Datenbanken

Benno Stein

Raum:	110b, B11
Tel:	3795
URL:	http://www.uni-weimar.de/medien/webis
E-Mail:	benno.stein@medien.uni-weimar.de
Sprechstunde:	nach Vereinbarung

Inhalt

Übungen

Organisation:

- Übungsaufgaben werden im Netz zur Verfügung gestellt
- fristgerechte Lösung bestimmter Aufgaben berechtigt zur Klausurteilnahme
- Teams von 2 bis 3 Studenten/innen möglich
- Übungen teilweise als Präsenzübungen
- Beginn der Übungen: Anfang November

Aufgaben:

- theoretisch
- praktisch

Tutoren:

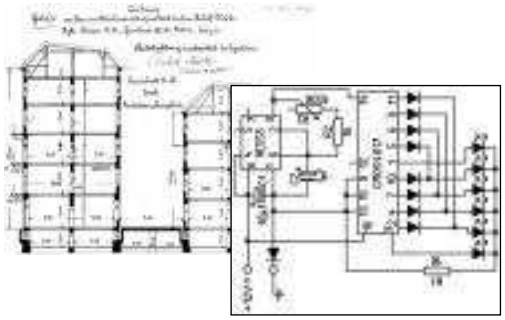
- Tsvetomira Palakarska tsvetomira.palakarska@medien.uni-weimar.de

Ziele

- Grundbegriffe von Datenbanken kennen und einordnen
- charakteristische Eigenschaften von Datenbanken kennen
- Techniken zur Modellierung anwenden
- relationale Datenbanken und die Sprache SQL verwenden
- Umgang mit formalen Methoden üben
- sich selbst weiterbilden können

Angrenzende Gebiete

Ingenieur-Datenbanken



Multimedia-Datenbanken

Data Warehouses

Geo-Informationssysteme

deduktive Datenbanken,
Wissensbanken

"very large data bases"

Google

Angrenzende Gebiete

1. Syntaktische Web-Technologien [Modellierung]
   - Dokumentsprachen: HTML, XML, DTD, XML-Schema
   - Style-Sprachen: CSS, XSL, XSLT
   - APIs und Retrieval: DOM, SAX, JAXB, XPath, XQuery
   - Client-Technologien: JavaScript, Java-Applets
   - Server-Technologien: Java Servlets, JSP, PHP, Perl, Python
   - Middleware: SOAP, WSDL, UDDI, WSFL
2. Semantische Web-Technologien
   - RDF, RDF-Schema, OWL, Ontologien, Description Logics
3. Wissensverarbeitung [Algorithmen]
   - Automatisierung des Schlußfolgerungsprozesses
   - Regel- und Constraint-Verarbeitung
4. Information Retrieval und Data Mining
   - Retrieval-Algorithmen für Texte und Bilder
   - Suche nach Mustern und Zusammenhängen
5. CMS / Wissens- und Dokumenten-Management [Anwendungen]
6. Data Warehouses, eCommerce

Literatur

- Ramez Elmasri, Shamkant B. Navathe.
  Fundamentals of Database Systems
  4th edition, Pearson Addison Wesley, 2003, ISBN 0321122267.
- Abraham Silberschatz, Henry Korth, S. Sudarshan.
  Database System Concepts
  3rd edition, McGraw-Hill, 1997, ISBN 0072958863.
- Jeffrey D. Ullman, Jennifer Widom.
  A First Course in Database Systems
  2nd edition, Prentice Hall, 2001, ISBN 0130353000.

- Alfons Kemper, Andre Eickler.
  Datenbanksysteme - Eine Einführung
  5. Auflage, 2004, ISBN 3-486-27392-2.
- Andreas Heuer, Gunter Saake.
  Datenbanken: Konzepte und Sprachen
  2. Auflage, International Thomson Publishing, 2000, ISBN 3-8266-0619-1.
- Gottfried Vossen.
  Datenmodell, Datenbanksprachen und Datenbankmanagement-Systeme
  4. Auflage, Oldenbourg-Verlag, 2000, ISBN 3-486-25339-5.

Kapitel DB: I

I. Einführung und grundlegende Konzepte von Datenbanken
   - ❑ Datenintensive Anwendungen
   - ❑ Begriffsbildung, Definitionen
   - ❑ Datenbank-Management-Systeme
   - ❑ Relationale Datenbanksysteme

II. Datenbankentwurf und Datenbankmodelle

III. Konzeptueller Datenbankentwurf

IV. Logischer Datenbankentwurf mit dem relationalen Modell

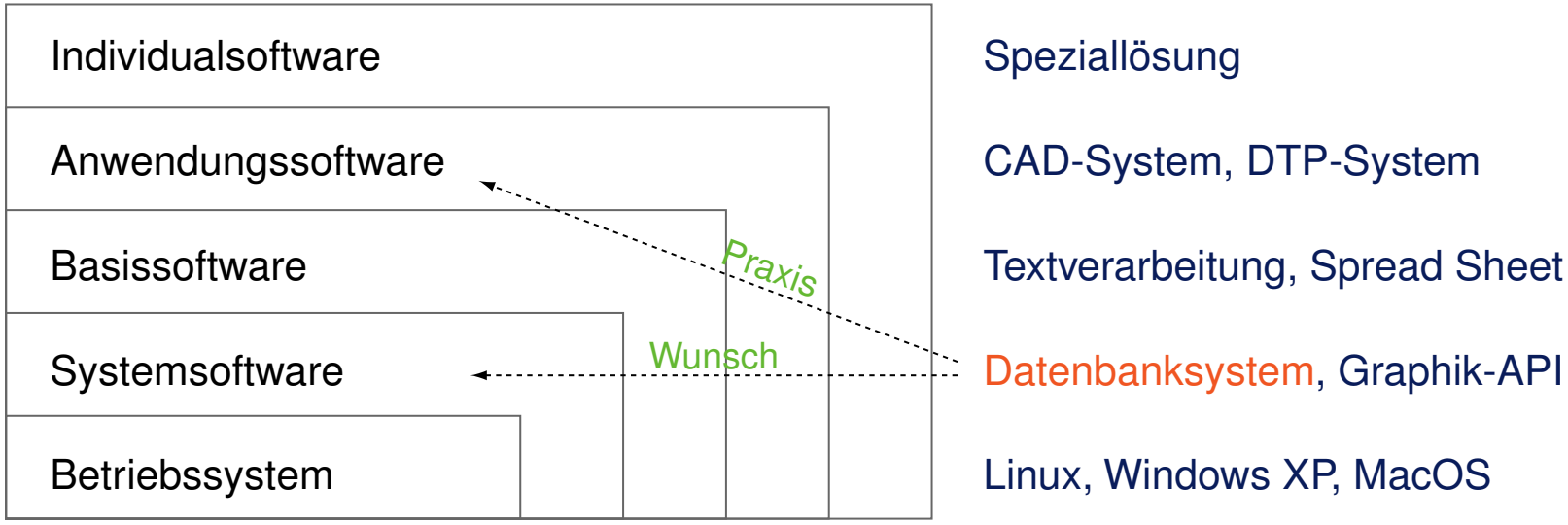
V. Grundlagen relationaler Anfragesprachen

VI. SQL

VII. Entwurfstheorie relationaler Datenbanken

Datenintensive Anwendungen

Ein Blick auf die Software eines Computer-Systems.



Datenintensive Anwendungen

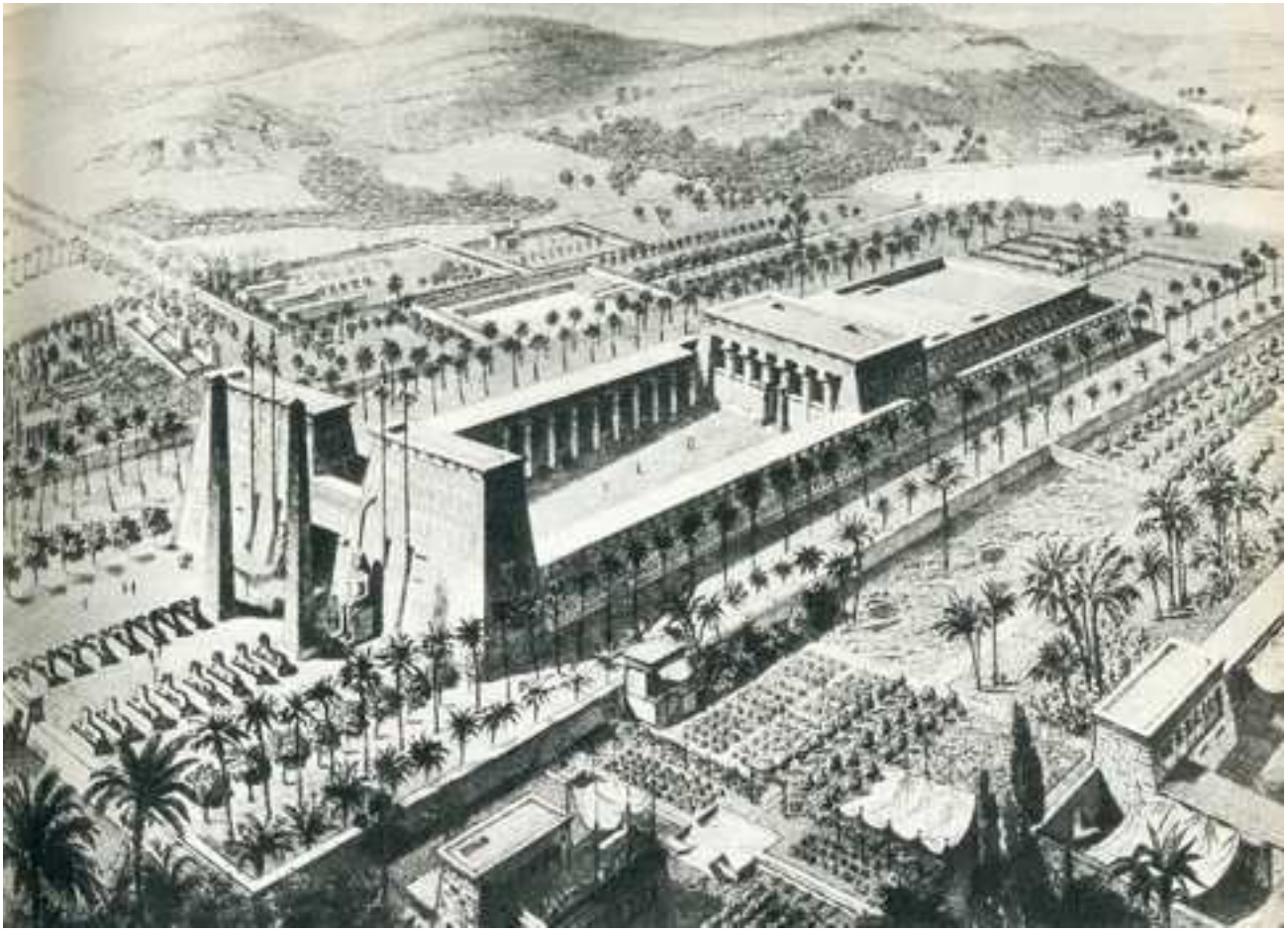
- Textverarbeitung: Textbausteine, Adressen
- Content-Management-System: Text-Dokumente, Bilder, sonstige „Assets“
- Buchhaltung: Personaldaten, Lohndaten, Kostenstellen
- Auftragsverwaltung: Aufträge, Kundenadressen, Artikelnummern
- Lagerverwaltung: Bestellungen, Teilenummern, Lagerbestand
- Konfigurationssystem: Komponenten, Baupläne
- CAD-System: Zeichnungen, Makro-Bibliotheken, technische Daten
- PPS-System: Maschinenbelegungspläne, Produktionspläne, Teilenummern
- Studentenverwaltung: Prüfungsdaten, Matrikelnummern, Belegungsdaten
- Online-Shop: Produktbeschreibungen, Kundendaten
- . . .

Bemerkungen:

- Jede Softwareschicht im Computer baut auf den weiter innenliegenden Schichten auf.
- Idealerweise sollte Basissoftware, Anwendungssoftware oder Indivudalsoftware ihre Datenverwaltung über ein standardisiertes Application-Programming-Interface (API) auf der Systemsoftwareebene realisieren. Stichwort: Datenbanksystem

Datenintensive Anwendungen

Entstehung von Datenbanksystemen



Bemerkungen:

- Die alte Bibliothek von Alexandrien wurde von Ptolemeus I im Jahre 288 v. Chr. gegründet. Sie war als Treffpunkt für die Weisen und großen Geister der Zeit gedacht. Gelehrte, Intellektuelle, Wissenschaftler und Schüler fanden hier ein Umfeld, um über das damalige Wissen zu diskutieren und zu lernen.
- Aristarchus war einer der ersten, der behauptete, dass sich die Erde um die Sonne dreht. Hipparchus war der erste, der das Solarjahr mit einer Abweichung von $6\frac{1}{2}$ Minuten messen konnte. Eratosthenes war der erste, der den Durchmesser der Erde berechnete. Euclid, der Vater der Geometrie, war ebenfalls hier tätig. Archimedes, der größte Mathematiker der alten Welt, lehrte hier. Der Gründer des Bibliothekswesens, Callimachus, erfand hier ein Sortierungssystem: einen Katalog, um Schriftrollen nach Thema oder Autor zu finden.
- Die Bibliothek war offen für jedermann und die besten Werke der Zeit wurden hier gesammelt. Das Alte Testament wurde erstmalig aus dem Hebräischen ins Griechische übersetzt. Es gab eine Mischung von diversen Kulturen und Sprachen unter den Gelehrten, aber die griechische Sprache setzte sich durch, da für die griechischen Philosophen Alexandrien mit ihrer Bibliothek die intellektuelle Hauptstadt der Zeit war.
- Die alte Bibliothek (es waren eigentlich zwei) wurde durch die Römer 49 v. Chr. und ein Feuer zerstört. Hier war das gesamte Wissen der Antike vorhanden; es soll an die 700 000 Papyrusschriftrollen gegeben haben.

[www.bibliothek-alexandria.de/sites/altebibliothek.html]

Datenintensive Anwendungen

Entstehung von Datenbanksystemen

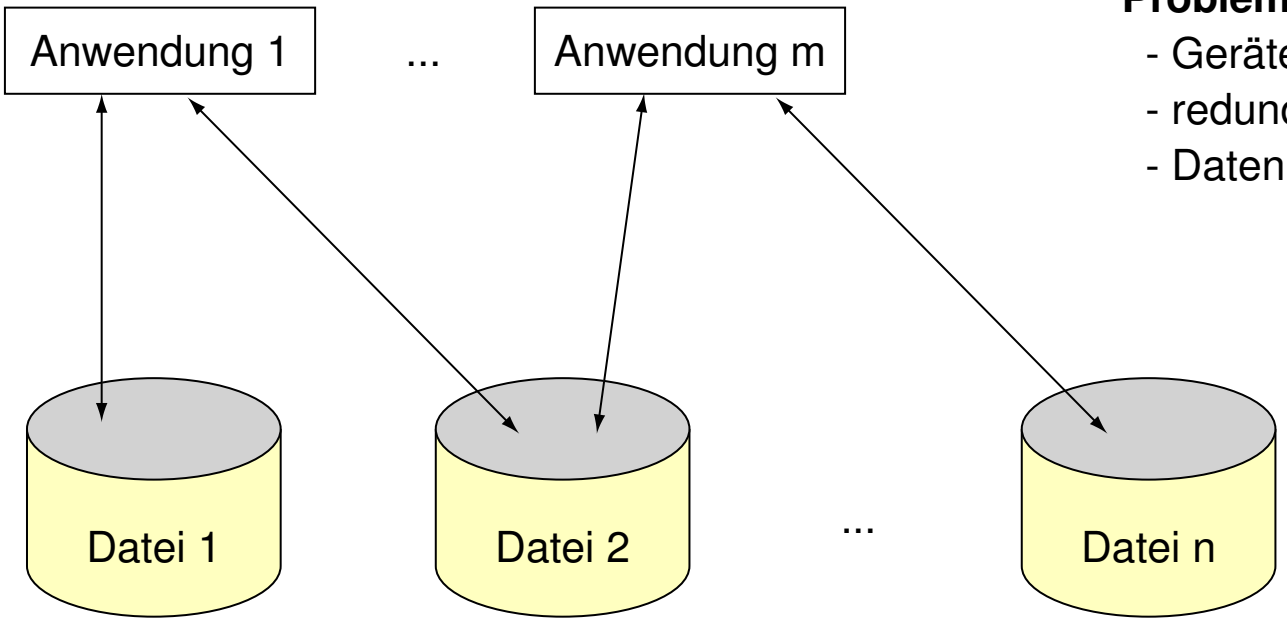
Bemerkungen:

- Die UNESCO setzte sich für den Plan ein, eine neue Bibliothek aufzubauen. Dem Appell der UNESCO von 1987 folgte eine Einigung im Jahre 1989 mit der ägyptischen Regierung. Im Jahre 1990 bei der Assuan Konferenz beteiligte sich eine Ehrenkommision, bei der arabische Länder und Privatpersonen 65 Millionen Dollar beisteuerten. 1995 war der Baubeginn und der Komplex wurde im Jahre 2001 fertig. Die Gesamtkosten betrugen ca. 250 Millionen Dollar.
- Die Gesamtfläche der Bibliothek beträgt 45 000 Quadratmeter. Ihre Kapazität ist auf 8 Millionen Bücher ausgelegt; im Jahr 2004 waren 200 000 Bücher vorhanden.

[www.bibliothek-alexandria.de/sites/neuebibliothek.html]

Datenintensive Anwendungen

Entstehung von Datenbanksystemen



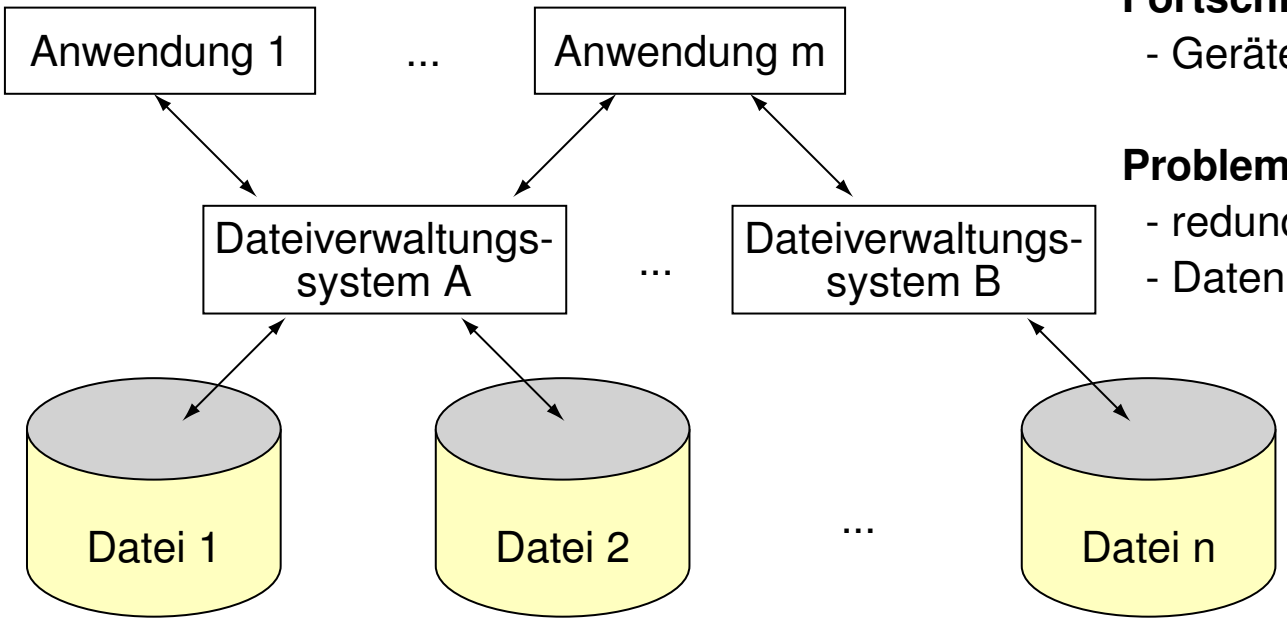
Probleme:

- Geräteabhängigkeit
- redundante Speicherung
- Dateninkonsistenz

Zugriff auf Dateien ohne spezielle Verwaltung, Anfang der 60er Jahre.

Datenintensive Anwendungen

Entstehung von Datenbanksystemen



Fortschritt:
- Geräteunabhängigkeit

Probleme:
- redundante Speicherung
- Dateninkonsistenz

Datenverwaltungssoftware für Dateien, Ende der 60er Jahre.

Datenintensive Anwendungen

Probleme der frühen Datenverarbeitung

- Redundanz
  Wiederholtes Speichern gleicher Daten aus verschiedenen Anwendungsprogrammen in verschiedenen Dateien; Verschwendung von Speicher; erhöhter Verwaltungs- und Verarbeitungsaufwand.

- Inkonsistenz
  Fehlende logische Übereinstimmung der Dateiinhalte – insbesondere bedingt durch Änderungen.

- Datenverteilung
  Daten werden an verschiedenen Orten benötigt; gezieltes Anlegen und Pflegen von Kopien wird nicht unterstützt.

- Daten-Programm-Abhängigkeit
  Direktes Erzeugen und Ansprechen der Daten durch Anwendungsprogramm; Änderungen am Dateiaufbau bedingen Änderungen des Anwendungsprogramms; Erweiterungen der Funktionalität des Anwendungsprogramms bedingen neue Anforderungen an den Dateiaufbau und eine Restrukturierung von Dateien; Anwendungsprogrammierer oder Benutzer kennen und benutzen internen Aufbau der gespeicherten Daten.

Datenintensive Anwendungen

Probleme der frühen Datenverarbeitung (Fortsetzung)

- Inflexibilität
  Auswertung der Daten sowie Realisierung neuer Anwendungen ist problematisch; Daten aus mehreren Dateien nur mit hohem Aufwand kombinierbar.

- große Datenmengen
  Oft können Softwaresysteme (Dateiverwaltung, Tabellenkalkulation, ...) große Datenmengen nicht effizient verwalten.

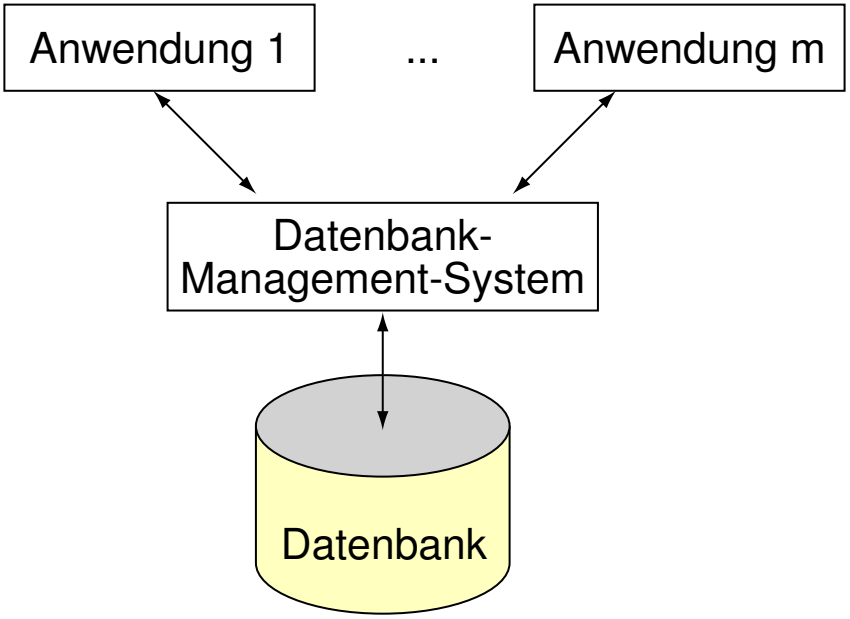
- paralleler Zugriff
  Mehrere Benutzer oder Anwendungen können nicht ohne gegenseitige Beeinflussung gleichzeitig auf denselben Daten arbeiten.

- Datenschutz
  Mechanismen zum Schutz der Daten von Benutzergruppen vor unberechtigten Zugriffen fehlen.

- Datensicherheit
  Mit dem Absturz von Anwendungen (auch Stromausfall, Platten-Crash) können Daten verloren gehen.

Datenintensive Anwendungen

Entstehung von Datenbanksystemen



Fortschritt:
- Datenintegration
  (statt Redundanz)
- einheitliche Operatoren
- Konsistenz

Herausforderungen:
- Transaktionsabwicklung
- Effizienz
- Datenschutz

  ...

Datenbank-Management-Systeme, Mitte der 70er Jahre.

Begriffsbildung, Definitionen

Definition 1 (Datenbank = Datenbasis, DB)

Eine Datenbank ist eine strukturierte Sammlung einer umfangreichen Menge persistenter (= dauerhaft zur Verfügung stehender) Daten in elektronischer Form. Diese Daten sind nach bestimmten Merkmalen und Regeln erfasst, geordnet und abgelegt. Der Zugriff auf die Daten und deren Änderung ist ohne großen Aufwand möglich.

Begriffsbildung, Definitionen

Definition 1 (Datenbank = Datenbasis, DB)
Eine Datenbank ist eine strukturierte Sammlung einer umfangreichen Menge persistenter (= dauerhaft zur Verfügung stehender) Daten in elektronischer Form. Diese Daten sind nach bestimmten Merkmalen und Regeln erfasst, geordnet und abgelegt. Der Zugriff auf die Daten und deren Änderung ist ohne großen Aufwand möglich.

Definition 2 (Datenbank-Management-System, DBMS)
Gesamtheit der Programme zum Zugriff auf die Datenbasis, zur Datenmodifikation und zur Gewährleistung der Konsistenz. Das Datenbank-Management-System bildet eine Softwareschicht zwischen dem Benutzer und der physischen Darstellung der Daten.

Definition 3 (Datenbanksystem, DBS)
Datenbanksystem = DB + DBMS

Datenbank-Management-Systeme, DBMS

Datenbank-Management-Systeme, DBMS

Neun Anforderungen an ein DBMS [Edgar Frank Codd 1982]

1. Integration

   Einheitliche (➜ nicht-redundante) Verwaltung der Daten aller Anwendungen.

2. Operationen

   Operationen zum Speichern, Suchen und Ändern des Datenbankinhaltes existieren.

3. Katalog bzw. Data-Dictionary

   Der Katalog ermöglicht den Zugriff auf die Datenbankbeschreibung.

4. Benutzersichten

   Realisierbarkeit unterschiedlicher Sichten auf den Datenbankinhalt.

5. Konsistenzüberwachung bzw. Integritätssicherung

   Gewährleistung der Korrektheit des Datenbankinhaltes.

6. Zugriffskontrolle bzw. Datenschutz

   Verhinderung unautorisierter Zugriffe auf den Datenbankinhalt.

Datenbank-Management-Systeme, DBMS

Neun Anforderungen an ein DBMS

7. Transaktionen

   Zusammenfassung mehrerer Operationen zu einer Funktionseinheit, die unteilbar ausgeführt wird.

8. Synchronisation

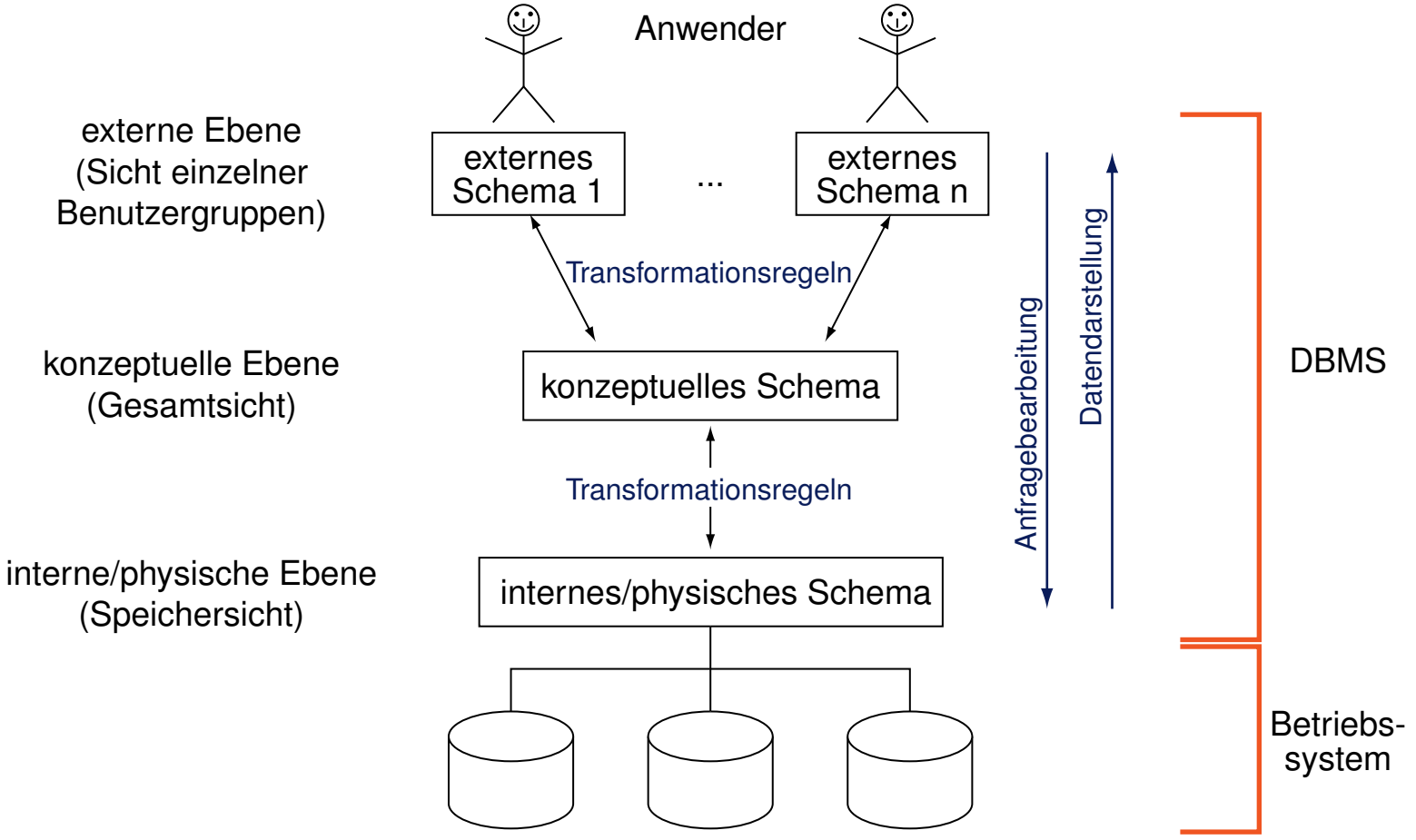
   Koordination konkurrierender Transaktionen mehrerer Benutzer.

9. Datensicherung

   Wiederherstellung von Daten nach Systemfehlern.

Datenbank-Management-Systeme

Drei-Schichten-Schema-Architektur [ANSI/SPARC 1975]



Bemerkungen:

- ANSI = American National Standards Institute
- SPARC = Standards Planning and Requirements Committee
- konzeptuell im Sinne von „Paradigmen-unabhängig“

Datenbank-Management-Systeme

Drei-Schichten-Schema-Architektur

Die Schema-Architektur nach ANSI/SPARC garantiert Datenunabhängigkeit.

Datenunabhängigkeit bezeichnet die Eigenschaft, dass höhere Ebenen des Modells unbeeinflusst von Änderungen auf niedrigeren Ebenen bleiben.

- logische Datenunabhängigkeit (Anwendungsunabhängigkeit)

- physische Datenunabhängigkeit

Datenbank-Management-Systeme

Drei-Schichten-Schema-Architektur

Die Schema-Architektur nach ANSI/SPARC garantiert Datenunabhängigkeit.

Datenunabhängigkeit bezeichnet die Eigenschaft, dass höhere Ebenen des Modells unbeeinflusst von Änderungen auf niedrigeren Ebenen bleiben.

- logische Datenunabhängigkeit (Anwendungsunabhängigkeit)

  Änderungen des konzeptuellen Schemas (z. B. Informationen über neue Typen von Objekten, weitere Eigenschaften für existierende Objekte) haben keine Auswirkungen auf externe Schemata (z. B. existierende Anwendungsprogramme).

- physische Datenunabhängigkeit

  Änderungen des physischen Schemas (z. B. Wechsel von einer Zugriffsstruktur zu einer effizienteren, Benutzung anderer Datenstrukturen, Austausch von Algorithmen) lassen die konzeptuelle Ebene unverändert und haben somit auch keine Auswirkungen auf externe Schemata.

Datenbank-Management-Systeme

Was ein DBMS zu tun hat

Beispielanfrage mittels SQL an eine relationale DB:

```
select  Titel
from    Buch
where   Autor = Pearl
```

Buch			
Inv_Nr	Titel	ISBN	Autor
0110	Lesebuch	2-341...	Popper
1201	C++	2-123...	Stroustrup
3309	Längengrad	2-123...	Sobel
4711	Glücksformel	2-679...	Klein
7510	Heuristics	9-212...	Pearl

Datenbank-Management-Systeme

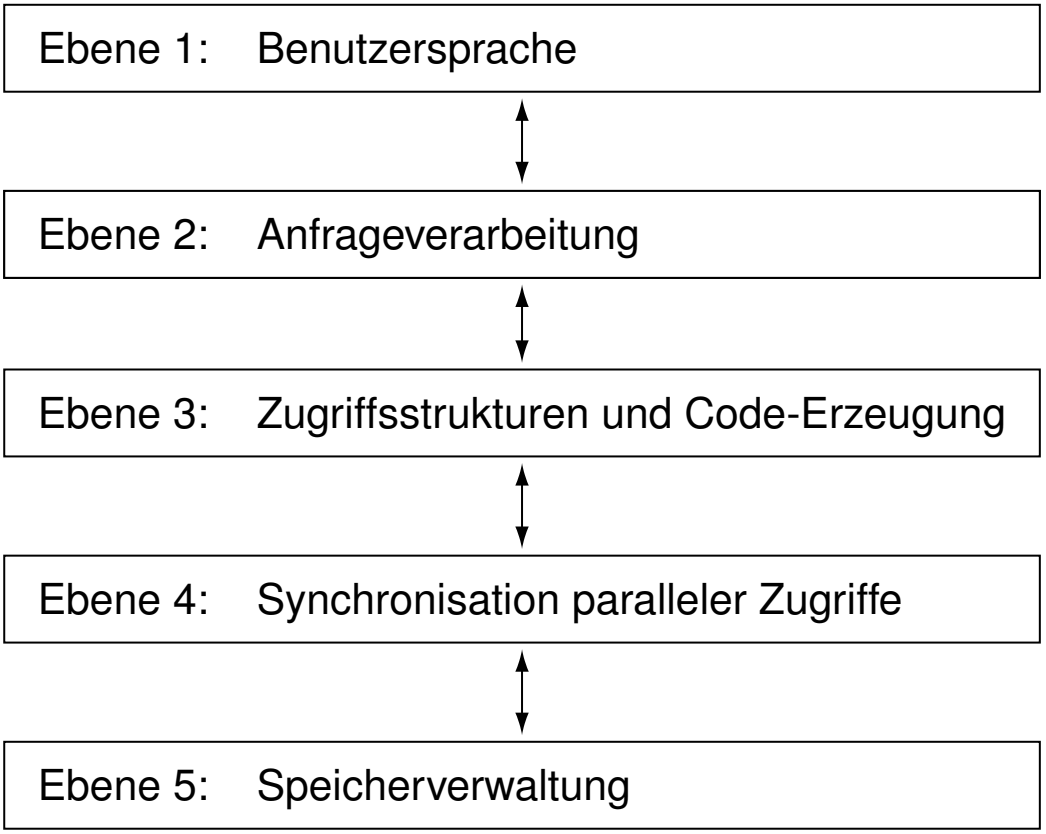
Was ein DBMS zu tun hat (Fortsetzung)

1. Überprüfen der Syntax.
2. Feststellen, ob die entsprechende Relation in der Datenbank definiert ist und ob der fragende Benutzer deren Information lesen darf.
3. Feststellen, welche Operationen zur Beantwortung der Anfrage intern auszuführen sind und wie der Anfrageoperand, die Relation Buch, gespeichert ist.
4. Erstellen eines (effizienten) Programms zur Berechnung der Antwort.
5. Holen des Operanden aus der Datenbank.
6. Aufbereiten des Operanden zum Zweck der Ausgabe.
7. Sicherstellen, dass der Operand während der Ausführung dieses Programms nicht durch eine andere Operation verändert wird.

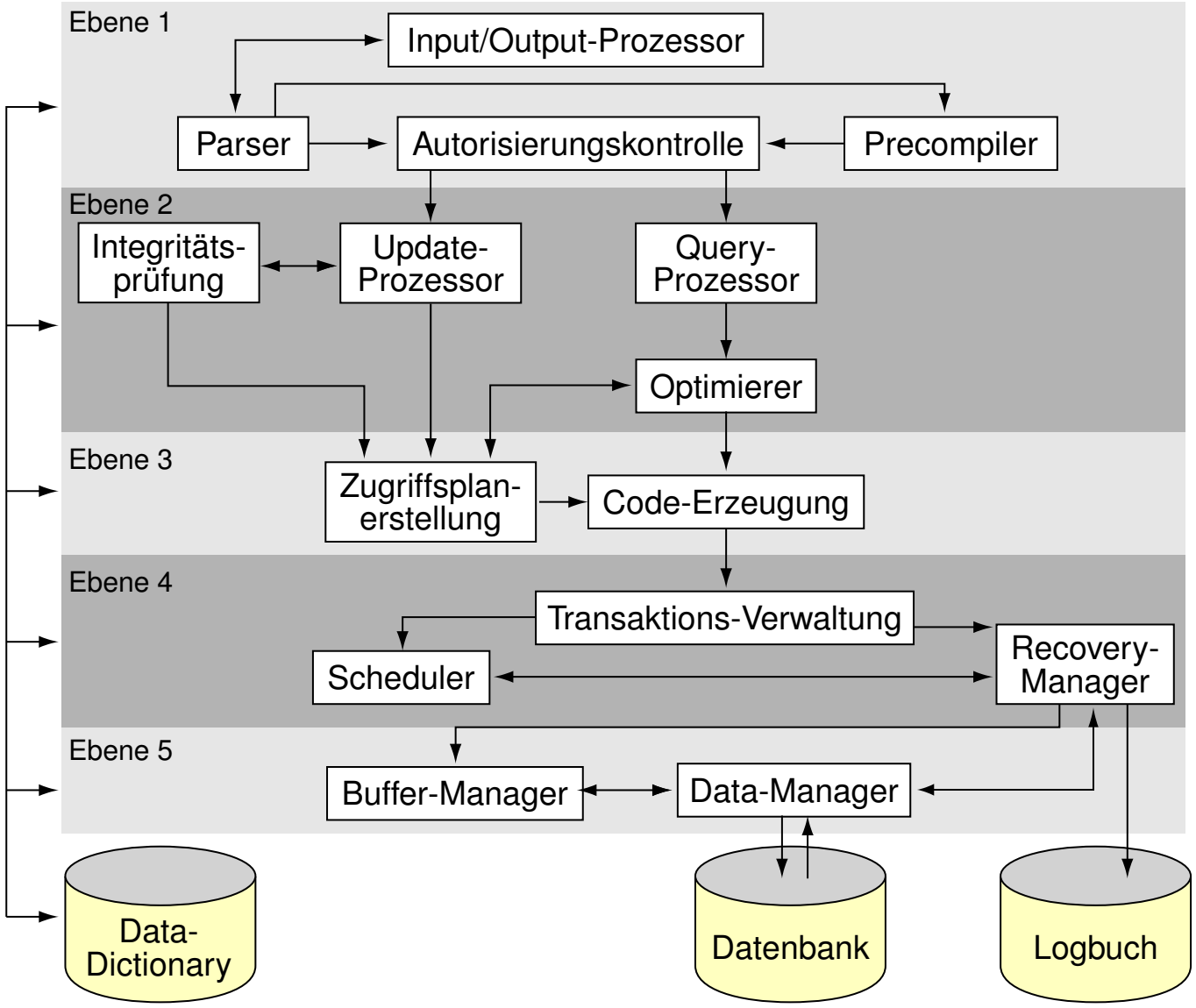
Datenbank-Management-Systeme

Systemarchitektur und Komponenten eines DBMS

➜ Unterscheidung von fünf Verarbeitungsschichten in einem DBMS:



Datenbank-Management-Systeme



[Vossen 2000]

Bemerkungen:

- Das Data-Dictionary enthält die drei Schemata der Ebenen der ANSI/SPARC-Architektur.
- Das Logbuch dient zum Zweck des Wiederanlaufs nach Systemfehlern.
- Der Parser macht eine syntaktische Analyse der Kommandos.
- Der Precompiler ist u. a. für die Verarbeitung eingebetteter Kommandos zuständig.
- Update-Operationen erfordern eine Integritätsprüfung, um die semantische Korrektheit der Datenbank zu gewährleisten.
- Der Optimierer untersucht die Formulierung von Anfragen dahingehend, ob sie sich in eine effizientere Form bringen lassen.
- Die Zugriffsplanerstellung enthält das Finden und die Auswahl möglichst effizienter Zugriffspfade auf die benötigten Daten (Stichwort: Index).
- Bei der Code-Erzeugung wird der bisher generierte Zwischen-Code in eine Folge von Lese- und Schreibbefehlen für den Sekundärspeicher übersetzt.
- Die Transaktionsverwaltung erledigt die Synchronisation von quasi parallel ablaufenden Transaktionen mehrerer Benutzer. Jede Transaktion wird entweder vollständig oder gar nicht ausgeführt.
- Eine Transaktion, die nicht erfolgreich zu Ende gebracht werden kann, wird dem Recovery-Manager übergeben. Er setzt die Datenbank in den Zustand zurück, in dem sie sich vor dem Start der Transaktion befand. ➜ Logbuch
- Buffer-Manager und Data-Manager realisieren die Speicherverwaltung des Systems. Der Buffer-Manager verwaltet im Hauptspeicher des Rechners den für jede Transaktion bereitgestellten Puffer; der Data-Manager verwaltet die dem DBMS zur Verfügung gestellten Hardware-Betriebsmittel.

Datenbank-Management-Systeme

Systemarchitektur und Komponenten eines DBMS (Fortsetzung)

Ein DBMS stellt bestimmte Sprachschnittstellen bereit:

- Datendefinitionssprache *(Data Definition Language)*, DDL
- Datenmanipulationssprache *(Data Manipulation Language)*, DML
- Datenverwaltungssprache *(Data Administration Language)*, DAL

Datenbank-Management-Systeme

Systemarchitektur und Komponenten eines DBMS (Fortsetzung)

Ein DBMS stellt bestimmte Sprachschnittstellen bereit:

- Datendefinitionssprache *(Data Definition Language)*, DDL

  Dient dem Benutzer, der die Schemata der Datenbank definieren möchte.
  Manchmal auch Unterscheidung einer Subschema-DDL (externe Ebene), einer Schema-DDL (konzeptuelle Ebene) und einer Data-Storage-Definition-Language, DSDL, (interne Ebene).

- Datenmanipulationssprache *(Data Manipulation Language)*, DML

  Sprache, die zum Arbeiten mit einer Datenbank zur Verfügung steht.
  Die „eigentliche“ Datenmanipulationssprache zur Änderung von gespeicherten Daten, zum Einfügen von neuen Daten und zum Löschen von gespeicherten Daten.
  Im Allgemeinen realisiert als nicht-prozedurale Sprache: Der Benutzer spezifiziert, *was* für Daten gesucht werden, aber nicht *wie* die Daten aufgefunden werden sollen.

- Datenverwaltungssprache *(Data Administration Language)*, DAL

  Sprache zur Manipulation der internen Ebene.

Datenbank-Management-Systeme

Historie

- 60er Jahre

  Datenbanksysteme mit hierarchischem oder Netzwerkmodell:
  - Zeigerstrukturen zwischen Daten
  - schwache Trennung zwischen `interner und konzeptueller` Ebene
  - navigierende Anfragesprachen entlang Zeigerstrukturen
  - Datenmanipulationssprache von der Programmiersprache getrennt (problematisch bei der Kopplung beider Sprachwelten)

- 70er Jahre

  relationale Datenbanksysteme:
  - Daten in Tabellenstruktur
  - Drei-Schichten-Schema-Architektur nach `ANSI/SPARC`
  - deklarative, standardisierte Datenbanksprache
  - Datenmanipulationssprache von Programmiersprache getrennt

Datenbank-Management-Systeme

Historie

- 80er Jahre

  Wissensbanksysteme bzw. deduktive Datenbanken:
  - Daten in Tabellenstruktur
  - logikbasierte, deklarative Anfragesprache
  - integrierte Datenbankprogrammiersprache

  objektorientierte Datenbanksysteme:
  - Daten in komplexen Objektstrukturen
  - navigierende oder deklarative Anfragesprache
  - objektorientierte Programmiersprache als Datenbankprogrammiersprache
  - keine vollständige Trennung der drei Ebenen

  geographische Datenbanksysteme:
  - spezielle Erweiterungen relationaler oder objektorientierter Systeme
  - neben „normalen“ Daten gibt es geometrische Daten (2D, 3D)
  - spezielle Indexstrukturen (QuadTrees, R-Trees, etc.) unterstützen räumliche Anfragen

Relationale Datenbanksysteme

Relationale Datenbanksysteme

Merkmale relationaler Datenbank-Management-Systeme, RDBMS

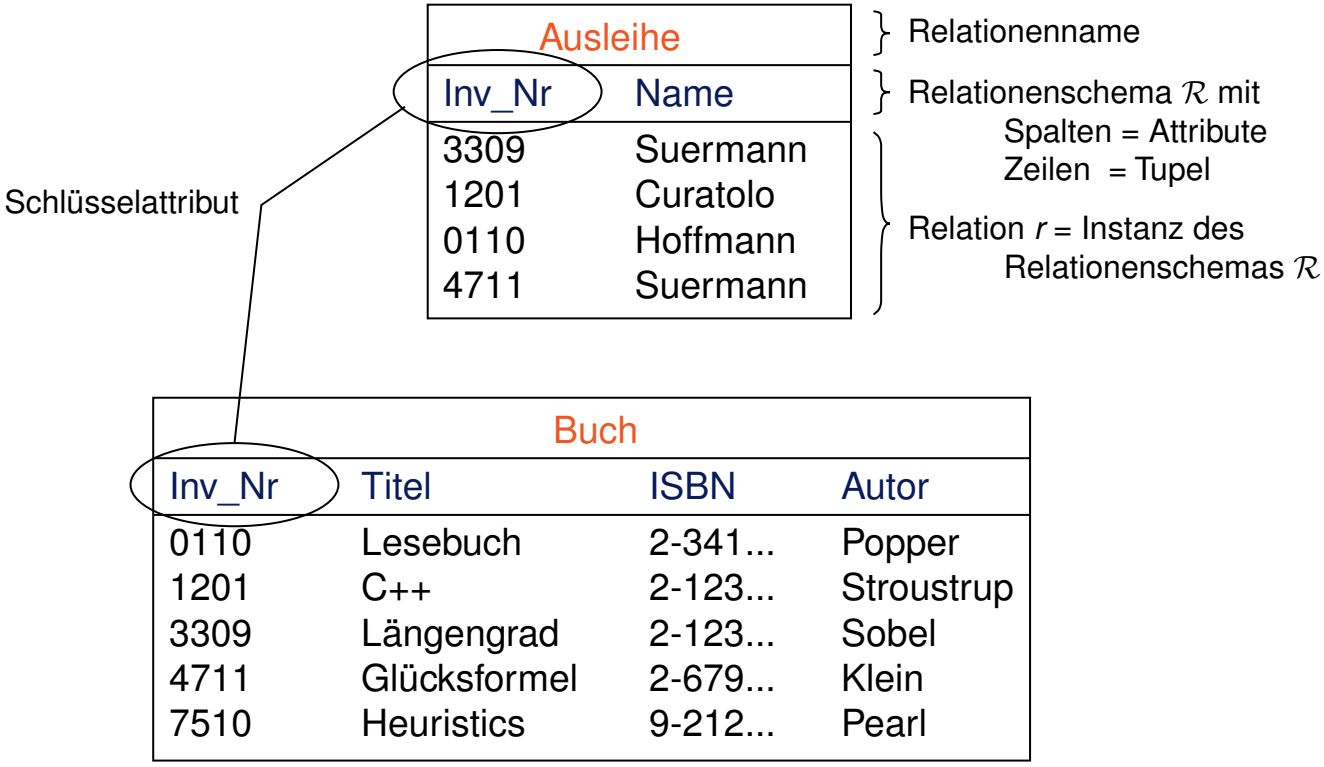
- Drei-Schichten-Schema-Architektur nach ANSI/SPARC
- standardisierte Datenbanksprache SQL *(Structured Query Language)*
- Einbettung von SQL in kommerzielle Programmiersprachen
- Werkzeuge:
  - für die interaktive Definition/Anfrage/Darstellung von Daten
  - für den Entwurf von Anwendungsprogrammen
  - für Benutzeroberflächen
- kontrollierter Mehrbenutzerbetrieb
- Gewährleistung von Datenschutz- und Datensicherheit

Bemerkungen:

- Beispiele „echter“ RDBMS: Sybase, ORACLE, DB2, Ingres, Informix
- Beispiele für „Pseudo“-RDBMS: MS-Access, dBASE. Hier fehlen u. a. eine explizite Abbildung der Drei-Schichten-Schema-Architektur und leistungsfähige Algorithmen zur Optimierung.

Relationale Datenbanksysteme

Beispiel: Daten- und Datenbankdefinition



- `Datenbank` = Menge von Relationen
- `Datenbankschema` = Menge von Relationenschemata

Relationale Datenbanksysteme

Beispiel: Integritätsbedingungen (Konsistenzregeln)

lokale Integritätsbedingungen:

- zu jedem Attribut in jedem Relationenschema gibt es eine Typdefinition, die die zulässigen Werte der entsprechenden Spalte festlegt (String, Integer).
- Attribut Inv_Nr ist (Primär)Schlüssel von Ausleihe:
  in Ausleihe existieren keine zwei Tupel mit demselben Inv_Nr-Wert.
- Attribut Inv_Nr ist (Primär)Schlüssel von Buch:
  in Buch existieren keine zwei Tupel mit demselben Inv_Nr-Wert.

globale Integritätsbedingung:

- Jeder Inv_Nr-Wert in der Relation Ausleihe muss in der Relation Buch auftreten und eindeutig sein. Stichwort: Fremdschlüsselbedingung

Relationale Datenbanksysteme

Beispiel: Anfrageoperationen

- Selektion = Auswahl von Zeilen bzw. Tupeln:

$\sigma_{\text{Name}='\text{Suermann}'}(\text{Ausleihe})$ ➜

Inv_Nr	Name
3309	Suermann
4711	Suermann

- Projektion = Auswahl von Spalten bzw. Attributen:

$\pi_{\text{Inv\_Nr,Titel}}(\text{Buch})$ ➜

Inv_Nr	Titel
0110	Lesebuch
1201	C++
3309	Längengrad
4711	Glücksformel
7510	Heuristics

Relationale Datenbanksysteme

Beispiel: Anfrageoperationen (Fortsetzung)

- Verbund *(Natural Join)* = Verknüpfung von Relationen über gleichbenannte Spalten mit gleichen Werten:

$$\pi_{\text{Inv\_Nr,Titel}}(\text{Buch}) \bowtie \sigma_{\text{Name}='\text{Suermann}'}(\text{Ausleihe})$$

➜

Inv_Nr	Titel	Name
3309	Längengrad	Suermann
4711	Glücksformel	Suermann

- Vereinigung, Durchschnitt und Differenz von gleich strukturierten Relationen

- Umbenennung von Spalten bzw. Attributen

Die genannten Operationen lassen sich beliebig kombinieren.
Stichwort: Relationenalgebra

Relationale Datenbanksysteme

Beispiel: Anfragen mittels SQL

Deklarativ-mengenorientierte Formulierung:

```
select  Buch.Inv_Nr, Titel, Name
from    Buch, Ausleihe
where   Name = Suermann
        and
        Buch.Inv_Nr = Ausleihe.Inv_Nr
```

Weiterhin können definiert werden:

- Relationen und Sichten
- Konsistenzbedingungen
- Update-Operationen
- Trigger

Relationale Datenbanksysteme

Anfrageoptimierung

Gegeben sei ein Relationenalgebra-Ausdruck α.

Finde einen zu α äquivalenten Ausdruck β, der effizienter als α auswertbar ist.

Beispiel:

a) $\sigma_{\text{Attribut}=\text{Wert}}(r_1 \bowtie r_2)$, mit Attribut $\in \mathcal{R}_1$

b) $\sigma_{\text{Attribut}=\text{Wert}}(r_1) \bowtie r_2$

$|r_1|$ = 100
$|r_2|$ = 50
10 Tupel in r_1 erfüllen Attribut=Wert

Relationale Datenbanksysteme

Anfrageoptimierung

Gegeben sei ein Relationenalgebra-Ausdruck α.

Finde einen zu α äquivalenten Ausdruck β, der effizienter als α auswertbar ist.

Beispiel:

a) $\sigma_{\text{Attribut}=\text{Wert}}(r_1 \bowtie r_2)$, mit Attribut $\in \mathcal{R}_1$

b) $\sigma_{\text{Attribut}=\text{Wert}}(r_1) \bowtie r_2$

$|r_1|$ = 100
$|r_2|$ = 50
10 Tupel in r_1 erfüllen Attribut=Wert

Feststellung: (a) und (b) sind äquivalente Anfragen

Auswertungsaufwand:

a) 5000 Join-Operationen + 5000 Select-Operation = 10000 Operationen

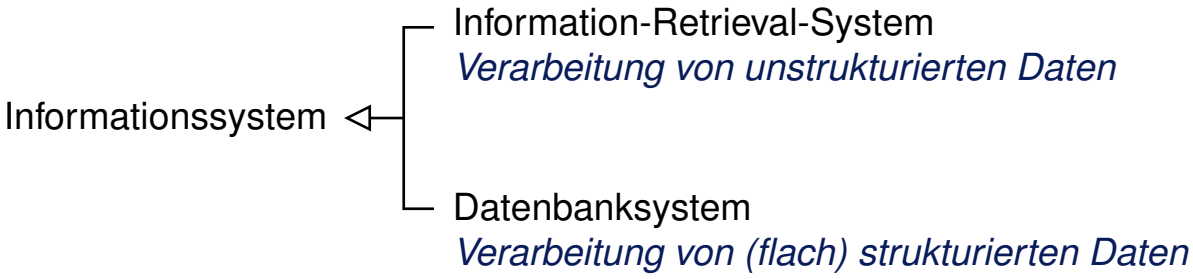
b) 100 Select-Operationen + 10·50 Join-Operationen = 600 Operationen

Bemerkungen:

1. Algebraische Optimierung bedeutet die Umformung von Ausdrücken.
2. Generelle Regel: Selektion-Operation vor Join-Operation durchführen.

Relationale Datenbanksysteme

Grenzen von RDBMS



- Realzeitanwendungen (Steuer- und Regelaufgaben)
  verlangen Reaktionszeitgarantien ➜ Echtzeit-DBS
- Speicherung und Verarbeitung von Expertenwissen
  wenige Objekte, viele Objekttypen, Anfragen ≈ Inferenzoperationen ➜ deduktive DBS
- Speicherung von Landkarten, annotierte Gebäudeinformationen
  komplex strukturierte Objekte, räumliche Informationen ➜ Geo-Informationsysteme
- Verarbeitung von Zeitreihen
  Informationen über regelmäßig wiederkehrende Ereignisse (tägliche Umsatzzahlen, etc.)
- Data Mining spielt eine zentrale Rolle
  Beispiel: Management-Informationssystem ➜ Data Warehouse, OLAP

Relationale Datenbanksysteme

Erstellung von RDBMS

Aufgabe	Tätigkeit	Sprache, Werkzeug
konzeptueller Entwurf	Beschreibung der Daten unabhängig von der Realisierung	Entity-Relationship-Modell, UML
relationaler Entwurf	Umsetzung in Relationendefinitionen	Relationenmodell
Datendefinition	Beschreibung der Relationen in einem SQL-Dialekt	SQL, DDL
Implementierung	Realisieren von Anfragen und Änderungen in SQL	SQL, DQL und DML
Qualitätssicherung	Analyse, Beweis, Dokumentation	formale Kriterien für Konsistenz, Effizienz, Wartbarkeit

Bemerkungen:

- Die Begriffe „Datenbankentwurf“, „konzeptueller Datenbankentwurf“ und „konzeptioneller Datenbankentwurf“ werden in der Literatur oft synonym benutzt.
- Die industrielle und universitäre Forschung beschäftigt sich mit der Weiterentwicklung klassischer Datenbanktechnologie, um die genannten `Nicht-Standardaufgaben` adäquat – d. h. effizient und auf die Daten abgestimmt – zu unterstützen.