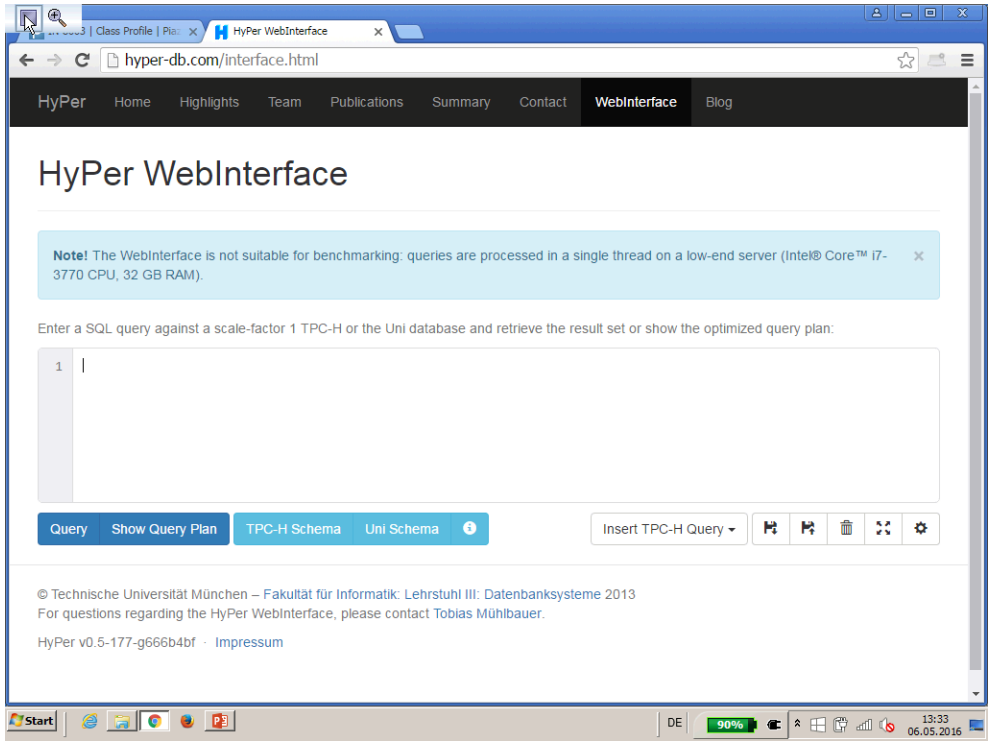
Script **generated by TTT**

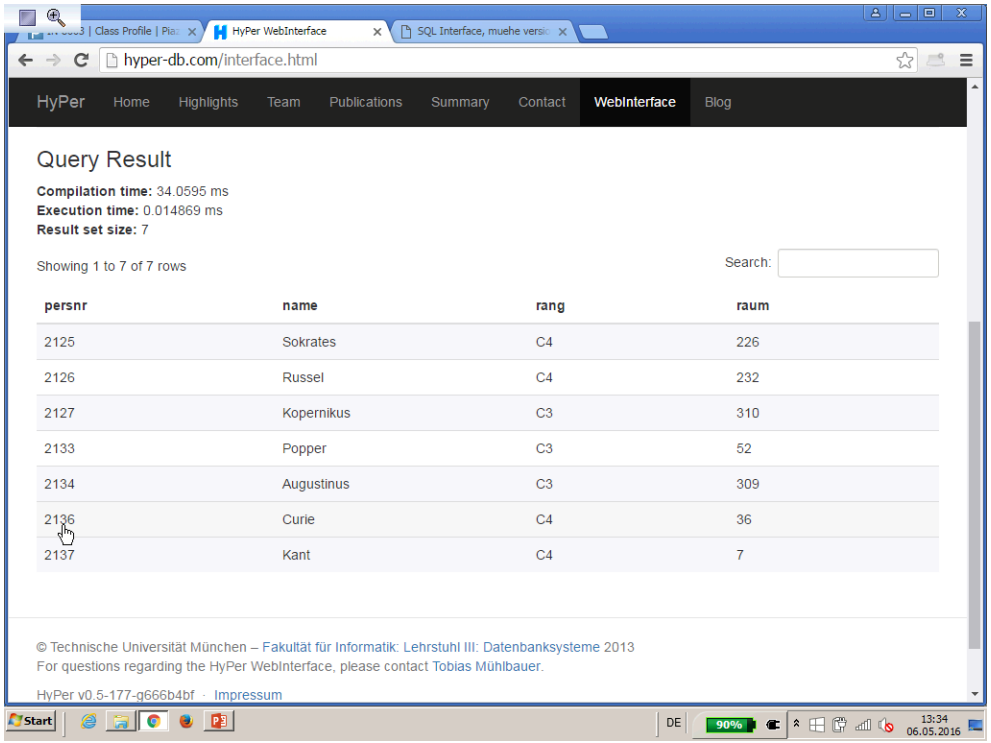
Title: groh: profile1 (06.05.2016)

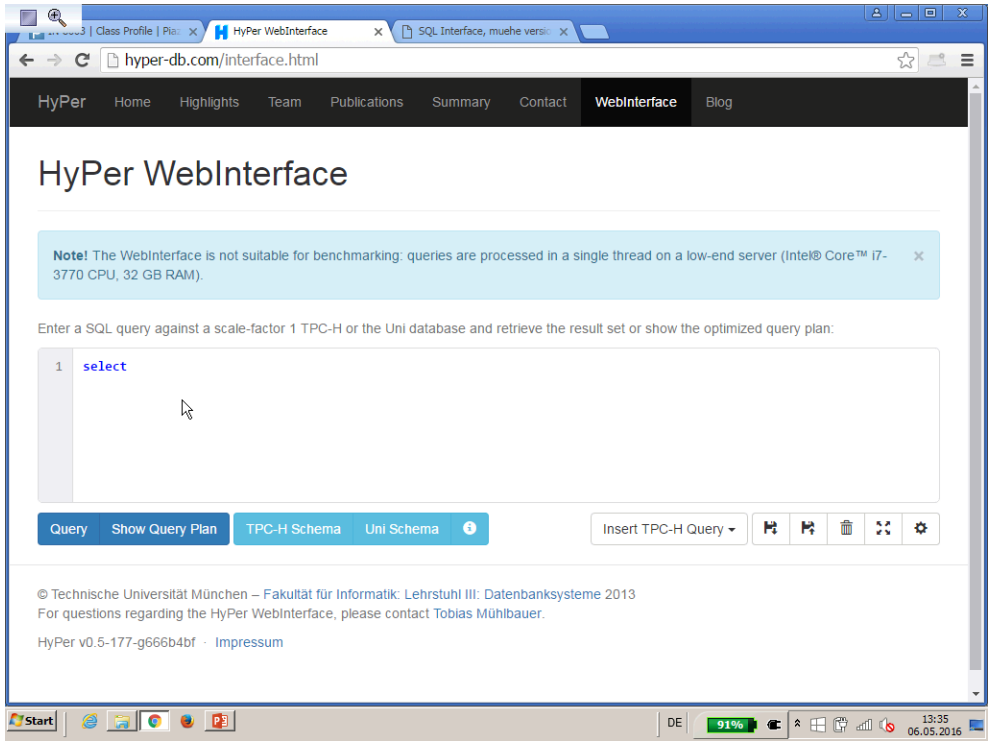
Date: Fri May 06 13:33:51 CEST 2016

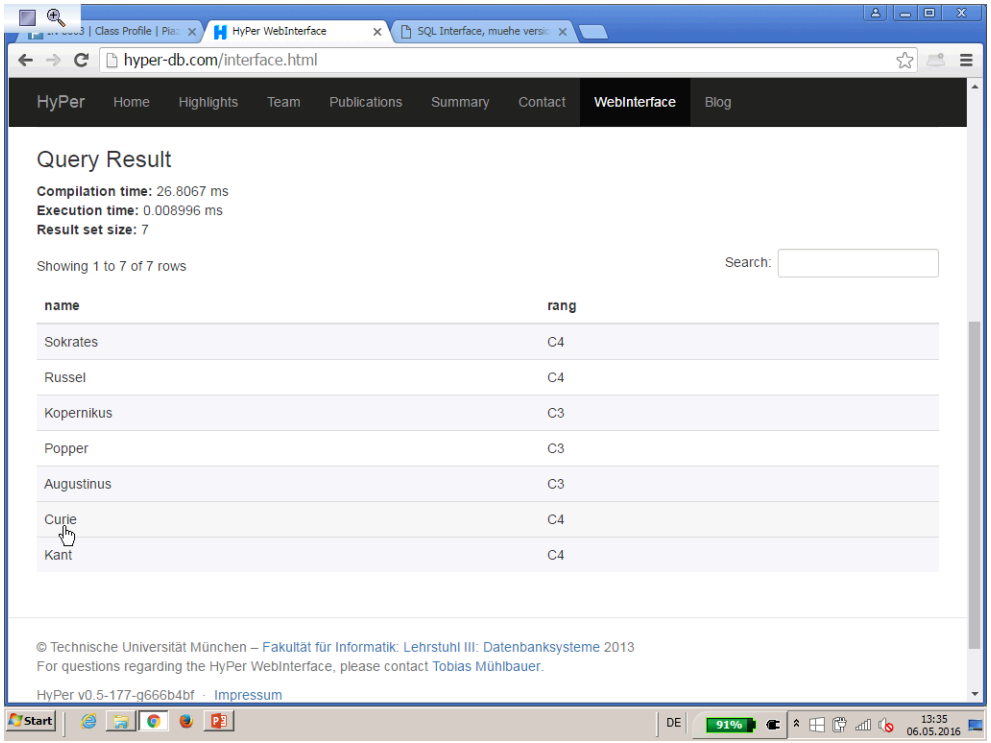
Duration: 87:19 min

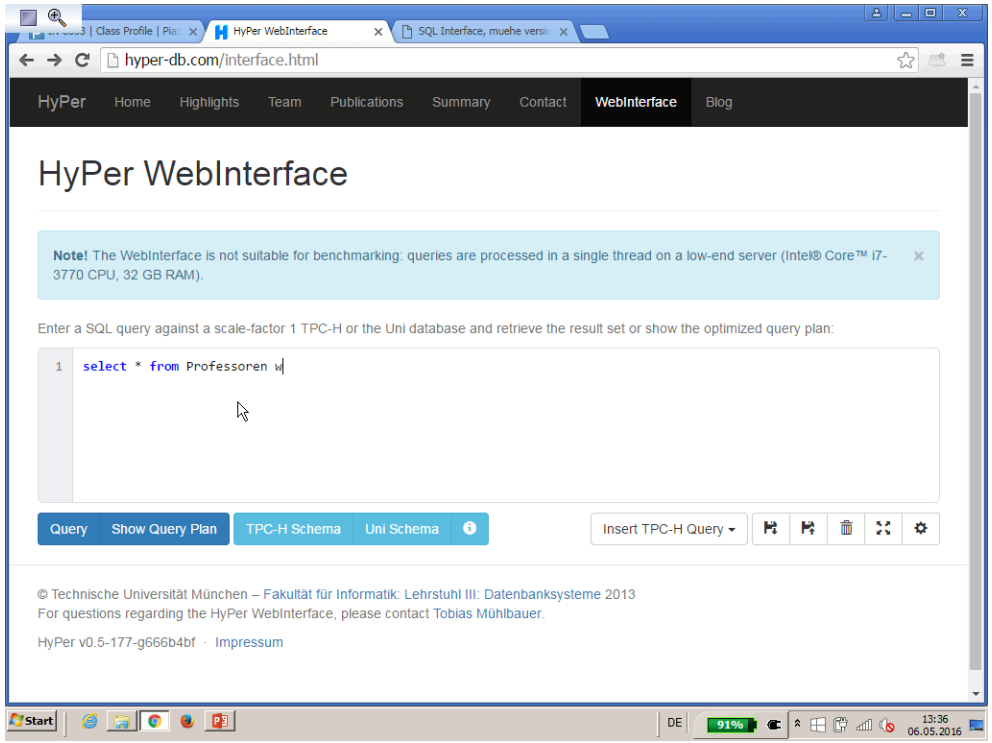
Pages: 77

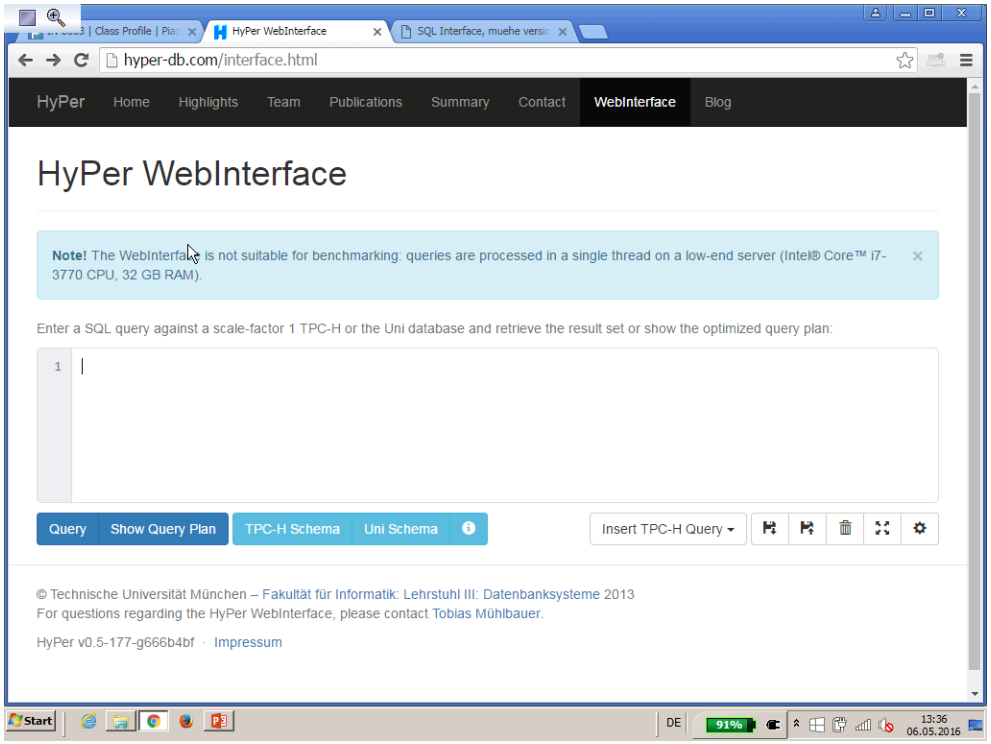


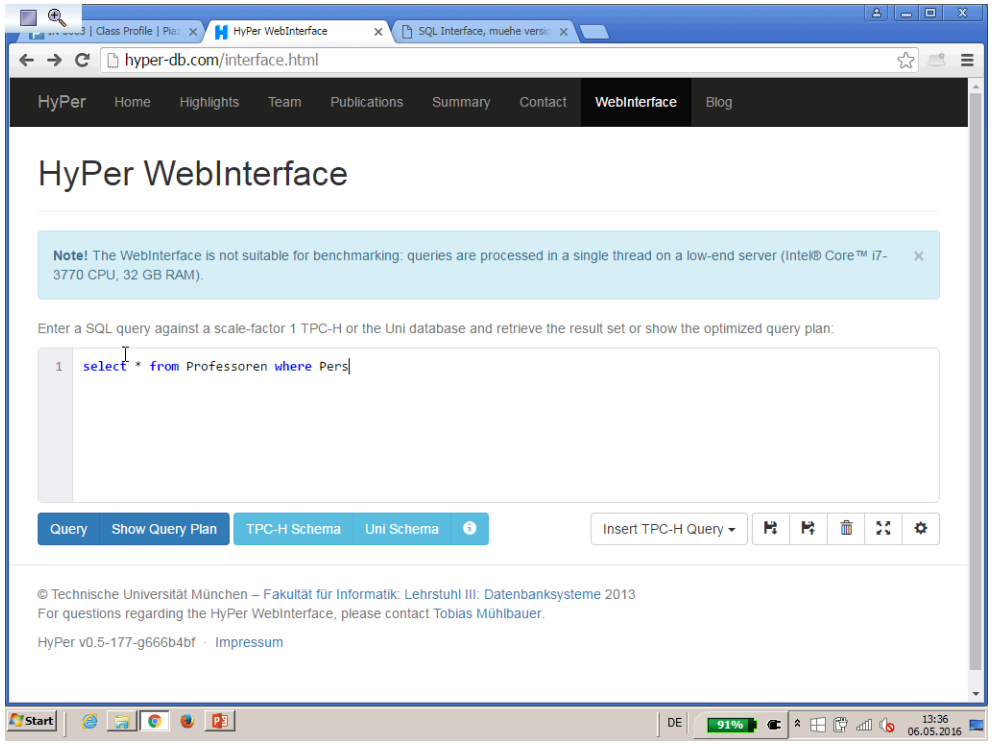





hyper-db.com/interface.html
HyPer
Home
Highlights
Team
Publications
Summary
Contact
WebInterface
Blog
Query Result
Compilation time: 26.8067 ms
Execution time: 0.008996 ms
Result set size: 7
Showing 1 to 7 of 7 rows
Search:
name
rang
Sokrates C4
Russel C4
Kopernikus C3
Popper C3
Augustinus C3
Curie C4
Kant C4
© Technische Universität München – Fakultät für Informatik: Lehrstuhl III: Datenbanksysteme 2013
For questions regarding the HyPer WebInterface, please contact Tobias Mühlbauer.
HyPer v0.5-177-g666b4bf · Impressum


hyper-db.com/interface.html
HyPer
Home
Highlights
Team
Publications
Summary
Contact
WebInterface
Blog
HyPer WebInterface
Note! The WebInterface is not suitable for benchmarking: queries are processed in a single thread on a low-end server (Intel® Core™ i7-3770 CPU, 32 GB RAM).
Enter a SQL query against a scale-factor 1 TPC-H or the Uni database and retrieve the result set or show the optimized query plan:
1 select * from Professoren w
Query
Show Query Plan
TPC-H Schema
Uni Schema
Insert TPC-H Query
© Technische Universität München – Fakultät für Informatik: Lehrstuhl III: Datenbanksysteme 2013
For questions regarding the HyPer WebInterface, please contact Tobias Mühlbauer.
HyPer v0.5-177-g666b4bf · Impressum


hyper-db.com/interface.html
HyPer
Home
Highlights
Team
Publications
Summary
Contact
WebInterface
Blog
HyPer WebInterface
Note! The WebInterface is not suitable for benchmarking: queries are processed in a single thread on a low-end server (Intel® Core™ i7-3770 CPU, 32 GB RAM).
Enter a SQL query against a scale-factor 1 TPC-H or the Uni database and retrieve the result set or show the optimized query plan:
1
Query
Show Query Plan
TPC-H Schema
Uni Schema
Insert TPC-H Query
© Technische Universität München – Fakultät für Informatik: Lehrstuhl III: Datenbanksysteme 2013
For questions regarding the HyPer WebInterface, please contact Tobias Mühlbauer.
HyPer v0.5-177-g666b4bf · Impressum


hyper-db.com/interface.html
HyPer
Home
Highlights
Team
Publications
Summary
Contact
WebInterface
Blog
HyPer WebInterface
Note! The WebInterface is not suitable for benchmarking: queries are processed in a single thread on a low-end server (Intel® Core™ i7-3770 CPU, 32 GB RAM).
Enter a SQL query against a scale-factor 1 TPC-H or the Uni database and retrieve the result set or show the optimized query plan:
1 select * from Professoren where Pers
Query
Show Query Plan
TPC-H Schema
Uni Schema
Insert TPC-H Query
© Technische Universität München – Fakultät für Informatik: Lehrstuhl III: Datenbanksysteme 2013
For questions regarding the HyPer WebInterface, please contact Tobias Mühlbauer.
HyPer v0.5-177-g666b4bf · Impressum

Anfragen über mehrere Relationen

Beispiel: Welcher Prof liest Mäeutik?

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

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

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

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

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

Vorlesungen			
VorlNr	Titel	SWS	gelesenVon
5001	Grundzüge	4	2137
5041	Ethik	4	2125
...	...	...	...
5049	Mäeutik	2	2125
...	...	...	...
4630	Die 3 Kritiken	4	2137

X

Professoren x Vorlesungen							
PersNr	Name	Rang	Raum	VorlNr	Titel	SWS	gelesenVon
2125	Sokrates	C4	226	5001	Grundzüge	4	2137
2125	Sokrates	C4	226	5041	Ethik	4	2125
...	...	...	...	...	...	...	...
2125	Sokrates	C4	226	5049	Mäeutik	2	2125
2126	Russel	C4	232	5001	Grundzüge	4	2137
...	...	...	...	...	...	...	...
2137	Kant	C4	7	4630	Die 3 Kritiken	4	2137

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

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

Professoren x Vorlesungen							
PersNr	Name	Rang	Raum	VorlNr	Titel	SWS	gelesenVon
2125	Sokrates	C4	226	5001	Grundzüge	4	2137
2125	Sokrates	C4	226	5041	Ethik	4	2125
...	...	...	...	...	...	...	...
2125	Sokrates	C4	226	5049	Mäeutik	2	2125
2126	Russel	C4	232	5001	Grundzüge	4	2137
...	...	...	...	...	...	...	...
2137	Kant	C4	7	4630	Die 3 Kritiken	4	2137

↓ Selektion

$\sigma_{PersNr=gelesenVon\ \wedge Titel='Mäeutik'}$(Professoren x Vorlesungen)							
PersNr	Name	Rang	Raum	VorlNr	Titel	SWS	gelesenVon
2125	Sokrates	C4	226	5049	Mäeutik	2	2125

↓ Projektion

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

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

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

Professoren x Vorlesungen							
PersNr	Name	Rang	Raum	VorlNr	Titel	SWS	gelesenVon
2125	Sokrates	C4	226	5001	Grundzüge	4	2137
2125	Sokrates	C4	226	5041	Ethik	4	2125
...	...	...	...	...	...	...	...
2125	Sokrates	C4	226	5049	Mäeutik	2	2125
2126	Russel	C4	232	5001	Grundzüge	4	2137
...	...	...	...	...	...	...	...
2137	Kant	C4	7	4630	Die 3 Kritiken	4	2137

↓ Selektion

$\sigma_{PersNr=gelesenVon\ \wedge Titel='Mäeutik'}$(Professoren x Vorlesungen)							
PersNr	Name	Rang	Raum	VorlNr	Titel	SWS	gelesenVon
2125	Sokrates	C4	226	5049	Mäeutik	2	2125

↓ Projektion

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

Name, Titel **from** Professoren, Vorlesungen **where** PersNr=gelesenVon and Titel='Mäeutik';	$\Pi_{Name,Titel}\,\sigma_{PersNr=gelesenVon\ \wedge Titel='Mäeutik'}$(Professoren x Vorlesungen)

Professoren x Vorlesungen							
PersNr	Name	Rang	Raum	VorlNr	Titel	SWS	gelesenVon
2125	Sokrates	C4	226	5001	Grundzüge	4	2137
2125	Sokrates	C4	226	5041	Ethik	4	2125
...	...	...	...	...	...	...	...
2125	Sokrates	C4	226	5049	Mäeutik	2	2125
2126	Russel	C4	232	5001	Grundzüge	4	2137
...	...	...	...	...	...	...	...
2137	Kant	C4	7	4630	Die 3 Kritiken	4	2137

↓ Selektion

$\sigma_{PersNr=gelesenVon\ \wedge Titel='Mäeutik'}$(Professoren x Vorlesungen)							
PersNr	Name	Rang	Raum	VorlNr	Titel	SWS	gelesenVon
2125	Sokrates	C4	226	5049	Mäeutik	2	2125

↓ Projektion

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

Aufgabe

Geben Sie ein `select` statement an, das **äquivalent** zu folgendem (sehr einfachen !) **relationale Algebra Ausruck** ist!

$$\Pi_{v1.Vorgaenger}\left(\sigma_{v.VorlNr=v2.Nachfolger \wedge v2.Vorgaenger=v1.Nachfolger \wedge v.Titel='Bioethik'}\left(\rho_v(Vorlesungen) \times \rho_{v2}(voraussetzen) \times \rho_{v1}(voraussetzen)\right)\right)$$

```
select v1.Vorgänger
  from voraussetzen v1, voraussetzen v2,Vorlesungen v
    where v.VorlNr = v2.Nachfolger
      and v2.Vorgänger = v1.Nachfolger
      and v.Titel = 'Bioethik';
```

Aufgabe

Geben Sie ein `select` statement an, das **äquivalent** zu folgendem (sehr einfachen !) **relationale Algebra Ausruck** ist!

$$\Pi_{v1.Vorgaenger}\left(\sigma_{v.VorlNr=v2.Nachfolger \wedge v2.Vorgaenger=v1.Nachfolger \wedge v.Titel='Bioethik'}\left(\rho_v(Vorlesungen) \times \rho_{v2}(voraussetzen) \times \rho_{v1}(voraussetzen)\right)\right)$$

```
select v1.Vorgänger
  from voraussetzen v1, voraussetzen v2,Vorlesungen v
    where v.VorlNr = v2.Nachfolger
      and v2.Vorgänger = v1.Nachfolger
      and v.Titel = 'Bioethik';
```

Aufgabe

Geben Sie ein `select` statement an, das **äquivalent** zu folgendem (sehr einfachen !) **relationale Algebra Ausruck** ist!

$$\Pi_{v1.Vorgaenger}\left(\sigma_{v.VorlNr=v2.Nachfolger \wedge v2.Vorgaenger=v1.Nachfolger \wedge v.Titel='Bioethik'}\left(\rho_v(Vorlesungen) \times \rho_{v2}(voraussetzen) \times \rho_{v1}(voraussetzen)\right)\right)$$

```
select v1.Vorgänger
  from voraussetzen v1, voraussetzen v2,Vorlesungen v
    where v.VorlNr = v2.Nachfolger
      and v2.Vorgänger = v1.Nachfolger
      and v.Titel = 'Bioethik';
```

Anfragen über mehrere Relationen: Tupelvariablen

Bsp.: Welche Studenten hören welche Vorlesungen?

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

alternativ (empfohlen):Einsatz von Tupelvariablen):

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

(entspricht ρ Operator)

Anfragen über mehrere Relationen: Tupelvariablen

Bsp.: Welche Studenten hören welche Vorlesungen?

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

alternativ (empfohlen):Einsatz von Tupelvariablen):

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

(entspricht ρ Operator)

Anfragen über mehrere Relationen: Tupelvariablen

Bsp.: Welche Studenten hören welche Vorlesungen?

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

alternativ (empfohlen):Einsatz von Tupelvariablen):

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

(entspricht ρ Operator)

select <---> Übersetzung in die relationale Algebra

Allgemeine (ungeschachtelte) SQL-Anfrage :

select $A_1, \ldots, A_n$
from $R_1, \ldots, R_k$
where P;

$\Leftrightarrow$

Übersetzung in die relationale Algebra:

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

Anfragen über mehrere Relationen: Tupelvariablen

Bsp.: Welche Studenten hören welche Vorlesungen?

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

alternativ (empfohlen):Einsatz von Tupelvariablen):

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

(entspricht ρ Operator)

Mengenoperationen union, intersect, minus

- Beispiel für union:

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

$\Pi_{Name}(Assistenten) \cup \Pi_{Name}(Professoren)$

- intersect, minus: analog

Geschachtelte Unteranfragen: exists

Existenzquantor exists

Bsp.: Profs die keine Vorlesung halten:

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


Geschachtelte Unteranfragen: exists

Existenzquantor exists

Bsp.: Profs die keine Vorlesung halten:

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

Geschachtelte Unteranfragen: exists

Existenzquantor exists

Bsp.: Profs die keine Vorlesung halten:

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


Geschachtelte Unteranfragen: exists

Existenzquantor exists

Bsp.: Profs die keine Vorlesung halten:

select p.Name
from Professoren p
where not exists (**select** *
from Vorlesungen v
where v.gelesenVon = p.PersNr);

Geschachtelte Unteranfragen: exists

Existenzquantor exists

Bsp.: Profs die keine Vorlesung halten:

select p.Name
from Professoren p
where not exists (**select** *
from Vorlesungen v
where v.gelesenVon = p.PersNr);

Geschachtelte Unteranfragen: exists

Existenzquantor exists

Bsp.: Profs die keine Vorlesung halten:

select p.Name
from Professoren p
where not exists (**select** *
from Vorlesungen v
where v.gelesenVon = p.PersNr);

Geschachtelte Unteranfragen: exists

Existenzquantor exists

Bsp.: Profs die keine Vorlesung halten:

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


Geschachtelte Unteranfragen: in

Bsp.: Profs die keine Vorlesung halten:

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

Korrelation

Bsp.: Profs die keine Vorlesung halten: Variante mit in

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


Geschachtelte Unteranfragen: in

Bsp.: Profs die keine Vorlesung halten:

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

Korrelation

Bsp.: Profs die keine Vorlesung halten: Variante mit in

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


Geschachtelte Unteranfragen: in

Bsp.: Profs die keine Vorlesung halten:

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

Korrelation

Bsp.: Profs die keine Vorlesung halten: Variante mit in

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

Keine Korrelation (effizienter)

Geschachtelte Unteranfragen: in

Bsp.: Profs die keine Vorlesung halten:

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

Korrelation

Bsp.: Profs die keine Vorlesung halten: Variante mit in

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

Keine Korrelation (effizienter)

Der Vergleich mit "all"

Bsp: Finde die Studis, die am längsten studieren!

```
select Name
from Studenten
where Semester >= all ( select Semester
                        from Studenten);
```


Der Vergleich mit "all"

Bsp: Finde die Studis, die am längsten studieren!

```
select Name
from Studenten
where Semester >= all ( select Semester
                        from Studenten);
```


Der Vergleich mit "all"

Bsp: Finde die Studis, die am längsten studieren!

```
select Name
from Studenten
where Semester >= all ( select Semester
                        from Studenten);
```

Gruppierung und Aggregation

- γ nicht mehr Teil der relationalen Algebra.
  Dennoch sehr wichtiger Operator (siehe `select` mit `groupby`)

- $\gamma_{A;\,f}(E)$ gruppiert nach Attribut(menge) A und wendet auf jede Gruppe (eine Menge von) Aggregatfunktion(en) f (count(), max(), min(), sum(), avg() etc.) an

$\gamma_{Semester;\ count(*)}(Studenten)$

$\gamma_{Semester;count(*)}(Studenten)$	
Semester	count(*)
18	1
12	1
10	1
8	1
6	1
3	1
2	2

$\gamma_{gelesenVon;\ count(*),sum(SWS)}(Vorlesungen)$

$\gamma_{gelesenVon;\ count(*),sum(SWS)}(Vorlesungen)$		
gelesenVon	count(*)	sum(SWS)
2125	3	10
2126	3	8
2133	1	2
2134	1	2
2137	2	8

Der Vergleich mit "all"

Bsp: Finde die Studis, die am längsten studieren!

```
select Name
from Studenten
where Semester >= all ( select Semester
                        from Studenten);
```

effizienter:

```
select Name
from Studenten
where Semester >= ( select max(Semester)
                    from Studenten);
```


Der Vergleich mit "all"

Bsp: Finde die Studis, die am längsten studieren!

```
select Name
from Studenten
where Semester >= all ( select Semester
                        from Studenten);
```

effizienter:

```
select Name
from Studenten
where Semester >= ( select max(Semester)
                    from Studenten);
```

Bemerkung (1) : neben **all** gibt es auch **any**

Bemerkung (2): entspricht NICHT dem Allquantor;
"Studis die **alle** 4std Vorl. hoeren" geht damit NICHT

Bemerkung (1) : neben **all** gibt es auch **any**

Bemerkung (2): entspricht NICHT dem Allquantor;
"Studis die **alle** 4std Vorl. hoeren" geht damit NICHT

Aggregatfunktionen und Gruppierung

- Aggregatfunktionen: **avg, max, min, count, sum, median**

  Beispiel:

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

- Gruppierung:

  Beispiel:

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

Aggregatfunktionen und Gruppierung

- Aggregatfunktionen: **avg, max, min, count, sum, median**

  Beispiel:

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

- Gruppierung:

  Beispiel:

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

Aggregatfunktionen und Gruppierung

- Aggregatfunktionen: **avg, max, min, count, sum, median**

  Beispiel:

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

- Gruppierung:

  Beispiel:

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

Aggregatfunktionen und Gruppierung

Gruppierung: Beispiel 2: Welcher C4 Professor liest im Durchschnitt lange Vorlesungen?

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

Gruppierung: Beispiel 2: Welcher C4 Professor liest im Durchschnitt lange Vorlesungen?

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


Aggregatfunktionen und Gruppierung

Gruppierung: Beispiel 2: Welcher C4 Professor liest im Durchschnitt lange Vorlesungen?

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

194

Aggregatfunktionen und Gruppierung

Gruppierung: Beispiel 2: Welcher C4 Professor liest im Durchschnitt lange Vorlesungen?

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

194

Aggregatfunktionen und Gruppierung

Gruppierung: Beispiel 2: Welcher C4 Professor liest im Durchschnitt lange Vorlesungen?

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

194

Gruppierung: Beispiel 2: Welcher C4 Professor liest im Durchschnitt lange Vorlesungen?

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


Aggregatfunktionen und Gruppierung

Gruppierung: Beispiel 2: Welcher C4 Professor liest im Durchschnitt lange Vorlesungen?

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


Aggregatfunktionen und Gruppierung

Gruppierung: Beispiel 2: Welcher C4 Professor liest im Durchschnitt lange Vorlesungen?


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


Aggregatfkt.: wird für jede Gruppe getrennt berechnet

wie gehabt: sortiert Tupel aus Kreuzprodukt aus (Theta Join bzw. Selektion)

sortiert bstimmte Gruppen aus

Vorlesung x Professoren

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum
5001	Grundzüge	4	2137	2125	Sokrates	C4	226
5001	Grundzüge	4	2137	2126	Russel	C4	232
...	...	...	...	...	...	...	...
5001	Grundzüge	4	2137	2137	Kant	C4	7
5041	Ethik	4	2125	2125	Sokrates	C4	226
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7

Selektion mit where Bedingung

$\sigma_{\text{gelesenVon = PersNr} \wedge \text{Rang = 'C4'}}$(Vorlesung x Professoren)

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


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

$\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}$(Vorlesung x Professoren)

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

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

Gruppierung mit groupBy

$\gamma_{gelesenVon, Name}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}$(Vorlesung x Professoren))

VorlNr	Titel	SWS	gelesen Von	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	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
5001	Grundzüge	4	2137	2137	Kant	C4	7

$\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}$(Vorlesung x Professoren)

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

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

Gruppierung mit groupBy

$\gamma_{gelesenVon, Name}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}$(Vorlesung x Professoren))

VorlNr	Titel	SWS	gelesen Von	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	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
5001	Grundzüge	4	2137	2137	Kant	C4	7

$\gamma_{gelesenVon, Name}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}$(Vorlesung x Professoren))

VorlNr	Titel	SWS	gelesen Von	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	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
5001	Grundzüge	4	2137	2137	Kant	C4	7

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

Berechnung der Aggregatfunktion avg

$\gamma_{gelesenVon, Name; avg(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}$(Vorlesung x Professoren))

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum	avg(SWS)
5041	Ethik	4	2125	2125	Sokrates	C4	226	3.333
5049	Mäeutik	2	2125	2125	Sokrates	C4	226	
4052	Logik	4	2125	2125	Sokrates	C4	226	
5043	Erkenntnistheorie	3	2126	2126	Russel	C4	232	2.666
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	4
5001	Grundzüge	4	2137	2137	Kant	C4	7	

$\gamma_{gelesenVon, Name; avg(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}$(Vorlesung x Professoren))

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum	avg(SWS)
5041	Ethik	4	2125	2125	Sokrates	C4	226	3.333
5049	Mäeutik	2	2125	2125	Sokrates	C4	226	
4052	Logik	4	2125	2125	Sokrates	C4	226	
5043	Erkenntnistheorie	3	2126	2126	Russel	C4	232	2.666
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	4
5001	Grundzüge	4	2137	2137	Kant	C4	7	

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

Selektion der Gruppen mit having Bedingung

$\sigma_{avg(SWS)>3}(\gamma_{gelesenVon, Name; avg(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}$(Vorlesung x Professoren)))

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum	avg(SWS)
5041	Ethik	4	2125	2125	Sokrates	C4	226	3.333
5049	Mäeutik	2	2125	2125	Sokrates	C4	226	
4052	Logik	4	2125	2125	Sokrates	C4	226	
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7	4
5001	Grundzüge	4	2137	2137	Kant	C4	7	

$\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren)))$

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

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum	avg(SWS)
5041	Ethik	4	2125	2125	Sokrates	C4	226	3.333
5049	Mäeutik	2	2125	2125	Sokrates	C4	226	
4052	Logik	4	2125	2125	Sokrates	C4	226	
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7	4
5001	Grundzüge	4	2137	2137	Kant	C4	7	

Berechnung von Aggregatfunktion sum

$\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS),sum(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren)))$

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum	avg(SWS)	sum(SWS)
5041	Ethik	4	2125	2125	Sokrates	C4	226	3.333	10
5049	Mäeutik	2	2125	2125	Sokrates	C4	226		
4052	Logik	4	2125	2125	Sokrates	C4	226		
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7	4	8
5001	Grundzüge	4	2137	2137	Kant	C4	7		

Projektion

$\Pi_{gelesenVon,sum(SWS)}(\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS),sum(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren))))$

gelesen Von	Name	sum(SWS)
2125	Sokrates	10
2137	Kant	8

$\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren)))$

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

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum	avg(SWS)
5041	Ethik	4	2125	2125	Sokrates	C4	226	3.333
5049	Mäeutik	2	2125	2125	Sokrates	C4	226	
4052	Logik	4	2125	2125	Sokrates	C4	226	
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7	4
5001	Grundzüge	4	2137	2137	Kant	C4	7	

Berechnung von Aggregatfunktion sum

$\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS),sum(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren)))$

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum	avg(SWS)	sum(SWS)
5041	Ethik	4	2125	2125	Sokrates	C4	226	3.333	10
5049	Mäeutik	2	2125	2125	Sokrates	C4	226		
4052	Logik	4	2125	2125	Sokrates	C4	226		
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7	4	8
5001	Grundzüge	4	2137	2137	Kant	C4	7		

Projektion

$\Pi_{gelesenVon,sum(SWS)}(\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS),sum(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren))))$

gelesen Von	Name	sum(SWS)
2125	Sokrates	10
2137	Kant	8

200

$\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren)))$

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

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum	avg(SWS)
5041	Ethik	4	2125	2125	Sokrates	C4	226	3.333
5049	Mäeutik	2	2125	2125	Sokrates	C4	226	
4052	Logik	4	2125	2125	Sokrates	C4	226	
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7	4
5001	Grundzüge	4	2137	2137	Kant	C4	7	

Berechnung von Aggregatfunktion sum

$\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS),sum(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren)))$

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum	avg(SWS)	sum(SWS)
5041	Ethik	4	2125	2125	Sokrates	C4	226	3.333	10
5049	Mäeutik	2	2125	2125	Sokrates	C4	226		
4052	Logik	4	2125	2125	Sokrates	C4	226		
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7	4	8
5001	Grundzüge	4	2137	2137	Kant	C4	7		

Projektion

$\Pi_{gelesenVon,sum(SWS)}(\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS),sum(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren))))$

gelesen Von	Name	sum(SWS)
2125	Sokrates	10
2137	Kant	8

200

Aggregatfunktionen und Gruppierung

- Eigentliche Verarbeitung ist nicht so schematisch
- Anwendung von Aggregatfunktionen: SQL erzeugt eigentlich pro Gruppe nur **ein** Ergebnistupel

  ⇒ Es dürfen - außer den aggregierten – in der **select**-Klausel nur Attribute aufgeführt werden, die auch in der **group by**-Klausel aufgeführt werden!
- Beispiel: Wie viele Vorlesungen besucht jeder Student?

```
select count(h.VorlNr), h.MatrNr, s.name
    from hoeren h, Studenten s
    where h.MatrNR = s.MatrNR
    group by h.MatrNr, s.Name
```


$\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren)))$

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum	avg(SWS)
5041	Ethik	4	2125	2125	Sokrates	C4	226	3.333
5049	Mäeutik	2	2125	2125	Sokrates	C4	226	
4052	Logik	4	2125	2125	Sokrates	C4	226	
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7	4
5001	Grundzüge	4	2137	2137	Kant	C4	7	

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

Berechnung von Aggregatfunktion sum

$\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS),sum(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren)))$

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum	avg(SWS)	sum(SWS)
5041	Ethik	4	2125	2125	Sokrates	C4	226	3.333	10
5049	Mäeutik	2	2125	2125	Sokrates	C4	226		
4052	Logik	4	2125	2125	Sokrates	C4	226		
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7	4	8
5001	Grundzüge	4	2137	2137	Kant	C4	7		

Projektion

$\Pi_{gelesenVon,sum(SWS)}(\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS),sum(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren))))$

gelesen Von	Name	sum(SWS)
2125	Sokrates	10
2137	Kant	8

$\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren)))$

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum	avg(SWS)
5041	Ethik	4	2125	2125	Sokrates	C4	226	3.333
5049	Mäeutik	2	2125	2125	Sokrates	C4	226	
4052	Logik	4	2125	2125	Sokrates	C4	226	
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7	4
5001	Grundzüge	4	2137	2137	Kant	C4	7	

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

Berechnung von Aggregatfunktion sum

$\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS),sum(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren)))$

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum	avg(SWS)	sum(SWS)
5041	Ethik	4	2125	2125	Sokrates	C4	226	3.333	10
5049	Mäeutik	2	2125	2125	Sokrates	C4	226		
4052	Logik	4	2125	2125	Sokrates	C4	226		
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7	4	8
5001	Grundzüge	4	2137	2137	Kant	C4	7		

Projektion

$\Pi_{gelesenVon,sum(SWS)}(\sigma_{avg(SWS)>3}(\gamma_{gelesenVon,\ Name;\ avg(SWS),sum(SWS)}(\sigma_{gelesenVon = PersNr \wedge Rang = 'C4'}(Vorlesung \times Professoren))))$

gelesen Von	Name	sum(SWS)
2125	Sokrates	10
2137	Kant	8

Aggregatfunktionen und Gruppierung

- Eigentliche Verarbeitung ist nicht so schematisch
- Anwendung von Aggregatfunktionen: SQL erzeugt eigentlich pro Gruppe nur **ein** Ergebnistupel

  ⇒ Es dürfen - außer den aggregierten – in der **select**-Klausel nur Attribute aufgeführt werden, die auch in der **group by**-Klausel aufgeführt werden!
- Beispiel: Wie viele Vorlesungen besucht jeder Student?

```
select count(h.VorlNr), h.MatrNr, s.name
    from hoeren h, Studenten s
    where h.MatrNR = s.MatrNR
    group by h.MatrNr, s.Name
```

Aggregatfunktionen und Gruppierung

- Eigentliche Verarbeitung ist nicht so schematisch
- Anwendung von Aggregatfunktionen: SQL erzeugt eigentlich pro Gruppe nur **ein** Ergebnistupel

  ⇒ Es dürfen - außer den aggregierten – in der **select**-Klausel nur Attribute aufgeführt werden, die auch in der **group by**-Klausel aufgeführt werden!
- Beispiel: Wie viele Vorlesungen besucht jeder Student?

```
select count(h.VorlNr), h.MatrNr, s.name
       from hoeren h, Studenten s
       where h.MatrNR = s.MatrNR
       group by h.MatrNr, s.Name
```


Aggregatfunktionen und Gruppierung

- Eigentliche Verarbeitung ist nicht so schematisch
- Anwendung von Aggregatfunktionen: SQL erzeugt eigentlich pro Gruppe nur **ein** Ergebnistupel

  ⇒ Es dürfen - außer den aggregierten – in der **select**-Klausel nur Attribute aufgeführt werden, die auch in der **group by**-Klausel aufgeführt werden!
- Beispiel: Wie viele Vorlesungen besucht jeder Student?

```
select count(h.VorlNr), h.MatrNr, s.name
       from hoeren h, Studenten s
       where h.MatrNR = s.MatrNR
       group by h.MatrNr, s.Name
```

201

Geschachtelte Anfragen

1. Unteranfrage in der **where**-Klausel

   Beispiel: Welche Prüfungen sind besser als durchschnittlich verlaufen?

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


Geschachtelte Anfragen

2. Unteranfrage in der **select**-Klausel

   - Für jedes Ergebnistupel wird die Unteranfrage ausgeführt

   - Beispiel: Ermittele Lehrbelastung der Profs:

```
select p.PersNr, p.Name, ( select sum (v.SWS) as Lehrbelastung
                           from Vorlesungen v
                           where v.gelesenVon=p.PersNr )
   from Professoren p;
```

   Unteranfrage ist hier korreliert (greift auf Attribute der umschließenden Anfrage zu)

Geschachtelte Anfragen

2. Unteranfrage in der **select**-Klausel

- Für jedes Ergebnistupel wird die Unteranfrage ausgeführt

- Beispiel: Ermittele Lehrbelastung der Profs:

```
select p.PersNr, p.Name, ( select sum (v.SWS) as Lehrbelastung
                           from Vorlesungen v
                           where v.gelesenVon=p.PersNr )
from Professoren p;
```

Unteranfrage ist hier korreliert (greift auf Attribute der umschließenden Anfrage zu)

$\gamma_{gelesenVon,\ Name}(\sigma_{gelesenVon\ =\ PersNr\ \wedge\ Rang\ =\ 'C4'}(Vorlesung \times Professoren))$

VorlNr	Titel	SWS	gelesen Von	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	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
5001	Grundzüge	4	2137	2137	Kant	C4	7

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

Berechnung der Aggregatfunktion avg

$\gamma_{gelesenVon,\ Name;\ avg(SWS)}(\sigma_{gelesenVon\ =\ PersNr\ \wedge\ Rang\ =\ 'C4'}(Vorlesung \times Professoren))$

VorlNr	Titel	SWS	gelesen Von	PersNr	Name	Rang	Raum	avg(SWS)
5041	Ethik	4	2125	2125	Sokrates	C4	226	
5049	Mäeutik	2	2125	2125	Sokrates	C4	226	3.333
4052	Logik	4	2125	2125	Sokrates	C4	226	
5043	Erkenntnistheorie	3	2126	2126	Russel	C4	232	
5052	Wissenschaftstheorie	3	2126	2126	Russel	C4	232	2.666
5216	Bioethik	2	2126	2126	Russel	C4	232	
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7	4
5001	Grundzüge	4	2137	2137	Kant	C4	7	

Geschachtelte Anfragen

2. Unteranfrage in der **select**-Klausel

- Für jedes Ergebnistupel wird die Unteranfrage ausgeführt

- Beispiel: Ermittele Lehrbelastung der Profs:

```
select p.PersNr, p.Name, ( select sum (v.SWS) as Lehrbelastung
                           from Vorlesungen v
                           where v.gelesenVon=p.PersNr )
from Professoren p;
```

Unteranfrage ist hier korreliert (greift auf Attribute der umschließenden Anfrage zu)

203

Geschachtelte Anfragen

2. Unteranfrage in der **select**-Klausel

- Für jedes Ergebnistupel wird die Unteranfrage ausgeführt

- Beispiel: Ermittele Lehrbelastung der Profs:

```
select p.PersNr, p.Name, ( select sum (v.SWS) as Lehrbelastung
                           from Vorlesungen v
                           where v.gelesenVon=p.PersNr )
from Professoren p;
```

Unteranfrage ist hier korreliert (greift auf Attribute der umschließenden Anfrage zu)

203

Geschachtelte Anfragen

3. Unteranfrage in der **from**-Klausel:

--> Verwertung der Ergebnismenge einer Unteranfrage:

Beispiel: Wer hört mehr als 2 Vorlesungen?

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


Geschachtelte Anfragen

3. Unteranfrage in der **from**-Klausel:

--> Verwertung der Ergebnismenge einer Unteranfrage:

Beispiel: Wer hört mehr als 2 Vorlesungen?

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

206

Geschachtelte Anfragen

3. Unteranfrage in der **from**-Klausel:

--> Verwertung der Ergebnismenge einer Unteranfrage:

Beispiel: Wer hört mehr als 2 Vorlesungen?

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

206

Korrelierte versus unkorrelierte Unteranfragen

Bsp: Welche Studenten sind älter als Professoren?

- korrelierte Formulierung:

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

- unkorrelierte Formulierung:

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

Vorteil: Unteranfrage muss nur einmal ausgewertet werden.

Entschachtelung korrelierter Unteranfragen durch Join

Bsp.: Welcher Assistent hat einen Boss der jünger ist als er selbst?

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

--> Entschachtelung durch Join:

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


Entschachtelung korrelierter Unteranfragen durch Join

Bsp.: Welcher Assistent hat einen Boss der jünger ist als er selbst?

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

--> Entschachtelung durch Join:

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

205