

Teil I

Was sind Datenbanken?

Was sind Datenbanken?

- 1 Überblick & Motivation
- 2 Architekturen
- 3 Einsatzgebiete
- 4 Historisches

Was sind Datenbanken?

- Daten = logisch gruppierte Informationseinheiten
- Bank =



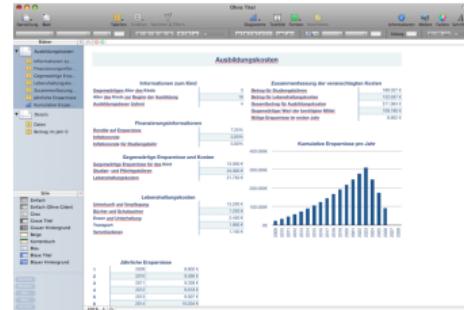
- Die **Sicherheit vor Verlusten** ist eine Hauptmotivation, etwas „auf die Bank zu bringen“.
- Eine Bank bietet **Dienstleistungen für mehrere Kunden** an, um effizient arbeiten zu können.
- Eine Datenbank hat die (langfristige) **Aufbewahrung** von Daten als Aufgabe.

Anwendungsbeispiele

The collage consists of five screenshots:

- A:** A screenshot of the Amazon.com shopping cart page showing items for purchase, including "Encyclopedia of Database Systems" and "Quicksilver (The Baroque Cycle, Vol. 1)". It highlights a \$30 rebate offer.
- B:** A screenshot of the SAP Crystal Reports Sales Performance Dashboard showing forecast vs. revenue for Q1-Q4 across various regions.
- C:** A screenshot of the Facebook Heidelberg 4 you page, featuring a banner image of a bridge at night and a bar chart showing user distribution by country.
- D:** A screenshot of a toll booth on a highway, with a sign indicating "Kein Bargeld!" (No cash!).
- E:** A screenshot of a Google search results page for "Langwiesener Straße 111" in Berlin, featuring a map of the area and a sidebar with local information.

Wie verwaltet man Daten?



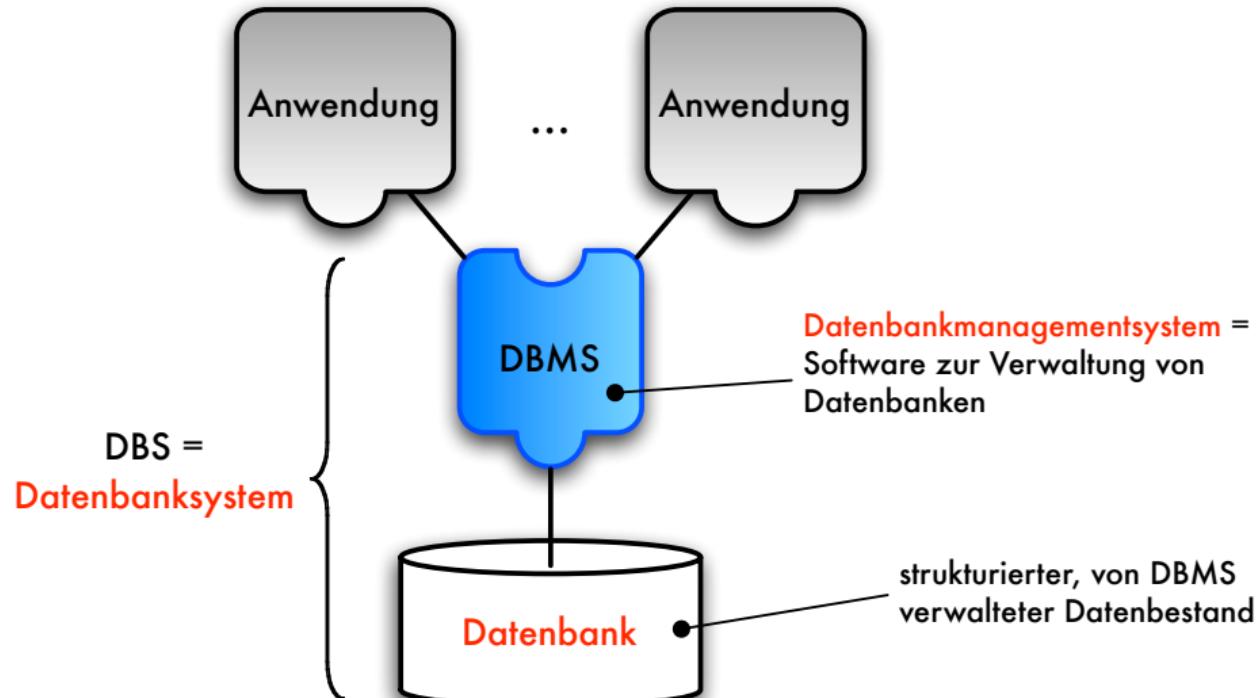
Ohne Datenbanken

- jedes Anwendungssystem verwaltet seine eigenen Daten
- Daten sind mehrfach gespeichert ↽ redundant
- Probleme
 - ▶ Verschwendung von Speicherplatz
 - ▶ „Vergessen“ von Änderungen
 - ▶ keine zentrale, „genormte“ Datenhaltung

Probleme der Datenredundanz

- Andere Softwaresysteme können große Mengen von Daten nicht effizient verarbeiten
- Mehrere Benutzer oder Anwendungen können nicht parallel auf den gleichen Daten arbeiten, ohne sich gegenseitig zu stören
- Anwendungsprogrammierer / Benutzer können Anwendungen nicht programmieren / benutzen, ohne
 - ▶ interne Darstellung der Daten
 - ▶ Speichermedien oder Rechner
- zu kennen (**Datenunabhängigkeit** nicht gewährleistet)
- Datenschutz und Datensicherheit sind nicht gewährleistet

Idee: Datenintegration durch Datenbanksysteme



Motivation

- Datenbanksysteme sind Herzstück heutiger IT-Infrastrukturen
- ... allgegenwärtig
- Datenbank- und Datenmanagementspezialisten sind gefragt



Der IT-Arbeitsmarkt: IT-Unternehmen

scrum

Für welche Arbeitsbereiche werden aktuell IT-Spezialisten gesucht?



Quelle: BITKOM

Fragestellungen

- ① Wie organisiert (modelliert und nutzt) man Daten?
- ② Wie werden Daten dauerhaft verlässlich gespeichert?
- ③ Wie kann man riesige Datenmengen (mehrere Peta/Terabyte) effizient verarbeiten?
- ④ Wie können viele Nutzer (≥ 10.000) gleichzeitig mit den Daten arbeiten?

Prinzipien: Die neun Codd'schen Regeln

- ① **Integration:** einheitliche, nichtredundante Datenverwaltung
- ② **Operationen:** Speichern, Suchen, Ändern
- ③ **Katalog:** Zugriffe auf Datenbankbeschreibungen im Data Dictionary
- ④ **Benutzersichten**
- ⑤ **Integritätssicherung:** Korrektheit des Datenbankinhalts
- ⑥ **Datenschutz:** Ausschluss unauthorisierter Zugriffe
- ⑦ **Transaktionen:** mehrere DB-Operationen als Funktionseinheit
- ⑧ **Synchronisation:** parallele Transaktionen koordinieren
- ⑨ **Datensicherung:** Wiederherstellung von Daten nach Systemfehlern

Datenunabhängigkeit und Schemata

- Basierend auf DBMS-Grobarchitektur
- Entkopplung von Benutzer- und Implementierungssicht
- Ziele u.a.:
 - ▶ Trennung von Modellierungssicht und interner Speicherung
 - ▶ Portierbarkeit
 - ▶ Tuning vereinfachen
 - ▶ standardisierte Schnittstellen

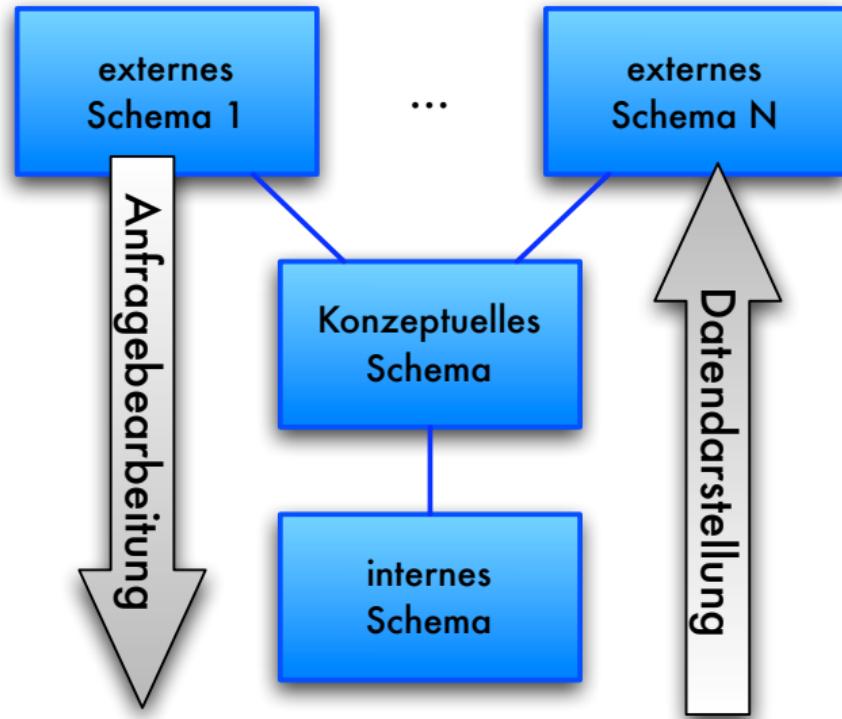
Schemaarchitektur

- Zusammenhang zwischen
 - ▶ Konzeptuellem Schema (Ergebnis der Datendefinition)
 - ▶ Internem Schema (Festlegung der Dateiorganisationen und Zugriffspfade)
 - ▶ Externen Schemata (Ergebnis der Sichtdefinition)
 - ▶ Anwendungsprogrammen (Ergebnis der Anwendungsprogrammierung)

Schemaarchitektur /2

- Trennung Schema — Instanz
 - ▶ Schema (Metadaten, Datenbeschreibungen)
 - ▶ Instanz (Anwenderdaten, Datenbankzustand oder -ausprägung)
- Datenbankschema besteht aus
 - ▶ internem, konzeptuellem, externen Schemata und den Anwendungsprogrammen
- im konzeptuellen Schema etwa:
 - ▶ Strukturbeschreibungen
 - ▶ Integritätsbedingungen
 - ▶ Autorisierungsregeln (pro Benutzer für erlaubte DB-Zugriffe)

Schemaarchitektur /3



Datenunabhängigkeit /1

- Stabilität der Benutzerschnittstelle gegen Änderungen
- **physisch**: Änderungen der Dateiorganisationen und Zugriffspfade haben keinen Einfluss auf das konzeptuelle Schema
- **logisch**: Änderungen am konzeptuellen und gewissen externen Schemata haben keine Auswirkungen auf andere externe Schemata und Anwendungsprogramme

Datenunabhängigkeit /2

- mögliche Auswirkungen von Änderungen am konzeptuellen Schema:
 - ▶ eventuell externe Schemata betroffen (Ändern von Attributen)
 - ▶ eventuell Anwendungsprogramme betroffen (Rekompilieren der Anwendungsprogramme, eventuell Änderungen nötig)
- nötige Änderungen werden jedoch vom DBMS erkannt und überwacht

Anwendungsbeispiel: Musikversand

The screenshot shows a music website interface. A red circle highlights the album cover of 'Living With War' by Neil Young. Red boxes labeled 'Musiker' (Artist), 'Titel' (Title), 'Jahr' (Year), 'Preis' (Price), and 'Rezension(en)' (Review(s)) are overlaid on the page. A large red oval encloses the tracklist table at the bottom.

Musiker: Neil Young

Titel: Living With War

Jahr: 2006

Preis: 9,99 €

Rezension(en):

Tracks:

#	Titel	Dauer	Interpret	Album	Preis	Aktion
1	After the Garden	3:21	Neil Young	Living With War	0,99 €	TITEL KAUFEN
2	Living With War	5:03	Neil Young	Living With War	0,99 €	TITEL KAUFEN
3	The Restless Consumer	5:46	Neil Young	Living With War	0,99 €	TITEL KAUFEN
4	Shock and Awe	4:51	Neil Young	Living With War	0,99 €	TITEL KAUFEN
5	Families	2:24	Neil Young	Living With War	0,99 €	TITEL KAUFEN
6	Flags of Freedom	3:40	Neil Young	Living With War	0,99 €	TITEL KAUFEN
7	Let's Impeach the President	5:07	Neil Young	Living With War	0,99 €	TITEL KAUFEN
8	Lookin' for a Leader	4:02	Neil Young	Living With War	0,99 €	TITEL KAUFEN
9	Roger and Out	4:23	Neil Young	Living With War	0,99 €	TITEL KAUFEN
10	America the Beautiful	2:56	Neil Young	Living With War	0,99 €	TITEL KAUFEN

Ebenen-Architektur am Beispiel

- Konzeptuelle Sicht: Darstellung in Tabellen (Relationen)

Musiker	MNr	Name	Land
	103	Apocalyptica	Finnland
	104	Subway To Sally	Deutschland
	105	Rammstein	Deutschland

Album	ANr	Titel	Jahr	Genre	MNr → Musiker
	1014	Amplified	2006	Rock	103
	1015	Nord Nord Ost	2005	Rock	104
	1016	Rosenrot	2005	Rock	105
	1021	Engelskrieger	2003	Rock	104
	1025	Reflections	2006	Rock	103

Ebenen-Architektur am Beispiel /2

- Externe Sicht: Daten in einer flachen Relation

ANr	Titel	Jahr	Genre	Musiker
1014	Amplified	2006	Rock	Apocalyptica
1015	Nord Nord Ost	2005	Rock	Subway To Sally
1016	Rosenrot	2005	Rock	Rammstein
1021	Engelskrieger	2003	Rock	Subway To Sally
1025	Reflections	2006	Rock	Apocalyptica

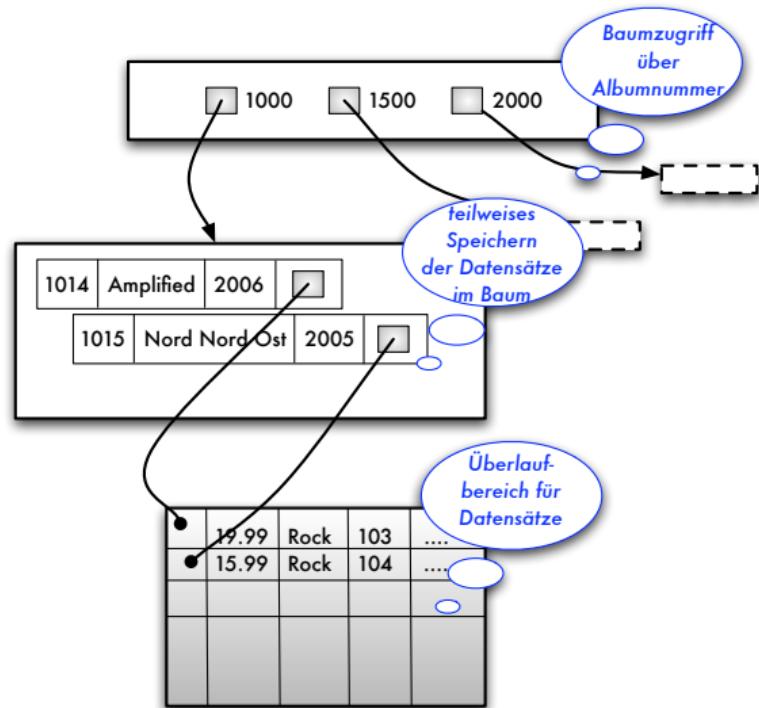
Ebenen-Architektur am Beispiel /3

- Externe Sicht: Daten in einer hierarchisch aufgebauten Relation

Musiker	Album		
	Titel	Jahr	Genre
Apolcalyptica	Amplified	2006	Rock
	Reflections	2003	Rock
Subway To Sally	Nord Nord Ost	2005	Metal
	Engelskrieger	2003	Rock
Rammstein	Rosenrot	2005	Rock

Ebenen-Architektur am Beispiel /4

- Interne Darstellung



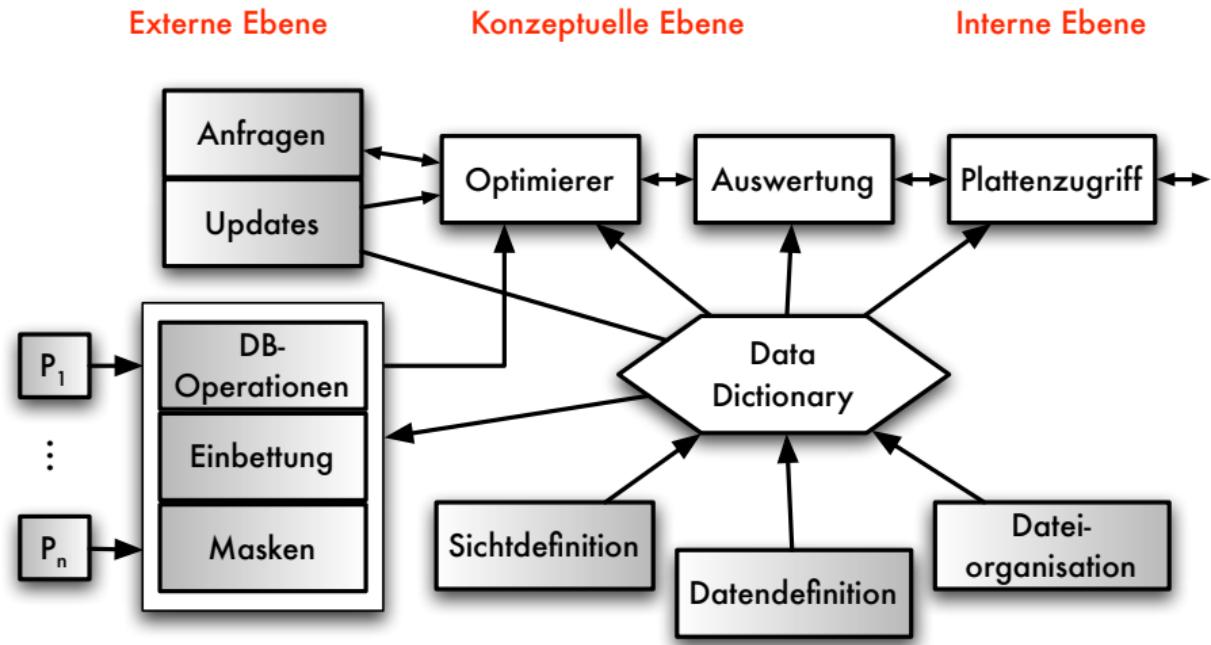
System-Architekturen

- Beschreibung der Komponenten eines Datenbanksystems
- Standardisierung der Schnittstellen zwischen Komponenten
- Architekturvorschläge
 - ▶ ANSI-SPARC-Architektur
 - ~~> Drei-Ebenen-Architektur
 - ▶ Fünf-Schichten-Architektur
 - ~~> beschreibt Transformationskomponenten im Detail

ANSI-SPARC-Architektur

- ANSI: American National Standards Institute
- SPARC: Standards Planning and Requirement Committee
- Vorschlag von 1978
- Im Wesentlichen Grobarchitektur verfeinert
 - ▶ Interne Ebene / Betriebssystem verfeinert
 - ▶ Mehr Interaktive und Programmier-Komponenten
 - ▶ Schnittstellen bezeichnet und normiert

ANSI-SPARC-Architektur /2

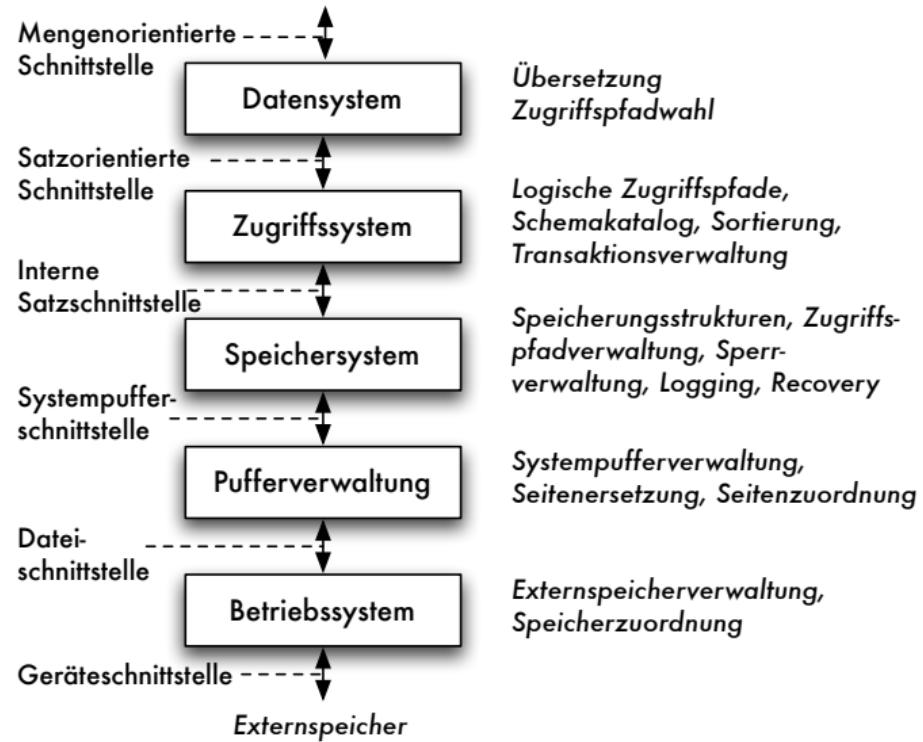


Klassifizierung der Komponenten

- Definitionskomponenten: Datendefinition, Dateiorganisation, Sichtdefinition
- Programmierkomponenten: DB-Programmierung mit eingebetteten DB-Operationen
- Benutzerkomponenten: Anwendungsprogramme, Anfrage und Update interaktiv
- Transformationskomponenten: Optimierer, Auswertung, Plattenzugriffssteuerung
- Data Dictionary (Datenwörterbuch): Aufnahme der Daten aus Definitionskomponenten, Versorgung der anderen Komponenten

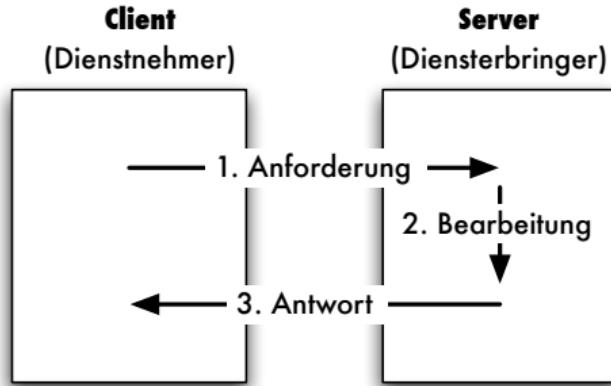
Fünf-Schichten-Architektur

- Verfeinerung der Transformationsschritte



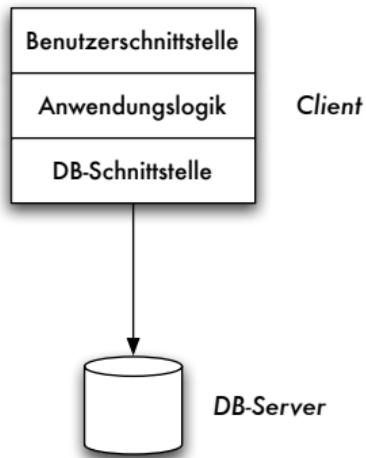
Anwendungsarchitekturen

- Architektur von Datenbankanwendungen typischerweise auf Basis des Client-Server-Modells: Server \equiv Datenbanksystem

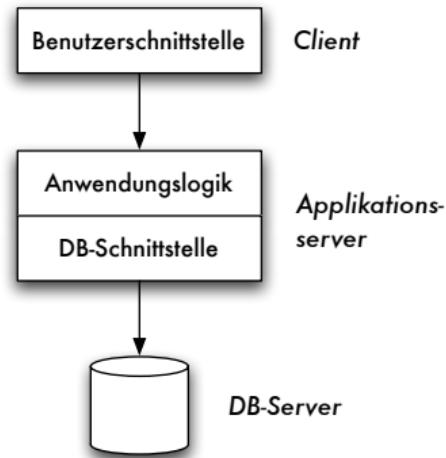


Anwendungsarchitekturen /2

- Aufteilung der Funktionalitäten einer Anwendung
 - ▶ Präsentation und Benutzerinteraktion
 - ▶ Anwendungslogik („Business“-Logik)
 - ▶ Datenmanagementfunktionen (Speichern, Anfragen, ...).



Zwei-Schichten-Architektur



Drei-Schichten-Architektur

Einige konkrete Systeme

- (Objekt-)Relationale DBMS
 - ▶ Oracle12c Release 2, IBM DB2 V12, Microsoft SQL Server 2017, SAP HANA 2.0 SPS03
 - ▶ MySQL (<https://dev.mysql.com/>), PostgreSQL (www.postgresql.org), Ingres (www.ingres.com,
<http://www.actian.com/products/operational-databases/ingres/>)
- Pseudo-DBMS
 - ▶ Microsoft Access
- Objektorientierte DBMS
 - ▶ Versant Object Database/Versant FastObjects/db4, VelocityDB
- XML-DBMS
 - ▶ eXist, BaseX, MonetDB/XQuery
- NoSQL Datenbanken
 - ▶ MongoDB, Solr, Elasticsearch, Cassandra, HBase, ArangoDB, ...

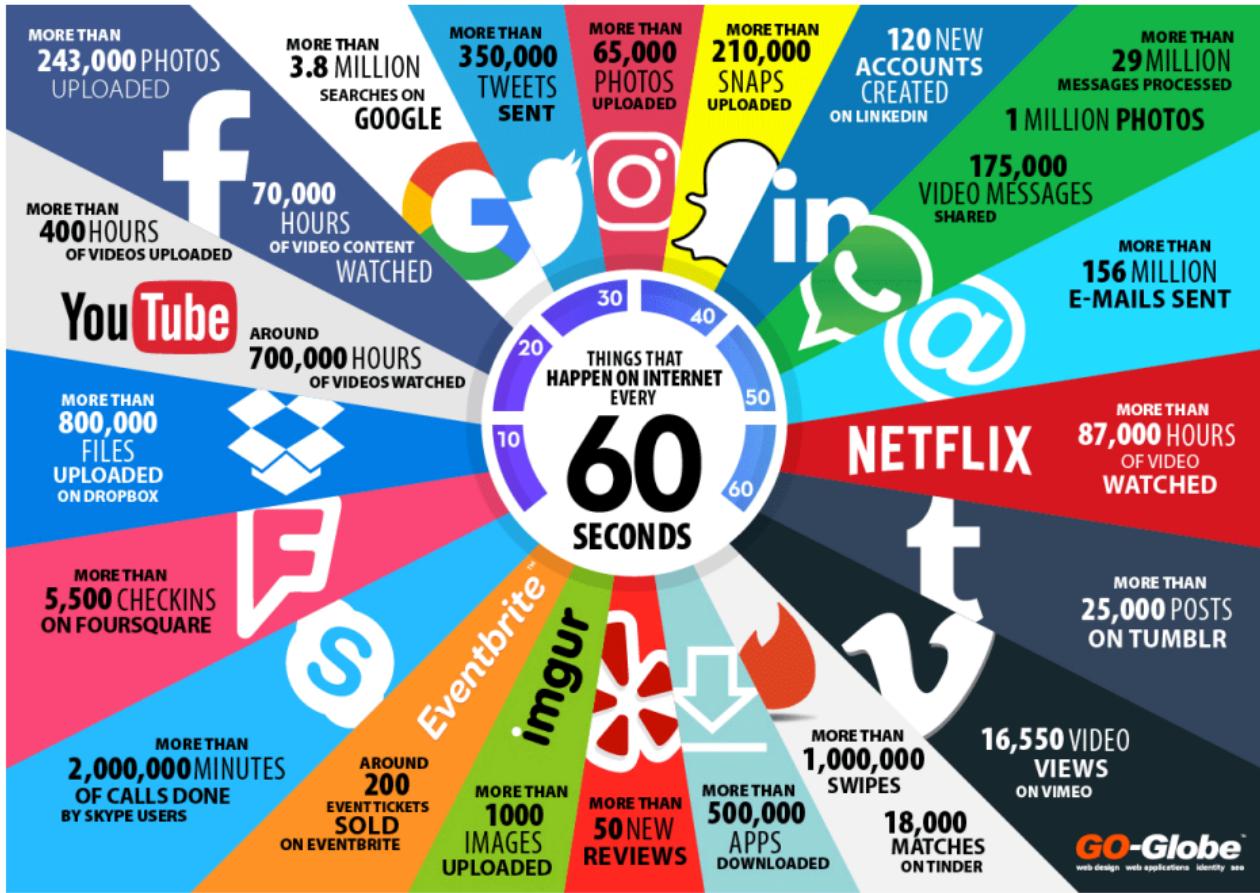
Einsatzgebiete

- Klassische Einsatzgebiete:

- ▶ viele Objekte (15.000.000 Bücher, 80.000 Benutzer, 4500 Ausleihvorgänge pro Tag, ...)
- ▶ wenige Objekttypen (BUCH, BENUTZER, AUSLEIHE)
- ▶ etwa Buchhaltungssysteme, Auftragserfassungssysteme, Bibliothekssysteme, ...

- Aktuelle Anwendungen (u.a.):

- ▶ E-Commerce, entscheidungsunterstützende Systeme (Data Warehouses, OLAP),
Banken und Versicherungen, Transportwesen, Enterprise-Resource-Planning (ERP), ...
- ▶ und darauf aufsetzend: Data Mining, Machine Learning, Visualisierung, ...



Source: <https://www.go-globe.com/blog/things-that-happen-every-60-seconds/>

Datenbankgrößen

eBay Data Warehouse 90 PB (= $9 \cdot 10^{16}$ Byte)

Teradata, commodity Hadoop clusters, Billiarden Datensätze,
120 TB/Tag an neuen Daten

WalMart Data Warehouse 2,8 PB

NCR TeraData;
Produktinfos (Verkäufe etc.) von >3.000 Märkten;
70.000 Anfragen/Woche

Facebook 1200 PB

Hadoop/Hive, 700 TB/Tag

US Library of Congress 10-20 TB

(noch) nicht alles digitalisiert

- PB für Petabyte entspricht der Größenordnung 10^{15}

Entwicklungslienien: 60er Jahre

- Anfang 60er Jahre: elementare Dateien, anwendungsspezifische Datenorganisation (geräteabhängig, redundant, inkonsistent)
- Ende 60er Jahre: Dateiverwaltungssysteme (SAM, ISAM) mit Dienstprogrammen (Sortieren) (geräteunabhängig, aber redundant und inkonsistent)
- DBS basierend auf **hierarchischem Modell, Netzwerkmodell**
 - ▶ Zeigerstrukturen zwischen Daten
 - ▶ Schwache Trennung interne / konzeptuelle Ebene
 - ▶ Navigierende DML
 - ▶ Trennung DML / Programmiersprache

Entwicklungslienien: 70er und 80er Jahre

- 70er Jahre: Datenbanksysteme (Geräte- und Datenunabhängigkeit, redundanzfrei, konsistent)
- **Relationale Datenbanksysteme**
 - ▶ Daten in Tabellenstrukturen
 - ▶ 3-Ebenen-Konzept
 - ▶ Deklarative DML
 - ▶ Trennung DML / Programmiersprache

Historie von RDBMS

- 1970: Ted Codd (IBM) → Relationenmodell als konzeptionelle Grundlage relationaler DBS
- 1974: System R (IBM) → erster Prototyp eines RDBMS
 - ▶ zwei Module: RDS, RSS; ca. 80.000 LOC (PL/1, PL/S, Assembler), ca. 1,2 MB Codegröße
 - ▶ Anfragesprache SEQUEL
 - ▶ erste Installation 1977
- 1975: University of California at Berkeley (UCB) → Ingres
 - ▶ Anfragesprache QUEL
 - ▶ Vorgänger von Postgres, Sybase, ...
- 1979: Oracle Version 2

Entwicklungslienien: (80er und) 90er Jahre

- **Wissensbanksysteme**

- ▶ Daten in Tabellenstrukturen
- ▶ Stark deklarative DML, integrierte Datenbankprogrammiersprache

- **Objektorientierte Datenbanksysteme**

- ▶ Daten in komplexeren Objektstrukturen (Trennung Objekt und seine Daten)
- ▶ Deklarative oder navigierende DML
- ▶ Oft integrierte Datenbankprogrammiersprache
- ▶ Oft keine vollständige Ebenentrennung

Entwicklungslienien: heute

- Unterstützung für spezielle Anwendungen
 - ▶ Hochskalierbare, parallele Datenbanksysteme: Umgang mit Datenmengen im PB-Bereich, “Big Data”
 - ▶ Datenstromverarbeitung: Online-Verarbeitung von Live-Daten (Börseninfos, Sensordaten, RFID-Daten, ...)
 - ▶ Hauptspeicherdatenbanken und Column Stores
 - ▶ XML/JSON-Datenbanken: Verwaltung semistrukturierter Daten (XML-Dokumente, JSON-Objekte)
 - ▶ Multimediadatenbanken: Verwaltung multimedialer Objekte (Bilder, Audio, Video)
 - ▶ Verteilte (Cloud) Datenbanken: Verteilung von Daten auf verschiedene Rechnerknoten
 - ▶ Mobile Datenbanken: Datenverwaltung auf Kleinstgeräten (PDA, Handy, ...)

Trends

- Nutzergenerierte Inhalte, z.B. Google:
 - ▶ Verarbeitung von 25 PB täglich (2+ Mio. Anfragen/Min.)
 - ▶ 400h Video-Upload auf YouTube in jeder Minute
 - ▶ Lesen von 20 PB würde bei 100 MB/s-Festplatte etwa 7 Jahre benötigen
- Linked Data und Data Web
 - ▶ Bereitstellung, Austausch und Verknüpfung von strukturierten Daten im Web
 - ▶ ermöglicht Abfrage (mit Anfragesprachen wie SPARQL) und Weiterverarbeitung
 - ▶ Beispiele: DBpedia, GeoNames

Zusammenfassung

- Motivation für Einsatz von Datenbanksystemen
- Codd'sche Regeln
- 3-Ebenen-Schemaarchitektur & Datenunabhängigkeit
- Einsatzgebiete