

# **Datenbanksysteme (DBS) (080041)**

Nebenfach alle Studiengänge / Zertifikat, 2 SWS, ECTS-Credits: 2,  
Zertifikat Praktische Informatik

**Prof. Dr. B. Thalheim**

Sommersemester 2004, Di 16:15 - 17:45, CAP3 - R.I

Termine: 6.4., 13.4., 20.4., 27.4., 4.5., 11.5., 18.5., 23.5. (PS), 1.6. (PS), 8.6., 15.6.,  
22.6., 29.6., 6.7., 13.7.

## **Programm:**

- Grundlagen von Informationssystemen
- Datenbankmodelle (Relationales, Entity-Relationship-Modell)
- Datenbanksprachen (SQL, QBE)
- Datenbankprogrammierung (DBD, DBM)
- Datenbankmodellierung (Prinzipien)
- Information-Retrieval-Systeme (Übersicht)
- Hypertext- Informationssysteme (HTML, XML, ...)
- Verteilte Informationssysteme
- Informationsdienste im Internet
- Management-Informationssysteme (Anwendungen)

# 1. Grundlagen und Ziele

*Manuelle Produktion* - *Industriezeitalter* - *Informationszeitalter*  
*versus Verarbeitungskapazität, Formulierungsfähigkeiten des Menschen,*  
*Gewohnheiten des Menschen*

**Ziel: Informationen zur rechten Zeit, der richtigen Dosis, der richtigen Sorte,  
in der richtigen Form, in vollem Umfang, zu akzeptablen Kosten, für alle Benutzer**

## **Unterschiedliche Arten von Informationssystemen (CIS)**

- *Management-Informationssysteme*
- *Entscheidungsunterstützungssystem*
- *(Hyper-)Textsysteme*
- *Information-Retrieval-Systeme*
- *Dokumentationssysteme*
- *Büroinformationssysteme*
- *Datenbanksysteme*
- *Wissensbanksysteme*

**Datenbanksysteme zunehmend mit Funktionalität aller anderen CIS !!**

## Kennzeichen von Datenbanksystemen

- große Datenmengen *large amount*
- dauerhaft *persistent*
- verlässlich *dependable*
- verschiedene Benutzer *shared*
- Sicherheit der Daten *security*
- effizient (verwalten, anfragen, ändern) *storage, query, update*

⇒ `Modell` der Welt (Strukturierung, Verhalten, Semantik)

unterschiedliche Benutzung (Zusammenwirken, Oberflächen, Dauerhaftigkeit)

3-Ebenen-Architektur (interne, konzeptionelle, externe (Sichten))

Parallelität, Sekundär- und Tertiärspeicher, Wiederanlauf, Sicherheitskonzepte,  
Optimierung, (einfache) (deklarative) Programmiersprache

**Benutzer:** (Datenbank-)Administrator, Entwerfer (Designer), Systemanalytiker,  
**Endbenutzer:** Zufällige, naiver (parametrischer), Experte

## Nutzen von DBS

- Kontrollierte Redundanz
- Daten-Sharing
- Eingeschränkter Benutzerzugriff
- Verschiedene Interfaces
- Darstellung komplizierter Berechnungen der Daten
- Integritätserzwingung
- Robustheit, Wiederanlauf

## Einsatz von DBS

- Heterogene Benutzergruppen, gemeinsame Datenhaltung
- Keine oder kontrollierte Redundanz
- Benutzerverwaltung, Authorisierung
- Unterschiedliche Schnittstellen für unterschiedlich geschulte Benutzer
- Komplexe Daten, komplexe Beziehungen zwischen den Daten
- Integritätsmechanismen
- Backup, Fehlerbehandlung
- Geringe Entwicklungszeit für sich ändernde Anwendungen

*Somit*

## •Vorteile von DBS

- Standarderzwingung
- Flexibilität
- Verringerte Anwendungsentwicklung
- Up-to-date-Information für alle
- Aufgabenverteilung

DBS sollten **nicht benutzt** werden, wenn:

- Aufwand zu hoch
  - initialer Aufwand
  - Allgemeinheit vorh. Fkten.
  - Sicherheit ,... zu hoch
- Einfache Anwendung
- Real-time-Forderungen
- Kein Parallelzugriff

Annahmen:

1. DBS = DBMS + DB'en
2. Dreiebenenarchitektur einer DB-Anwendung

Damit:

### **5-Schichten-Architektur eines DBMS**

1. Ebene der Benutzersprache
2. Ebene der Anfrageverarbeitung
3. Ebene der Zugriffsstrukturen und Code-Erzeugung
4. Ebene der Synchronisation paralleler Zugriffe
5. Ebene der Speicherverwaltung

**Benutzungsschnittstelle**

**Geräteschnittstelle**

1 ⇔ 2:	Mengen-Schnittstelle	<i>select, insert, delete, update</i>	<b>view, relation</b>
2 ⇔ 3:	Tupel-Schnittstelle	<i>fetch, store, erase, modify</i>	<b>cursor, tuple</b>
3 ⇔ 4:	Record-Schnittstelle	<i>retrieve, add change, dispose, change</i>	(indexed) <b>record</b>
4 ⇔ 5:	Seiten-Schnittstelle	<i>read, write</i>	<b>Seite, Segment</b>

## Arbeitsweise eines DBMS am Beispiel der Anfrageverarbeitung

1. Überprüfung der Syntax
2. Authorisierungs- und Existenztest
3. Generieren der notwendigen internen Operationen und Speichermechanismen
4. Generieren eines effizienten Programmes
5. Berechnen der Anfrage in der DB
6. Aufbereiten der Anfrageergebnisse
7. Sicherstellen, daß das Ergebnis während der Ausführung nicht geändert wird

## Sprachen zur Benutzung

1. **Data Definition Language (DDL)**: Definition des konzeptionellen und der externen Schemata (daraus wird Speicherbelegung abgeleitet (**Storage Definition Lang.**))  
mitunter extra: **View Definition Language**
2. **Data Manipulation Language (DML)**: Manipulations- und Anfragesprache  
**Retrieval** (Selektion) für die Anfrage  
**Insert** für das Einfügen von (noch nicht vorhandenen) Daten  
**Delete** für das Streichen von (vorhandenen) Daten  
**Update** für die Modifikation von (lokalisierbaren) Daten  
i.a. *nicht prozedural, mengenorientiert, deklarativ*
3. **Host language** (Wirtssprache) des unterlegten Programmiersystemes
4. **Data control language (DCL)**: zur Steuerung der Datenbank  
grant, revoke, commit, rollback, assign

Zur Vereinfachung der Arbeitsweise

## **3-Ebenen-Architektur**

auf der Grundlage

von Unabhängigkeitskonzepten

### **Externe Ebene**

Benutzergruppen mit ihren Sichtweisen auf die Datenbank

*Unabhängigkeit der Benutzer von Integrationsmechanismus*

Benutzer sieht Datenbanksystem und Anwendung

### **Konzeptionelle Ebene**

Integriertes Schema der Datenbank mit ableitbaren Sichten

*Logische Unabhängigkeit von unterlegtem DBMS*

Programmierer, erfahrener Benutzer

### **Logische Ebene**

Schema zur Implementation, internen Darstellung, mit adäquater Repräsentation des konzeptionellen Schemas

*Physische Unabhängigkeit von unterlegter Speicherarchitektur*

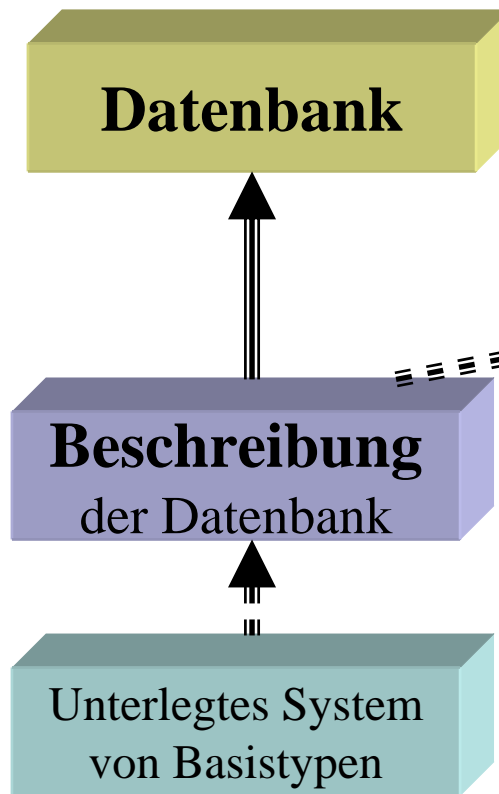
Datenbankadministrator, Datenbankprogrammierer

### **(Physische Ebene)**

Implementation, Speicherstruktur  
Betriebssystem

# 3 ★ Schichten einer Datenbank

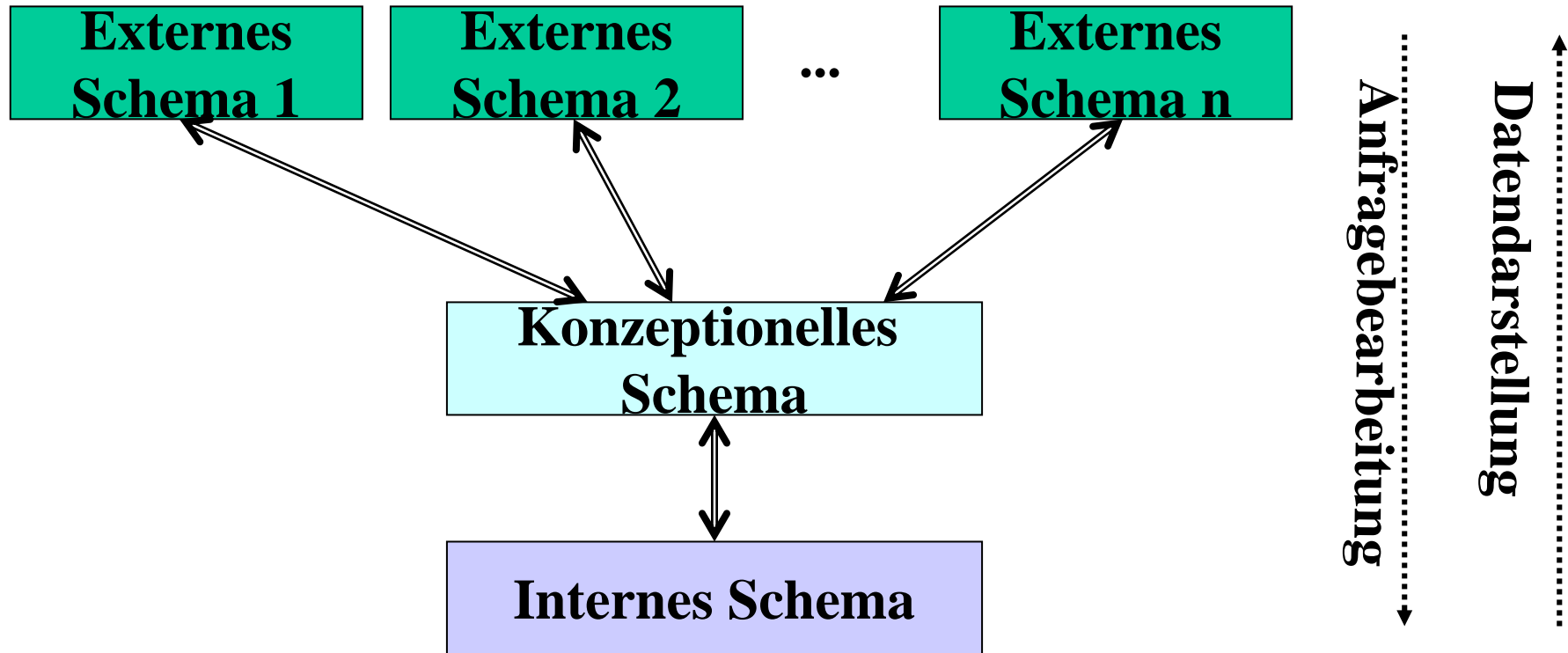
**Data Dictionary**



integer, real, float, decimal,  
Boolean, date, time

Konzeptionelle Schicht:  
(E)ER-Modell  
Logische (Implem.-)Schicht:  
relationales,  
hierarchisches  
Netzwerk-Modell  
Implementationsschicht:  
Physische Modelle

# 3-Ebenen-Architektur für Schemata von Datenbankanwendungen



**Modellierungsobligationen: a) konzeptionelles Schema  
b) Sichten**

# Einige Leistungen der Datenbanktechnologie:

## I. Technologie auf sicherer theoretischer Basis

Mit der Entwicklung des relationalen Datenbankmodelles

Strukturierung von Datenbanken

Semantik von Datenbanken

Generische Operationen

Optimierungstheorie

Interoperabilität, Verteilung, middleware

Dynamik von Datenbankanwendungen

Benutzerunterstützung

Ausgereifte Datenbankmodelle

Darauf aufsetzend:

Theorie semantischer und

objekt-orientierter Datenbanken

# Einige Leistungen der Datenbanktechnologie: (Fortsetzung)

## II. Ausgereifte Technologie

Ausgereifte, standardisierte Programmiersprachen

SQL92 (SQL2), Einbettungsprinzipien in Wirtssprachen

Ausgereifte Architektur von Systemen

Einbettung in Betriebssysteme, graphische Systeme, ...

Komponentenarchitektur, 3-Ebenen-Architektur

Verteilung (two tier, three tier, trader)

Benutzbarkeit für jedermann

Benutzungsoberflächen

Vielfältige Einsatzgebiete durch Erweiterungen der Technologie

Informationssysteme, Wissensbanken,

Information-Retrieval-Systeme, Suchmaschinen,

Dokumenten-Management-Systeme,

Entscheidungsunterstützungssysteme, MIS, EIS,

Data Warehouse, Internet-DB, Multimedia-DB, Dienste

# Einige Leistungen der Datenbanktechnologie: Fortsetzung

## III. Sichere Technologie

Sicher für den Anwender

Verhalten der Datenbank gesichert

Daten gegen Schäden der HW/SW gesichert

Sicher gegen Fremdbenutzer

Einhaltung der Sicherheitsforderungen

Veränderungssicherheit

Nachvollziehbarkeit

Sicherheitsnormen auf B-, sogar A-Niveau

Sicher gegen den Anwender

und seine Fehler (Konsistenz erzwingung),

seine Nachlässigkeiten (Transaktionskonzept),

seine Neugier (Sichten)

# Einige Leistungen der Datenbanktechnologie: (Fortsetzung)

## IV. Ausgereifte Pragmatik vorhanden:

Methodiken und Vorgehensmodelle zum Entwurf und zur  
Implementation von Datenbank Anwendungen  
insbesondere für relationale Anwendungen

Beherrschung des Anwendungs-Engineering über die volle  
Lebenszeit der Anwendung  
mit Redesign, Re-Engineering

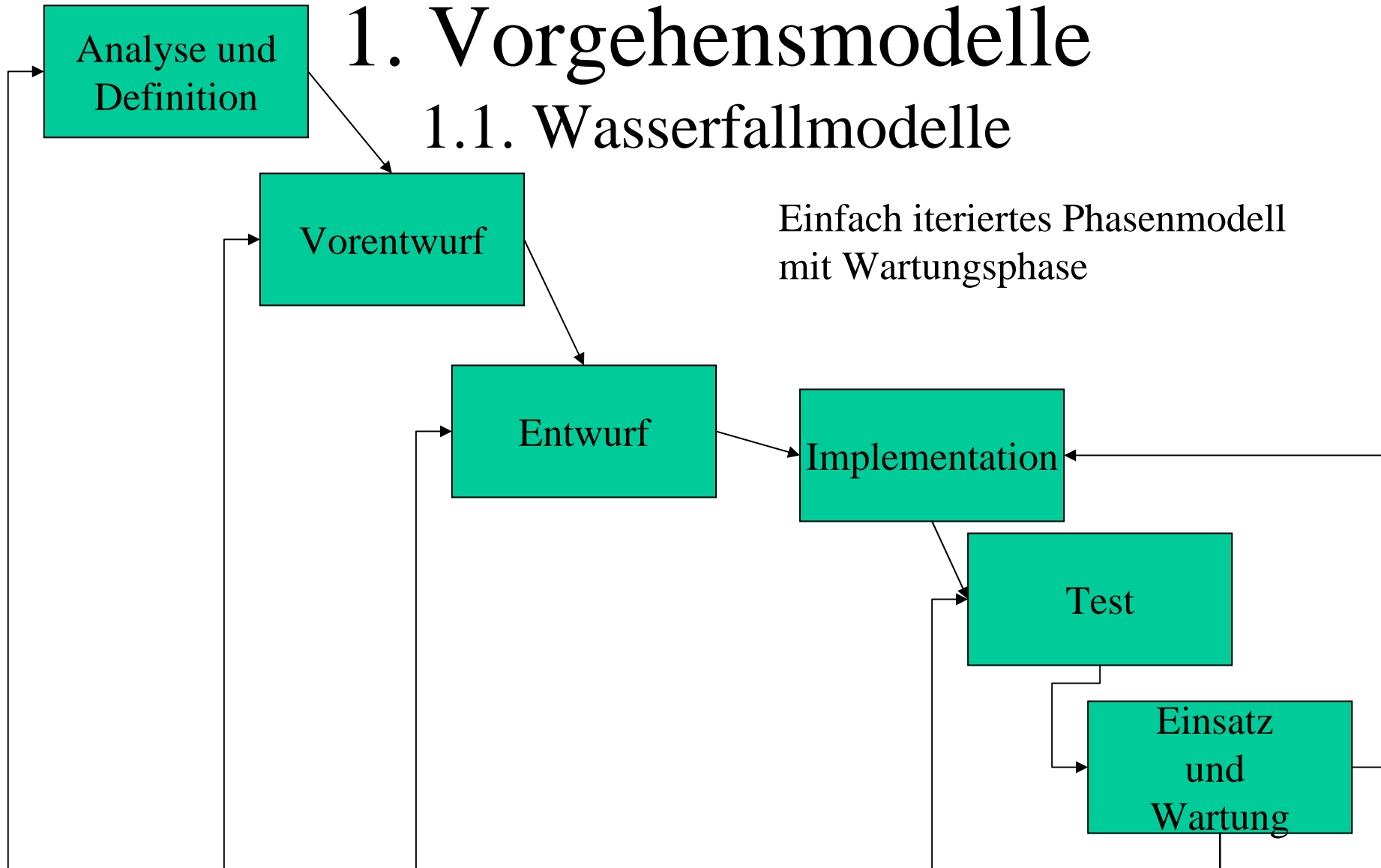
Beherrschung der Abstraktion

Somit:           Regelmäßiges semiotisches Dreieck  
                  (Syntax, Semantik, Pragmatik)

# 1. Vorgehensmodelle

## 1.1. Wasserfallmodelle

Einfach iteriertes Phasenmodell  
mit Wartungsphase



## Phasenmodell

**A1.** Problemstellung

**A2.** Studie zur Problemanalyse

**A3.** Anforderungsdefinition

Plichtenheft, Lastenheft

**B.** Grobentwurf

**C.** Architektur, Entwurf

**D.** Bausteinspezifikation

**E.** Fertiges System

**F.** System in der Zielumgebung

### *Phasenübergreifendes:*

1. Leiten (managen)

2. Qualität

3. Dokumentieren

## Grundstruktur

*analysieren*

*definieren*

*entwerfen, spezifizieren*

*spezifizieren, entwerfen i.w.S.,  
validieren, Alternativen ausw.*

*entwerfen*

*implementieren, integrieren*

*installieren*

*betreiben, betreuen*

*planen, führen, überwachen*

*sichern*