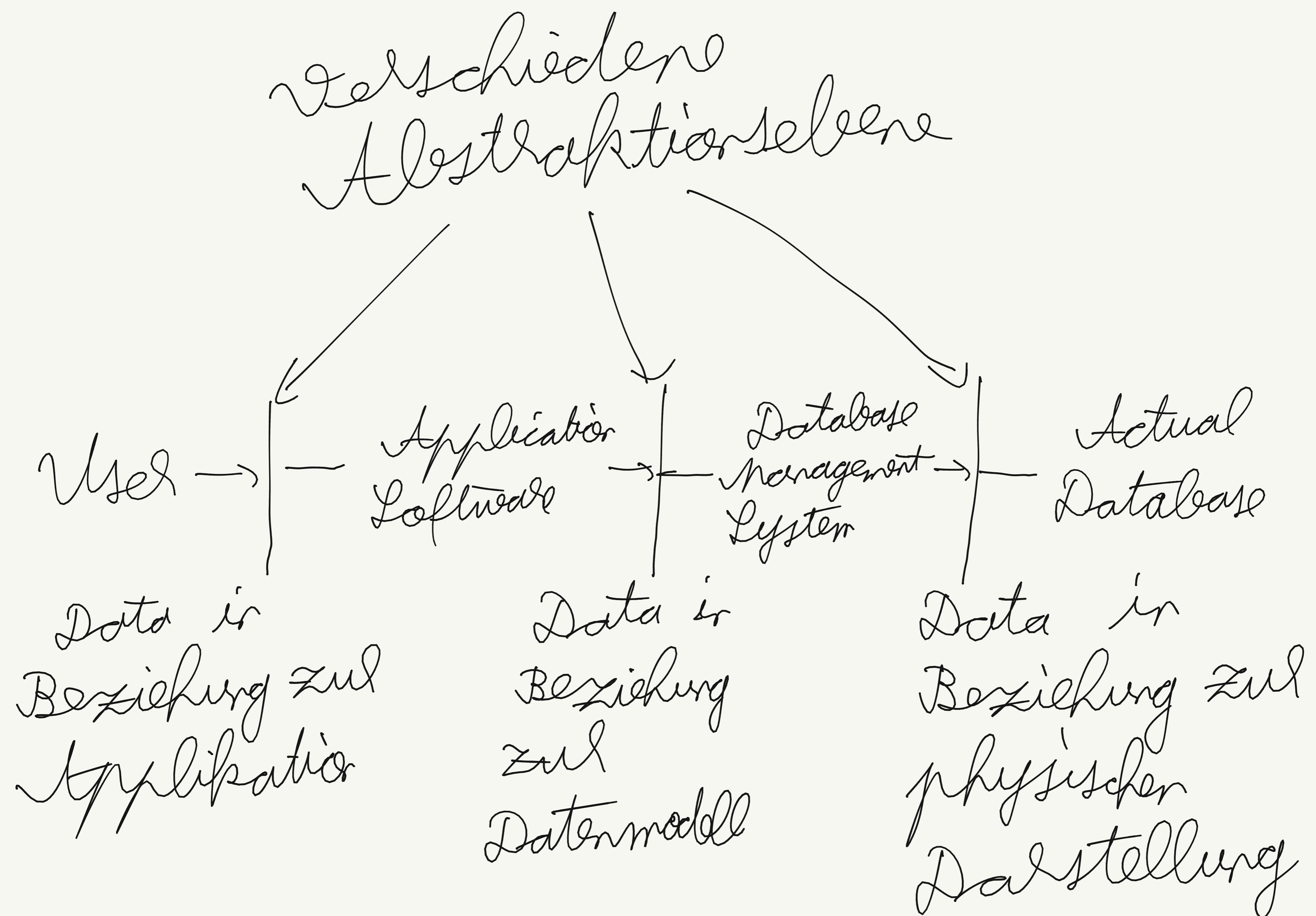


## Vorlesung 3

### Abstraktionssschichter. Das Relational Datenmodell



# Datenunabhängigkeit

- Ziel: die Datendefinition in einer Schicht zu ändern, ohne dabei die Definition der Daten in der darüber liegende Schicht zu beeinflussen
- Logische Datenstrukturen unabhängig von physischer Datenstrukturen
- Robustheit der Anwendungen gegenüber Änderungen

## 3-Schichten nach ANSI-SPARC

- Externe Ebene:
  - Anwendungsspezifisch / Anwendungen
  - Ebene der Anwendungen und Verwendung der Daten
  - Beschreibt wie der Benutzer die Daten sieht
- Logische Ebene
  - Definition der logischen Datenstrukturen: die Beziehungen zwischen den Daten unabhängig von der physischen Repräsentation
- Interne / Physische Ebene:
  - Definition des physischen Schemas
  - Beschreibt die Dateien und Indexstrukturen
  - Es geht um Leistungsfähigkeit der Datenbankanwendungen

## Physische Datenunabhängigkeit

- Änderungen an der Art der Datenspeicherung und der Zugriffstechniken haben keinen Einfluss auf Anwendungsprogramme
- Programme sind von interner Datenorganisation unabhängig
- Physische Datenunabhängigkeit wird durch relationale DBMS weitestgehend hergestellt

## Logische Datenunabhängigkeit

- ermöglicht durch externe Ebene
- Änderungen an logischer Schema haben nur geringe Auswirkungen auf die Anwendung
- ist nur begrenzt möglich, denn Anwendungen basieren sich auf dem logischen Schema
- Sichtkonzept (Views) ermöglicht Verbergen von Details des logischen Datenmodells

## Queries (Abfragen) im DBMS

→ Frager, die sich auf Daten gespeichert im DBMS beziehen, sind Datenbank-Abfrager

## Datenbanksplache

- Formale Sprache
- hat folgende Komponente:
  - Data Definition Language (DDL):
    - Befehle zur Definition des Datenbankschemas
    - Constraint Definition Language (CDL)
    - Storage Definition Language (SDL)
  - Data Manipulation Language (DML)
    - Befehle zur Datenmanipulation
    - Procedural DML (wie) vs.  
Declarative DML (wie)

## Abfragesplache für relationale Datenbasen

- SQL
  - z.B.:  $\text{SELECT name FROM studenter}$   
 $\text{WHERE age} > 20$
- Relationale Algebra
  - z.B.:  $\Pi_{\text{name}} (\sigma_{\text{age} > 20} (\text{Studenter}))$
- Domänenkalkül (Domain Calculus)
  - z.B.:  $\{ \exists X \geq 1 \vee \exists Y \exists Z \exists T : \text{Studenter}(V, X, Y, Z, T) \wedge Z > 20 \}$
- Tupelkalkül
  - z.B.:  $\{ X \exists Y : Y \in \text{Studenter} \wedge Y.\text{age} > 20 \wedge X.\text{name} = Y.\text{name} \}$

## Relational Query Language

- Vorteil des relationalen Modells:  
unterstützt einfache Datenbankfragenstellungen
- Abfragen können intuitiv formuliert werden und der DBMS ist zuständig für die optimale Abfrage Bearbeitung
- manchmal müssen wir die Abfrage optimal formulieren, um die gewünschte Zeitkomplexität zu erreichen

→ SQL