Fortsetzung: Anfragen mit SQL

Bisher:

- Projektion
- Selektion
- Duplikatbehandlung
- NULL Werte

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

hören

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

voraussetzen

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

Assistenten

PersNr	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

Anfragen über mehrere Relationen: Kreuzprodukt

• Falls mehrere Relationen in der from-Klausel auftauchen, werden sie mit einem Kreuzprodukt verbunden

• Beispiel:
Anfrage: "Gib alle Professoren und Vorlesungen aus„

select *
from Vorlesung, Professor**;**

Ergebnis???

Anfragen über mehrere Relationen: Joins

- Kreuzprodukte machen meistens keinen Sinn, interessanter sind Joins
- Joinprädikate werden in der where-Klausel angegeben:

```
select *
from Vorlesung, Professor
where gelesenVon = PersNr;
```

Anfragen über mehrere Relationen: Joins cont.

Welcher Professor liest "Mäeutik"?

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

Beispiel

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

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

PersN	Name	Rang	Raum	VorlNr	Titel	SWS	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

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

↓ Projektion

Name	Titel
Sokrates	Mäeutik

Namenskollision

- gleichnamige Attribute in verschiedenen Relationen müssen aufgelöst werden

Beispiel:
Welche Studenten hören welche Vorlesungen?

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

Namenskollision cont.

Welche Studenten hören welche Vorlesungen?

Alternativ:

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

Mengenoperationen

- In SQL gibt es auch die üblichen Operationen auf Mengen:
  Vereinigung, Schnitt und Differenz
- Setzen wie in der relationalen Algebra gleiches Schema der verknüpften Ausgabe-Relationen voraus

```
( select Name
  from Assistenten )
union
( select Name
  from Professoren);
```

Duplikateliminierung

- Im Gegensatz zu **select** eliminiert **union** automatisch Duplikate

- Falls Duplikate im Ergebnis erwünscht sind, muss der **union all**-Operator benutzt werden

Schnitt, Mengendifferenz

Professoren **und** Assistenten

```
select Name  from Professoren
intersect
select Name from Assistenten;
```

Professoren, **aber nicht** Assistenten

```
select Name from Professoren
except
select Name from Assistenten;
```

Sortierung

- Tupel in einer Relation sind nicht (automatisch) sortiert
- Ergebnis einer Anfrage kann mit Hilfe der **order by**-Klausel sortiert werden
- Es kann aufsteigend oder absteigend sortiert werden
- Default Sortierung: aufsteigend

Beispiel

```
select *
from Studenten
order by Name, Semester desc;
```

Geschachtelte Anfragen

- Anfragen können in anderen Anfragen geschachtelt sein, d.h. es kann mehr als eine select-Klausel geben
- Geschachteltes select kann in der where-Klausel, in der from-Klausel und sogar in einer select-Klausel selbst auftauchen
- Im Prinzip wird in der "inneren" Anfrage ein Zwischenergebnis berechnet, das in der "äußeren" benutzt wird

Select in Where-Klausel

- Zwei verschiedene Arten von Unteranfragen: korrelierte und unkorrelierte
- unkorreliert: Unteranfrage bezieht sich nur auf "eigene" Attribute
- korreliert: Unteranfrage referenziert auch Attribute der äußeren Anfrage

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

hören

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

voraussetzen

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

Assistenten

PersNr	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

Unkorrelierte Unteranfrage

Namen aller Studenten, die VorlNr 5041 hören

```
select S.Name
from Studenten S
where S.MatrNr in
(select h.MatrNr
from hoeren h
where h.VorlNr = 5041);
```

- Unteranfrage wird einmal ausgewertet
- für jedes Tupel der äußeren Anfrage wird geprüft, ob die MatrNr im Ergebnis der Unteranfrage vorkommt

Korrelierte Unteranfrage

Finde alle Professoren, für die Assistenten mit voneinander unterschiedlichen Fachgebieten arbeiten

```
select distinct P.Name
from Professoren P, Assistenten A
where A.Boss = P.PersNr
and exists
(select *
from Assistent B
where B.Boss = P.PersNr and A.Fachgebiet <> B.Fachgebiet);
```

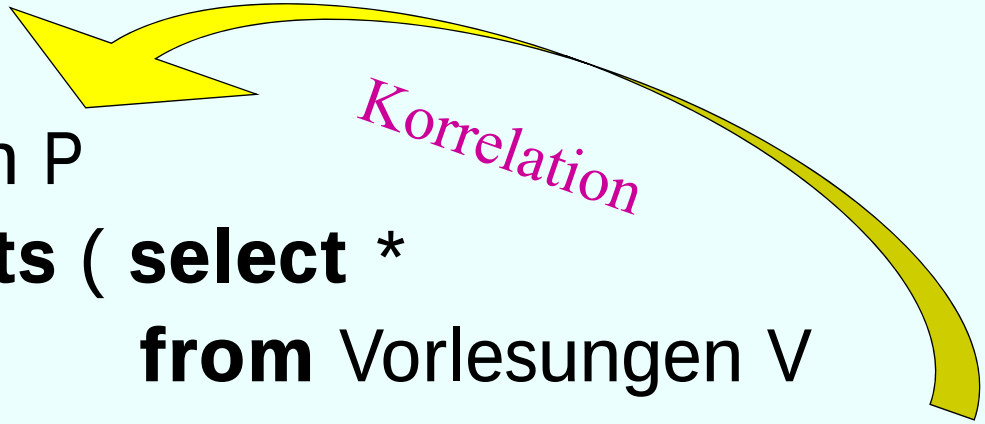
- Für jedes Tupel der äußeren Anfrage hat innere Anfrage verschiedene Werte
- das exists-Prädikat ist wahr, wenn die Unteranfrage mind. ein Tupel enthält

Existenzquantor exists

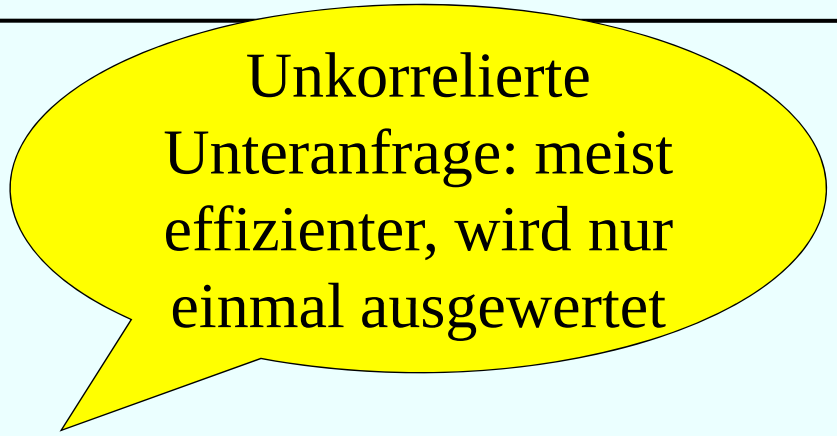
```
select P.Name
from Professoren P
where not exists ( select *
                   from Vorlesungen V
                   where V.gelesenVon = P.PersNr );
```

Existenzquantor exists

```
select P.Name
from Professoren P
where not exists ( select *
                   from Vorlesungen V
                   where V.gelesenVon = P.PersNr );
```



Mengenvergleich




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

Unkorrelierte versus korrelierte Unteranfragen

- korrelierte Formulierung

```
select s.*
from Studenten s
where exists
        (select p.*
        from Professoren p
        where p.GebDatum > s.GebDatum);
```

Umformulierung

- Äquivalente unkorrelierte Formulierung

```
select s.*
from Studenten s
where s.GebDatum <
        (select max (p.GebDatum)
         from Professoren p);
```

- Vorteil: Unteranfrageergebnis kann materialisiert werden
- Unteranfrage braucht nur einmal ausgewertet zu werden

Entschachtelung korrelierter Unteranfragen -- Forts.

```
select a.*
from Assistenten a
where exists
    ( select p.*
      from Professoren p
      where a.Boss = p.PersNr and p.GebDatum>a.GebDatum);
```

- Entschachtelung durch Join

```
select a.*
from Assistenten a, Professoren p
where a.Boss=p.PersNr and p.GebDatum > a.GebDatum;
```

Aggregatfunktion und Gruppierung

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

```
select avg (Semester)
   from Studenten ;

select gelesenVon, sum (SWS)
   from Vorlesungen
   group by gelesenVon;
```

Aggregatfunktion und Gruppierung

```
select gelesenVon, Name, sum (SWS)
   from Vorlesungen, Professoren
   where gelesenVon = PersNr and Rang = ´C4´
   group by gelesenVon, Name
       having avg (SWS) >= 3;
```

Besonderheiten bei Aggregatoperationen

- SQL erzeugt pro Gruppe ein Ergebnistupel
- alle in der **select**-Klausel aufgeführten Attribute - außer den aggregierten – müssen auch in der **group by**-Klausel aufgeführt werden
- Nur so kann SQL sicherstellen, dass sich das Attribut nicht innerhalb der Gruppe ändert
- NULL Wert ist eigene Gruppe

Anfrage mit group by (Equi-Join, Selektion Rang=‚C4‘)

Vorlesung x Professoren							
Vorl Nr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum
5001	Grundzüge	4	2137	2125	Sokrates	C4	226
5041	Ethik	4	2125	2125	Sokrates	C4	226
...	...	...	...	...	...	C3	...
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7

⇓ **where**-Bedingung

Gruppieren nach gelesenVon, Name

VorlNr	Titel	SWS	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

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

hören

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

voraussetzen

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

Assistenten

PersNr	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

Nur Gruppen mit mindestens 3 SWS im Schnitt

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

⇓ **having**-Bedingung

Summenbildung über SWS und Projektion

VorlNr	Titel	SWS	gelesenVon	PersNr	Name	Rang	Raum
5041	Ethik	4	2125	2125	Sokrates	C4	226
5049	Mäeutik	2	2125	2125	Sokrates	C4	226
4052	Logik	4	2125	2125	Sokrates	C4	226
5001	Grundzüge	4	2137	2137	Kant	C4	7
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7

⇓ Aggregation (**sum**) und Projektion

Ergebnis

gelesenVon	Name	**sum** (SWS)
2125	Sokrates	10
2137	Kant	8

Maximum / Minimum

Gib mir den Studenten mir der größten MatrNr

```
select  MatrNr, Name
from Student
where MatrNr =
   (select max(MatrNr)
    from Student);
```

NICHT

```
select Name, max(MatrNr)
from Student;
```

Verwertung der Ergebnismenge einer Unteranfrage

```
select tmp.MatrNr, tmp.Name, tmp.VorlAnzahl
from (select s.MatrNr, s.Name, count(*) as VorlAnzahl
        from Studenten s, hoeren h
        where s.MatrNr=h.MatrNr
        group by s.MatrNr, s.Name)  tmp
where tmp.VorlAnzahl > 2;
```

MatrNr	Name	VorlAnzahl
28106	Carnap	4
29120	Theophrastos	3

… oder auch

```
select tmp.MatrNr, tmp.Name, tmp.VorlAnzahl
from (select s.MatrNr, s.Name, count(*) as VorlAnzahl
        from Studenten s, hoeren h
        where s.MatrNr = h.MatrNr
        group by s.MatrNr, s.Name
        having count(*) > 2)  tmp;
```

Decision-Support-Anfrage mit geschachtelten Unteranfragen

```
select  h.VorlNr, h.AnzProVorl, g.GesamtAnz,
        h.AnzProVorl/g.GesamtAnz as Marktanteil
from    ( select VorlNr, count(*) as AnzProVorl
          from hoeren
          group by VorlNr ) h,
        ( select count (*) as GesamtAnz
          from Studenten) g;
```

Das Ergebnis der Anfrage ?!

VorlNr	AnzProVorl	GesamtAnz	Marktanteil
4052	1	8	0
5001	3	8	0
5022	2	8	0
...	...	...	...

Casting der Integer zu Decimal

```
select  h.VorlNr, h.AnzProVorl, g.GesamtAnz,
        cast(h.AnzProVorl as decimal(6,2)) / g.GesamtAnz
        as Marktanteil
from    ( select VorlNr, count(*) as AnzProVorl
          from hören
          group by VorlNr ) h,
        ( select count (*) as GesamtAnz
          from Studenten) g;
```

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

hören

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

voraussetzen

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

Assistenten

PersNr	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

Das Ergebnis der Anfrage

VorlNr	AnzProVorl	GesamtAnz	Marktanteil
4052	1	8	.125
5001	3	8	.375
5022	2	8	.25
...	...	...	...