# Datenbankanwendung

Wintersemester 2014/15

Prof. Dr.-Ing. Sebastian Michel
TU Kaiserslautern

smichel@cs.uni-kl.de

# Vorlesung Datenbankanwendung

# Übersicht

Planung

## Übersicht, Motivation, Wiederholung

- Motivation
- Organisation der VL und Regeln zum Übungsbetrieb
- Wiederholung aus Infosys (Tabellen, relationale Algebra, SQL)

## Entwurfstheorie

- Konzeptioneller Entwurf
- Normalformenlehre
- Synthese von Relationen

# Übersicht

## Anfrageverarbeitung

- Implementierung von Operatoren (Joins, externes Sortieren, ...)
- Kostenschätzung
- Histogramme, Sketches

## Indexstrukturen

- Wiederholung B+ Baum und Varianten
- Multidimensionale Indexstrukturen
- Hashing
- Ähnlichkeitssuche

# Übersicht

## Anfrageoptimierung

- Prinzip und Regelbasierte Optimierung (Wiederholung Infosys)
- Dynamische Programmierung, Greedy Algorithmen

## Weitere SQL Konzepte, JDBC, PL/pgSQL, Trigger

- SQL Fensteranfragen und Rekursion
- JDBC
- Trigger und PL/pgSQL

# Übersicht

## Transaktionsverwaltung

- Transaktionskonzept, Ablauf von Transaktionen
- Commit-Protokolle

## Serialisierbarkeit

- Anomalien im Mehrbenutzerbetrieb
- Theorie der Serialisierbarkeit
- Klassen von Historien

## Synchronisation und Sperrverfahren

- Sperrprotokolle
- Nicht sperrende Protokolle
- Zweiphasen-Sperrprotokolle

# Übersicht

## Logging und Recovery

- Fehlermodelle und Recovery-Arten
- Logging-Strategien
- ...

## (Materialisierte) Sichten

- Sichten
- Materialisierte Sichten und Updates

## Datenschutz und Zugriffskontrolle

- Technische Probleme des Datenschutzes
- Konzepte der Zugriffskontrolle, Zugriffskontrolle in SQL
- Sicherheitsprobleme in statischen Datenbanken

# Literaturliste


- A. Kemper und A. Eickler. Datenbanksysteme. Oldenbourg-Verlag, 2012.
- T. Härder und E. Rahm. Datenbanksysteme. Springer. 2011.
- H. Samet. Foundations of Multidimensional and Metric Data Structures. Morgan Kaufmann Publishers. 2006.
- G. Weikum und G. Vossen. Transactional Information Systems. Morgan Kaufmann Publishers. 2002.

# Organisatorisches

## Vorlesung

- 4 SWS Vorlesung: Wöchentlich, Dienstags und Freitags.
  - Dienstags, Raum 42-110, 11:45 bis 13:15 Uhr
  - Freitags, Raum 46-210, 11:45 bis 13:15 Uhr
- 2 SWS Übung: Wöchentlich, Donnerstags, 15:30 bis 17:00. Raum 13-222. **Übungsgruppenleiter Johannes Schildgen**.

## Klausur

- Abschlussklausur am 20. März 2015 (Stand der Planung)
- Für die Zulassung zur Klausur bedarf es einer erfolgreichen Teilnahme am Übungsbetrieb.

# Regeln zum Übungsbetrieb

- Es gibt 13 Übungsblätter, die in der wöchentlich stattfindenden Übung besprochen werden.
- Auf diesen Übungsblättern gibt es 40 Aufgaben, die Sie in Vorbereitung auf die Übung lösen müssen.
- Auf diese 40 Aufgaben gibt es jeweils einen Punkt.
- Sie müssen insgesamt 20 Punkte erreichen, um für die Klausur zugelassen zu werden. Die Punktevergabe funktioniert wie folgt:
- Zu Beginn der Übung markieren Sie auf einer Teilnehmerliste, welche der verbindlichen Aufgaben Sie gelöst haben und in der Übungsstunde präsentieren können.
- In der Übungsstunde wählt der Übungsleiter unter allen markierten Lösungen einen oder mehrere Studierende aus, die entsprechende Aufgabe zu präsentieren. Die Auswahl erfolgt auf Gutdünken des Übungsgruppenleiters.

# Regeln zum Übungsbetrieb (2)

- Falls eine Aufgabe durch einen Studierenden als gelöst markiert wird, dieser bei der Aufforderung zur Präsentation passt oder die Präsentation der Aufgabe zeigt, dass sich der Studierende nicht ausreichend mit dem Thema befasst hat, erhält der Studierende 3 Minuspunkte.
- Ist die Lösung korrekt oder hat nur leichte Mängel, ist aber auf dem richtigen Weg zur korrekten Lösung, gibt es den vollen Punkt.
- Die Vergabe der Punkte erfolgt durch subjektive Einschätzung des Übungsleiters und muss nicht begründet werden.

# Regeln zum Übungsbetrieb (3)

- Gruppenarbeit generell sinnvoll. Aber: Trotzdem ist die "Abgabe" der Lösungen nicht als Gruppe möglich, sondern pro Studierendem. Jeder Teilnehmer einer Gruppe muss in der Lage sein, die als gelöst markierten Aufgaben zu präsentieren.
- Neben den 40 verbindlichen Aufgaben gibt es weitere Aufgaben, deren Bearbeitung freiwillig ist. Wir wünschen und natürlich trotzdem rege Beteiligung bei der Präsentation der Lösungen. **Alle Aufgaben sind verbindlich sofern nicht als freiwillig markiert.**
- Für Aufgaben bei denen nur ein oder zwei Studierende eine Lösung markiert haben, gibt es zwei zusätzliche Bonuspunkte.

# Vorlesungsfolien und Übungsblätter

## Vorlesungsfolien

- Werden in der Regel einen Tag vor der Vorlesung online gestellt.
- Werden nach der Vorlesung evtl. aktualisiert (Typos, Anmerkungen, ...) .

## Übungsblätter

- Ausgabe Dienstags in der Vorlesung.
- Besprechung in Übung der folgenden Woche (auf Übungsblatt angegeben)

# Wiederholung Infosys: So ging es los ...

## Abbilden eines Teilaspekts der realen Welt

- Was wollen wir Abbilden?
- Welcher Grad an Details?
- Welche Entitäten spielen eine Rolle?
- Und welche Rolle spielen sie?
- Wie sind Entitäten miteinander verbunden?

# Anforderungsalyse

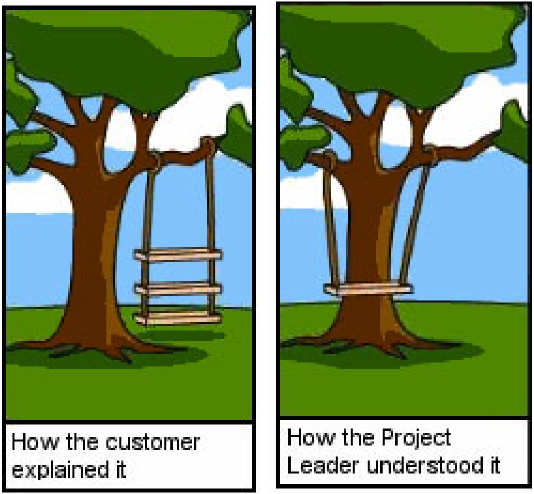
## Reale Welt: Universität

→ Anforderungsanalyse

## Pflichtenheft

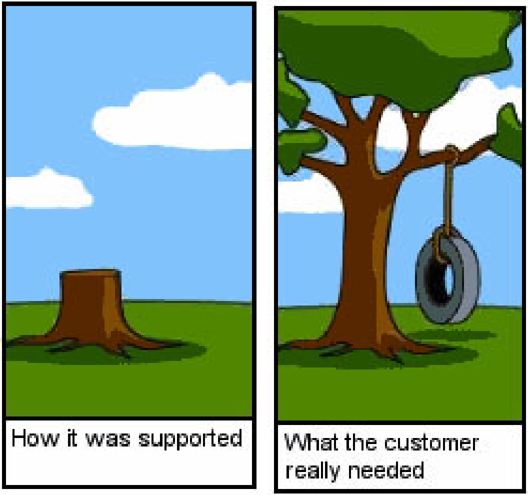
- „Studenten hören Vorlesungen”
- „Professoren halten Vorlesungen”
- „Studenten könnten anhand ihrer Matrikelnummer eindeutig identifiziert werden”
- ...

# Entity/Relationship-Modellierung




Quelle: www.fuki.ch

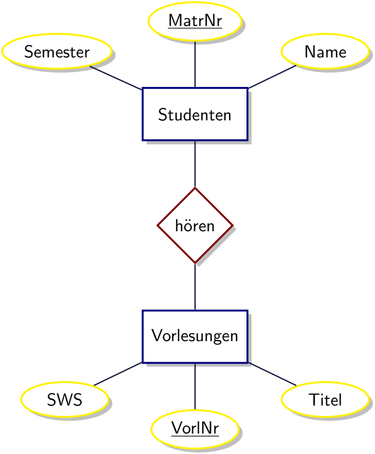
# Entity/Relationship-Modellierung



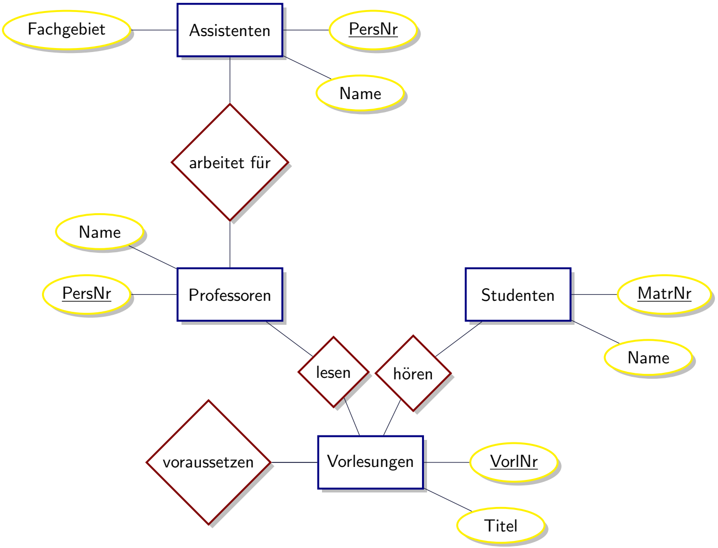

Quelle: www.fuki.ch

# Entity/Relationship-Modellierung

1. Entität → Entitätstyp
2. Beziehung → Beziehungstyp
3. Attribut (Eigenschaft)
4. Schlüssel (Identifikation)
5. Rolle



# Beispiel: Universitätsschema



# Relationales Modell

Abbilden des ER-Modells in Relationen:

Zum Beispiel:

- **Kunden**(ID, Telefon, Adresse, Name)
- **Kaufen**(Wert, Datum, Preis, Verkäufer, Auto, Kunde)
- **Verkaufen**(Datum, Wert, Kommission, Kunde, Verkäufer, Auto)
- ...

oder ...

- **Studenten** (MatrNr, Name)
- **Hören** (MatrNr, VorlNr)
- **Vorlesungen** (VorlNr, Titel)

# Beispiel: Die relationale Uni-DB

**Professoren**

| PersNr | Name | Rang | Raum |
|---|---|---|---|
| 2125 | Sokrates | C4 | 226 |
| 2126 | Russel | C4 | 232 |
| 2127 | Kopernikus | C3 | 310 |
| 2133 | Popper | C3 | 52 |
| 2134 | Augustinus | C3 | 309 |
| 2136 | Curie | C4 | 36 |
| 2137 | Kant | C4 | 7 |

**Studenten**

| MatrNr | Name | Semester |
|---|---|---|
| 24002 | Xenokrates | 18 |
| 25403 | Jonas | 12 |
| 26120 | Fichte | 10 |
| 26830 | Aristoxenos | 8 |
| 27550 | Schopenhauer | 6 |
| 28106 | Carnap | 3 |
| 29120 | Theophrastos | 2 |
| 29555 | Feuerbach | 2 |

**Vorlesungen**

| VorlNr | Titel | SWS | gelesen von |
|---|---|---|---|
| 5001 | Grundzüge | 4 | 2137 |
| 5041 | Ethik | 4 | 2125 |
| 5043 | Erkenntnistheorie | 3 | 2126 |
| 5049 | Mäeutik | 2 | 2125 |
| 4052 | Logik | 4 | 2125 |
| 5052 | Wissenschaftstheorie | 3 | 2126 |
| 5216 | Bioethik | 2 | 2126 |
| 5259 | Der Wiener Kreis | 2 | 2133 |
| 5022 | Glaube und Wissen | 2 | 2134 |
| 4630 | Die 3 Kritiken | 4 | 2137 |

**voraussetzen**

| Vorgänger | Nachfolger |
|---|---|
| 5001 | 5041 |
| 5001 | 5043 |
| 5001 | 5049 |
| 5041 | 5216 |
| 5043 | 5052 |
| 5041 | 5052 |
| 5052 | 5259 |

**hören**

| MatrNr | VorlNr |
|---|---|
| 26120 | 5001 |
| 27550 | 5001 |
| 27550 | 4052 |
| 28106 | 5041 |
| 28106 | 5052 |
| 28106 | 5216 |
| 28106 | 5259 |
| 29120 | 5001 |
| 29120 | 5041 |
| 29120 | 5049 |
| 29555 | 5022 |
| 25403 | 5022 |

**Assistenten**

| PerslNr | Name | Fachgebiet | Boss |
|---|---|---|---|
| 3002 | Platon | Ideenlehre | 2125 |
| 3003 | Aristoteles | Syllogistik | 2125 |
| 3004 | Wittgenstein | Sprachtheorie | 2126 |
| 3005 | Rhetikus | Planetenbewegung | 2127 |
| 3006 | Newton | Keplersche Gesetze | 2127 |
| 3007 | Spinoza | Gott und Natur | 2126 |

**prüfen**

| MatrNr | VorlNr | PersNr | Note |
|---|---|---|---|
| 28106 | 5001 | 2126 | 1 |
| 25403 | 5041 | 2125 | 2 |
| 27550 | 4630 | 2137 | 2 |

# Die Operatoren der relationalen Algebra

- Selektion $\sigma$
- Projektion $\pi$
- Kreuzprodukt $\times$
- Join (Verbund) $\bowtie$
- Umbenennung $\rho$
- Differenz $-$
- Division $\div$

- Vereinigung $\cup$
- Schnitt $\cap$
- Semi-Join (linker) $\ltimes$
- Semi-Join (rechter) $\rtimes$
- linker äußerer Join ⟕
- rechter äußerer Join ⟖
- äußerer Join ⟗

# Grundlagen der relationalen Algebra

Achtung:

Es gibt Relationen (Ausprägungen: $R \subseteq D_1 \times D_2 \times ... \times D_n$)
und Relationenschema:{[ attributname1: datentyp1, attributname2: datetyp2, ...]}.

Das Schema einer Relation wird auch mit $sch(R)$ oder $\mathcal{R}$ beschrieben.

Die Operatoren und ihre Verwendung:

- Eingabe eines Operators ist eine oder mehrere Relationen.
- Ausgabe ist auch wieder eine Relation.
- D.h., Operatoren sind kombinierbar (mit gewissen Regeln).

# Z.B. Selektion

**Selektion**

$\sigma_{Semester>10}(Studenten)$

| $\sigma_{Semester>10}(Studenten)$ | | |
|---|---|---|
| MatrNr | Name | Semester |
| 24002 | Xenokrates | 18 |
| 25403 | Jonas | 12 |

- Allgemein: $\sigma_F$
- Selektionsprädikat $F$ besteht aus
  - Operatoren: $\vee$ (oder) $\wedge$ (und) $\neg$ (nicht)
  - Arithmetischen Vergleichsoperatoren: $<, \leq, =, >, \geq, \neq$
  - ... und natürlich aus Attributnamen der Argumentrelation oder Konstanten als Operanden

# Kartesisches Produkt (Kreuzprodukt)

Gegeben zwei Relationen $R$ und $S$:

$$R \times S$$

beinhaltet **ALLE!** $|R| * |S|$ möglichen Paare von Tupeln aus $R$ und $S$.

Enthält viele (oft auch viele unsinnige) Kombinationen!

Das resultierende Schema $sch(R \times S) = sch(R) \cup sch(S) = \mathcal{R} \cup \mathcal{S}$.

Beim Referenzieren der Attribute des resultierenden Schemas wird R.X und S.Y verwendet, insbesondere bei Überlappungen der einzelnen Schemata (=keine Unklarheiten was gemeint ist).

# Der natürliche Verbund (Join)

Gegeben zwei Relationen ( + Schemata)

- $R(A_1, ..., A_m, \mathbf{B_1}, ..., \mathbf{B_k})$
- $S(\mathbf{B_1}, ..., \mathbf{B_k}, C_1, ..., C_n)$

$$R \bowtie S = \pi_{A_1,...,A_m,R.B_1,...,R.B_k,C_1,...,C_n}(\sigma_{R.B_1=S.B_1 \wedge ... \wedge R.B_k=S.B_k}(R \times S))$$

| $R \bowtie S$ | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $\mathcal{R} - \mathcal{S}$ | | | | $\mathcal{R} \cap \mathcal{S}$ | | | | $\mathcal{S} - \mathcal{R}$ | | | |
| $A_1$ | $A_2$ | ... | $A_m$ | $B_1$ | $B_2$ | ... | $B_k$ | $C_1$ | $C_2$ | ... | $C_n$ |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |

# Mehrere Joins zusammen

- Es können beliebig viele Relationen durch Join Operatoren verknüpft werden.
- Die Reihenfolge spielt dabei keine Rolle (Joins sind kommutativ und assoziativ).

$$(Studenten \bowtie hoeren) \bowtie Vorlesungen$$

| $(Studenten \bowtie hoeren) \bowtie Vorlesungen$ | | | | | | |
|---|---|---|---|---|---|---|
| MatrNr | Name | Semester | VorlNr | Titel | SWS | gelesenVon |
| 26120 | Fichte | 10 | 5001 | Grundzüge | 4 | 2137 |
| 27550 | Jonas | 12 | 5022 | Glaube und Wissen | 2 | 2134 |
| 28106 | Carnap | 3 | 4052 | Wissenschaftstheorie | 3 | 2126 |
| ... | ... | ... | ... | ... | ... | ... |

# Allgemeiner Join (Theta-Join)

Gegeben zwei Relationen ( + Schemata)

- $R(A_1, ..., A_n)$
- $S(B_1, ..., B_m)$

$\theta$ ist ein beliebiges Prädikat über den beteiligten Attributen.

$$R \bowtie_\theta S = \sigma_\theta(R \times S)$$

| $R \bowtie S$ | | | | | | | |
|---|---|---|---|---|---|---|---|
| $\mathcal{R}$ | | | | $\mathcal{S}$ | | | |
| $A_1$ | $A_2$ | ... | $A_n$ | $B_1$ | $B_2$ | ... | $B_m$ |
| ... | ... | ... | ... | ... | ... | ... | ... |

Equi-Join: Join-Prädikat $\theta$ darf nur auf Gleichheit ($=$) prüfen.

# Weitere Join Typen: Äußere Joins

Diese Join Typen spezifizieren wie mit Tupeln umgegangen wird, die keinen Joinpartner gefunden haben.

- linker äußerer Join ($⟕$): auch “partnerlose” Tupel der linken Relation bleiben erhalten
- rechter äußerer Join ($⟖$): auch “partnerlose” Tupel der rechten Relation bleiben erhalten
- vollständiger äußerer Join ($⟗$): die “partnerlosen” Tupel beider Relationen bleiben erhalten

# Natürlicher Join und linker äußerer Join

## Natürlicher Join

| L | | |
|---|---|---|
| A | B | C |
| $a_1$ | $b_1$ | $c_1$ |
| $a_2$ | $b_2$ | $c_2$ |

$\bowtie$

| R | | |
|---|---|---|
| C | D | E |
| $c_1$ | $d_1$ | $e_1$ |
| $c_3$ | $d_2$ | $e_2$ |

$=$

| Resultat | | | | |
|---|---|---|---|---|
| A | B | C | D | E |
| $a_1$ | $b_1$ | $c_1$ | $d_1$ | $e_1$ |

## Linker äußerer Join

| L | | |
|---|---|---|
| A | B | C |
| $a_1$ | $b_1$ | $c_1$ |
| $a_2$ | $b_2$ | $c_2$ |

$⟕$

| R | | |
|---|---|---|
| C | D | E |
| $c_1$ | $d_1$ | $e_1$ |
| $c_3$ | $d_2$ | $e_2$ |

$=$

| Resultat | | | | |
|---|---|---|---|---|
| A | B | C | D | E |
| $a_1$ | $b_1$ | $c_1$ | $d_1$ | $e_1$ |
| $a_2$ | $b_2$ | $c_2$ | — | — |

# Rechter äußerer und äußerer Join

## Rechter äußerer Join

| **L** | | |
|---|---|---|
| A | B | C |
| $a_1$ | $b_1$ | $c_1$ |
| $a_2$ | $b_2$ | $c_2$ |

$⟖$

| **R** | | |
|---|---|---|
| C | D | E |
| $c_1$ | $d_1$ | $e_1$ |
| $c_3$ | $d_2$ | $e_2$ |

$=$

| **Resultat** | | | | |
|---|---|---|---|---|
| A | B | C | D | E |
| $a_1$ | $b_1$ | $c_1$ | $d_1$ | $e_1$ |
| — | — | $c_3$ | $d_2$ | $e_2$ |

## Äußerer Join

| **L** | | |
|---|---|---|
| A | B | C |
| $a_1$ | $b_1$ | $c_1$ |
| $a_2$ | $b_2$ | $c_2$ |

$⟗$

| **R** | | |
|---|---|---|
| C | D | E |
| $c_1$ | $d_1$ | $e_1$ |
| $c_3$ | $d_2$ | $e_2$ |

$=$

| **Resultat** | | | | |
|---|---|---|---|---|
| A | B | C | D | E |
| $a_1$ | $b_1$ | $c_1$ | $d_1$ | $e_1$ |
| $a_2$ | $b_2$ | $c_2$ | — | — |
| — | — | $c_3$ | $d_2$ | $e_2$ |

# Weitere Join Typen: Semi Joins

Idee: Finde alle Tupel der linken Relation die Joinpartner in der rechten Relation haben.

$$L \ltimes R = \pi_{\mathcal{L}}(L \bowtie R)$$

$L \rtimes R$ ist analog definiert.

Es gilt:

$$L \rtimes R = R \ltimes L$$

allerdings sind Semi-Joins sowie die linken und rechten äußeren Joins **nicht kommutativ!**

# Semi-Joins

## Semi-Join von $L$ mit $R$

| L | | |
|---|---|---|
| A | B | C |
| $a_1$ | $b_1$ | $c_1$ |
| $a_2$ | $b_2$ | $c_2$ |

$\ltimes$

| R | | |
|---|---|---|
| C | D | E |
| $c_1$ | $d_1$ | $e_1$ |
| $c_3$ | $d_2$ | $e_2$ |

$=$

| Resultat | | |
|---|---|---|
| A | B | C |
| $a_1$ | $b_1$ | $c_1$ |

## Semi-Join von $R$ und $L$

| L | | |
|---|---|---|
| A | B | C |
| $a_1$ | $b_1$ | $c_1$ |
| $a_2$ | $b_2$ | $c_2$ |

$\rtimes$

| R | | |
|---|---|---|
| C | D | E |
| $c_1$ | $d_1$ | $e_1$ |
| $c_3$ | $d_2$ | $e_2$ |

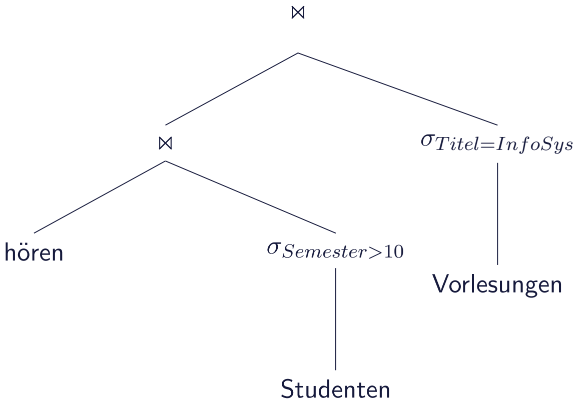
$=$

| Resultat | | |
|---|---|---|
| C | D | E |
| $c_1$ | $d_1$ | $e_1$ |

# Operatorbaum-Darstellung

Alternative Darstellung von Ausdrücken der relationalen Algebra. Gerade für größere Ausdrücke viel übersichtlicher.

Beispiel:



# SQL

- **deklarative** Anfragesprache ("was" nicht "wie")
- setzte sich aus drei Sprachen zusammen:
- Datendefinitions (DDL)-Sprache:
    - erstellt/ändert das Schema
    - create, alter, drop
- Datenmanipulations (DML)-Sprache
    - ändert Ausprägungen
    - insert, update, delete
- Anfragesprache:
    - berechnet Anfragen auf den Ausprägungen
    - select * from where ...

# Relationen vs. Tabellen

- bisher hatten wir Relationen (=Mengen) betrachtet
- in SQL betrachten wir aber Tabellen (=Bags)
- was ist der Unterschied?
- Relationen versus Tabellen:
  - Tabellen können Duplikate enthalten
  - Tabellen können eine Ordnung haben
  - Tabellen haben nicht unbedingt einen Schlüssel

# Einfache Datendefinitionen in SQL

## Datentypen

- **character** (n), **char** (n)
- **character varying** (n), **varchar** (n)
- **numeric** (p,s), **integer**
- **blob** oder **raw** für sehr große binäre Daten
- **clob** für sehr große String-Attribute
- **date** für Datumsangaben
- **xml** für XML-Dokumente

# Datendefinitions (DDL) Sprache

## Tabellen erstellen:

```
create table Professoren(
        PersNr    integer not null,
        Name      varchar (10) not null
        Rang      character (2) );
```

## Tabellen verändern:

```
alter table Professoren
      add (Raum integer);

alter table Professoren
      modify (Name varchar(30));
```

# Datendefinitions (DDL) Sprache

Tabellen löschen:

**drop table** Professoren;

# Datenmanipulationssprache: insert

```
insert into Studenten (MatrNr, Name)
            values (28121, 'Archimedes');
```

```
insert into Studenten (Name)
            values ('Meier');
```

```
insert into hören
            select MatrNr, VorlNr
            from Studenten, Vorlesungen
            where Titel = 'Logik';
```

# Anfragesprache

select <Liste von Spalten>
from <Liste von Tabellen>
where <Bedingung>;

**select *** wählt alle verfügbaren Spalten aus!

Relationale Algebra $\rightarrow$ SQL

- Projektion $\pi$ $\rightarrow$ select
- Kreuzprodukt $\times$ $\rightarrow$ from
- Selektion $\sigma$ $\rightarrow$ where

# Select from where ...

```
select PersNr, Name
from   Professoren
where  Rang = 'C4';
```

| Professoren | |
|---|---|
| **PersNr** | **Name** |
| 2125 | Sokrates |
| 2126 | Russel |
| 2136 | Curie |
| 2137 | Kant |

| Professoren | | | |
|---|---|---|---|
| **PersNr** | **Name** | **Rang** | **Raum** |
| 7 2125 | Sokrates | C4 | 226 |
| 2126 | Russel | C4 | 232 |
| 2127 | Kopernikus | C3 | 310 |
| 2133 | Popper | C3 | 52 |
| 2134 | Augustinus | C3 | 309 |
| 2136 | Curie | C4 | 36 |
| 2137 | Kant | C4 | 7 |

## Anmerkung:

SQL legt nicht fest in welcher Reihenfolge Selektion, Projektion oder Join ausgeführt werden.

# Anfragen über mehrere Relationen

Welcher Professor liest "Mäeutik"?

**select** Name, Titel
**from** Professoren, Vorlesungen
**where** PersNr = gelesenVon **and** Titel = 'Mäeutik'

In der relationalen Algebra sieht das wie folgt aus:

$$\pi_{Name,Titel}(\sigma_{PersNr=gelesenVon \wedge Titel=\text{'Mäeutik'}}(Professoren \times Vorlesungen))$$

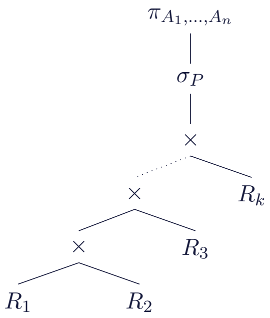
# Übersetzung von SQL in die relationale Algebra

Allgemeine Form einer (ungeschachtelten) SQL-Anfrage:

**select** $A_1$, ..., $A_n$
**from** $R_1$, ..., $R_k$
**where** $P$;

Übersetzung in die relationale Algebra:

$$\pi_{A_1,\ldots,A_n}(\sigma_P(R_1 \times \ldots \times R_k))$$



# Anfragen über mehrere Relationen

Welche Studenten hören welche Vorlesungen?

**select** Name, Titel
**from** Studenten, hören, Vorlesungen
**where** Studenten.MatrNr=hören.MatrNr **and**
hören.VorlNr = Vorlesungen.VorlNr;

Alternativ:

**select** **s**.Name, **v**.Titel
**from** Studenten **s**, hören **h**, Vorlesungen **v**
**where** **s**.MatrNr=**h**.MatrNr **and**
**h**.VorlNr = **v**.VorlNr;

# Relationale Entwurfstheorie

| ProfVorl | | | | | | |
|---|---|---|---|---|---|---|
| PersNr | Name | Rang | Raum | VorlNr | Titel | SWS |
| 2125 | Sokrates | C4 | 226 | 5041 | Ethik | 4 |
| 2125 | Sokrates | C4 | 226 | 5049 | Mäeutik | 2 |
| 2125 | Sokrates | C4 | 226 | 4052 | Logik | 4 |
| ... | ... | ... | ... | ... | ... | ... |
| 2132 | Popper | C3 | 52 | 5259 | Der Wiener Kreis | 2 |
| 2137 | Kant | C4 | 7 | 4630 | Die 3 Kritiken | 4 |

Wieso ist dies ein schlechtes Schema?

| ProfVorl | | | | | | |
|---|---|---|---|---|---|---|
| PersNr | Name | Rang | Raum | VorlNr | Titel | SWS |
| 2125 | Sokrates | C4 | 226 | 5041 | Ethik | 4 |
| 2125 | Sokrates | C4 | 226 | 5049 | Mäeutik | 2 |
| 2125 | Sokrates | C4 | 226 | 4052 | Logik | 4 |
| ... | ... | ... | ... | ... | ... | ... |
| 2132 | Popper | C3 | 52 | 5259 | Der Wiener Kreis | 2 |
| 2137 | Kant | C4 | 7 | 4630 | Die 3 Kritiken | 4 |

## Wieso ist dies ein schlechtes Schema?

- **Änderungsanomalien:** Sokrates zieht um, von Raum 226 in Raum 338. Was passiert?
- **Einfügeanomalien:** Neuer Professor ohne Vorlesungen?
- **Löschanomalien:** Letzte Vorlesung eines Profs wird gelöscht. Was passiert?

# Funktionale Abhängigkeiten

## Functional Dependency (FD)

- Schema $\mathcal{R} = \{A, B, C, D\}$
- Ausprägung $R$
- Seien $\alpha \subseteq \mathcal{R}$ und $\beta \subseteq \mathcal{R}$
- $\alpha \rightarrow \beta$ genau dann, wenn $\forall r, s \in R$ gilt: $r.\alpha = s.\alpha \Rightarrow r.\beta = s.\beta$
- D.h. die $\alpha$-Werte bestimmen die $\beta$-Werte funktional (=eindeutig)

| R | | | |
|---|---|---|---|
| A | B | C | D |
| a4 | b2 | c4 | d3 |
| a1 | b1 | c1 | d1 |
| a1 | b1 | c1 | d2 |
| a2 | b2 | c3 | d2 |
| a3 | b2 | c4 | d3 |

$$\{A\} \rightarrow \{B\}$$

$$\{A\} \rightarrow \{C\}$$

$$\{C, D\} \rightarrow \{B\}$$

Aber nicht $\{B\} \rightarrow \{C\}$

Notation: $CD \rightarrow B$

# Einhaltung funktionaler Abhängigkeiten

## Alternative Formulierung

- Die FD $\alpha \rightarrow \beta$ ist in $R$ erfüllt, wenn für jede mögliche Ausprägung von $\alpha$ gilt:
$$|\pi_\beta(\sigma_{\alpha=c}(R))| \leq 1$$
- "die Menge aller Tupel von $R$ mit $\alpha = c$ (für beliebiges $c$) projiziert auf $\beta$ enthält keinen oder genau einen Wert"
- ansonsten wäre es auch keine "Funktion"

## Daraus folgt ein einfacher Algorithmus:

boolean giltFD(Relation $R$, FD $\alpha \rightarrow \beta$){
    sortiere $R$ nach $\alpha$ Werten
    für jede Gruppe $G_i \subseteq R$ von Tupeln mit Wert $\alpha_i$ aus $\alpha$:
        falls nicht alle $\beta_i$ aus $\beta$ identisch sind:
            return falsch;
    return wahr;
}

# Schlüssel

- $\alpha \subseteq \mathcal{R}$ ist ein Superschlüssel, falls gilt: $\alpha \rightarrow \mathcal{R}$
- trivial: es gilt $\mathcal{R} \rightarrow \mathcal{R}$
- Allerdings sind Superschlüssel nicht notwendigerweise minimal!

## Volle funktionale Abhängigkeit:

- $\beta$ ist voll funktional abhängig von $\alpha$ (Notation: $\stackrel{\bullet}{\rightarrow}$) genau dann wenn gilt:
  - $\alpha \rightarrow \beta$
  - $\alpha$ kann nicht mehr verkleinert (=linksreduziert) werden, d.h.
    - $\forall A \in \alpha$ folgt, dass $(\alpha - \{A\}) \rightarrow \beta$ nicht gilt, bzw. alternativ:
    - $\forall A \in \alpha : \neg((\alpha - \{A\}) \rightarrow \beta)$
- $\alpha \subseteq \mathcal{R}$ ist ein **Kandidatenschlüssel**, falls gilt: $\alpha \stackrel{\bullet}{\rightarrow} \mathcal{R}$

Ein Kandidatenschlüssel wird als **Primärschlüssel** ausgewählt!

# Beispiel zur Schlüsselbestimmung

| Städte | | | |
|---|---|---|---|
| Name | BLand | Vorwahl | EW |
| Frankfurt | Hessen | 69 | 650000 |
| Frankfurt | Brandenburg | 335 | 84000 |
| München | Bayern | 89 | 1200000 |
| Passau | Bayern | 851 | 50000 |
| Saarbrücken | Saarland | 681 | 175000 |
| Kaiserslautern | Rheinland-Pfalz | 631 | 100000 |
| ... | ... | ... | ... |

## Kandidatenschlüssel von Städte:

- {Name, BLand}
- {Name, Vorwahl}

# Bestimmung funktionaler Abhängigkeiten

- Tabelle **Professoren**: {[PersNr, Name, Rang, Raum, Ort, Straße, PLZ, Vorwahl, BLand, EW, Landesregierung]}

- {PersNr} $\rightarrow$ {PersNr, Name, Rang, Raum, Ort, Straße, PLZ, Vorwahl, BLand, EW, Landesregierung}
- {Ort,BLand} $\rightarrow$ {EW, Vorwahl}
- {PLZ} $\rightarrow$ {BLand, Ort, EW}
- {BLand, Ort, Straße} $\rightarrow$ {PLZ}
- {BLand} $\rightarrow$ {Landesregierung}
- {Raum} $\rightarrow$ {PersNr}

Außerdem kann abgeleitet werden:

- {Raum} $\rightarrow$ {PersNr, Name, Rang, Raum, Ort, Straße, PLZ, Vorwahl, BLand, EW, Landesregierung}
- {PLZ} $\rightarrow$ {Landesregierung}

# Herleitung weiterer FDs

- Aus einer Menge $F$ von FDs sind weitere FDs herleitbar
- $F^+$ wird Hülle (closure) von $F$ genannt.
- $F^+$ besteht aus allen aus $F$ herleitbaren FDs
- Inferenzregeln, die Amstrong Axiome, beschreiben die Herleitung

# Die Armstrong-Axiome

$\alpha$,$\beta$, $\gamma$, $\delta$ sind Teilmengen der Attribute aus $\mathcal{R}$

- **Reflexivität:** Falls $\beta$ eine Teilmenge von $\alpha$ ist ($\beta \subseteq \alpha$) dann gilt immer $\alpha \rightarrow \beta$.
  Insbesondere gilt also immer $\alpha \rightarrow \alpha$.
- **Verstärkung:** Falls $\alpha \rightarrow \beta$ gilt, dann gilt auch $\alpha\gamma \rightarrow \beta\gamma$. Hierbei stehe z.B. $\alpha\gamma$ für $\alpha \cup \gamma$
- **Transitivität:** Falls $\alpha \rightarrow \beta$ und $\beta \rightarrow \gamma$ gilt, dann gilt auch $\alpha \rightarrow \gamma$.

Die Armstrong-Axiome sind **korrekt** und **vollständig**.

D.h. Mit Hilfe dieser Axiome können alle gültigen FDs hergeleitet werden.

# Die Armstrong-Axiome (2)

Nicht notwendig, aber oftmals komfortabel für Herleitungen:

- **Vereinigungsregel:**
  Wenn $\alpha \rightarrow \beta$ und $\alpha \rightarrow \gamma$ gelten, dann auch $\alpha \rightarrow \beta\gamma$.
- **Dekompositionsregel:**
  Wenn $\alpha \rightarrow \beta\gamma$ gilt, dann gelten auch $\alpha \rightarrow \beta$ und $\alpha \rightarrow \gamma$.
- **Pseudotransitivitätsregel:**
  Wenn $\alpha \rightarrow \beta$ und $\gamma\beta \rightarrow \delta$, dann gilt auch $\alpha\gamma \rightarrow \delta$

# Attributhülle

Die Attributhülle AttrHülle($F$,$\alpha$) einer Attributmenge $\alpha$ bzgl. FDs $F$ ist die Menge von Attributen $\alpha^+$, für die gilt $\alpha \rightarrow \alpha^+$.

Gegeben:

- eine Menge F von FDs
- eine Menge von Attributen $\alpha \subseteq \mathcal{R}$

## Algorithmus:

Erg := $\alpha$;<br>
**while** (Änderungen an $Erg$) **do**<br>
    **for each** FD $\beta \rightarrow \gamma$ **in** $F$ **do**<br>
        **if** $\beta \subseteq Erg$ **then** $Erg := Erg \cup \gamma$;<br>
$\alpha^+ := Erg$;

# Beispielanwendung

**Ziel:**

Bestimme, ob $\kappa$ einen Superschlüssel einer Relation $\mathcal{R}$ bzgl. der FDs in $F$ bildet.

**Lösung:**

Durch Aufruf AttrHülle($F$,$\kappa$) erhalten wir $\kappa^+$.
Falls $\kappa^+ = \mathcal{R}$: $\kappa$ ist Superschlüssel von $\mathcal{R}$.

# Äquivalente FD-Mengen

- Im Allgemeinen: es gibt viele unterschiedliche äquivalente Mengen von funktionalen Abhängigkeiten.
- Zwei Mengen $F$ und $G$ heißen äquivalent (Notation: $F \equiv G$), wenn ihre Hüllen gleich sind, d.h. $F^+ = G^+$.
- Bedeutung: die gleichen Mengen von FDs müssen herleitbar sein.

## Beobachtung:

- $F^+$ kann sehr groß sein
- viele redundanten Abhängigkeiten
- in der Praxis unübersichtlich

Ziel: **kleinstmögliche** Menge $F_c$ finden, so dass immer noch gilt: $F_c^+ = F^+$.

# Kanonische Überdeckung $F_c$

Folgende Eigenschaften müssen erfüllt sein:

1. $F_c \equiv F$, d.h., $F_c^+ = F^+$
2. In $F_c$ existieren keine FDs $\alpha \to \beta$, bei denen $\alpha$ oder $\beta$ überflüssige Attribute enthalten. D.h. es muss gelten:
   (a) $\forall A \in \alpha : (F_c - (\alpha \to \beta) \cup ((\alpha - A) \to \beta)) \not\equiv F_c$
   (b) $\forall B \in \beta : (F_c - (\alpha \to \beta) \cup (\alpha \to (\beta - B))) \not\equiv F_c$
3. Jede linke Seite einer funktionalen Abhängigkeit in $F_c$ ist einzigartig. Dies kann durch sukzessive Anwendung der Vereinigungsregel auf FDs der Art $\alpha \to \beta$ und $\alpha \to \gamma$ erzielt werden, so dass die beiden FDs durch $\alpha \to \beta\gamma$ ersetzt werden.

# Algorithmus zur Bestimmung von $F_c$

1. Führe für jede FD $\alpha \to \beta \in F$ die Linksreduktion durch, also:
   - Überprüfe für alle $A \in \alpha$, ob $A$ überflüssig ist, d.h. ob $\beta \subseteq AttrHülle(F, \alpha - A)$
2. Führe für jede (verbliebene) FD die Rechtsreduktion durch:
   - Überprüfe für alle $B \in \beta$, ob $B \in \mathit{AttrHülle}(F - (\alpha \to \beta) \cup (\alpha \to (\beta - B)), \alpha)$ gilt. In diesem Fall ist $B$ auf der rechten Seite überflüssig und kann eliminiert werden, d.h. $\alpha \to \beta$ wird durch $\alpha \to (\beta - B)$ ersetzt.
3. Entferne die FDs der Form $\alpha \to \emptyset$, die im 2. Schritt entstanden sind.
4. Fasse mittels Vereinigungsregel FDs der Form $\alpha \to \beta_1, ..., \alpha \to \beta_n$ zusammen, so dass $\alpha \to (\beta_1 \cup ...\beta_n)$ verbleibt.

# Beispiel für Herleitung einer kanonischen Überdeckung

- $F = \{A \rightarrow B, B \rightarrow C, AB \rightarrow C\}$
- **Schritt 1:** Linksreduktion ersetzt $AB \rightarrow C$ durch $A \rightarrow C$
- **Schritt 2:** Rechtsreduktion ersetzt $A \rightarrow C$ durch $A \rightarrow \emptyset$
- **Schritt 3:** eliminiere $A \rightarrow \emptyset$
- **Schritt 4:** keine Zusammenfassungen notwendig
- Damit ist $F_c = \{A \rightarrow B, B \rightarrow C\}$

# Normalisierung von Relationen

Um Qualitätsprobleme im ursprünglichen Entwuft zu beheben, wird das bestehende Relationenschema $\mathcal{R}$ wird in mehrere Relationenschemata $\mathcal{R}_1, ..., \mathcal{R}_n$ zerlegt, die dann "besser" sind.

- Die Güte einer Zerlegung wird mit **Normalformen** beschrieben.
- Normalformen: 1NF, 2NF, 3NF, BCNF, 4NF, ...

## Korrektheitskriterien für Zerlegung:

- **Verlustlosigkeit:** Die in der ursprünglichen Relationenausprägung $R$ des Schemas $\mathcal{R}$ enthaltenen Daten müssen aus den Ausprägungen $R_1, ..., R_n$ der neuen Relationenschemata $\mathcal{R}_1, ..., \mathcal{R}_n$ rekonstruierbar sein.
- **Abhängigkeitsbewahrung:** Alle FDs in $F_R$ sollten in den $F_{\mathcal{R}_1}, F_{\mathcal{R}_2}, ....F_{\mathcal{R}_n}$ bewahrt bleiben.

# Verlustlosigkeit

- Zerlegung ist gültig, wenn: $\mathcal{R} = \mathcal{R}_1 \cup \mathcal{R}_2$
- D.h. alle Attribute aus $\mathcal{R}$ bleiben in der Zerlegung erhalten
- Wir definieren:
  - $R_1 := \pi_{\mathcal{R}_1}(R)$
  - $R_2 := \pi_{\mathcal{R}_2}(R)$
- die Zerlegung von $\mathcal{R}$ in $\mathcal{R}_1$ und $\mathcal{R}_2$ ist **verlustlos**, falls für jede mögliche (gültige) Ausprägung $R$ von $\mathcal{R}$ gilt:

$$R = R_1 \bowtie R_2$$

Die Zerlegung muss also durch einen natürlichen Verbund (Join) rekonstruierbar sein.

D.h. $\mathcal{R}_1 \cap \mathcal{R}_2 \neq \emptyset$ “sinnvoller” Schlüssel existiert

# Formale Charakterisierung Verlustloser Zerlegungen

- Gegeben Zerlegung von $\mathcal{R}$ in $\mathcal{R}_1$ und $\mathcal{R}_2$
- $F_{\mathcal{R}}$ ist die Menge der FDs in $\mathcal{R}$
- Zerlegung ist verlustlos, wenn mindestens **eine** der folgenden FDs herleitbar ist:
  - $(\mathcal{R}_1 \cap \mathcal{R}_2) \rightarrow \mathcal{R}_1 \in F_{\mathcal{R}}^+$ **oder** d.h. Schlüssel bestimmt $\mathcal{R}_1$
  - $(\mathcal{R}_1 \cap \mathcal{R}_2) \rightarrow \mathcal{R}_2 \in F_{\mathcal{R}}^+$ d.h. Schlüssel bestimmt $\mathcal{R}_2$

## Beispiel:

- Seien $\alpha, \beta$ und $\gamma$ paarweise disjunkte Attributmengen
- $\mathcal{R} = \alpha \cup \beta \cup \gamma$, $\mathcal{R}_1 = \alpha \cup \beta$, $\mathcal{R}_2 = \alpha \cup \gamma$, $\mathcal{R}_1 \cap \mathcal{R}_2 = \alpha$
- Dann muss eine der Bedingungen gelten:
  - $\beta \subseteq$ *AttrHülle*$(F_{\mathcal{R}}, \alpha)$ **oder**
  - $\gamma \subseteq$ *AttrHülle*$(F_{\mathcal{R}}, \alpha)$

*D.h. die gemeinsamen Joinattribute $\alpha$ müssen $\mathcal{R}_1$ oder $\mathcal{R}_2$ bestimmen.*

Dies ist eine hinreichende aber nicht notwendige Bedingung!

# Beispiel für Verlust

| Biertrinker | | |
|---|---|---|
| **Kneipe** | **Gast** | **Bier** |
| Kowalski | Kemper | Pils |
| Kowalski | Eickler | Hefeweizen |
| Innsteg | Kemper | Hefeweizen |

**wird zerlegt in ...**

$\pi_{Kneipe,Gast}$

| Besucht | |
|---|---|
| **Kneipe** | **Gast** |
| Kowalski | Kemper |
| Kowalski | Eickler |
| Innsteg | Kemper |

$\pi_{Gast,Bier}$

| Trinkt | |
|---|---|
| **Gast** | **Bier** |
| Kemper | Pils |
| Eickler | Hefeweizen |
| Kemper | Hefeweizen |

# Beispiel für Verlust: Wiederherstellung

$\pi_{Kneipe,Gast}$

| Besucht | |
|---|---|
| **Kneipe** | **Gast** |
| Kowalski | Kemper |
| Kowalski | Eickler |
| Innsteg | Kemper |

$\pi_{Gast,Bier}$

| Trinkt | |
|---|---|
| **Gast** | **Bier** |
| Kemper | Pils |
| Eickler | Hefeweizen |
| Kemper | Hefeweizen |

**Mit Join verbunden gibt ..**

| Biertrinker | | |
|---|---|---|
| **Kneipe** | **Gast** | **Bier** |
| Kowalski | Kemper | Pils |
| **Kowalski** | **Kemper** | **Hefeweizen** |
| Kowalski | Eickler | Hefeweizen |
| **Innsteg** | **Kemper** | **Pils** |
| Innsteg | Kemper | Hefeweizen |

# Erläuterung des Biertrinker-Beispiels

- User Biertrinker-Beispiel war **keine verlustlose Zerlegung**.
- Es gibt in Biertrinker nämlich nur die folgende nicht-triviale funktionale Abhängigkeit:

$$\{Kneipe, Gast\} \rightarrow \{Bier\}$$

- $\mathcal{R} = \{Kneipe\} \cup \{Gast\} \cup \{Bier\}$
- $\mathcal{R}_1 = \{Kneipe\} \cup \{Gast\}$, $\mathcal{R}_2 = \{Gast\} \cup \{Bier\}$, $\mathcal{R}_1 \cap \mathcal{R}_2 = \{Gast\}$.
- Dann muss eine der Bedingungen gelten:
  - $\{Kneipe\} \subseteq \mathit{AttrHülle}(\{Kneipe, Gast\} \rightarrow \{Bier\}, \{Gast\})$ **oder**
  - $\{Bier\} \subseteq \mathit{AttrHülle}(\{Kneipe, Gast\} \rightarrow \{Bier\}, \{Gast\})$
- Das ist nicht der Fall!
- Keine der zwei möglichen, die Verlustlosigkeit garantierenden FDs gilt:

$$\{Gast\} \rightarrow \{Bier\}$$
$$\{Gast\} \rightarrow \{Kneipe\}$$