WWU MÜNSTER

Einführung

Datenbanken

Gestell	Sorte	Jahrgang	Anzahl_Flaschen
2	Franken	2009	5
1	Baden	2006	3
4	Rheinhessen	2007	10
1	Mosel	2013	2
2	Franken	2010	10

```
SELECT Gestell, Sorte, Jahrgang
FROM Weinkeller
WHERE Anzahl_Flaschen >= 4
```


Tanya Braun
Arbeitsgruppe Data Science, Institut für Informatik

Inhalte: Datenbanken (DBs)

1. **Einführung**
   - Anwendungen
   - Datenbankmanagementsysteme
2. **Datenbank-Modellierung**
   - Entity-Relationship-Modell (ER-Modell)
   - Beziehung zwischen ER und UML
3. **Das relationale Modell**
   - Relationales Datenmodell (RM)
   - Vom ER-Modell zum RM
   - Relationale Algebra als Anfragesprache
4. **Relationale Entwurfstheorie**
   - Funktionale Abhängigkeiten
   - Normalformen
5. **Structured Query Language (SQL)**
   - Datendefinition
   - Datenmanipulation
6. **Anfrageverarbeitung**
   - Architektur
   - Indexierung
   - Anfragepläne, Optimierung
7. **Transaktionen**
   - Transaktionsverarbeitung, Schedules, Sperren
   - Wiederherstellung
8. **Verteilte Datenbanken**
   - Fragmentierung, Replikation, Allokation; CAP
   - Anfragebeantwortung, föderierte Systeme

Überblick: 1. Einführung

A. Datenbanken

- Datenbank (DB)
- Datenabstraktion, Datenmodelle, Datenunabhängigkeit
- Mehrbenutzersysteme

B. DB-Umgebungen

- DB-Sprachen
- Datenbanksystem (DBS)
- Datenbankmanagementsystem (DBMS)

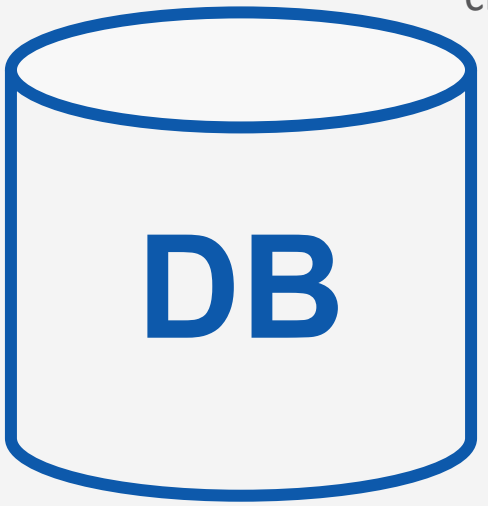
C. Phasen des DB-Entwurfs

- Anforderung, Modellierung

Datenbank (DB)

Sammlung von Daten
- in der Regel logisch zusammenhängend

“closed world assumption”



Miniwelt oder
Universe of Discourse

Daten → Information
(Interpretation)

Kennzeichen von Daten in DBs
- Lange Lebensdauer (Jahre, Jahrzehnte)
- Reguläre Strukturen
- Große Datenobjekte, große Datenmengen
- Stetig anwachsende integrierte Bestände (Giga-, Tera-, Petabyte)

Erstes Beispiel einer (relationalen) DB und einer Anfrage

- DB für Inventar eines Weinkellers
  - Tabelle Weinkeller

Gestell	Sorte	Jahrgang	Anzahl_Flaschen
2	Franken	2009	5
1	Baden	2006	3
4	Rheinhessen	2007	10
1	Mosel	2013	2
2	Franken	2010	10

- Anfrage: Alle Informationen zu Weinen, von denen es mindestens 4 Flaschen gibt

```
SELECT Gestell, Sorte, Jahrgang
FROM Weinkeller
WHERE Anzahl_Flaschen >= 4
```

- Ergebnis:

Gestell	Sorte	Jahrgang
2	Franken	2009
4	Rheinhessen	2007
2	Franken	2010

E. F. Codd: A Relational Model of Data for Large Shared Data Bank. In: *Communications of the ACM*, Vol. 13, No. 6, pages 377-387, 1970.

Datenabstraktion

- abs-trahere = "abziehen, entfernen"
  → Weglassen von Einzelheiten, Überführen in etwas Einfacheres
- Abstraktion: ein Grundprinzip der Informatik!
  - Abstraktionsschicht verbirgt Einzelheiten/Aufgaben
    → Unabhängigkeit für darüber liegende Schichten
- Unterschiedliche Dimensionen:
  - Abstraktion von der Speicherung
  - Abstraktion von Anwendungen (= mehrere Anwendungen möglich)
  - Konzeptuelle Sicht auf die Datenbanken

- Basis für Abstraktion: Datenmodell

Datenmodell

- DB: Sammlung von Daten
- DB-Struktur/Schema: Elemente zur Definition, welche Daten möglich sind
  - Datentypen (z.B. String, Integer, ...)
  - Beziehungen (z.B. „Jeder Mitarbeiter hat einen Vorgesetzten.“)
  - Einschränkungen (z.B. Das Geburtsdatum muss in der Vergangenheit liegen)

- Datenmodell
  - Elemente zur Definition einer Datenbankstruktur
  - Basis-Operationen zur Abfrage und Änderung von Daten
  - Erweiterungen durch benutzerdefinierte Operationen

- Populäres Beispiel: das **relationale Datenmodell**

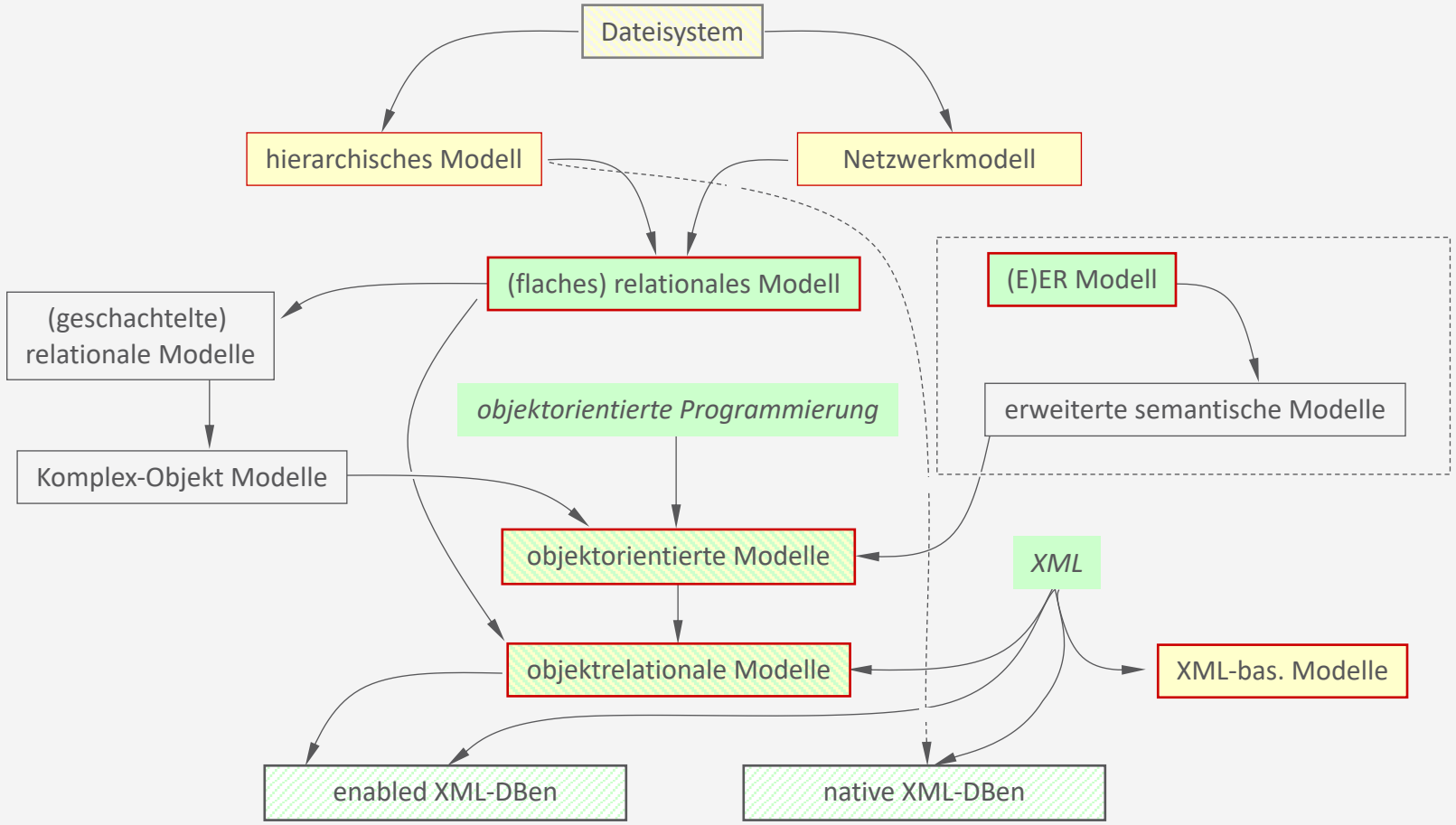
Modellierungsebenen von Datenmodellen

- Konzeptuelle Datenmodelle
  - Zur Definition der Miniwelt (→ Kundenanforderungen)
  - Beispiel: Entity-Relationship-Modell (ER-Modell)

- Logische Datenmodelle
  - Spezifikation, die leicht implementiert werden kann (→ für Entwickler, unabhängig vom DBMS)
  - Beispiel: Relationales Datenmodell

- Physische Datenmodelle
  - Spezifikation der konkreten Datenspeicherung (für ein konkretes DBMS)

Eine kurze Geschichte der Datenmodelle



Hierarchisches Modell

- Ältestes klassisches Datenmodell
- Ein Datensatz und alle von ihm abhängigen Datensätze → hierarchische Einheit

- Natürliche Hierarchie:
  - Unistrukturdatei (Universität -> Fakultät -> Institut -> Department)

- Künstliche Hierarchie:
  - Artikeldatei (Artikel -> Lieferant)

  könnte auch sein:
  - Artikeldatei (Lieferant -> Artikel)

Eigenschaften des Hierarchischen Modells

- Modellierung von Beziehungen:
  - 1 : 1 - Elternelement wird Kindelement zuordnet
  - 1 : n - Elternelement wird mehreren Kindelementen zugeordnet
  - m : n - Nicht direkt darstellbar
  - Keine Zyklen

- Zugriff
  - Entlang der Hierarchie sehr effizient
  - „Quer dazu“ sehr ineffizient
    - Beispiel: Teilnehmerdatei (Vorlesung -> Student)
      - Alle Studierenden der Vorlesung Datenbanken → prima
      - Alle Vorlesungen von Martha Mustermann → eeek …

Fazit:
- Gut bei klar definiertem Einsatz (z.B. directories, index-Verwaltung, …)
- Ineffizient bei allgemeinem Einsatz (wg. mangelnder Flexibilität)

Netzwerkmodell

- Schreibt das hierarchische Modell fort:
  - Ein Element kann mehreren Gruppen zugeordnet werden
    - Student in Vorlesung, in Studiengang, in Semester, …
  - Mehrere Wurzelelemente möglich
    - Jetzt auch n:m Beziehungen; allerdings nicht direkt:
      Kursbelegung (Student → Belegung → Kurs)

- Nachteile:
  - Wird leicht unübersichtlich
  - Für Abfragen ist genaue Kenntnis der Struktur nötig
  - Sequentielles Lesen („alle Studierenden einer Vorlesung“) wird ineffizienter

Das (flache) relationale Modell

- Für die Praxis das wichtigste Datenmodell!
- Tabellen (mit Attributen = Spaltenüberschriften) sind Container für komplexe Datenobjekte

Teilnehmer

MatrikelNr	Vorname	Nachname	Semester
4711	Martha	Mustermann	3
42	Arthur	Dent	21

- Beziehungen:
  - Durch Gruppierung in Tabellen
  - Durch wertebasierte Zusammenhänge zwischen Tabellen:

Vorlesung

VorlesungNr	Titel
42	Datenbanken
48151623	Altrömisches Abwasserecht

Belegung

VorlesungNr	MatrikelNr
42	4711
48151623	42

Schema / Instanz / DB-Zustand

- Schema (Intension)
  - Beschreibung der kompletten Struktur einer DB
  - Ändert sich (hoffentlich) selten
- Instanz (elementare Extension)
  - *Einzelne*, der vorgegebenen DB-Struktur entsprechende, aus konkreten Datenelementen bestehende Datensätze
- DB-Zustand
  (Gesamt-Extension oder Snapshot)
  - Die Gesamtheit der aktuell in einer DB gespeicherten Daten

Teilnehmer

MatrikelNr	Vorname	Nachname	Semester
4711	Martha	Mustermann	3

Vorlesung

VorlesungNr	Titel

Belegung

VorlesungNr	MatrikelNr

Physische Datenmodelle

- Physische Datenmodelle beschreiben konkret, wie die Daten anhand von Angaben zu
  - Datensatzformaten,
  - Datensatzanordnungen und
  - Zugriffspfaden

  physisch gespeichert werden (sollen)

- Ein Zugriffspfad ist eine Datenstruktur, welche die Suche nach Datensätzen in einer DB unterstützt/beschleunigt

- Beispiele: B-Bäume, B*-Bäume, R-Bäume, …

- Davon soll der*die Nutzer*in nichts mitbekommen!

Datenunabhängigkeit

- Ein Schema kann geändert werden, ohne zwangsläufig auch auf der nächsthöheren Ebene Änderungen vornehmen zu müssen.

- Logische Datenunabhängigkeit
  - Ein konzeptuelles/logisches Schema ändern, ohne immer auch externe Schemata oder Applikationen ändern zu müssen.

Teilnehmer

MatrikelNr	Vorname	Nachname	Semester

- Physische Datenunabhängigkeit
  - Ein internes Schema ändern, ohne konzeptuelle/ logische und externe Schemata sowie Applikationen ändern zu müssen.

Beispiele für Änderungen

- Logische Datenunabhängigkeit
  - Ein konzeptuelles/logisches Schema ändern
  - Hinzufügen von Attributen und Tabellen zum konzeptionellen Schema
  - Verändern der Tabellenstruktur
  - DB-Erweiterung durch neue Datensatztypen/Datenfelder
  - DB-Reduktion/Streichung bestehender Datensatztypen oder Datenfelder
  - Erweiterung („Verschärfung“) oder Reduktion („Entschärfung“) von Einschränkungen der Schemata

  →Wirkt sich nur auf externe Schemata aus, die sie nutzen

Beispiele für Änderungen

- Physische Datenunabhängigkeit
  - Ein internes Schema ändern
  - Veränderung des Speicherortes
  - Änderung des Speicherformates
  - Anlegen/Löschen von Indizes (für Anfrageoptimierung)

  →Verbleiben die gleichen Daten in der DB, so muss das konzeptuelle/logische DB-Schema i.d.R. nicht angepasst werden.

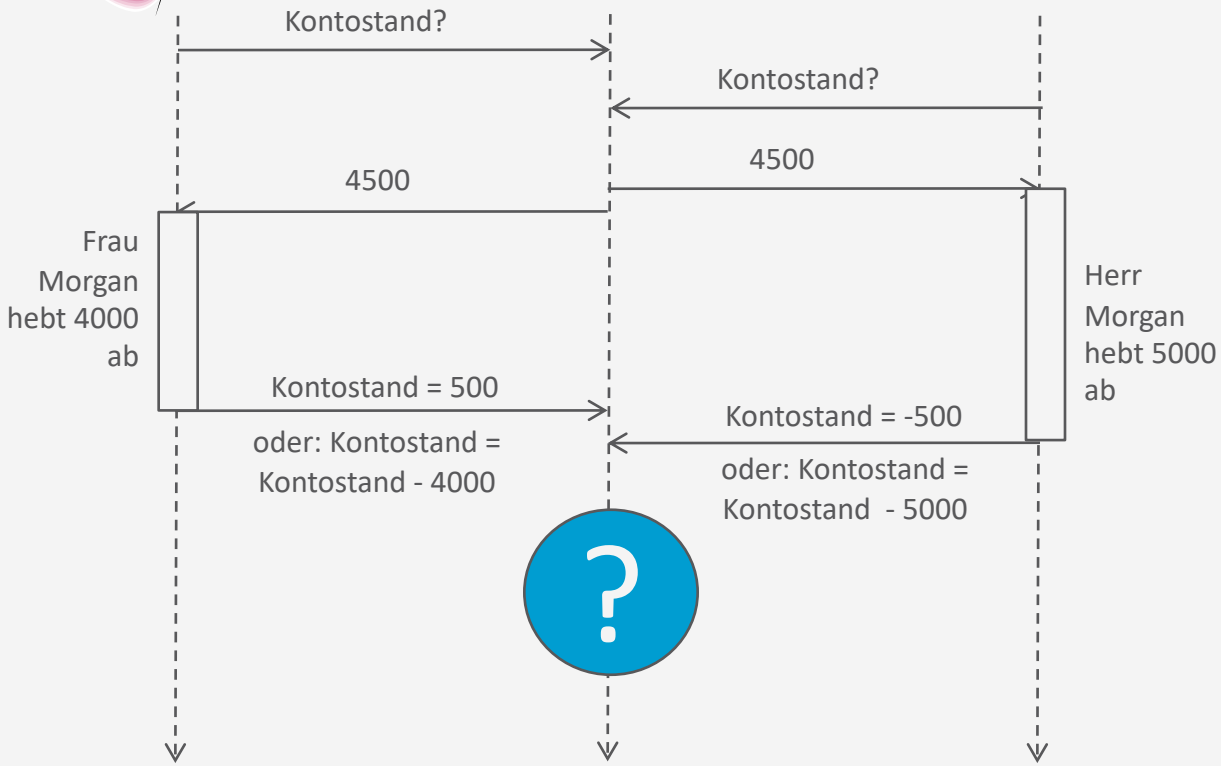
Viele gleichzeitige Benutzer...

- Jede*r Nutzer*in
  - Stellt Anfrage an den Kontostand
  - Verändert den Kontostand

  → Abfolge von DB-Befehlen = Transaktion

Welche Anforderungen ergeben sich?

Was sollte bei einem Systemabsturz passieren?

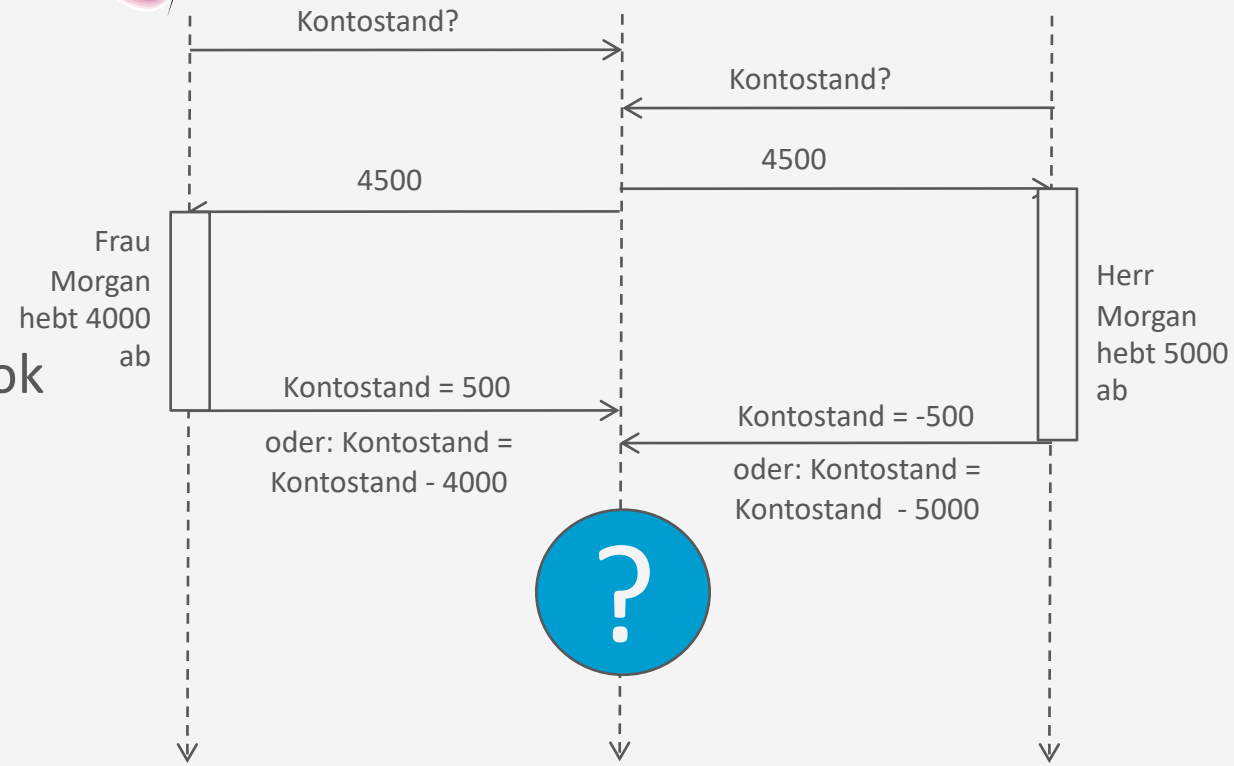


Viele gleichzeitige Benutzer...

- Jede*r Nutzer*in
  - Stellt Anfrage an den Kontostand
  - Verändert den Kontostand

  → Abfolge von DB-Befehlen = Transaktion
- Anforderungen:
  - **A**tomicity (Atomarität): Alles oder nichts
  - **C**onsistency (Konsistenz): Vorher ok, hinterher ok
  - **I**solation (Isolation): Jede*r denkt, er sei alleine auf der DB
  - **D**urability (Dauerhaftigkeit): Transaktionen bestätigt? Dann sind die Daten jetzt sicher

ACID-Eigenschaften (später mehr)



Zwischenzusammenfassung

- Datenbank
  - Persistente Speicherung großer Datenmengen
  - Logisch zusammenhängende Sammlung von Daten mit inhärenter Bedeutung
- Datenabstraktion
  - Datenmodelle: konzeptuell, logisch und physisch
  - Schema (Intension), Instanz und DB-Zustand (Extension)
  - Datenunabhängigkeit
    - Möglichkeit, ein Schema auf einer Ebene zu ändern unabhängig von anderen Ebenen
    - Genauere Betrachtung: logische und physische Datenunabhängigkeit
- Mehrbenutzung
  - Transaktionen als eine Abfolge von Datenbankbefehlen
  - Anforderungen: ACID (atomicity, consistency, isolation, durability)

Überblick: 1. Einführung

A. Datenbanken
- Datenbank (DB)
- Datenabstraktion, Datenmodelle, Datenunabhängigkeit
- Mehrbenutzersysteme

B. DB-Umgebungen
- DB-Sprachen
- Datenbanksystem (DBS)
- Datenbankmanagementsystem (DBMS)

C. Phasen des DB-Entwurfs
- Anforderung, Modellierung

DB-Sprachen

- Definition von DBs:
  - View Definition Languages (VDLs): extern
  - Data Definition Languages (DDLs): logisch
  - Storage Definition Languages (SDLs): intern

- Zugriff auf DBs
  (Einfügen, Ändern, Löschen und Anfragen von Datensätzen):
  - Data Manipulation Languages (DMLs)
    - Einfüge-, Änderungs- und Löschoperationen: Updates
    - Reine Anfragen: „Queries“
    - Alle Zugriffsarten: „Manipulation“
  - Data Control Languages (DCLs)

SQL : Structured Query Language

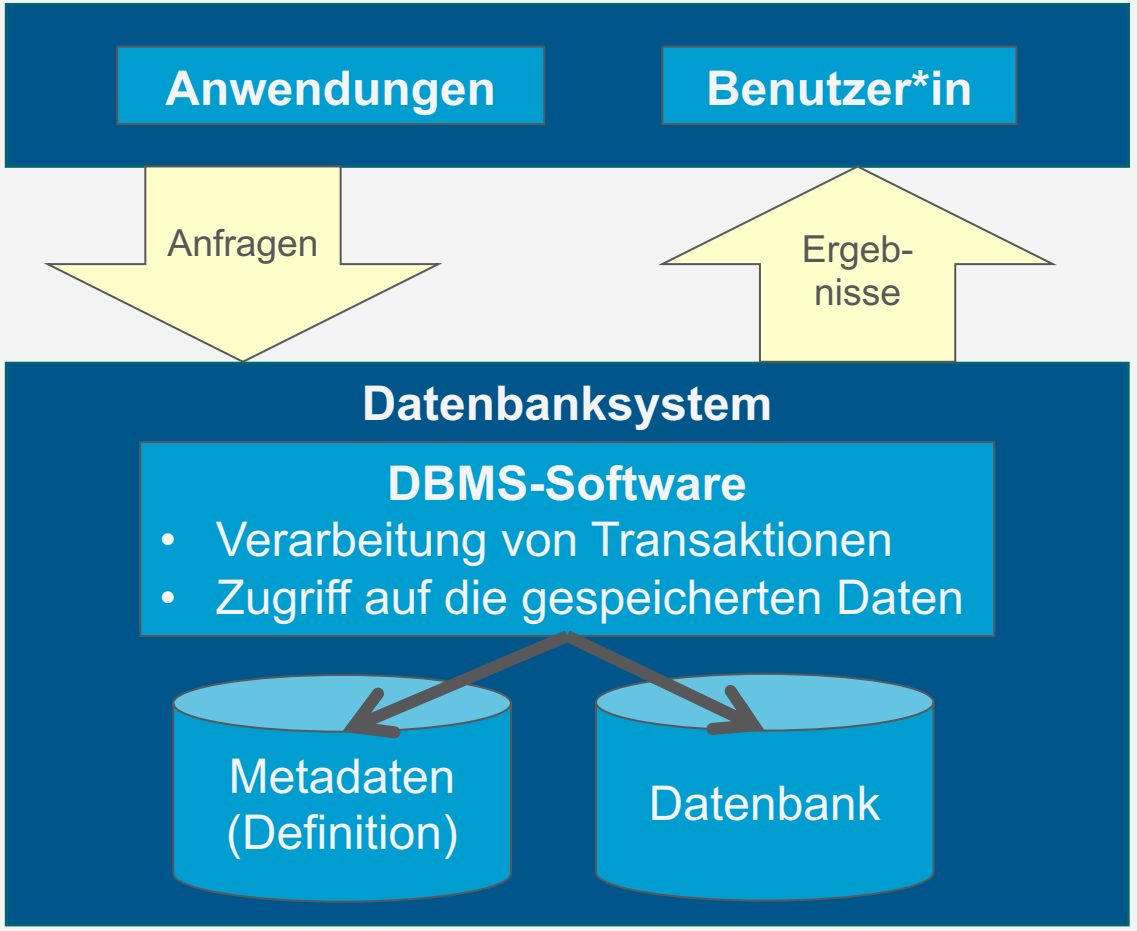
- Universal-Sprache für Datenbanken
  - VDL, DDL, SDL und DML in einem
- Haupteigenschaften:
  - Mengenorientiertes Arbeiten
    - Adressierung einer Menge von Datensätzen (zurückgegeben, geändert, gelöscht)
    - Menge kann auch nur ein Element enthalten
  - Deklarativ
    - Angabe darüber, welche Daten man möchte
    - Keine Angabe darüber wie der Zugriff erfolgen soll (Abstraktion; → Optimierungsmöglichkeit für das DBMS!)

```
CREATE TABLE Student (
  Name          VARCHAR2(100) NOT NULL,
  StudentNumber NUMBER(10)    PRIMARY KEY,
  Class         NUMBER(2)     DEFAULT 1 NOT NULL,
  Major         VARCHAR2(10)
);

INSERT INTO Student (Name, StudentNumber, Class, Major)
VALUES ('Smith', 17, 1, 'CS');

INSERT INTO Student (Name, StudentNumber, Class, Major)
VALUES ('Brown', 8, 2, 'CS');

SELECT Name
FROM Student
WHERE Major = 'CS';
```

SQL ist ein Standard. Konkrete Implementierungen von SQL sind z.B. MySQL oder PostgreSQL, die voneinander abweichen können.

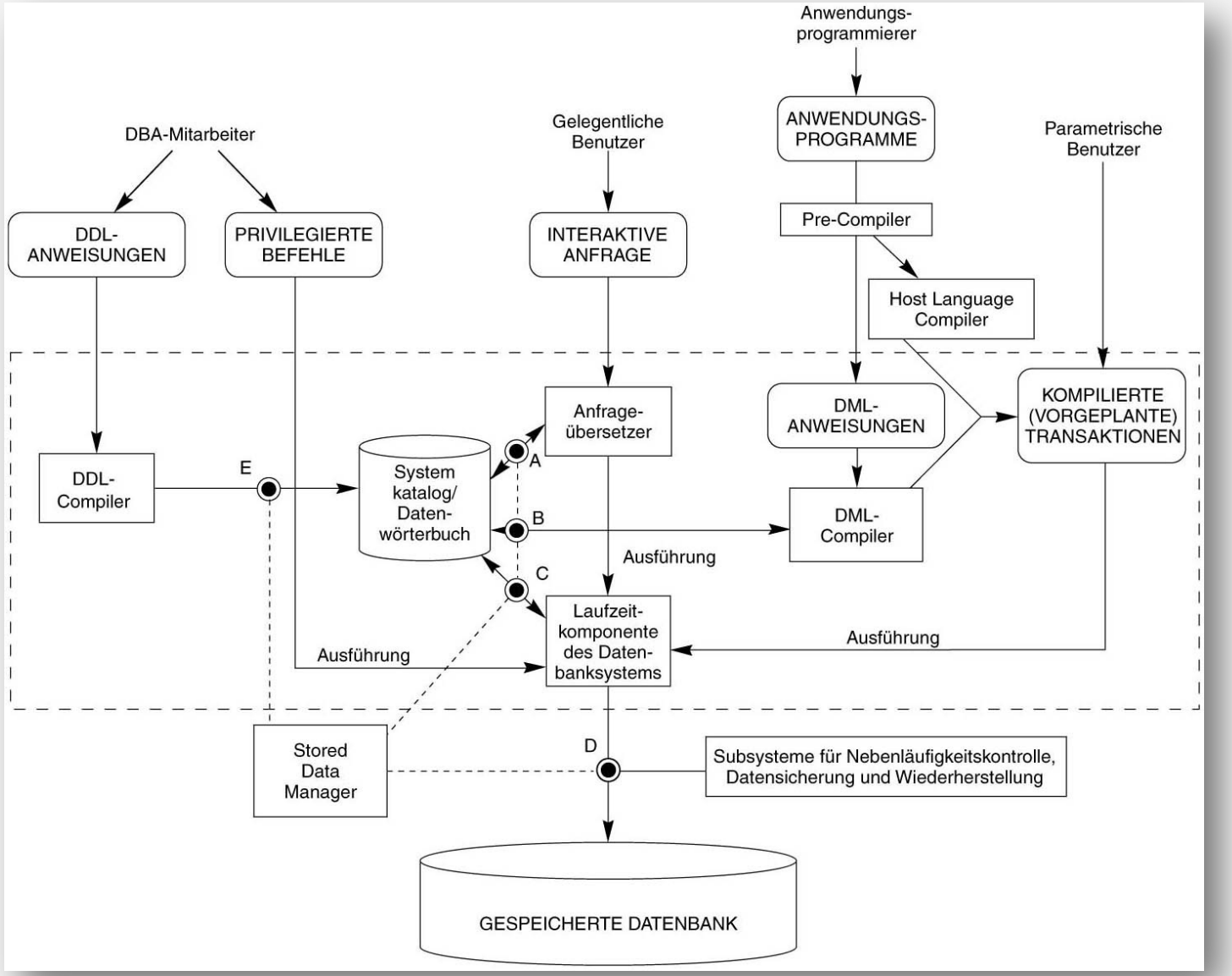
Datenbankmanagementsystem (DBMS) & Datenbanksystem (DBS)

- DBMS: Software-System, um DBs zu verwalten
  - Was muss es leisten?
    - Interfaces für Nutzung der DB
      - DB definieren, Daten ablegen + verändern, Anfragen
    - Anfrageverarbeitung / Datenmanipulationen
    - Unter Anforderungen an Korrektheit (ACID), Effizienz (Nutzbarkeit)
      - Auch bei Mehrbenutzersystemen
    - Weiteres Funktionen: Sicherheitsmechanismen, Fehlerbehandlung, Integritätskonzepte, verschiedene Sichten auf die DB, Optimierung von Anfragen
- DBS = DB + DBMS



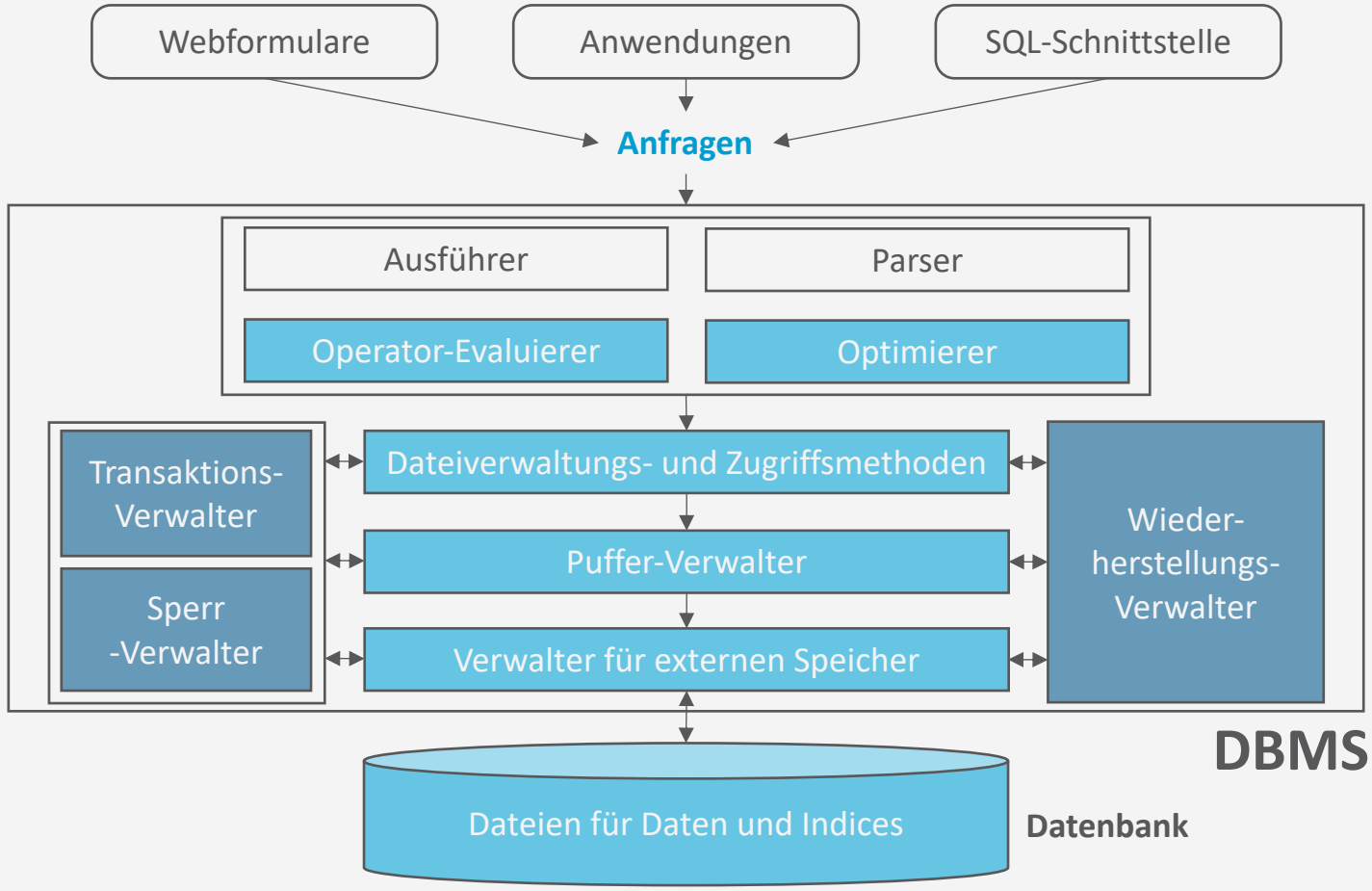
Erweiterte Systemumgebung

- Zugriffe von
  - DB-Administratoren
  - Anwendungen
  - Nutzer*innen
    - Direkt auf DB
      - Gelegentlich, interaktiv
    - Parametrisch
      - Vorgefertigte Anwendungsprogramme mit beschränktem Kommandovorrat (routinierte, wohldefinierte, formalisierte Befehle)



Architektur eines DBMS

- Ausblick: Hinter den Kulissen
  - Teil von 4. Relational Datenmodell – Relationale Algebra und 5. SQL
    - Anfragen stellen
    - Daten ändern
  - Teil von 6. Anfrageverarbeitung
    - Speicherung und Verwaltung der DB
    - Effiziente Umsetzung der SQL Befehle im DBMS
  - Teil von 7. Transaktionen
    - ACID-Umsetzung



Zwischenzusammenfassung

- DB-Sprachen
  - VDL, DDL, SDL und DML
  - SQL als Universalsprache
- DB-Systemumgebung
  - Manipulationen von Administrator*innen, Programmen, Nutzer*innen (gelegentlich, parametrisch)
  - DBMS
    - Anfrage: Ausführer, Parser, Operator-Evaluierer, Optimierer
    - Daten: Dateien, Verwalter für externen Speicher, Puffer, Dateiverwaltung, Zugriffsmethoden
    - (Mehr)benutzung: Transaktionen-, Sperr-, Wiederherstellungs-Verwalter
  - DBS = DBMS + DB

Überblick: 1. Einführung

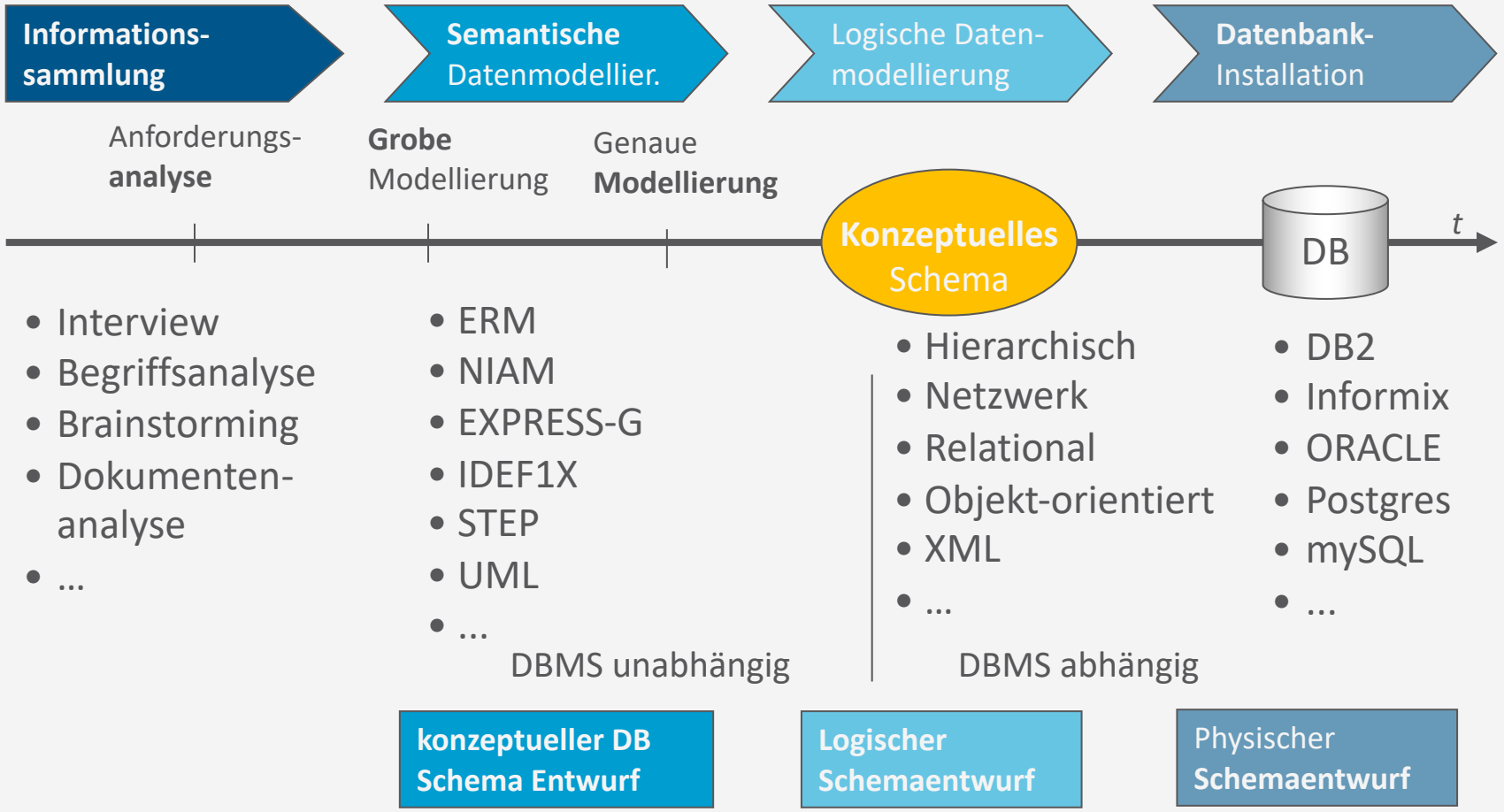
A. Datenbanken
- Datenbank (DB)
- Datenabstraktion, Datenmodelle, Datenunabhängigkeit
- Mehrbenutzersysteme

B. DB-Umgebungen
- DB-Sprachen
- Datenbanksystem (DBS)
- Datenbankmanagementsystem (DBMS)

C. Phasen des DB-Entwurfs
- Anforderung, Modellierung

Phasen des DB-Entwurfs

- Ausblick: Von der Anwendung her
  - Teil von 2. DB-Modellierung
    - Methode: ERM
  - Teil von 3. Das relationale Datenmodell
    - Methode: relationale Modellierung
  - Teil von 4. DB-Entwurf
  - Teil von 5. SQL & Übergang zu „Hinter den Kulissen“



Zwischenzusammenfassung

- Phasen des DB-Entwurfs: Der Weg von den Anforderungen zur DB-Anwendung
- Von der Anwendung her
  - Informationssammlung
  - Semantische Modellierung
  - Logische Modellierung
  - Datenbankinstallation
    - Übergang zu hinter den Kulissen

Inhalte: Datenbanken (DBs)

1. **Einführung**
   - Anwendungen
   - Datenbankmanagementsysteme
2. **Datenbank-Modellierung**
   - Entity-Relationship-Modell (ER-Modell)
   - Beziehung zwischen ER und UML
3. **Das relationale Modell**
   - Relationales Datenmodell (RM)
   - Vom ER-Modell zum RM
   - Relationale Algebra als Anfragesprache
4. **Datenbank-Entwurf**
   - Funktionale Abhängigkeiten
   - Normalformen
5. **Structured Query Language (SQL)**
   - Datendefinition
   - Datenmanipulation
6. **Anfrageverarbeitung**
   - Architektur
   - Indexierung
   - Anfragepläne, Optimierung
7. **Transaktionen**
   - Transaktionsverarbeitung, Schedules, Sperren
   - Wiederherstellung
8. **Verteilte Datenbanken**
   - Fragmentierung, Replikation, Allokation; CAP
   - Anfragebeantwortung, föderierte Systeme

Überblick: 1. Einführung

A. *Datenbanken*
- Datenbank (DB)
- Datenabstraktion, Datenmodelle, Datenunabhängigkeit
- Mehrbenutzersysteme

B. *DB-Umgebungen*
- DB-Sprachen
- Datenbanksystem (DBS)
- Datenbankmanagementsystem (DBMS)

C. *Phasen des DB-Entwurfs*
- Anforderung, Modellierung

→ Datenbank-Modellierung