

Informatik II

Vorlesung am D-BAUG der ETH Zürich

Vorlesung 12, 2017

Datenbanksysteme: Datendefinition in SQL,
Kompliziertere Datenbankabfragen

Datendefinition (DDL) in SQL

Anlegen einer Tabelle

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

Tabelle löschen

```
drop table Professoren;
```

Tabellenstruktur anpassen

```
alter table Professoren add column Raum integer;
```

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

Veränderung am Datenbestand

Einfügen von Tupeln

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

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

Studenten		
MatrNr	Name	Semester
29120	Theophrastos	2
29555	Feuerbach	2
28121	Archimedes	-

Null-
Wert

Veränderungen am Datenbestand

Löschen von Tupeln

```
delete Studenten  
    where Semester > 100;
```

Verändern von Tupeln

```
update Studenten  
    set Semester= Semester + 1;
```

Sortieren

SELECT *Spalten* FROM *Tabellen* ORDER BY Name[ASC|DESC]

SELECT Name, Population, SurfaceArea FROM country ORDER BY SurfaceArea DESC

Name	Population	SurfaceArea
Russian Federation	146934000	17075400.00
Antarctica	0	13120000.00
Canada	31147000	9970610.00
China	1277558000	9572900.00
United States	278357000	9363520.00
Brazil	170115000	8547403.00
Australia	18886000	7741220.00
India	1013662000	3287263.00
Argentina	37032000	2780400.00

Kombination

```
SELECT Code, Name, Population, SurfaceArea
FROM country WHERE
Population > 7000000 AND Population < 8000000
ORDER BY SurfaceArea ASC
```

Code	Name	Population	SurfaceArea
RWA	Rwanda	7733000	26338.00
CHE	Switzerland	7160400	41284.00
AZE	Azerbaijan	7734000	86600.00
GIN	Guinea	7430000	245857.00
TCD	Chad	7651000	1284000.00

$\pi_{\text{Code,Name,Population,SurfaceArea}}(\sigma_{\text{Population}>7000000 \wedge \text{Population}<8000000}(\text{country}))$

Anfragen über mehrere Relationen

Welcher Professor liest "Mäeutik"?

```
select      Name, Titel
from        Professoren , Vorlesungen
where       PersNr = gelesenVon and Titel = "Mäeutik" ;
```

Projektion

Kreuzprodukt

Selektion

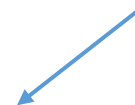
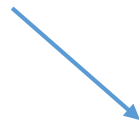
$\Pi_{\text{Name, Titel}}$
 $(\sigma_{\text{PersNr=gelesenVon} \wedge \text{Titel='Mäeutik'}}$
 $(\text{Professoren} \times \text{Vorlesungen}))$

Anfragen über mehrere Relationen

Professoren			
PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
⋮	⋮	⋮	⋮
2137	Kant	C4	7

Vorlesungen			
VorlNr	Titel	KP	gelesen Von
5001	Grundzüge	4	2137
5041	Ethik	4	2125
⋮	⋮	⋮	⋮
5049	Mäeutik	2	2125
⋮	⋮	⋮	⋮
4630	Die 3 Kritiken	4	2137

Verknüpfung X



Anfragen über mehrere Relationen (Fortsetzung)

PersNr	Name	Rang	Raum	VorlNr	Titel	KP	gelesen Von
2125	Sokrates	C4	226	5001	Grundzüge	4	2137
1225	Sokrates	C4	226	5041	Ethik	4	2125
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
2125	Sokrates	C4	226	5049	Mäeutik	2	2125
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
2126	Russel	C4	232	5001	Grundzüge	4	2137
2126	Russel	C4	232	5041	Ethik	4	2125
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
2137	Kant	C4	7	4630	Die 3 Kritiken	4	2137

↓ Auswahl σ

PersNr	Name	Rang	Raum	VorlNr	Titel	KP	gelesen Von
2125	Sokrates	C4	226	5049	Mäeutik	2	2125

↓ Projektion π

Name	Titel
Sokrates	Mäeutik

String-Vergleiche mit like

Platzhalter "%" ; "_"

"%" steht für beliebig viele (auch gar kein) Zeichen

"_" steht für genau ein Zeichen

select *

from Studenten

where Name **like** `T%eophrastos`;

select distinct s.Name

from Vorlesungen v, hören h, Studenten s

where s.MatrNr = h.MatrNr **and** h.VorlNr = v.VorlNr **and** v.Titel **like** `%thik%`;

Syntactic Sugar

```
select *  
from Studenten  
where Semester > = 1 and Semester < = 4;
```

```
select *  
from Studenten  
where Semester between 1 and 4;
```

```
select *  
from Studenten  
where Semester in (1,2,3,4);
```

Anfragen: weitere Beispiele

Welche Studenten hören welche Vorlesungen?

```
select  Name, Titel
from    Studenten, hören, Vorlesungen
where   Studenten.Legi = hören.Legi
          and hören.VorlNr = Vorlesungen.VorlNr
```

Alternative:

```
select  s.Name, v.Titel
from    Studenten s, hören h, Vorlesungen v
where   s. Legi= h.Legi and h.VorlNr = v.VorlNr
```

Anfragen: weitere Beispiele

Welche Studenten kennen sich aus Vorlesungen?

```
SELECT s1.name, s2.name
FROM   Studenten s1, Studenten s2, hören h1, hören h2
WHERE  s1.Legi = h1.Legi
      AND s2.Legi = h2.Legi
      AND h1.VorlNr = h2.VorlNr
      AND s1.Name < s2.Name
```

Anfragen: weitere Beispiele

In welchen Städten wird Deutsch verstanden?

```
SELECT s.name, c.name
FROM   city s, country c, countrylanguage l
WHERE  s.countrycode = c.code
      AND c.code = l.countrycode
      AND l.language = "German"
      AND l.isOfficial = TRUE
```

Anfragen: weitere Beispiele

In welchen Ländern gibt es Städte mittlerer Grösse (100-500k Einwohner), deren Namen mit "Z" beginnen ?

```
SELECT c.name  
FROM   city s, country c  
WHERE  s.countrycode = c.code  
       AND s.name like "Z%"  
       AND s.population between 100000 and 500000
```

Aggregatfunktion und Gruppierung

Aggregatfunktionen **avg, max, min, count, sum**

```
select avg(Semester)  
from Studenten;
```

```
select v.gelesenVon, sum(v.KP)  
from Vorlesungen v  
group by v.gelesenVon;
```


Aggregatfunktion und Gruppierung

```
select v.gelesenVon, p.Name, sum(v.KP)  
from Vorlesungen v, Professoren p  
where v.gelesenVon = p.PersNr and p.Rang = 'FP'  
group by v.gelesenVon, p.Name  
having avg(v.KP) >= 3;
```

SQL weiss nicht, dass sich der Name innerhalb der Gruppe nicht ändern kann, wenn nur gelesenVon angegeben ist

Besonderheiten bei Aggregatoperationen

SQL erzeugt pro Gruppe ein Ergebnistupel

Deshalb müssen alle in der **select**-Klausel aufgeführten Attribute - außer den aggregierten – auch in der **group by**-Klausel aufgeführt werden

Nur so kann SQL sicherstellen, dass sich das Attribut nicht innerhalb der Gruppe ändert

(Fast) Alles

select [distinct]

column1 [**as** *alias1*], *column2* [**as** *alias2*], [avg, max, min, count, sum]...

from *table1* [*alias1*], *table2* [*alias2*],...

[**where** *condition(s)*]

[**group by** *column(s)*]

[**having** *condition(s)*]

[**order by** *column(s)* [**DESC**]];

Projektion π
[+Umbenennung ρ]

Kreuzprodukt $\times$
[+Umbenennung ρ]

Selektion σ

Abarbeitung der Anfrage in SQL

1. Schritt: *Kreuzprodukt und Selektion*
from Vorlesungen, Professoren
where gelesenVon = PersNr **and** Rang = 'FP'
2. Schritt: *Gruppierung*
group by gelesenVon, Name
3. Schritt: *Selektion der Gruppierung*
having avg (KP) >= 3
4. Schritt: *Projektion*
select gelesenVon, Name, **sum** (KP)

Ausführen einer Anfrage mit group by

Vorlesung x Professoren							
VorlNr	Titel	KP	gelesen Von	PersNr	Name	Rang	Raum
5001	Grundzüge	4	2137	2125	Sokrates	FP	226
5041	Ethik	4	2125	2125	Sokrates	FP	226
...	...	...	...	...	...	...	...
4630	Die 3 Kritiken	4	2137	2137	Kant	FP	7

↓ **where-Bedingung**

nach Selektion

VorlNr	Titel	KP	gelesen Von	PersNr	Name	Rang	Raum
5001	Grundzüge	4	2137	2137	Kant	C4	7
5041	Ethik	4	2125	2125	Sokrates	C4	226
5043	Erkenntnistheorie	3	2126	2126	Russel	C4	232
5049	Mäeutik	2	2125	2125	Sokrates	C4	226
4052	Logik	4	2125	2125	Sokrates	C4	226
5052	Wissenschaftstheorie	3	2126	2126	Russel	C4	232
5216	Bioethik	2	2126	2126	Russel	C4	232
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7

↓ Gruppierung

nach Gruppierung

VorlNr	Titel	KP	gelesenVon	PersNr	Name	Rang	Raum
5041	Ethik	4	2125	2125	Sokrates	FP	226
5049	Mäeutik	2	2125	2125	Sokrates	FP	226
4052	Logik	4	2125	2125	Sokrates	FP	226
5043	Erkenntnistheorie	3	2126	2126	Russel	FP	232
5052	Wissenschaftstheo.	3	2126	2126	Russel	FP	232
5216	Bioethik	2	2126	2126	Russel	FP	232
5001	Grundzüge	4	2137	2137	Kant	FP	7
4630	Die 3 Kritiken	4	2137	2137	Kant	FP	7

↓ **having-Bedingung**

VorlNr	Titel	KP	gelesenVon	PersNr	Name	Rang	Raum
5041	Ethik	4	2125	2125	Sokrates	FP	226
5049	Mäeutik	2	2125	2125	Sokrates	FP	226
4052	Logik	4	2125	2125	Sokrates	FP	226
5001	Grundzüge	4	2137	2137	Kant	FP	7
4630	Die 3 Kritiken	4	2137	2137	Kant	FP	7

↓ **Aggregation (sum) und Projektion**

Ergebnis

gelesenVon	Name	sum (KP)
2125	Sokrates	10
2137	Kant	8

Geschachtelte Anfrage

Unteranfrage in der **where**-Klausel

Welche Prüfungen sind besser als durchschnittlich verlaufen?

```
select *  
  from prüfen  
 where Note < ( select avg (Note)  
                from prüfen );
```

Mengenvergleich

```
select p.Name  
from Professoren p  
where p.PersNr not in ( select v.gelesenVon  
                        from Vorlesungen v);
```

Das rechte Argument darf eine Tabelle mit maximal einer Spalte sein

transient erzeugte, separate Tabelle. Klammerung erlaubt Bezug.

Andere Beispiele

Erdbevölkerung

```
select  sum(c.population)  
from    country c
```

Andere Beispiele

Bevölkerung der Kontinente

```
select  c.continent, sum(c.population)
from    country c
group by c.continent
```

Andere Beispiele

Tabelle der durchschnittlichen Anzahl Bewohner der Städte eines Landes

```
select  c.name, avg(s.population) as "Stadtdurchschnitt"  
from    city s, country c  
where   c.code = s.countrycode  
group by c.name
```

Andere Beispiele

Tabelle aller Länder mit mehr als 10 Städten mit 1 Mio Einwohnern.
Tabelle soll auch die Anzahl Städte aufzählen.

```
select  c.name, count(c.name)
from    country c, city s
where   s.countrycode = c.code
        and s.population > 1000000
group   by c.name
having  count(c.name)>10
```