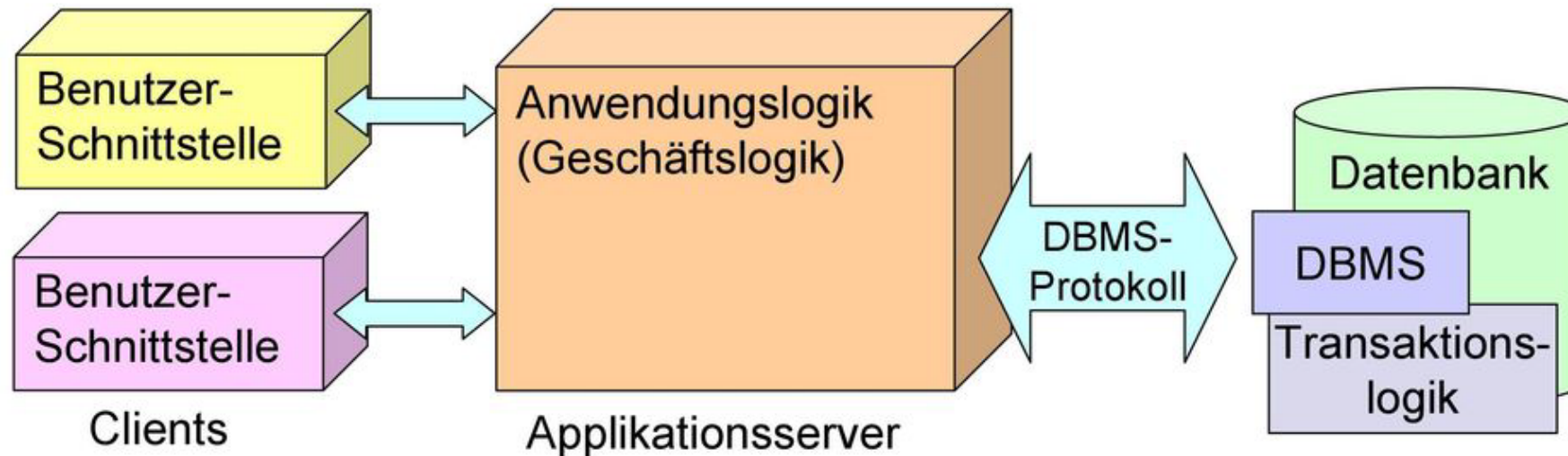
Architektur von Datenbanksystemen

- Client / Server Modell (2 Schichten Modell)



Architektur von Datenbanksystemen

- Das 3-Schichten-Modell



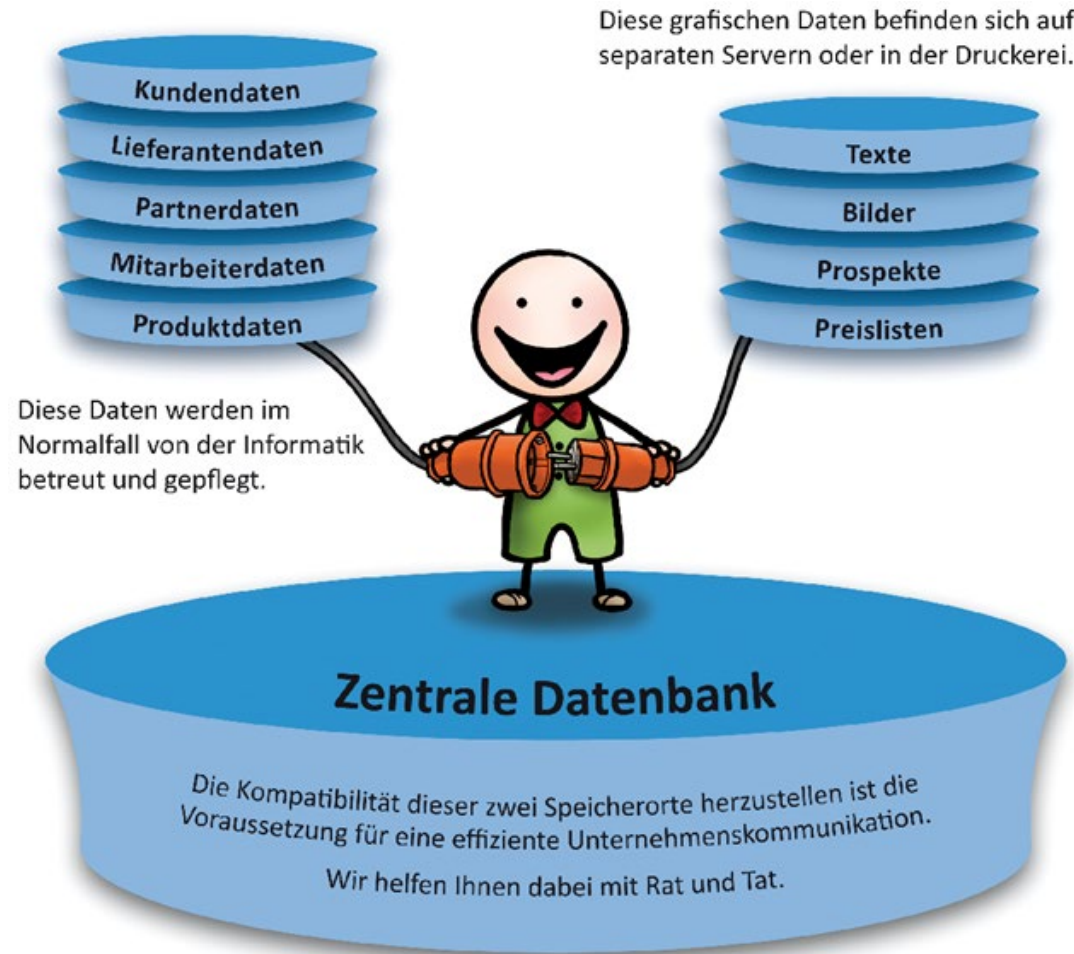
DBMS – Schichtenmodell

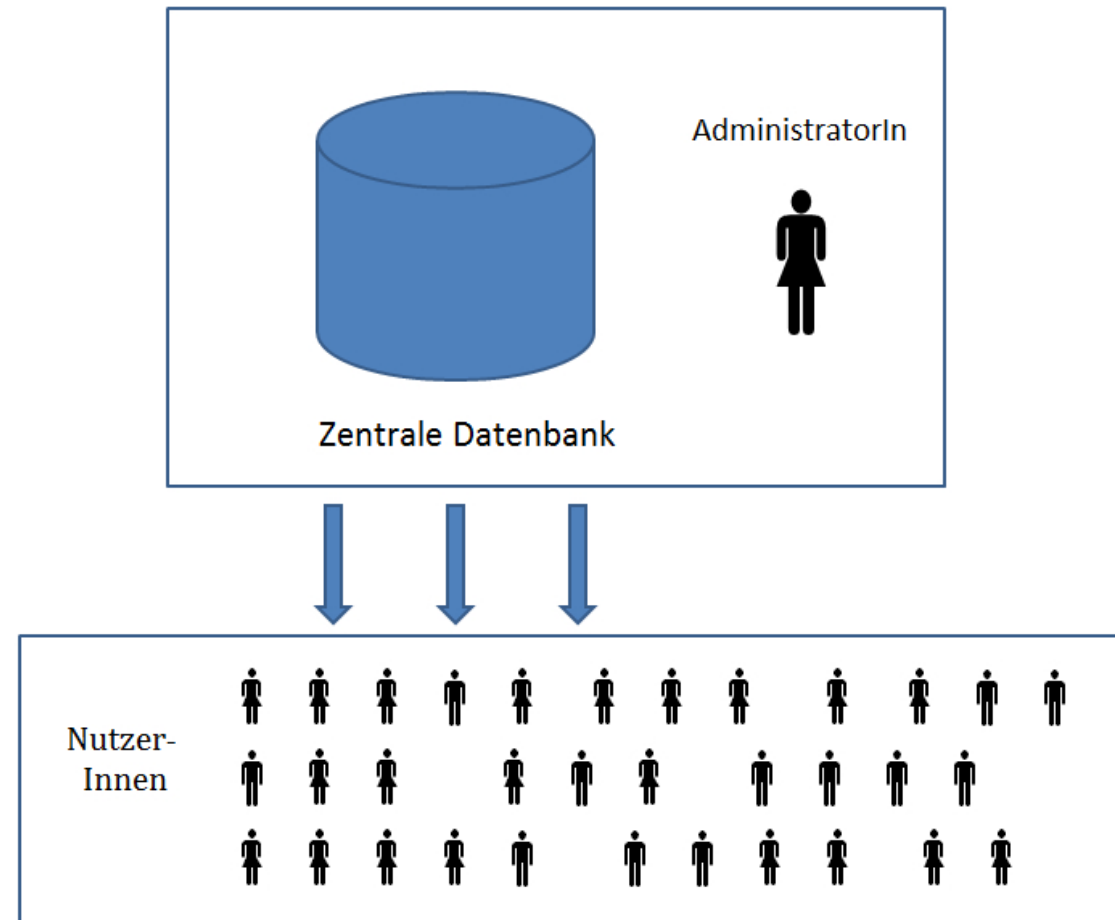
- Physische Schicht: Konkrete Umsetzung der Operationen (SW) und Speicherung auf dem Gerät, auf dem DB aufgesetzt/installiert ist.
- Logische Schicht: Ausprägung einer DB so wie sie von einem Superuser einsehbar ist. Es kann auf den gesamten Datenbestand zugegriffen werden.
- Sichtungsschicht: Nicht jeder Benutzer der DB benötigt Zugriff auf gesamte Informationen in der DB. Deshalb werden über Sichten die Datensätze eingeschränkt, auf die der Benutzer Zugriff hat.

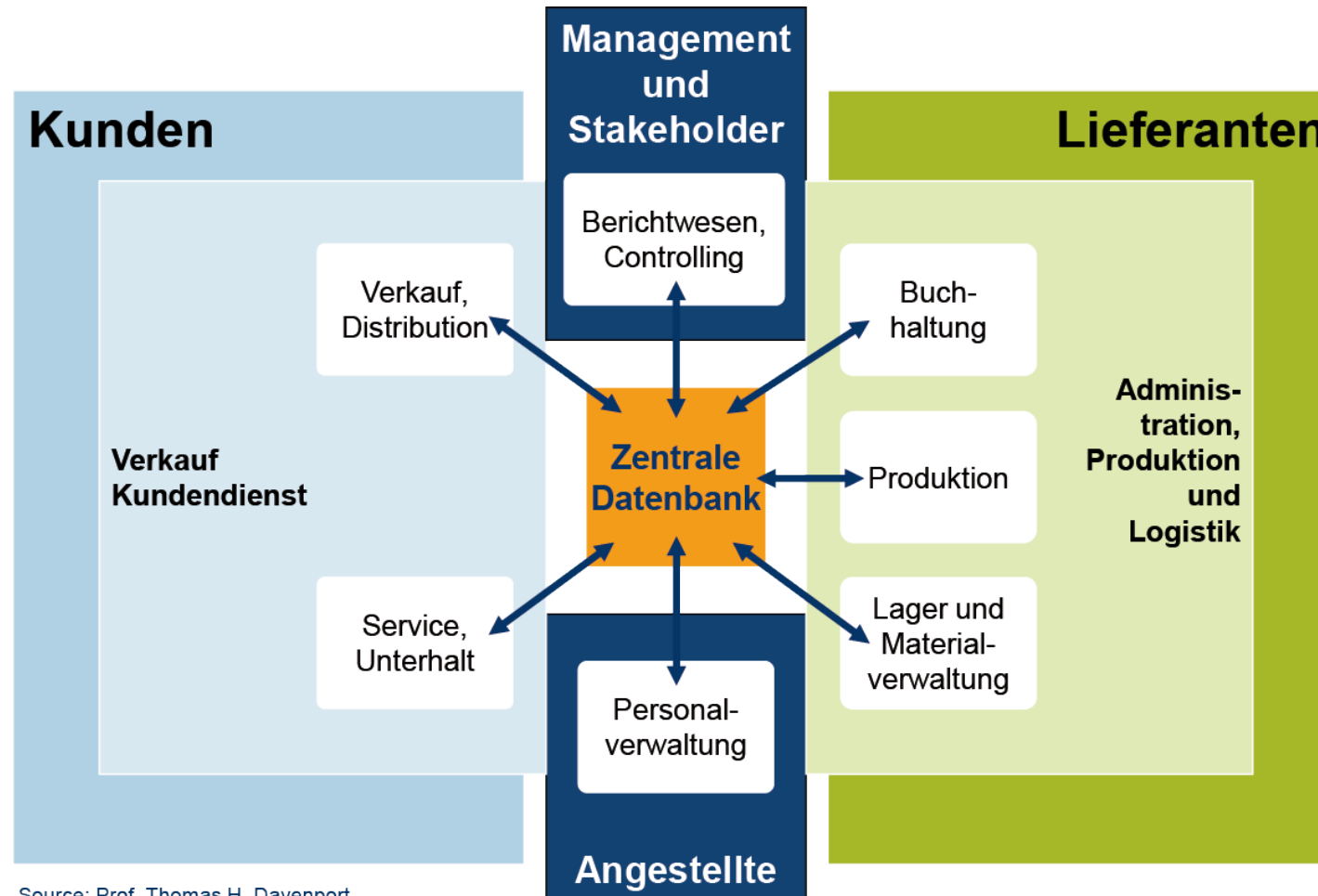
Schichten sind unabhängig voneinander. Eine Änderung in der einen Schicht macht keine Änderungen in einer anderen Schicht nötig.

Physische DB-Architektur

- Zentralisierte DBS
- Verteilte DBS
- Client-Server DBS
- Parallele DBS







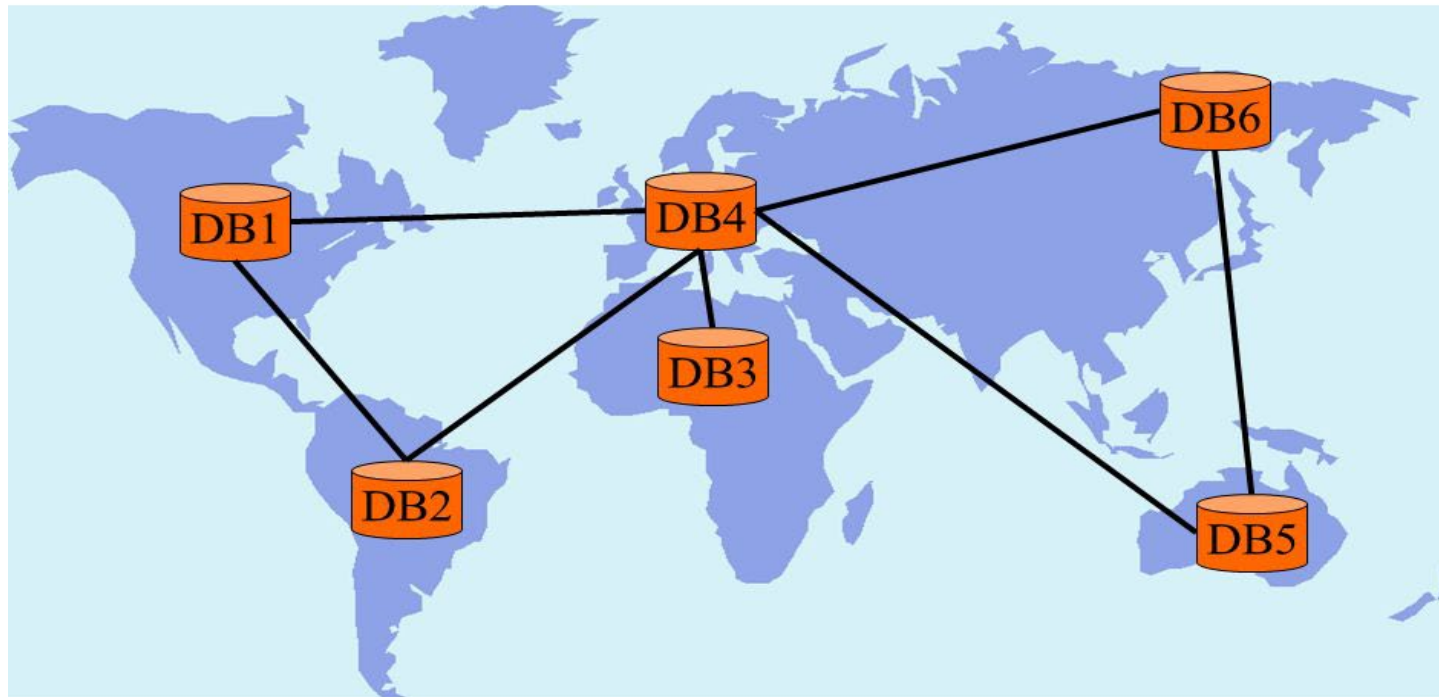
Source: Prof. Thomas H. Davenport,
Boston University School of Management

Verteilte Datenbanken

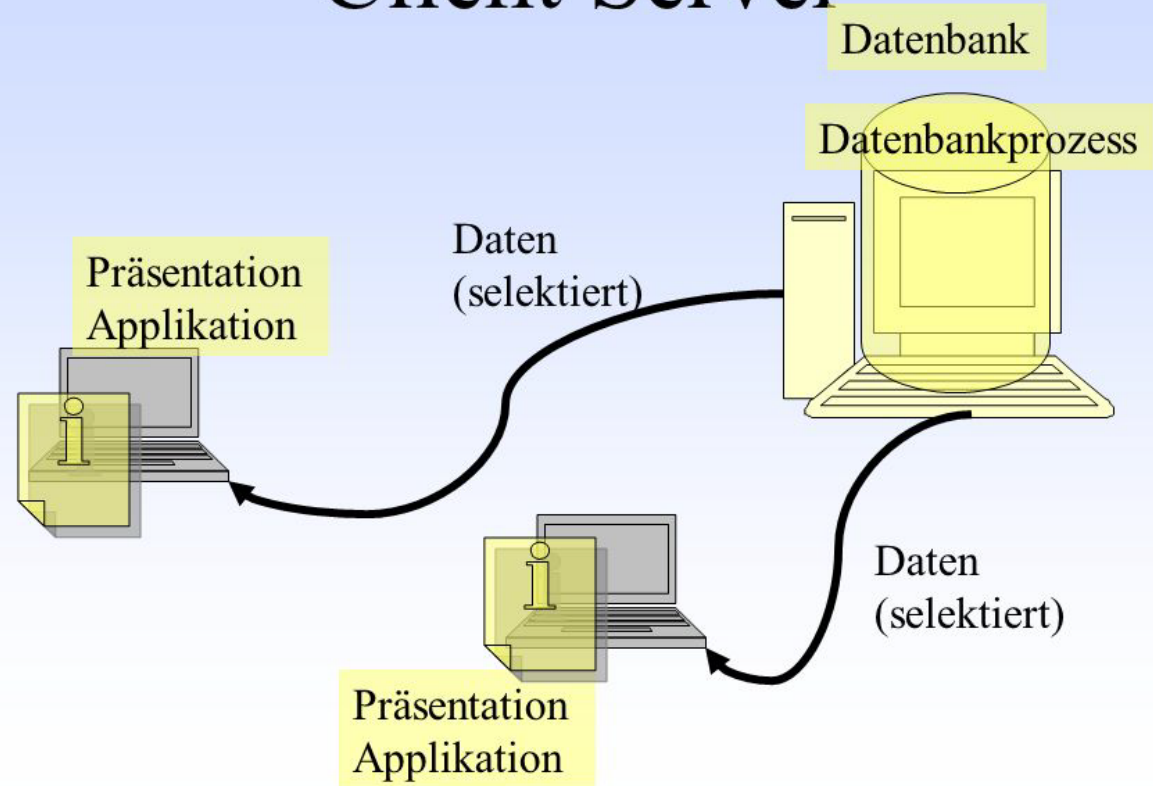
Szenario:

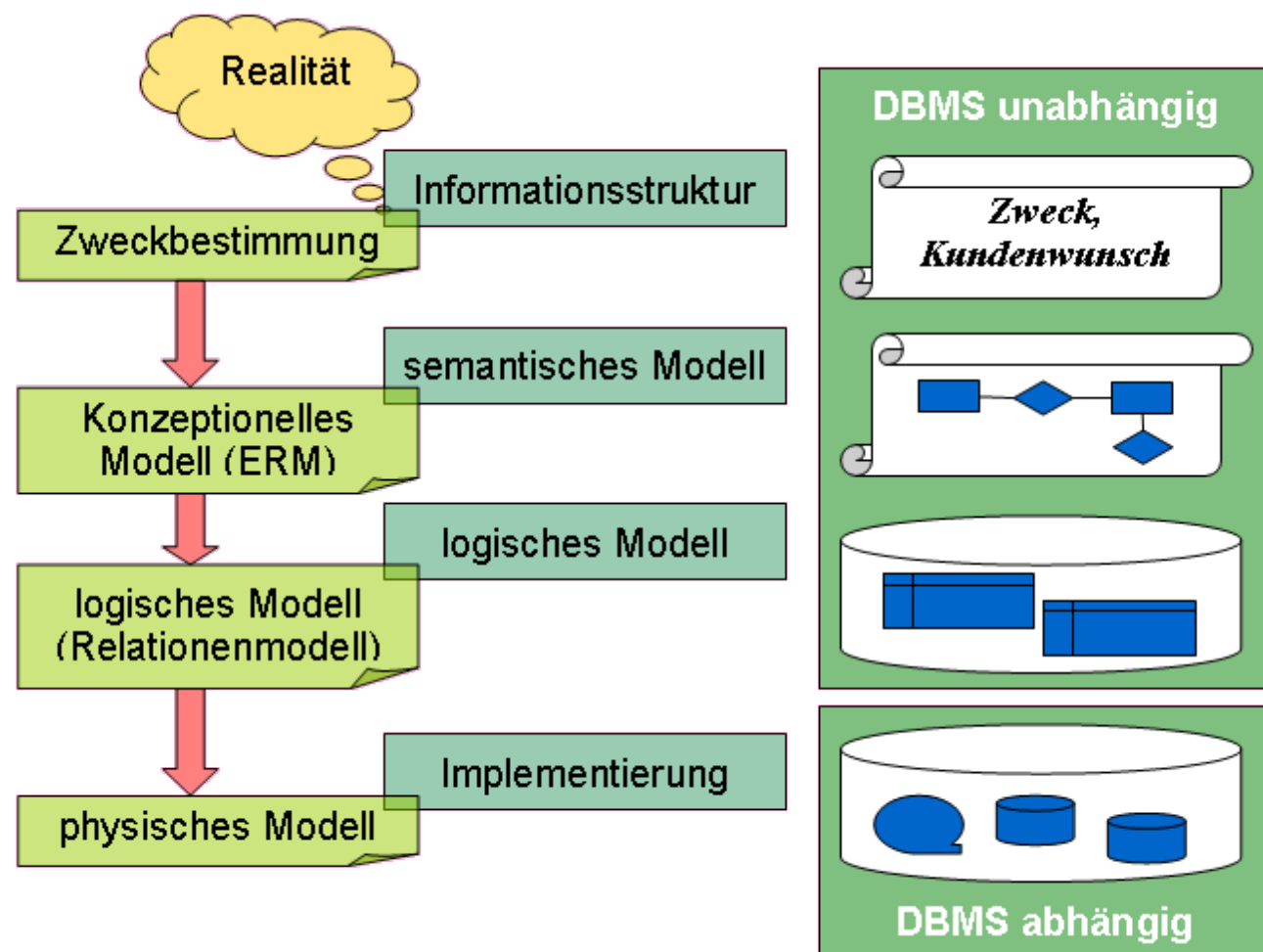
- Geographisch verteilte Organisationsform einer Bank mit ihren Filialen
- Filialen sollen Daten lokaler Kunden lokal bearbeiten können
- Zentrale soll Zugriff auf alle Daten haben (z.B. für Kontostandsüberprüfung bei Kreditvergabe)

Weiträumig verteilte Datenbanken



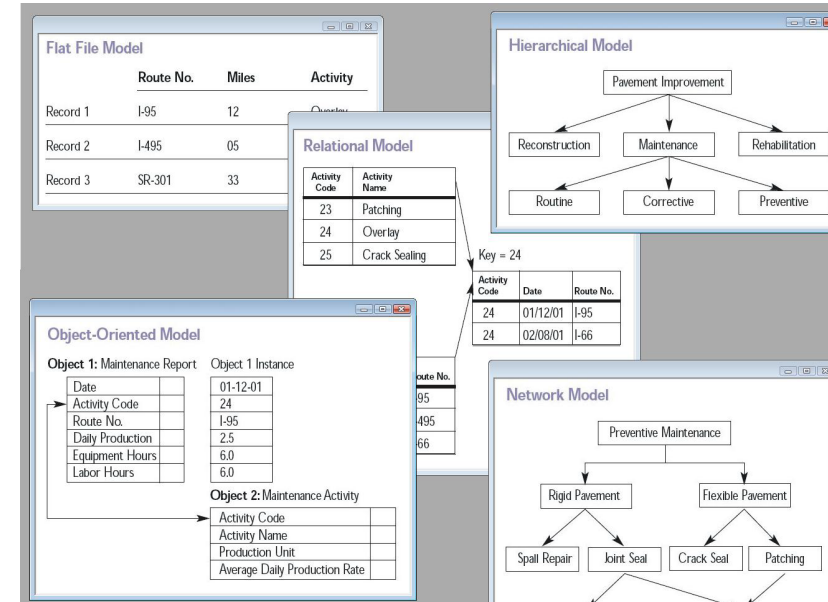
Client Server





Arten von DBMS

- Relational
- Objektorientiert
- Dokumentenorientiert und XML-DB
- NoSQL
- Elastic Search DB
- Historisch: Hierarchisch oder Netzwerkorientiert



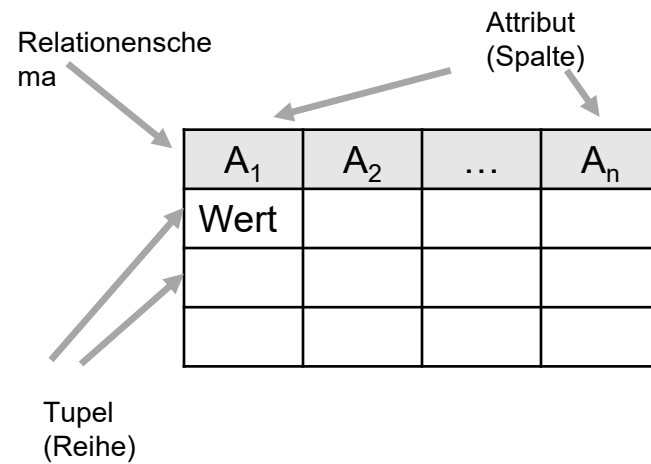
<https://en.wikipedia.org/wiki/Database> (abgerufen: 08.03.2018)

Referenzen:

- <https://db.in.tum.de/teaching/bookDBMSeinf/folien/index.shtml> (abgerufen: 08.03.2018)
- <https://de.wikipedia.org/wiki/Datenbank> (abgerufen: 08.03.2018)

Relationale Datenbanken

- Deklarative Methode um Daten und Abfragen anzugeben
- In der Praxis das meist verwendete Modell



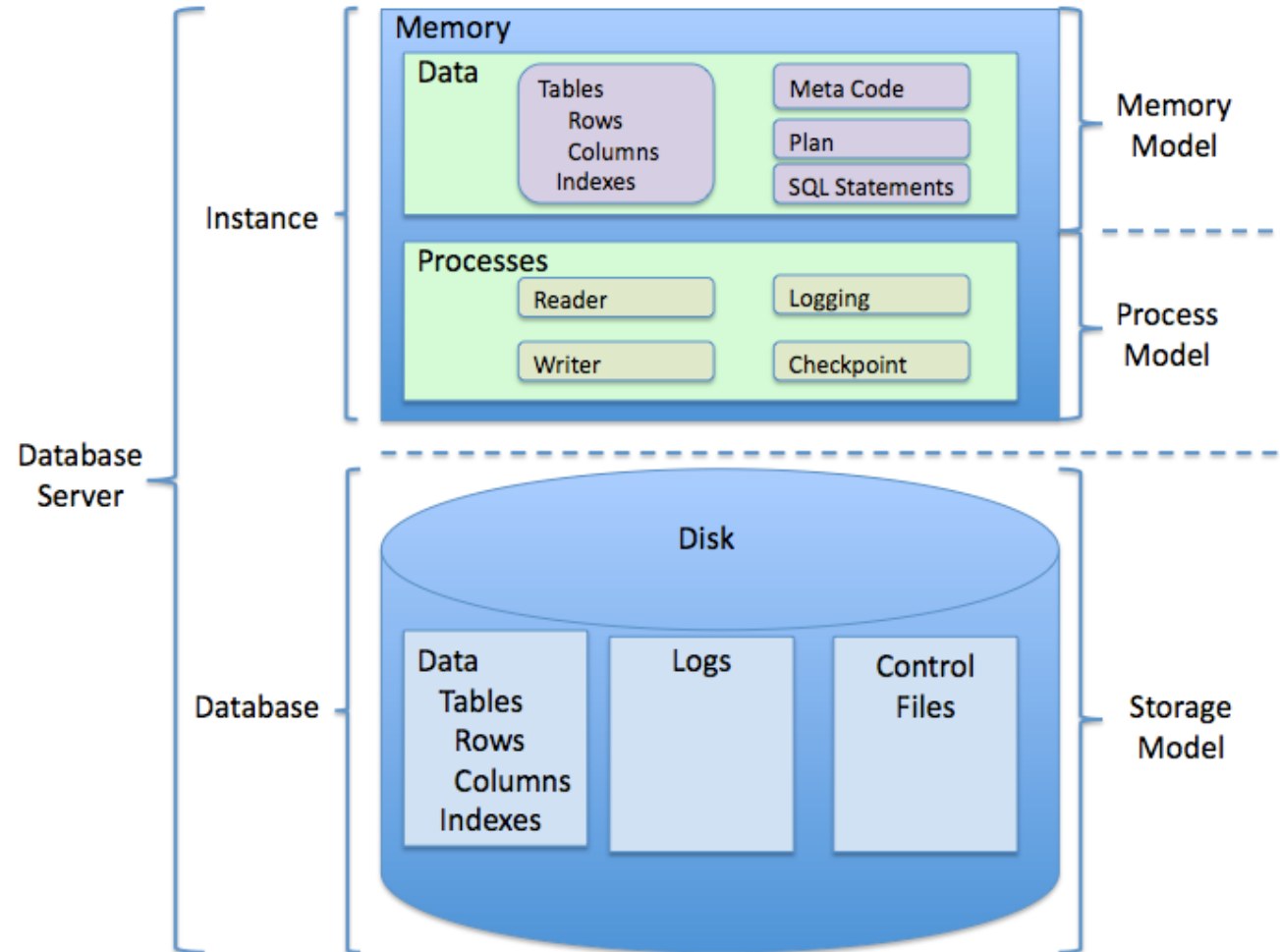
Activity Code	Activity Name
23	Patching
24	Overlay
25	Crack Sealing

Key = 24

Activity Code	Date	Route No.
24	01/12/01	I-95
24	02/08/01	I-66

Date	Activity Code	Route No.
01/12/01	24	I-95
01/15/01	23	I-495
02/08/01	24	I-66

https://en.wikipedia.org/wiki/Relational_model (abgerufen: 08.03.2018)



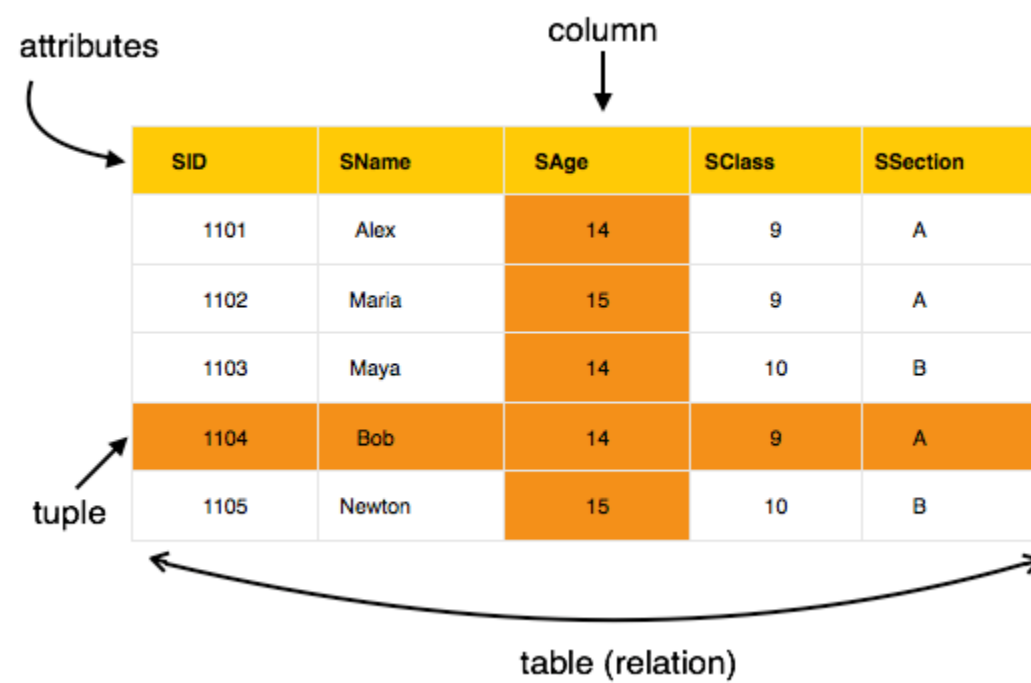
attributes

column

SID	SName	SAge	SClass	SSection
1101	Alex	14	9	A
1102	Maria	15	9	A
1103	Maya	14	10	B
1104	Bob	14	9	A
1105	Newton	15	10	B

tuple

table (relation)

The diagram shows a table with 5 columns and 6 rows. The first row is the header with attributes: SID, SName, SAge, SClass, and SSection. The following five rows represent tuples. Arrows point from the labels to the corresponding parts of the table: 'attributes' points to the header row, 'column' points to the SAge column, 'tuple' points to the 1104 row, and 'table (relation)' points to the entire table structure.

Relational Model

Activity Code	Activity Name
23	Patching
24	Overlay
25	Crack Sealing

Key = 24

Activity Code	Date	Route No.
24	01/12/01	I-95
24	02/08/01	I-66

Date	Activity Code	Route No.
01/12/01	24	I-95
01/15/01	23	I-495
02/08/01	24	I-66

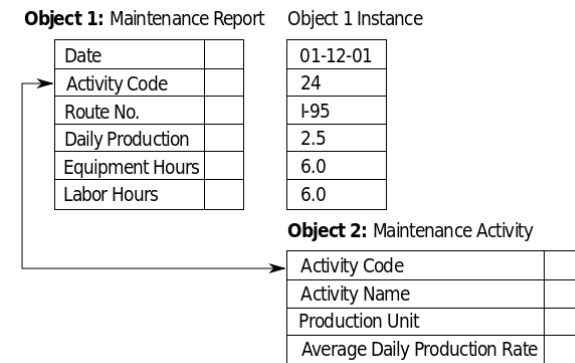
Relationale Datenbanken

- Ausprägung: der aktuelle Zustand der Datenbasis
- Schlüssel: minimale Menge von Attributen, deren Werte ein Tupel eindeutig identifizieren; mehrere Schlüsselkandidaten sind möglich
- Primärschlüssel: der ausgewählte Schlüsselkandidat
- Produkte: OracleDB, MS SQL Server, MySQL, PostgreSQL, SQLite3
- Objekte, Constraints ????

Objektorientierte und Objektrelationale DBs



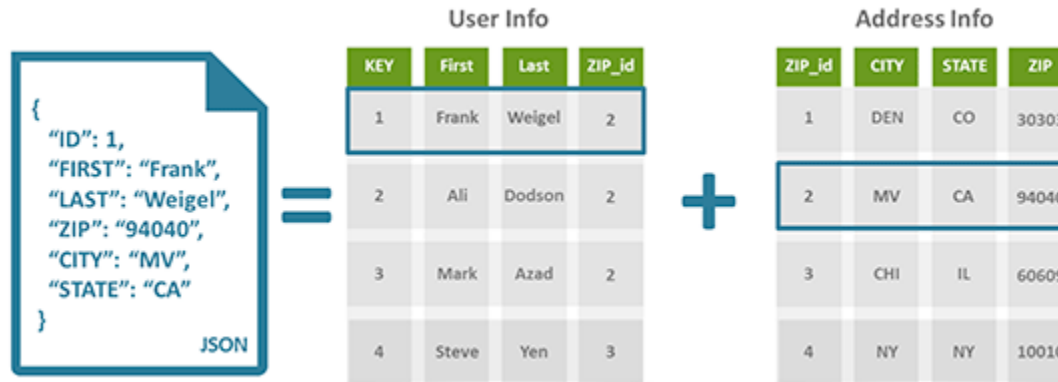
- Nachteile von Relationalen DBMS: Segmentierung, Künstliche Schlüsselattribute, Fehlendes Verhalten, Externe Programmierschnittstelle
- Objektorientiert:
 - Datenbeschreibung, -manipulation und -abfrage direkt über Programmiersprache
 - Objektkapselung und Wiederverwendbarkeit
- Objektrelational
 - Einbindung objektorientierter DB Funktionen in relationale DBs
 - Große Objekte – BLOBs, z.B. Mediendateien: Bilder, Audio, Video
 - Attribute können wieder aus Relationen bestehen
- Beispiele: Cache



https://en.wikipedia.org/wiki/Object_database (abgerufen: 08.03.2018)

Dokumentenorientiert und XML-DB

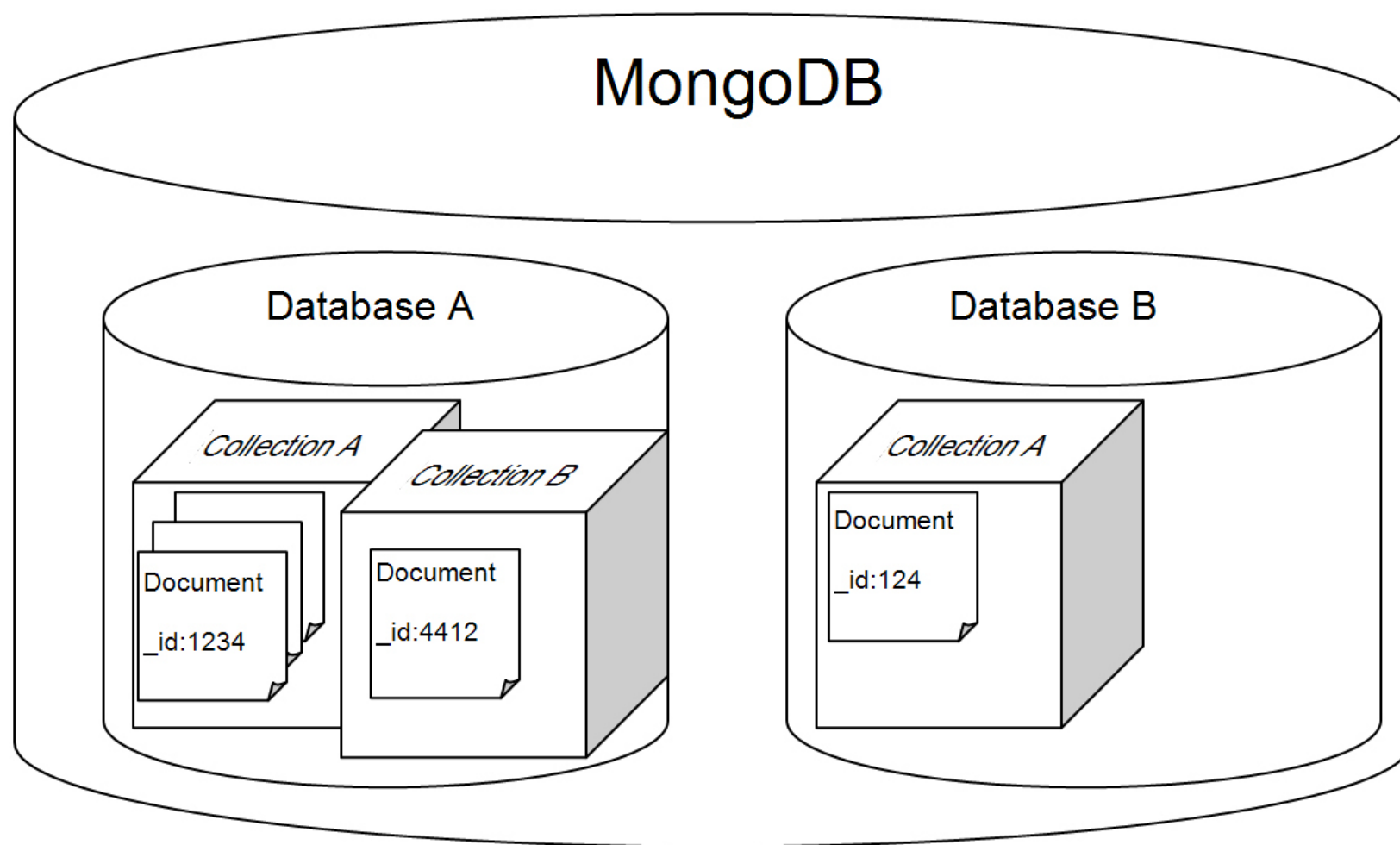
- Daten werden nicht in Form strukturierter Tabellen sondern als Dokumente abgelegt.
- XML DBs legen Dokumente im XML Format ab.
- XQuery, XPath, XSLT um Daten zu durchsuchen und modifizieren.
- Andere Art der Interpretation von Datenstrukturen und –zugriffe.
- Modelle trotzdem meist relational oder objektorientiert.
- Beispiele: nativ – Berkley DB XML; als Erweiterung: Oracle, MS SQL Server



Dokumentenorientierte Datenbanken

- Speichern Daten in Form von Dokumenten
- Semistrukturierte Inhalte
- JSON, YAML, XML

```
{
  "_id"      : ObjectId("42"),
  "firstname" : "John",
  "lastname"  : "Lennon",
  "address"   : { "city"   : "Liverpool",
                  "street" : "251 Menlove Avenue" }
}
```



Dokumente

- ❖ Entsprechen „in etwa“ einer Zeile in einer relationalen DB
- ❖ Haben „beliebig“ viele Attribute in „beliebiger“ Verschachtelung
- ❖ Normalisierung der Daten spielt eine geringere Rolle
- ❖ Dokumente kennen keine Relationen zu anderen Dokumenten (Foreign Keys oder Constraints)
- ❖ Viele Datenbanken können binäre Attachments verwalten

```
Source
{
  "_id": "d784be9e079d0100e0977e660d000748",
  "type": "order",
  "created": "2011-01-21",
  "items": [
    {
      "item_id": 1234325,
      "name": "Pizza A",
      "qty": 5,
      "ppi": 6,
      "sum": 30
    },
    {
      "item_id": 1255324,
      "name": "Pizza B",
      "qty": 1,
      "ppi": 12.5,
      "sum": 12.5
    }
  ],
  "sum": 42.5
}
```

Dokumentenorientierte Datenbanken

- Speicherung der Daten in Dokumenten
- Dokumente sind strukturierte Zusammenstellungen von Daten, z.B. JSON

```
{
  "vorname": "Dominik",
  "nachname": "Schmitz",
  "alter": 21,
  "adresse": {
    "strasse": "Aachener Str. 1",
    "stadt": "Alsdorf",
    "plz": 52477,
    "bundesland": "NRW"
  },
  "telefon": [
    {"typ": "festnetz", "nummer": "02404 32168"},
    {"typ": "mobil", "nummer": "0151 123456789"}
  ]
}
```

© 2015 TravelTainment

NoSQL = not only SQL

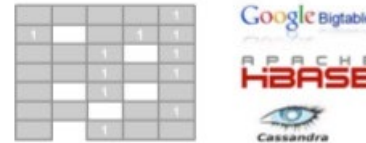
- <https://www.slideshare.net/ReginaHolzapfel/warum-nosql-wann-macht-der-einsatz-von-nosql-datenbanken-sinn-50890929>

NoSQL-Datenbanken

Key-Value Stores



Column Stores



Graph Databases



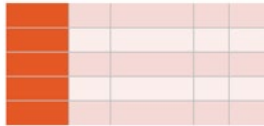
Document Stores



noSQL – not only SQL

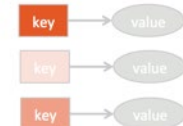
SQL-Datenbank-Modell

Relational:



noSQL-Datenbank-Modelle

Key-Value:



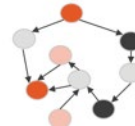
Wide-Column:



Document-Store:

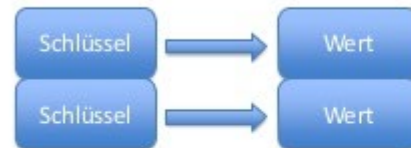


Graph:

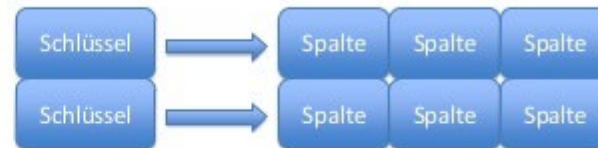


Typen von NoSQL Datenbanken

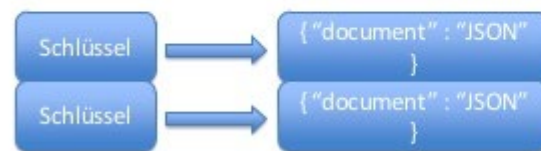
Stark vereinfacht, ohne Memory Caches & Analytische Datenbanken



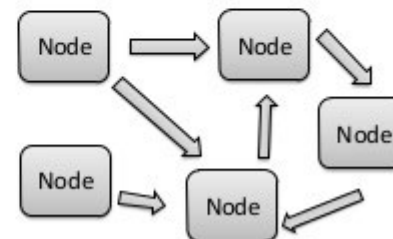
z.B. Riak



z.B. DynamoDB, Cassandra



z.B. MongoDB, ClouhDB



z. B. Neo4j

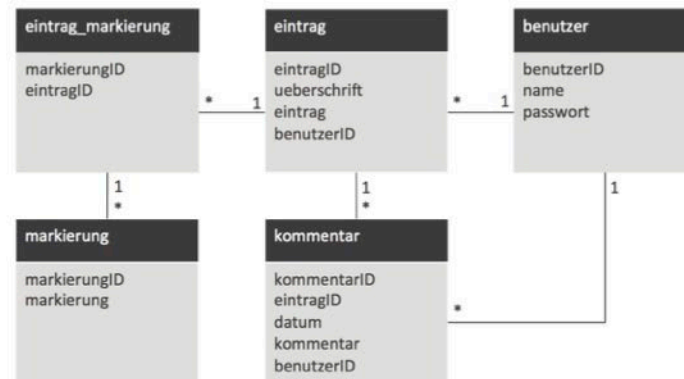
+ diverse andere Typen

Daten-Modellierung relational vs. noSQL

Am Beispiel eines Blog-Systems

Relationale Daten-Modellierung

Normalisierung zur Redundanz-Vermeidung



noSQL Daten-Modellierung

Aggregat¹-Orientierung behandelt verbundene Objekte als Einheit



NoSQL

- NoSQL – Not Only SQL
- Indexierung großer Anzahl von Dokumenten
- Webseiten mit hohem Lastaufkommen (z.B. YouTube)
- Streaming-Media-Applikationen
- Relational kann schlecht mit hohen Datenraten bei häufigen Datenänderungen umgehen
- Weniger Beschränkungen als Relational, z.B. Konsistenz, ACID nicht immer gegeben
- Beispiele: MongoDB, CouchDB, Apache Cassandra, Google Big Table

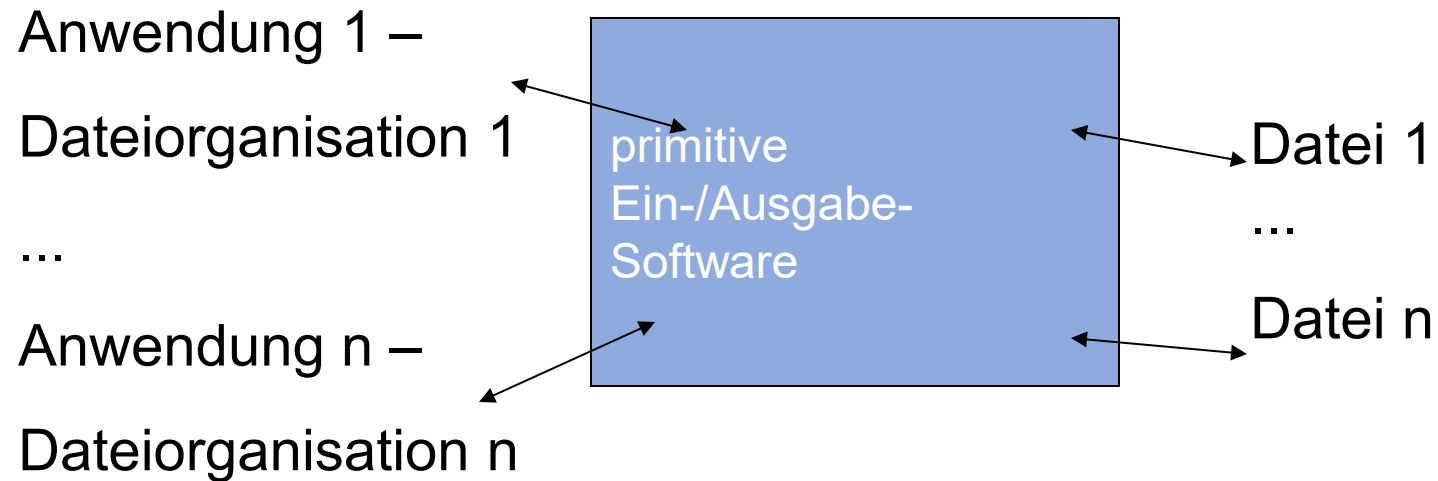
und sonst



Datenbanksystem-Generationen

1. 50er J. Dateisystem auf Band, nur sequentieller Zugriff, Batchbetrieb
2. 60er J. Dateisystem auf Platte, Random Access, Dialogbetrieb, Indexdateien, Dateiverwaltungssystem
3. 70er J. Prärelationale Systeme (Netzwerk-, hierarchische Systeme)
4. 80er J. Relationale Systeme
5. 90er J. objektbasierte Systeme

1. Generation (50er J.)



anwendungsspezifische Datenorganisation

Geräteabhängigkeit der Programme

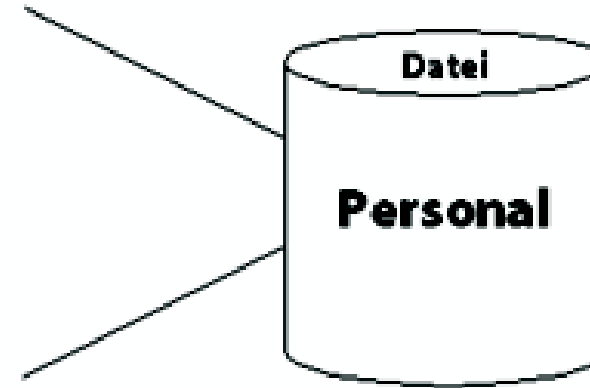
Redundanz, Inkonsistenz der Daten

50er Jahre: Dateisysteme

- Daten speichern in einzelne Dateien
- Dateiorganisation ist anwendungsspezifisch:
 - Öffnen von Dateien zum Lesen und/oder Schreiben
 - Positionieren innerhalb von Dateien auf bestimmte Sätze mit Hilfe von Dateizeigern
 - Lesen von Sätzen aus einer Datei
 - Schreiben von Sätzen in eine Datei
 - Erkennen des Dateiendes
 - Schließen von Dateien.

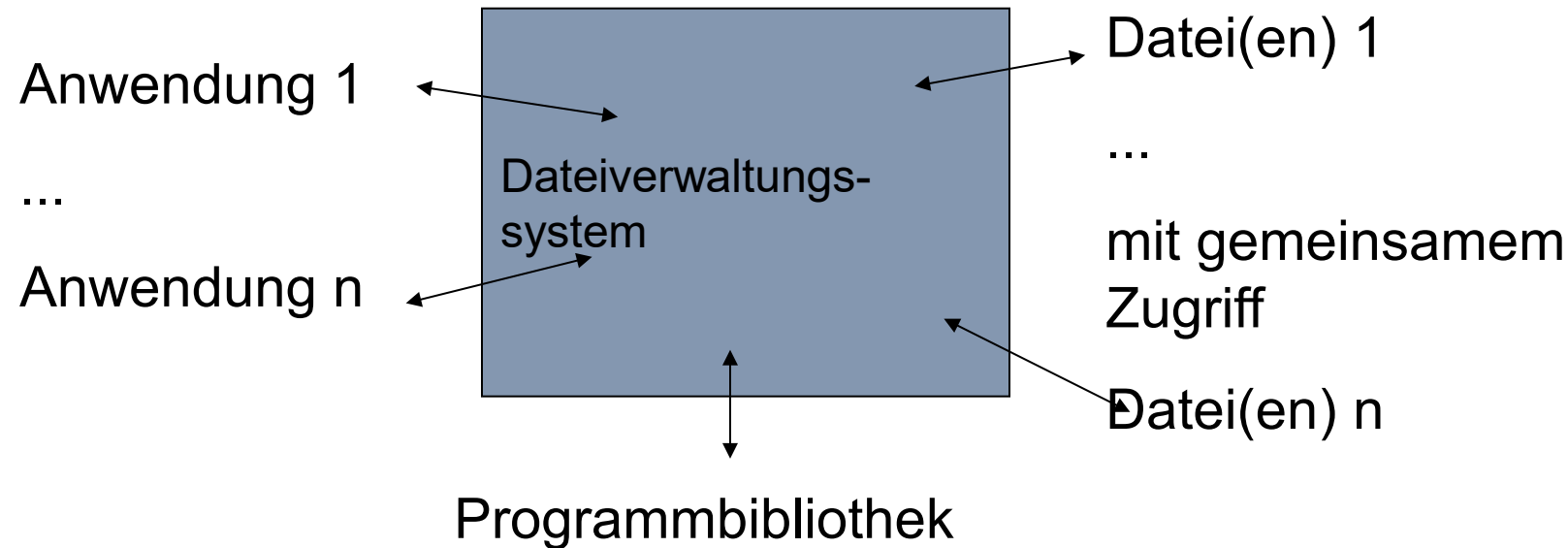
Beispiel Dateisystem

Persnr	Name	Abteilung	Gehalt
2450	Schulze	20	3500
2453	Müller	10	3700
2458	Meier	20	3600
2449	Schmidt	10	3500
2462	Meier-Schulze	20	3400



```
begin
  maxgehalt = 0
  öffne Datei Personal
  solange nicht Dateiende
    lies nächsten Satz
    falls (Abteilung = 20 und gehalt > maxgehalt)
      maxgehalt = gehalt
  schließe Datei Personal
  gib maxgehalt aus
end
```

2. Generation (60er J.)



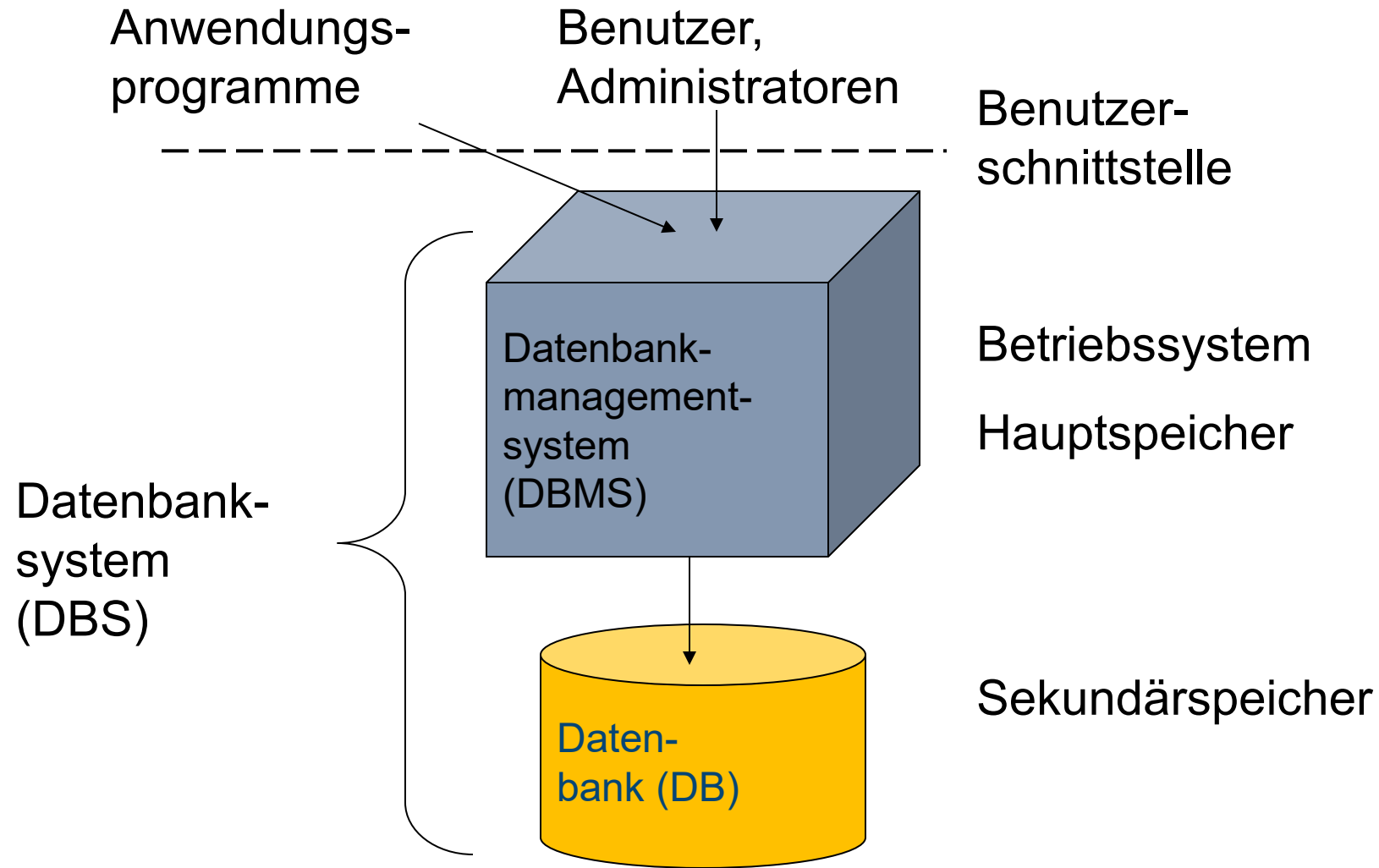
- teilw. standardisierte Datenorganisation
- Dienstprogramme wie z. B. Sortierer
- Geräteunabhängigkeit
- jedoch: Abhängigkeit von Speicherstruktur und Zugriffspfaden

Datenbanksysteme (seit 70er J.)

Datenbanksystem (DBS) besteht aus:

- Datenbankmanagementsystem (DBMS)
 - Software zur Verwaltung von Datenbeständen
 - Schnittstelle zwischen Benutzer und Datenbank, hierzu DB-Sprache, z.B. SQL
 - realisiert Konsistenz und Datenunabhängigkeit
- Datenbank (DB)
 - integrierter Datenbestand
 - einheitlich gemäß Datenmodell strukturiert

Aufbau von Datenbanksystemen



Aufgaben von DBMS

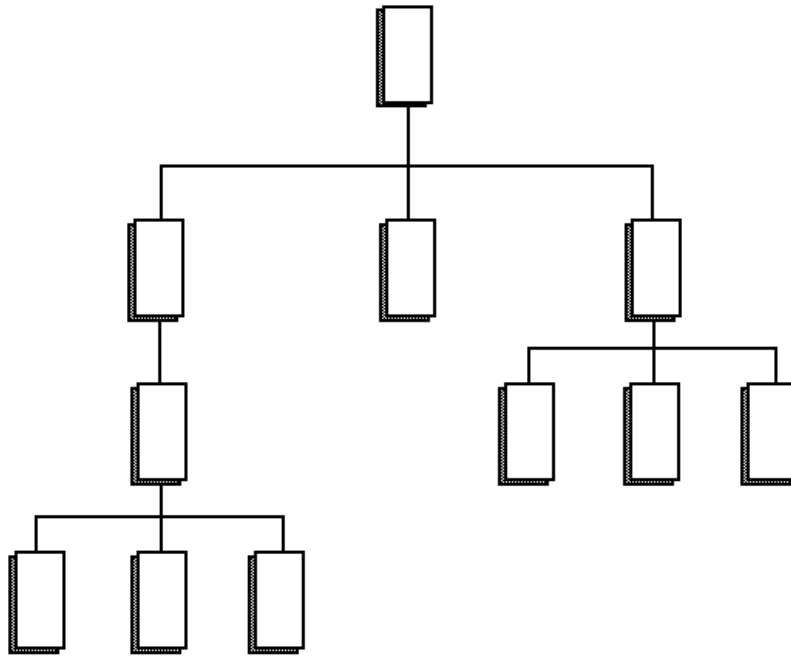


- Speicherung u. Verwaltung großer Datenbestände
- Speicherung dauerhaft, frei von Redundanzen, Konsistenzbedingungen genügend
- Daten vor unberechtigtem Zugriff geschützt
- Anfragen sowie Aktualisierungen möglich
- effizienter / schneller Zugriff auf die Daten
- mehrere Benutzer gleichzeitig aktiv
- Zugriffe über Netz oder auf mehrere Datenbanken
- mit Anwendungs-Software koppelbar

Datenmodelle

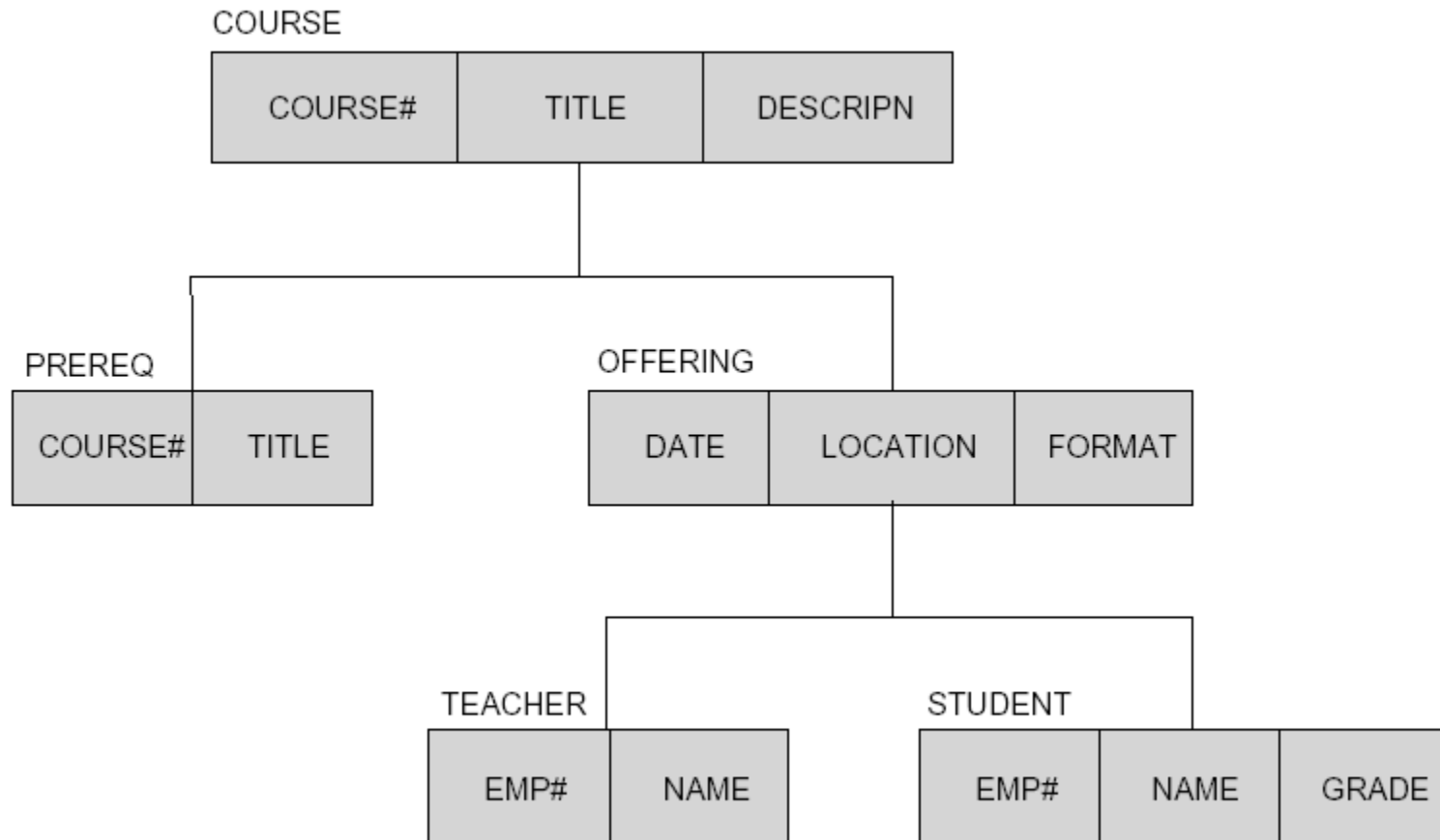
- Hierarchisches Datenmodell
IMS (Information Management System) IBM
- Netzwerkmodell
UDS von Siemens
- Relationales Datenmodell
dBase, MySQL, Access, SQL-Server, ...
- Objektorientiertes Datenmodell
teilw. Oracle, O₂ von O₂ Technology

Hierarchisches Datenmodell

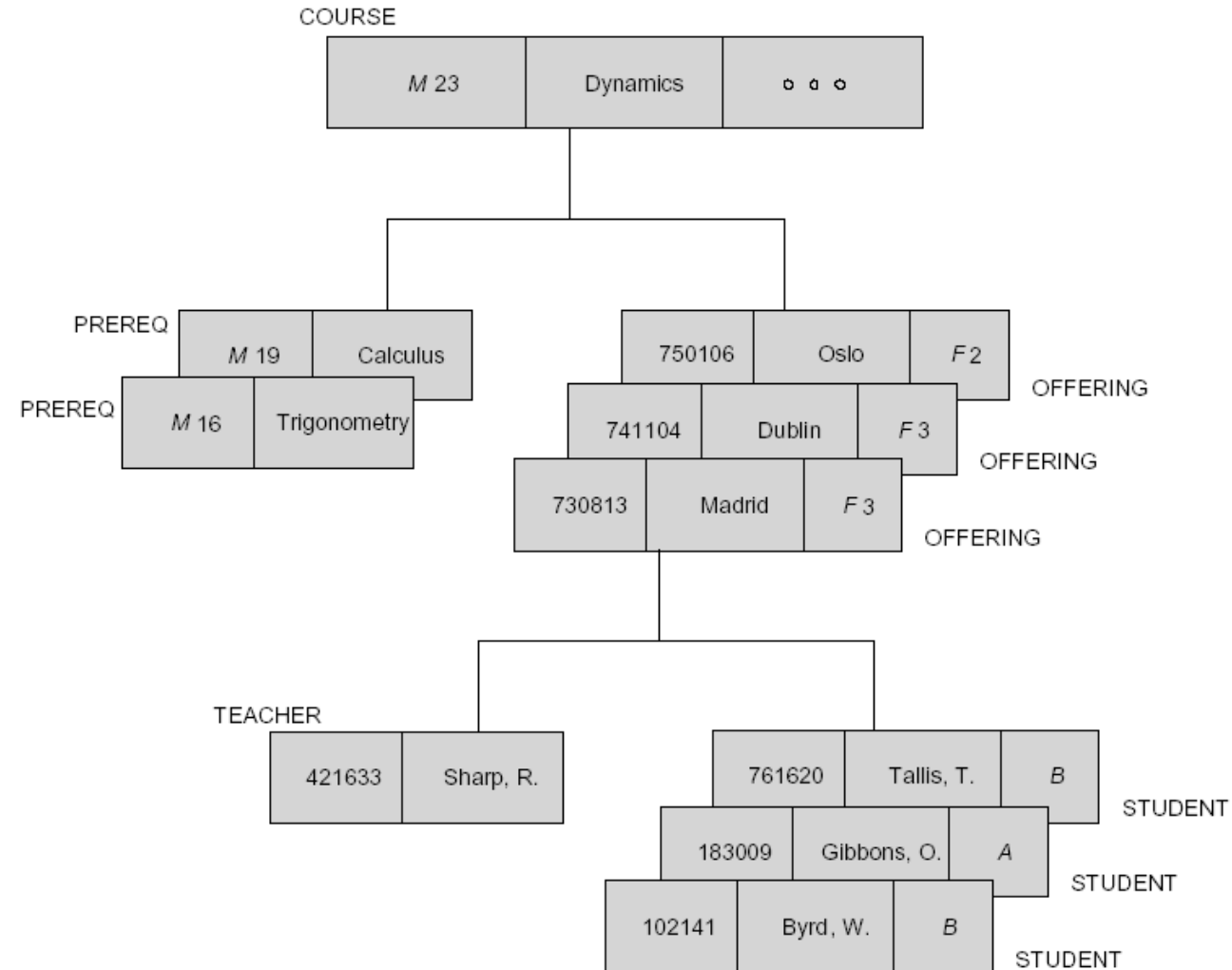


- 1960 entwickelt
- Baumstruktur mit einem Ausgangselement (Root)
- Jedes Element hat maximal einen Vorgänger (Parent)
- Jedes Element hat beliebig viele Nachfolger (Children)

Beispiel Hierarchisches Modell



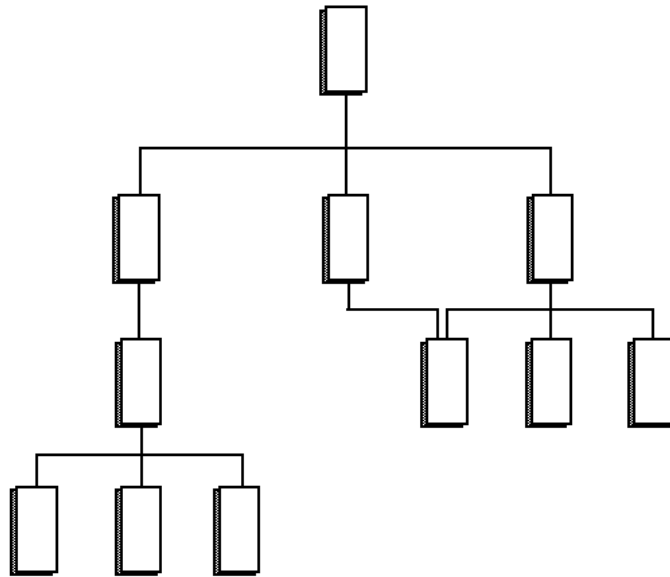
Ausprägung Hierarchisches Modell



Hierarchisches Modell heute

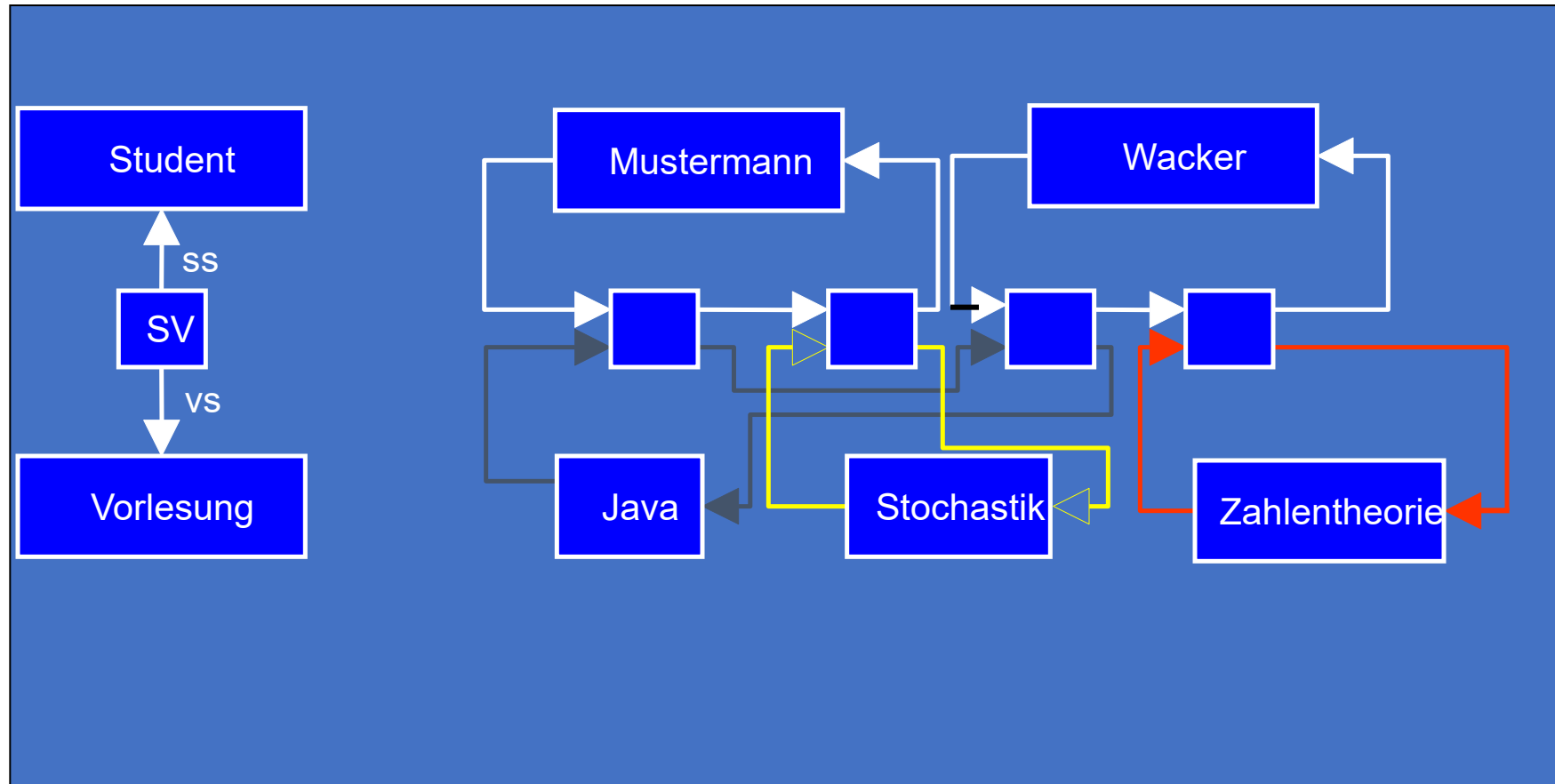
- LDAP (Lightweight Directory Access Protocol) organisiert Verzeichnisdienst
- Verzeichnis- und Dateistrukturen von Betriebssystemen
- HTML / XML
- IMS (Information Management System) von IBM

Netzwerkmodell



- Ab etwa 1970 fand das hierarchische Datenmodell seine Erweiterung im Netzwerkmodell, das nun auch Querverbindungen zwischen Baumstrukturen zuließ.
- Jeder Knoten kann mehrere übergeordnete Knoten haben.
- Jede Netzwerkstruktur lässt sich durch Einführung redundanter Knoten als hierarchische Baumstruktur darstellen.

Beispiel Netzwerk-Modell



Elastic Search DB

- Speichert Suchergebnisse in einem NoSQL-Format (JSON), Abfrage über REST Webinterface
- Einfacher Betrieb über mehrere Rechner (Server) verteilt.
- Dadurch höhere Verfügbarkeit bei großer Anzahl von Anfragen und Lastverteilung.
- Indizes werden auf Server aufgeteilt – d.h. nicht jeder Server hat komplette Daten/Information
- Beispiele: Suchmaschinen

- <https://crate.io/products/cratedb/>