Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

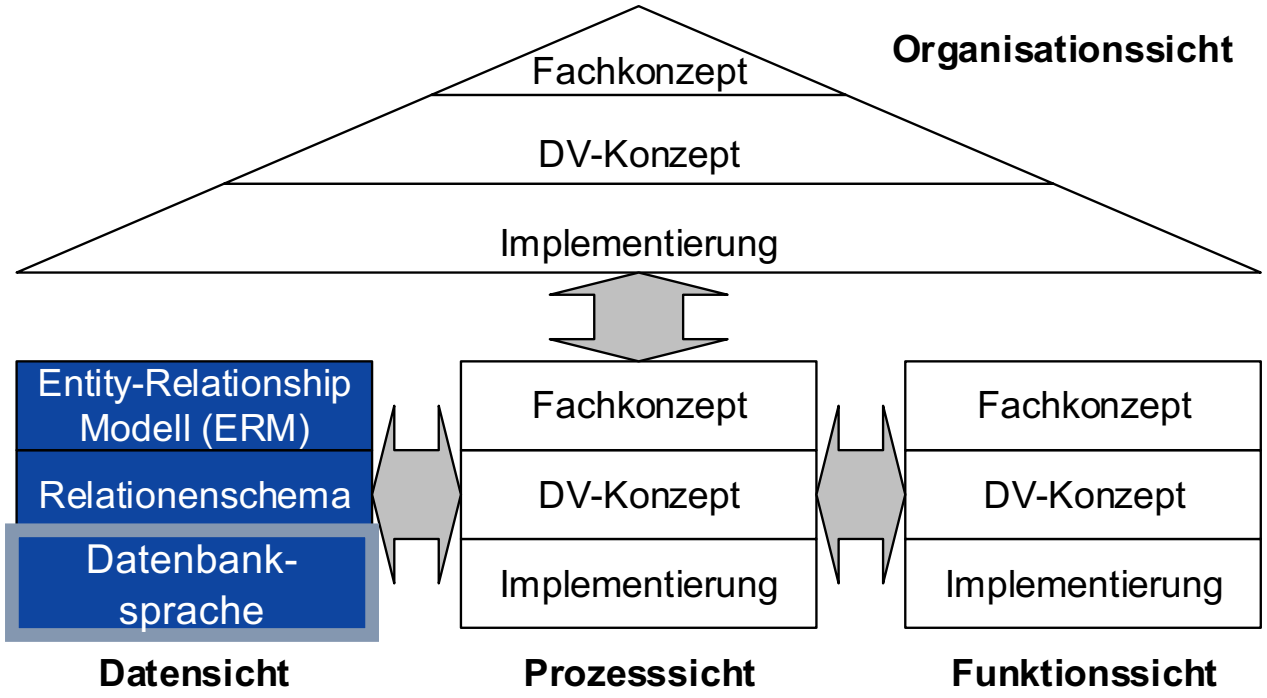
Datenmanagement & -analyse

Übung 4 – Datenbankabfragen

Dr. Nikolai Stein
Lehrstuhl für WI & BA
Julius-Maximilians-Universität Würzburg

Sommersemester 2021

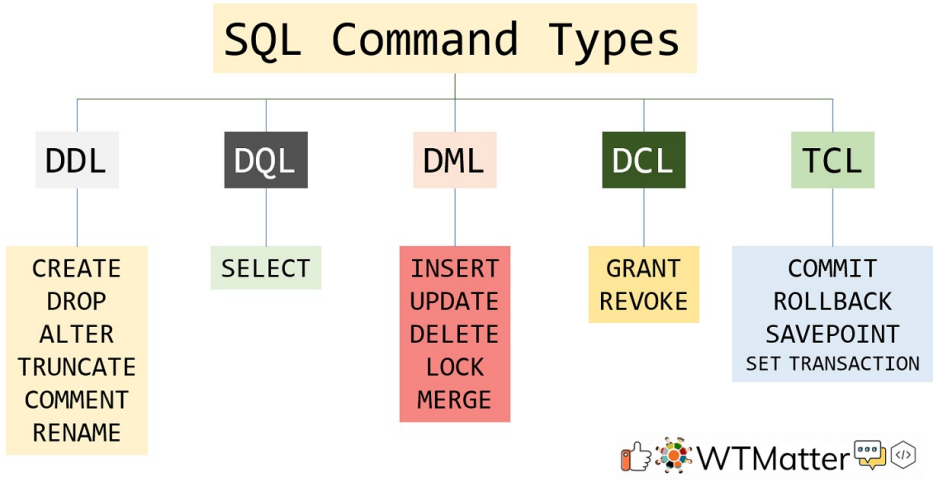
Architektur Integrierter Informationssysteme



Scheer (1992)

Befehlsklassen

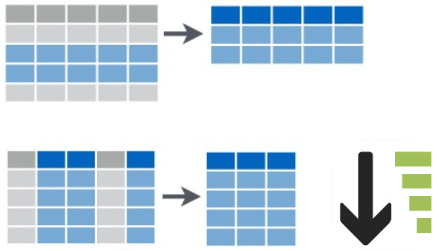
- SQL als Data **Definition** Language (DDL)
  - Anlegen, Ändern, Löschen von Tabellen
- SQL als Data **Query** Language (DQL)
  - Ausführen von Abfragen auf Datenbestand
- SQL als Data **Manipulation** Language (DML)
  - Einfügen, Ändern, Löschen von Daten
- SQL als Data **Control** Language (DCL)
  - Benutzer- und Transaktionsverwaltung
- SQL als **Transaction** Control Language (TCL)
  - Verwaltung von Transaktionen



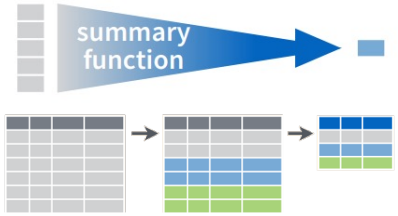
Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Die vereinfachte Welt der Abfragen

Einfache Tabellenoperationen

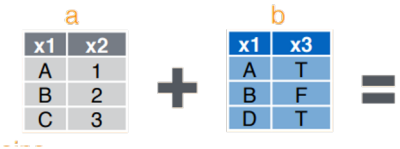


Gruppieren & Zusammenfassen



Fensteroperationen

Verbundoperationen



Funktionen auf einer Tabelle

Funktionen auf mehreren Tabellen

Die grundlegenden Verben von dplyr

- select: Spalten nach Namen auswählen
- filter: wählt Zeilen aus, die den Kriterien entsprechen
- arrange: Zeilen neu anordnen
- mutate: neue Variablen hinzufügen

Semantik

- Erstes Argument ist ein data.frame
- Nachfolgende Argumente sagen, was mit dem data.frame geschehen soll
- Gibt immer einen data.frame zurück, modifiziert niemals das ursprüngliche Objekt

df

color	value
blue	1
black	2
blue	3
blue	4
black	5

df2

color	value
4	1
1	2
5	3
3	4
2	5

Gruppieren und Zusammenfassen

- Mehrere Variablen mit Hilfer einer summary-Funktion auf einen einzigen Wert reduzieren
- Typische Funktionen
  - `min(x), median(x), max(x),`
  - `quantile(x, p)`
  - `n(), n_distinct(), sum(x), mean(x)`
  - `sum(x > 10), mean(x > 10)`
  - `sd(x), var(x), iqr(x), mad(x)`

```
summarise(df, total=sum(value))
```

df

color	value
blue	1
black	2
blue	3
blue	4
black	5

→

total
15

- Erstellen Sie einen neuen data.frame, der die zugrundeliegende Gruppierung explizit hinterlegt hat

```
grouped_by_color <- group_by(df,
color)
```

- Beim Zusammenfassen des gruppierten data.frame bleiben die Gruppierungen erhalten

```
summarise(grouped_by_color,
total=sum(value))
```

df

color	value
blue	1
black	2
blue	3
blue	4
black	5

→

color	total
blue	8
black	7

Pipelining nutzen die Struktur der dplyr Semantik

- Idee: `x %>% f(y)` ⇔ `f(x, y)`
  - Interpretation des Pipeline Operators `%>%`: links wird an rechts übergeben

- Einzeltabellen-Abfragen haben stets einen data.frame als erstes Argument und geben einen data.frame als Ergebnis zurück → perfekte Passung zur Pipeline-Logik

- Daher wird der Code aus der letzten Folie äquivalent durch die folgende Pipeline erfasst

```
df %>%
    filter(…) %>%
    group_by(…) %>%
    summarise(…) %>%
    arrange(…)
```

Fensterfunktionen

- Eine Fensterfunktion ist eine Variante einer Aggregationsfunktion
- Während eine Aggregationsfunktion, wie sum() und mean(), n Eingaben annimmt und einen einzigen Wert zurückgibt, gibt eine Fensterfunktion n Werte zurück
- Die Ausgabe einer Fensterfunktion kann von allen Eingabewerten abhängen

lead() und lag() erzeugen versetzte Versionen eines Eingangsvektors, die entweder vor oder hinter dem ursprünglichen Vektor liegen.

```
x <- c(1, 2, 3, 4, 5)
lead(x) #> [1] 2 3 4 5 NA
lag(x) #> [1] NA 1 2 3 4
```

Sie können sie verwenden, um:

- Gab es eine Änderung? `x != lag(x)`
- Absolute Änderung? `x - lag(x)`
- Prozentuale Änderung? `(x - lag(x)) / x`
- Verfielwachung? `x / lag(x)`
- Zuvor falsch, jetzt wahr? `!lag(x) & x`