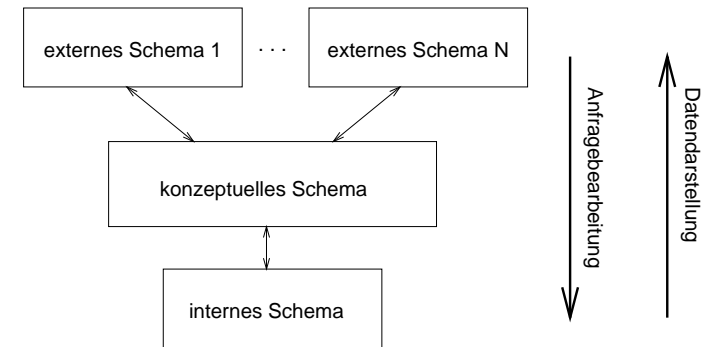


Schema-Architektur I

Zusammenhang zwischen

- Konzeptuellen Schema (Ergebnis der Datendefinition)
- Internen Schema (Festlegung der Dateiorganisationen und Zugriffspfade)
- Externen Schema (Ergebnis der Sichtdefinition)
- Anwendungsprogrammen (Ergebnis der Anwendungsprogrammierung)

Schema-Architektur III



2. Architekturen von DBS

- ➡ Schema-Architektur
- ➡ System-Architekturen
- ➡ Konkrete System-Architekturen
- ➡ Anwendungsarchitekturen

Schema-Architektur II

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

Ebenen-Architektur am Beispiel: Konzeptuelle Sicht

■ Relationale Darstellung

AUTOR	Name	Nr	BuchID => BUCH.BuchID
	Meier	1	4242
	Schulze	2	3745
	Ibsen	1	3745

BUCH	BuchID	Titel	Jahr	ISBN
	4242	Datenbasen I	1993	3-452-12
	3745	UNIX X	1998	1-424-11

Ebenen-Architektur am Beispiel: Externe Sicht II

■ Daten in einer hierarchisch aufgebauten Relation

TITEL	Autoren	Titel	Jahr	ISBN
	{ Autor }			
	Meier	Datenbasen I	1993	1-424-11
	Ibsen Schulze	UNIX X	1998	3-452-12

Datenunabhängigkeit

- Stabilität der Benutzerschnittstelle gegen Änderungen
- physisch: Änderungen der Dateioorganisationen 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
 - ◆ eventuell aber externe Schemata betroffen (Ändern von Attributen)
 - ◆ eventuell aber Anwendungsprogramme betroffen (Rekompilieren der Anwendungsprogramme, eventuell Änderungen nötig)
 - ◆ nötige Änderungen werden jedoch vom DBMS erkannt und überwacht

Ebenen-Architektur am Beispiel: Externe Sicht I

■ Daten in einer flachen Relation

TITEL	Name	Nr	Titel	Jahr	ISBN
	Meier	1	Datenbasen I	1993	1-424-11
	Schulze	2	UNIX X	1998	3-452-12
	Ibsen	1	UNIX X	1998	3-452-12

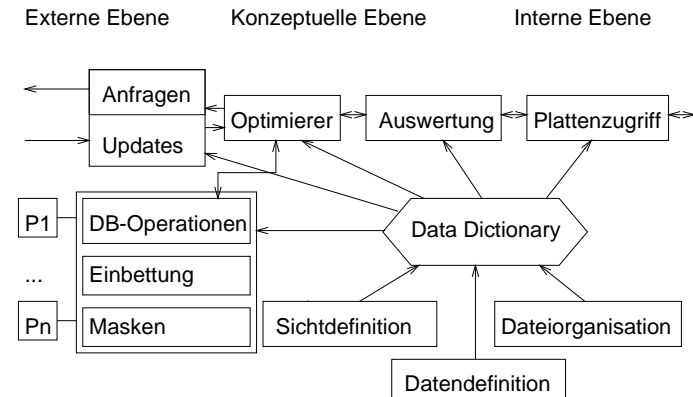
System-Architekturen

- Beschreibung der Komponenten eines Datenbanksystems
- Standardisierung der Schnittstellen zwischen Komponenten

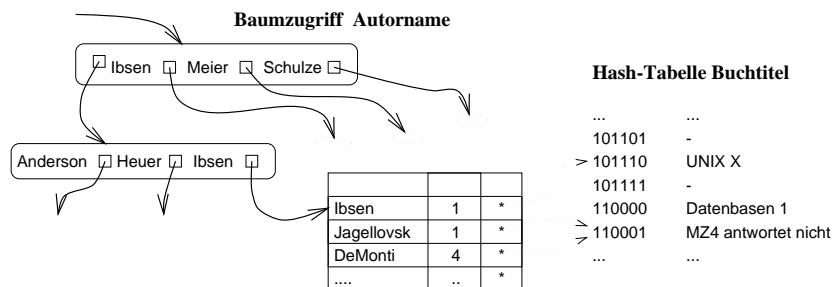
Architekturvorschläge

- ANSI-SPARC-Architektur
 - Drei-Ebenen-Architektur
- Fünf-Schichten-Architektur
 - beschreibt Transformationskomponenten

ANSI-SPARC-Architektur II



Ebenen-Architektur am Beispiel: Interne Darstellung



ANSI-SPARC-Architektur I

- 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

Fünf-Schichten-Architektur

- basierend auf Idee von Senko 1973
- Weiterentwicklung von Härder 1987
- Umsetzung im Rahmen des IBM-Prototyps *System R*
- genauere Beschreibung der Transformationskomponenten
 - ◆ schrittweise Transformation von Anfragen/Änderungen bis hin zu Zugriffen auf Speichermedien
 - ◆ Definition der Schnittstellen zwischen Komponenten

Fünf-Schichten-Architektur: Schnittstellen I

- Pufferschnittstelle
 - ◆ Seiten, Seitenadressen
 - ◆ Freigeben und Bereitstellen
- Datei- oder Seitenschnittstelle
 - ◆ Hole Seite, Schreibe Seite
- Geräteschnittstelle
 - ◆ Spuren, Zylinder
 - ◆ Armbewegungen

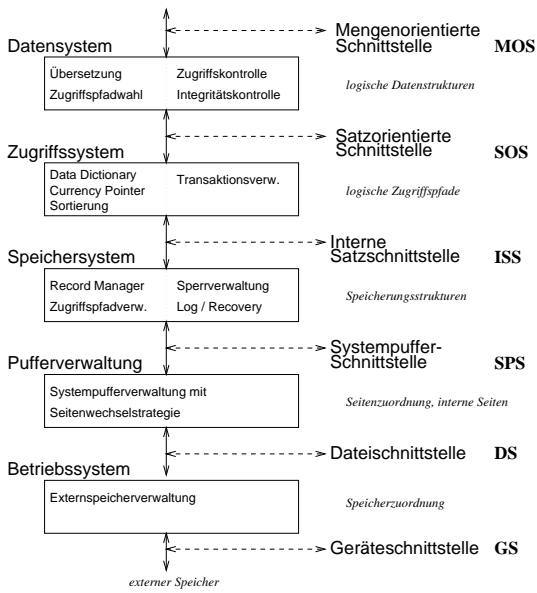
Klassifizierung der Komponenten

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

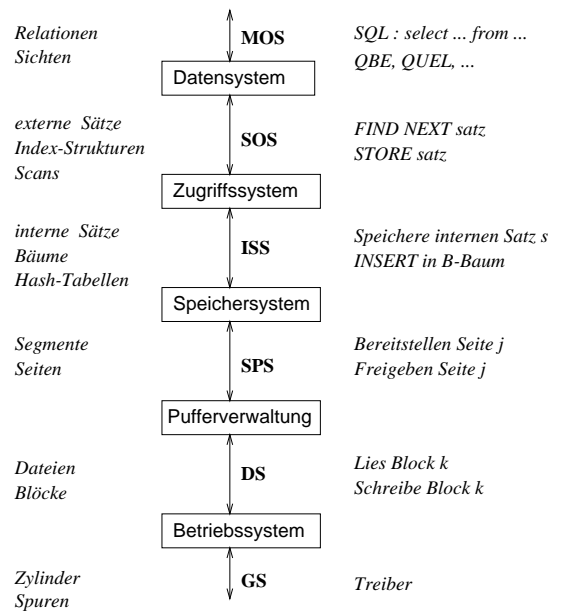
Fünf-Schichten-Architektur: Schnittstellen I

- mengenorientierte Schnittstelle:
 - ◆ deklarative DML auf Tabellen, Sichten, Zeilen
- satzorientierte Schnittstelle:
 - ◆ Sätze, logische Dateien, logische Zugriffspfade
 - ◆ navigierender Zugriff
- interne Satz Schnittstelle
 - ◆ Sätze, Zugriffspfade
 - ◆ Manipulation von Sätzen und Zugriffspfaden

Fünf-Schichten-Architektur: Funktionen



Fünf-Schichten-Architektur: Objekte, Operationen



IMS

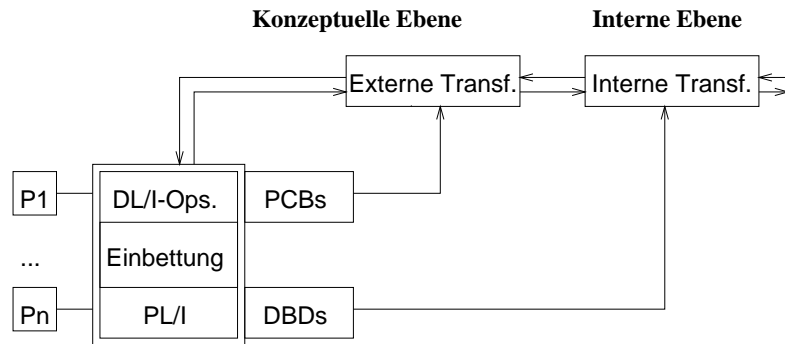
- IMS ≙ Information Management System (IBM, Mitte der 60er)
- Versionen
 - IMS/360
 - IMS/V/S (Virtual Storage)
 - ◆ IMS/V/S/DB (Data Base); nur batch-orientiert
 - ◆ IMS/V/S/DC (Data Communication); auch dialogorientiert
 - unter Betriebssystem DOS/VSE

Einige konkrete Systeme

Architekturüberblicke über

- IMS (hierarchisch)
- UDS (Netzwerk)
- relationale Systeme
 - ◆ DB2
 - ◆ SQL/DS
 - ◆ ORACLE
 - ◆ INGRES
 - ◆ dBASE
 - ◆ MS-Access

IMS (Skizze)



Relationale Systeme

Gemeinsame Merkmale

- Drei-Ebenen-Architektur nach ANSI-SPARC
- einheitliche Datenbanksprache (SQL; Structured Query Language)
- Einbettung dieser Sprache in kommerzielle Programmiersprachen
- diverse Werkzeuge für Definition, Anfrage und Darstellung von Daten; Entwurf von Datenbank-Anwendungsprogrammen; Benutzer-Interaktion
- kontrollierter Mehrbenutzerbetrieb, Datenschutz- und Datensicherheitsmechanismen

IMS II

■ Begriffseinordnung

Datenbank	≡	interne (physische) Datenbanken
internes Schema	≡	DBD (Data Base Description) für jede Datenbank
Sicht	≡	Menge logischer Datenbanken
Sichtdefinition	≡	PSB (Program Specification Block), bestehend aus mehreren PCBs (Program Communication Blocks)

Alles im Zugriff des Anwendungsprogrammierers

■ Sprache: DL/I (Data Language)

- ◆ eingebettet in PL/I, COBOL oder System/370 Assembler
- ◆ navigierend

UDS

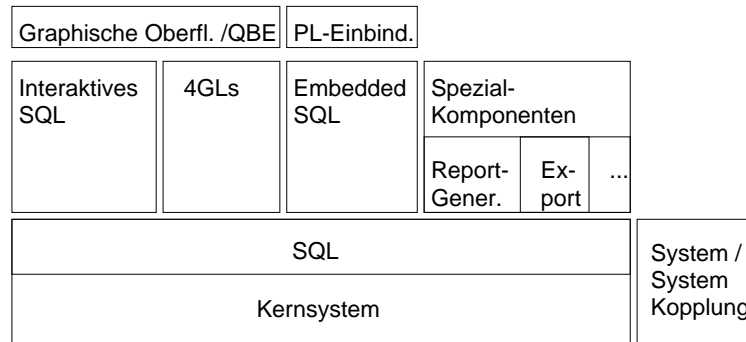
- UDS \cong Universelles Datenbank-System (Siemens, 1976)
- Betriebssystem BS 2000 für Siemens 7.100 bis 7.500
- Kein Optimierer, sonst Systemaufbau entsprechend ANSI-SPARC
- DML-Einbettung in COBOL, Assembler, FORTRAN, PASCAL

Begriffseinordnung

DDL	≡	Schema-DDL
VDL	≡	Subschema-DDL
SSL	≡	SSL
IQL	≡	IQS

Anwendungs-Architekturen

Benutzerkomponenten



Abarbeiten eines DB-Anwendungsprogrammes II

- Aus Benutzer- / Anwendungsprogrammierer-Sicht besteht DB2 aus
 - ◆ Precompiler
 - ◆ Binder
 - ◆ Laufzeitüberwachungssystem
 - ◆ Dateiverwaltungssystem

Relationale Systeme II

Beispiele

- DB2 (SQL/DS)
 - Entwicklung System R → DB2 ∼ □ □
- ORACLE
- INGRES

Pseudo-RDBS (keine drei Ebenen, kein Mehrbenutzerbetrieb, ...)

- dBASE
- MS-Access

Abarbeiten eines DB-Anwendungsprogrammes

am Beispiel DB2

- Tabelle AUSLEIHE
- Spalten: NAME, INVENTARNR
- SQL-Anfrage:

```
SELECT NAME
FROM   AUSLEIHE
WHERE  INVENTARNR = 82;
```

soll in C-Programm eingebettet werden.

DB2-Bausteine II

- Laufzeitüberwachungssystem
 - ◆ Überwacht die Ausführung des Anwendungsprogramms
 - ◆ Falls Anwendungsprogramm auf SQL-Statement stößt, wählt Laufzeitüberwachungssystem Teil eines passenden Anwendungsplanes aus
- Dateiverwaltungssystem
 - ◆ Verwirklicht schnellen Zugriff auf Daten
 - ◆ Z.B. Suchprozeduren, Updateoperationen, Indexverwaltung, ...

DB2-Bausteine

- Precompiler für C, PL/I, COBOL, ...
 - ◆ Quellprogramm analysieren
 - ◆ Alle SQL-Statements durch Unterprogrammaufrufe ersetzen
 - ◆ Konstruktion eines Database Request Modules (DBRM) für jedes SQL-Statement, der Eingabe für Binder sein wird
- Binder
 - ◆ Ein oder mehrere DBRMs zu einem "Anwendungsplan" übersetzen
 - ◆ Der Anwendungsplan enthält ausführbaren Code zur Verwirklichung des ursprünglichen SQL-Statements
 - ◆ Der ausführbare Code enthält Betriebssystemaufrufe zum Dateiverwaltungssystem

Abarbeitung Anwendungsprogramm (Skizze)

