Datenbanksysteme

Einführung und Grundlagen

Burkhardt Renz

Fachbereich MNI
Technische Hochschule Mittelhessen

Sommersemester 2021

Gliederung der Veranstaltung

- Grundlagen
  - Überblick über das Datenbank-Management
  - Datenbankarchitektur und Datenunabhängigkeit
  - Datenmodelle
- Das relationale Modell
  - SQL
  - Relationen und relationale Algebra
  - Datenbankintegrität
- Datenbank-Entwurf
  - Semantische Modellierung – Entity/Relationship-Modell
  - Schema-Entwurf
  - Funktionale Abhängigkeiten und Normalformen

Gliederung der Veranstaltung

- Transaktionen und Synchronisation konkurrierender Zugriffe
  - Transaktionen
  - Recovery
  - Isolationslevel in SQL
- Verwendung von Datenbanksystemen
  - Programmierung von Datenbank-Zugriffen mit JDBC
  - Zugriffsrechte und Datensicherheit

Ziele

- Konzepte von DBMS – was steckt dahinter?
  – keine speziellen Produkte
- SQL anwenden können
- Informationen strukturieren können
- Programmieren mit Datenbanken – erste Schritte

Literatur

- Gunter Saake, Kai-Uwe Sattler, Andreas Heuer: *Datenbanken – Konzepte und Sprachen* mitp
- Thomas Studer: *Relationale Datenbanken* Springer Vieweg
- Ramez Elmasri, Shamkant B. Navathe: *Grundlagen von Datenbanksystemen – Bachelorausgabe* Pearson Studium
- Matthias Schubert: *Datenbanken – Theorie, Entwurf und Programmierung relationaler Datenbanken* Teubner
- Wolfgang Gerken: *Datenbanksysteme für Dummies* Wiley-VCH
- Lehrbücher von Kifer, Bernstein und Lewis
  Elmasri und Navathe
  C.J. Date
  Silberschatz, Korth und Sudarshan
  Ullman, Garcia-Molina und Widom ...

Internet-Quellen

- Jennifer Widom: *Introduction to Databases* Online-Kurs der Stanford University `https://www.youtube.com/playlist?list=PLroEs25KGvwzmvIxYHRhoGTz9w8LeXek0`
- GNU: *SQLtutor* Interaktiver Web-basierter Tutor für SQL `http://geo102.fsv.cvut.cz/cgi-bin/cepek/sqltutor`
- Spiel zum Erlernen von SQL: `http://www.sql-island.de/`
- SQL-Tutorial: `https://sqlbolt.com/`
- Übungen zu SQL: `https://www.pgexercises.com/`

Übersicht

- Warum Datenbankmanagementsysteme?
  - Eine kleine Geschichte
  - Was ist ein DBMS?
  - Begriffe
- Datenunabhängigkeit und Datenbankarchitektur
  - Modell, Schema, Zustand
  - ANSI/SPARC-Architektur & Datenunabhängigkeit
  - Datenbanksprache
- Datenmodelle

Ein erfolgreicher Web-Shop

- Wir machen einen Web-Shop zum Verkauf von Spitzenweinen auf.
- Speichern von Daten wird notwendig: Artikel, Kunden, Aufträge...
- Speichern der Daten im Dateisystem, also: eine Datei für Artikel, eine Datei für Kunden, eine Datei für Bestellungen usw.
- Welche Probleme treten auf?
  - Datenredundanz
  - Anwendungen müssen Navigation programmieren
  - Abhängigkeit aller Verwender von der Repräsentation der Daten
  - Wenn mehrere Verwender die Daten gleichzeitig verändern wollen, können die Daten inkonsistent werden
- Das kann ja doch nicht die Wahrheit sein!

Ein Zitat

Das Problem wurde schon vor langer Zeit erkannt und auch gelöst:

> *Future users of large data banks must be protected from having to know how the data is organized in the machine (the internal representation). ... Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed and even when some aspects of the external representation are changed. Changes in data representation will often be needed as a result of changes in query, update, and report traffic and natural growth in the types of stored information.*
>
> *– E.F. Codd*

Quelle: E. F. Codd: *A Relational Model of Data for Large Shared Data Banks* Communications of the ACM Juni 1970

Wie sieht dann so eine relationale Datenbank aus?

Artikel:

ArtNr	Bez	Weingut	Jahrgang	Farbe	Preis
100001	Les Châteaux	Louis Max	2002	rot	17.90
100002	Chablis	Louis Max	2005	weiß	15.50
100003	Château Caraguilhes	Louis Max	2005	rosé	14.90
604851	Prosecco Val Monte	Cave Bellenda	<null>	weiß	7.60
145119	Le Cop de Cazes	Domaine Cazes	2004	rot	6.90

Weiter die Beispieldatenbank Wein

Lieferant:

LftNr	Firma	Postfach	PLZ	Ort
1	Weinimport Lehr	45367	F-68567	Colmar
2	Bremer Weinkontor	56	28195	Bremen

LieferBez:

LftNr	ArtNr
1	100001
1	100002
1	100003
2	100002
2	145119
2	604851

Weiter die Beispieldatenbank Wein

Kunde:

KndNr	Name	Vorname	Str	PLZ	Ort
100101	Kehl	Thomas	Weinstr. 3	79675	Kaiserstuhl
100102	Kehl	Thomas	Im Riesling 3	68734	Eltville
100105	Riesling	Karin	67, Rue du Château	F-68567	Colmar

Auftrag und AuftrPos:

AuftrNr	Datum	KndNr
1003	2007-03-01	100101
1001	2006-10-12	100101
1002	2006-02-12	100102
1004	2006-02-12	<null>

AuftrNr	Anzahl	ArtNr
1003	12	100001
1003	12	100002
1003	12	100003
1001	1	100001
1001	1	100002
1001	1	100003
1001	1	145119
1002	48	100003

Definition

Ein Datenbankmanagementsystem (DBMS) ist eine Software, die große Mengen von persistenten Daten für Speicherung und Zugriff verwaltet, und zwar

- *effizient,*
- *zweckmäßig,*
- *sicher und*
- *für den parallelen Zugriff vieler Anwender und Anwendungen.*

Diskussion, 1

- große Menge von Daten
  viel zu groß für den Hauptspeicher
- persistente Daten
  Daten bleiben erhalten, auch wenn Programme, die sie verwenden, beendet werden
- Multiuser-Zugriff
  Synchronisation der Zugriffe
  Verschiedene Sichten für verschiedene Anwendungen

Diskussion, 2

- sicher
  in Bezug auf Systemausfälle
  in Bezug auf Berechtigungen von Anwendern
- zweckmäßig
  einfache „Kommandos“ für den Zugriff
  aber auch: es ist einfach möglich, neue, bisher nicht vorgesehene Abfragen zu machen
- effizient
  Geschwindigkeit eines Zugriffs
  Zahl der Transaktionen in einer bestimmten Zeit

Arten von Datenbanksystemen

- Informations- und Verbuchungssysteme, z.B. Reisebuchung, Finanzielle Transaktionen etc.
  OLTP = Online Transaction Processing
- Multimedia-Datenbanken
- Geografische Informationssysteme (GIS)
- Data Warehouses, Informationssysteme für strategische Entscheidungen
  OLAP = Online Analytic Processing
- Echtzeit-Datenbanksysteme, z.B. zur Produktionssteuerung
- Internet-Suchmaschinen, Information Retrieval

Schwerpunkt der Vorlesung: *klassische Informationssysteme*

Einige Begriffe

- Daten = bekannte Tatsachen über die interessierende Domäne (Miniwelt)
- Datenbank = strukturierte Sammlung von Daten über eine Miniwelt, d.h.
  - logisch zusammenhängend
  - systematisch aufgezeichnet
  - gespeichert und gepflegt
  - zweckmäßig für (evtl. verschiedene) Anwender
- Datenbankmanagementsystem (DBMS) = Software zum Erstellen und Pflegen von Datenbanken; generisch, d.h. unabhängig von einem bestimmten Anwendungsgebiet
- Datenbanksystem = Einsatz eines DBMS für eine bestimmte Datenbank und bestimmte Anwendungen

Was leistet ein DBMS?

- Datenbank definieren
  Tabellen, Struktur der Datensätze in den Tabellen, Datentypen, Integritätsbedingungen
- Datenbank konstruieren
  Daten in die Datenstruktur einbringen und speichern, oft aus anderen elektronischen Quellen
- Datenbank verwenden
  Anfragen stellen = neue zutreffende Aussagen aus bekannten Fakten herleiten
  Daten pflegen = Aktualisieren der Daten, damit sie dem „Zustand“ der (Mini-)Welt immer entsprechen

Merkmale des Datenbankansatzes

- Gemeinsamer Datenbestand, integrierte Daten für verschiedene Anwendungen, gemeinsame Nutzung derselben Daten
- Datenunabhängigkeit = Immunität von Anwendungen in Bezug auf Änderungen der physischen Repräsentation der Daten und von Zugriffstechniken
- Unterstützung spezifischer Sichten auf die Daten, auch verschiedener Berechtigungen
- Steuerung des Mehrbenutzerbetriebs = Synchronisation konkurrierender Zugriffe.

Akteure im Zusammenhang mit Datenbanksystemen

Akteure auf der Bühne

- Datenadministrator/Datenbankdesigner, auch Datenarchitekt
- Datenbankadministrator
- Endbenutzer
- Anwendungsentwickler

Akteure hinter der Bühne

- Designer und Entwickler eines DBMS
- Werkzeug-Entwickler
- Operateure und Wartungsingenieure

Erstes Fazit

Vorteile von DBMS

1. Gemeinsame Nutzung von Daten
2. Kontrolle von Redundanz
3. Überwachung der Konsistenz der Daten
4. Sicherheit bzgl. von Berechtigungen
5. Sicherheit bzgl. der Persistenz der Daten
6. Synchronisation konkurrierender Datenzugriffe
7. Ausbalancieren konfligierender Anforderungen
8. Einhalten von Standards

Übersicht

- Warum Datenbankmanagementsysteme?
  - Eine kleine Geschichte
  - Was ist ein DBMS?
  - Begriffe
- Datenunabhängigkeit und Datenbankarchitektur
  - Modell, Schema, Zustand
  - ANSI/SPARC-Architektur & Datenunabhängigkeit
  - Datenbanksprache
- Datenmodelle

Datenmodell

Anwender und Anwendungen sollen die Datenbank unabhängig von ihrer physischen Repräsentation, ihrer Implementierung sehen. Wie sollen sie sie dann sehen?

Erster Begriff von Datenmodell

> *A data model is an abstract, self-contained, logical definition of the objects, operators, and so forth, that together constitute the abstract machine with which the users interact.*
> *The objects allow us to model the structure of data. The operators allow us to model its behavior.*
>
> *– C.J. Date*

Beispiel: das relationale Datenmodell
konkret: Wir organisieren die interessierenden Informationen als *Werte* in *Tupeln* (= Datensätzen), die *Relationen* (= Tabellen) bilden und geben an, wie diese Relationen *zusammenhängen*.

Datenbankschema

Beschreibung der Struktur der Daten für ein bestimmtes Anwendungsgebiet.
Metadaten sind die Informationen *über* den Aufbau der Daten – sie werden im relationalen Modell selbst wieder in Relationen gespeichert – Systemkatalog

Zweiter Begriff von Datenmodell (= Datenbankschema)

> *A data model is a model of the persistent data of some particular enterprise.*
>
> – *C.J. Date*

Beispiel: die Struktur unserer Datenbank für den Weinhandel

Datenbankzustand

Der Datenbankzustand ist

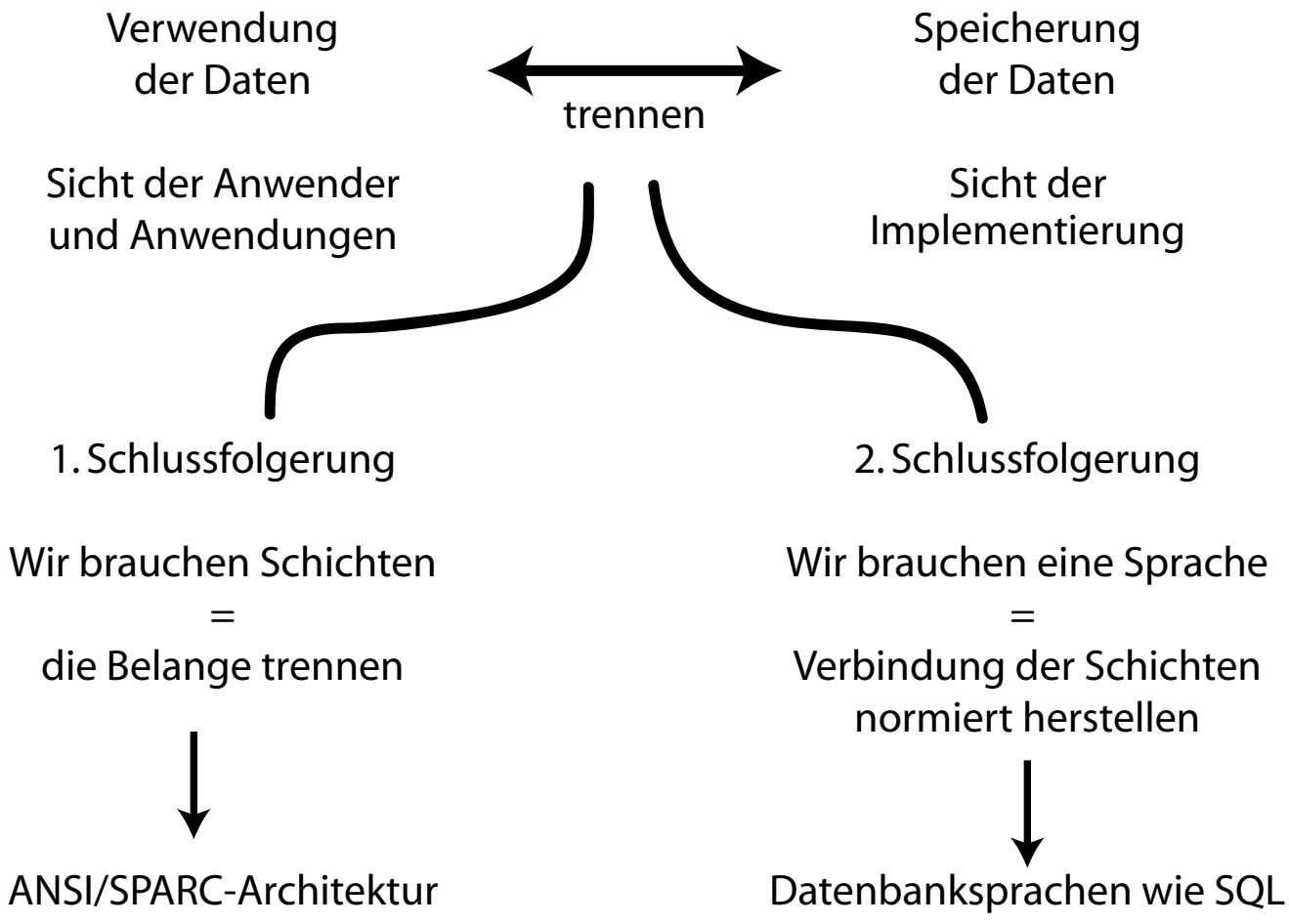
- der konkrete Inhalt der Datenbank zu einem bestimmten Zeitpunkt
- abgelegt in den Tabellen der Datenbank und dort gespeichert

Beispiel: der Inhalt unserer Datenbank für den Weinhandel

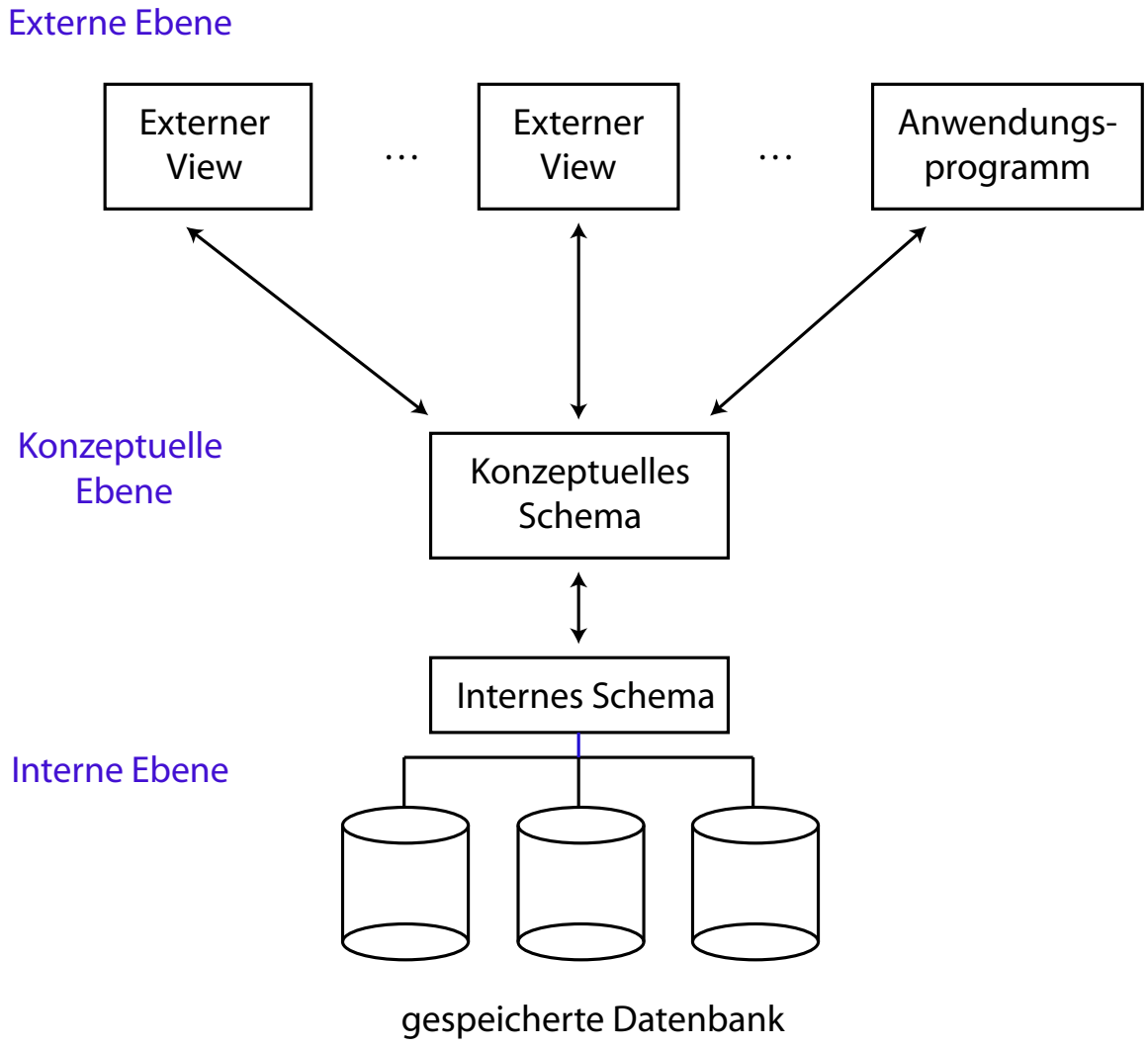
Fazit

- Datenmodell = Konzept der Strukturierung von Daten
- Datenbankschema = Struktur einer bestimmten Miniwelt (auch: Intension)
- Datenbankzustand = Inhalt einer Datenbank, also Fakten über eine Miniwelt (auch: Extension)

Wie erreicht man Datenunabhängigkeit?



Die ANSI/SPARC-Architektur



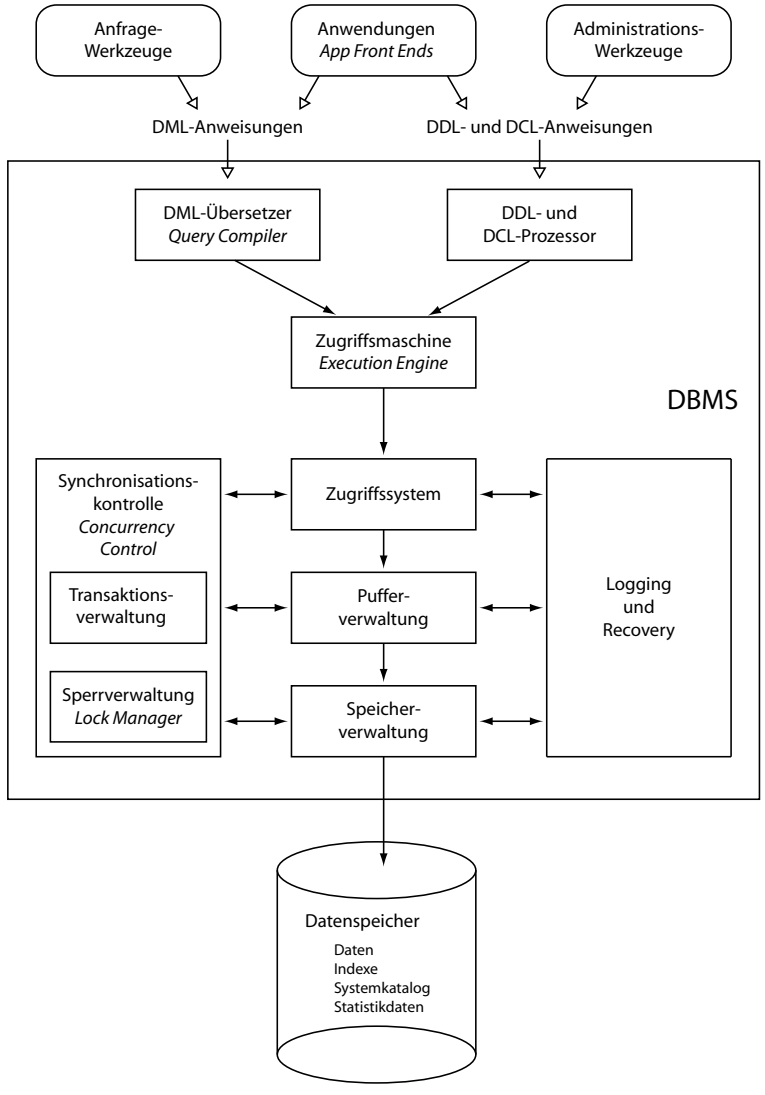
Datenunabhängigkeit

Datenunabhängigkeit Die Immunität von Anwendungsprogrammen gegenüber Änderungen des Datenbankmanagementsystems.
Änderungen des DBMS entkoppeln von den Anwendungen, die es benutzen.

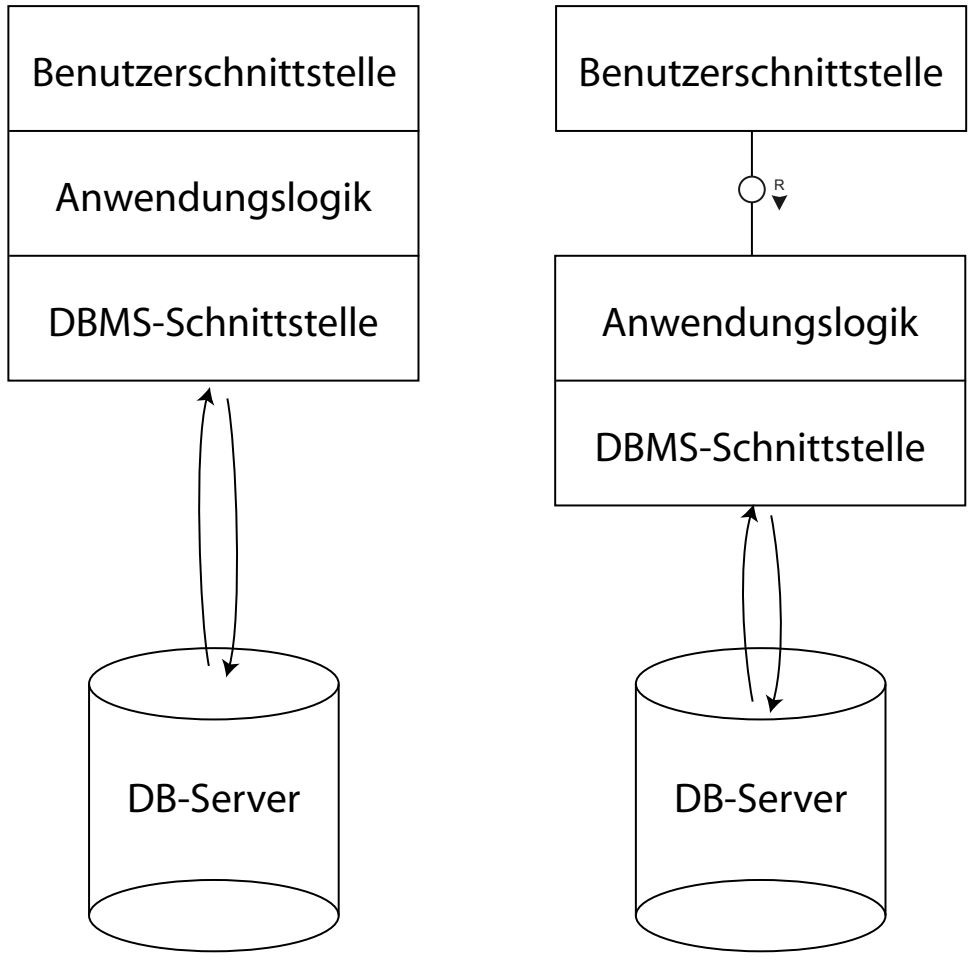
Physische Datenunabhängigkeit Änderungen an der Art der Datenspeicherung und den Zugriffstechniken haben keinen Einfluss auf Anwendungsprogramme.

Logische Datenunabhängigkeit Änderungen am konzeptuellen Schema haben nur erwünschten Einfluss auf Anwender und Anwendungsprogramme.

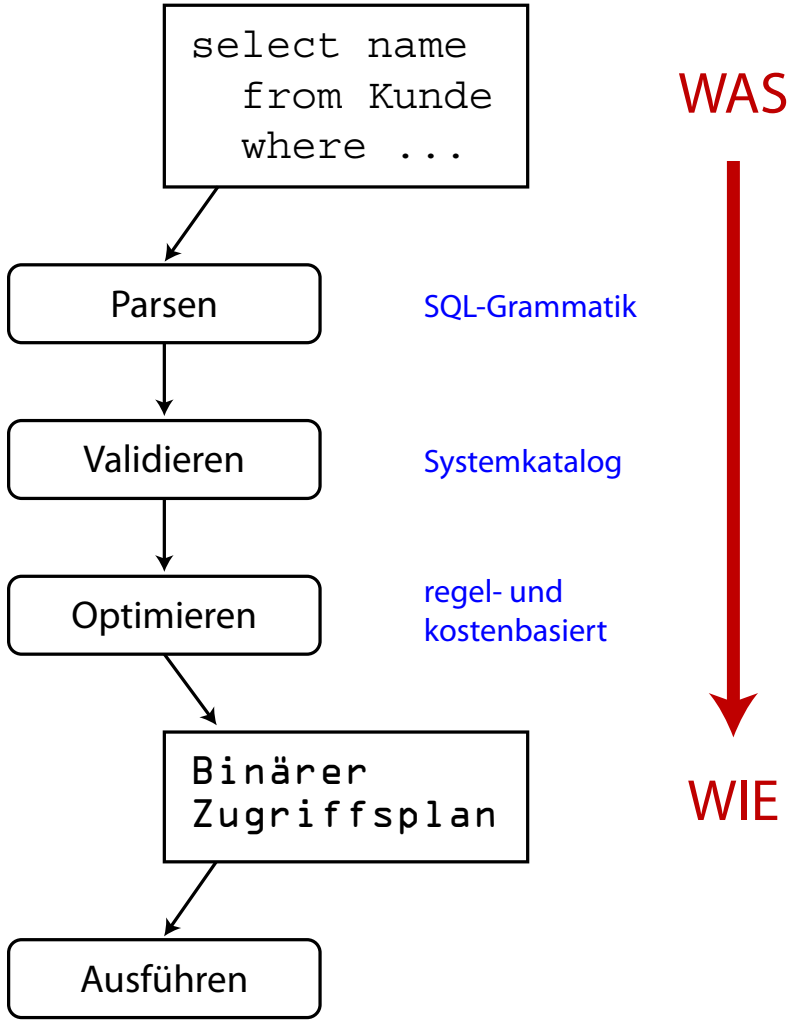
Komponenten eines DBMS



Anwendungsarchitektur



Die Datenbanksprache SQL



Sprachumfang

DDL *Data Definition Language*
Datendefinition
SQL: `create table...`, `alter table...`

DML *Data Manipulation Language*
Datenverwendung
SQL: `select...`, `insert...`, `update...`, `delete...`

DCL *Data Control Language*
Datenberechtigungen
SQL: `grant...`, `revoke...`

Übersicht

- Warum Datenbankmanagementsysteme?
  - Eine kleine Geschichte
  - Was ist ein DBMS?
  - Begriffe
- Datenunabhängigkeit und Datenbankarchitektur
  - Modell, Schema, Zustand
  - ANSI/SPARC-Architektur & Datenunabhängigkeit
  - Datenbanksprache
- Datenmodelle

Diskussion der Datenmodelle

Warum?

Wir brauchen logisches Modell, wie wir *über* Daten und ihre Struktur sprechen können wegen Datenunabhängigkeit.
$\rightarrow$ Datenmodelle

Leitbeispiel

Ein Lieferant (S = Supplier) hat eine Nummer <SNo>, einen Namen <SName> und einen Firmensitz <City>.
Ein Teil (P = Part) hat eine Nummer <PNo>, einen Namen <PName>, ein Gewicht <Weight> und wird in der Stadt <City> gelagert.
Ein Lieferant liefert Teile, ein Teil kann von verschiedenen Lieferanten kommen. Wir verzeichnen die Menge <Qty> des von einem Lieferanten gelieferten Teils.

Hierarchisches Datenmodell

Geschichte

- späte 60er Jahre
- 1968 IBM erste Version von *IMS* (Information Management System)
- heute Version 15

Konzept

- *Record Type* = Struktur eines Datensatzes mit benannten Felder und definierten Wertebereichen
- Jeder Record hat einen eindeutigen *Key*
- Alle Records sind in einem *Baum* organisiert durch sogenannte *Parent-Child-Relationships, PCR*

Netzwerk-Datenmodell

Geschichte

- 1971 CODASYL (Committee on Data System Languages)
- Implementierung z.B. UDS/SQL von Siemens/Fujitsu auf BS2000
- Datenmodell erlebt eine gewisse Renaissance durch das semantische Web (*semantisches Netz*, Graphdatenbanken)

Konzept

- *Record Type* – Datensatzstruktur
- Jeder Record hat einen eindeutigen *Key*
- Records sind organisiert als zusammenhängender *Graph* durch sogenannte *Owner-Child-Relationships*
- Für jede Datenbank gibt es einen oder mehrere *Entry Points*

Relationales Modell

Geschichte

- 1970 Edgar Frank „Ted“ Codd: Publikation des Modells
- erste Implementierungen etwa 1980: Oracle, INGRES
- IBM kündigt 1984 DB2 an
- heute vorherrschendes Datenmodell

Konzept

- *Relation* – Mengen von Tupeln von Werten
- *Informationsprinzip* – Die Daten sind in genau einer Weise organisiert, als Werte in Tupeln in Relationen
- Zugriff durch eine *deklarative* Sprache

Entity-Relationship-Modell

Geschichte

- 1974 Peter Chen Publikation des Modells
- Idee: Vereinheitlichung von Hierarchie – Netz – Relational

Konzept

- *Entitätstypen* – Klassen von „Dingen“, Objekten der Welt
- *Attribute* – Eigenschaften dieser Objekte
- *Beziehungstypen* zwischen den Entitätstypen
- Abbildung des ER-Modells in die anderen Datenmodelle

Objektorientiertes Datenmodell

Geschichte

- ab Mitte der 80er Jahre im Zuge der OO-Sprachen
- Ziel: Überwindung des Konzeptbruchs (*impedance mismatch*)
- ODMG (*Object Data Management Group*) Spezifikation 3.0 2000

Konzept

- *Objektmodell* – Objekte mit Methoden, Polymorphismus
- *Object Definition Language ODL*
- *Object Query Language OQL* – Sprache, die Navigation in Objektstruktur unterstützt
- *Sprachbindung* – für C++, Java, SmallTalk ...

Objekt-relationales Datenmodell

Geschichte

- Anfang der 80er
- SQL-Erweiterung in SQL:1999, SQL:2003, SQL:2008, SQL:2011
- Geografische Informationssysteme (PostGIS)

Konzept

Füge zu (klassischem) SQL hinzu:

- benutzerdefinierte Datentypen samt Methoden inklusive Vererbung und Polymorphismus
- benutzerdefinierte Operatoren
- benutzerdefinierte Zugriffsmethoden

Semistrukturierte Daten/XML

Geschichte

- HTML – Dokumentenstruktur auf Basis von SGML
- Übertragung auf Datenstruktur

Konzept

- Metadaten durch Tags mit Daten kombiniert
- Semantische Heterogenität möglich
- Hierarchische Struktur
- Links möglich – Netz-Datenmodell
- XML Schema – Vererbungsmechanismen aller Couleur
- XPath, XQuery – Navigation und Anfragesprache

NoSQL, Big Data

Geschichte

- XML – XML-Datenbanken
- Suchmaschinen im Internet
- Cloud

Konzept

- Verschiedene Datenmodelle
- Thema 0: Performanz bei großen Datenmengen für spezielle Anwendungen
- Thema 1: Verteilung der Daten
- Thema 2: Parallele Verarbeitung
- Thema 3: Konsistenz
- Thema 4: Heterogenität der Daten

NoSQL – Arten

Datenmodelle in NoSQL

- MapReduce Framework (z.B. Hadoop)
- Key-Value Store (z.B. Google Big Table, Amazon Dynamo, Cassandra)
- Dokumentenorientierte Datenbanken (z.B. Apache CouchDB, MongoDB)
- Graphendatenbanken (z.B. Neo4j)

Literatur und Links zum Thema Datenmodell

- Michael Stonebraker und Joseph M. Hellerstein: *What Goes Around Comes Around* `https://people.cs.umass.edu/~yanlei/courses/CS691LL-f06/papers/SH05.pdf`
- Jeffrey Dean und Sanjay Ghemawat: *MapReduce: Simplified Data Processing on Large Clusters* `https://static.googleusercontent.com/media/research.google.com/de//archive/mapreduce-osdi04.pdf`
- Pramod J. Sadalage und Martin Fowler: *NoSQL Distilled*, Addison-Wesley Professional, 2012
  Präsentation des Buchs `https://youtu.be/ASiU89Gl0F0`