

Dr. Christoph M.A.
Scheich

Angewandte Informatik
Hochschule Fulda

Christoph.scheich@infor
matik.hs-fulda.de

Datenbanksystem
e

Dr. Christoph M.A.

Scheich

Angewandte Informatik

Hochschule Fulda

Christoph.scheich@informatik.hs-fulda.de

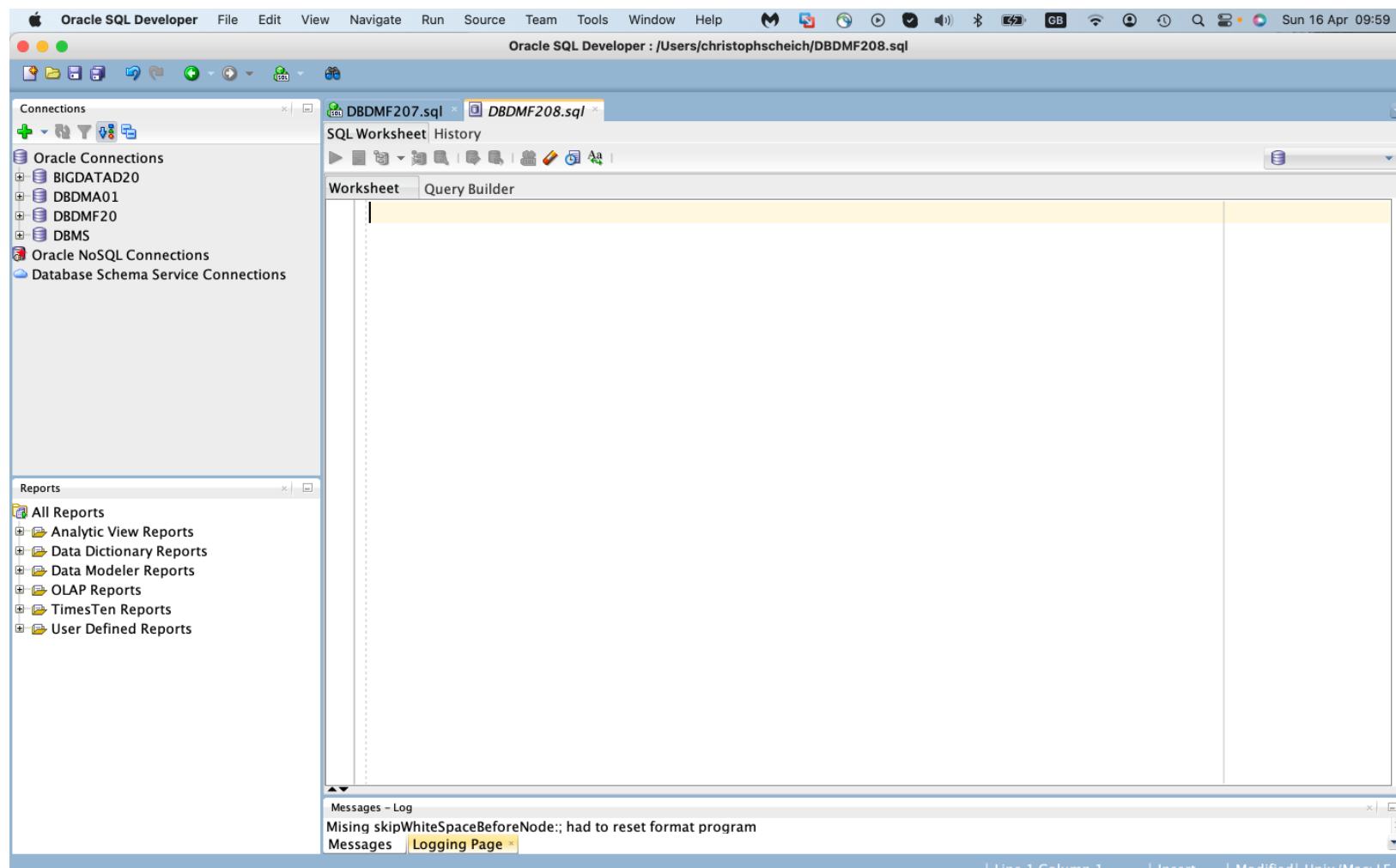
SQL – Structured
Query Language,
Teil 1

- Datentypen
- Veränderungen am Datenbestand
- Einfache Abfragen
- Abfragen über mehrere Relationen
- Geschachtelte Abfragen
- Aggregationen
- Entschachtelung von Abfragen

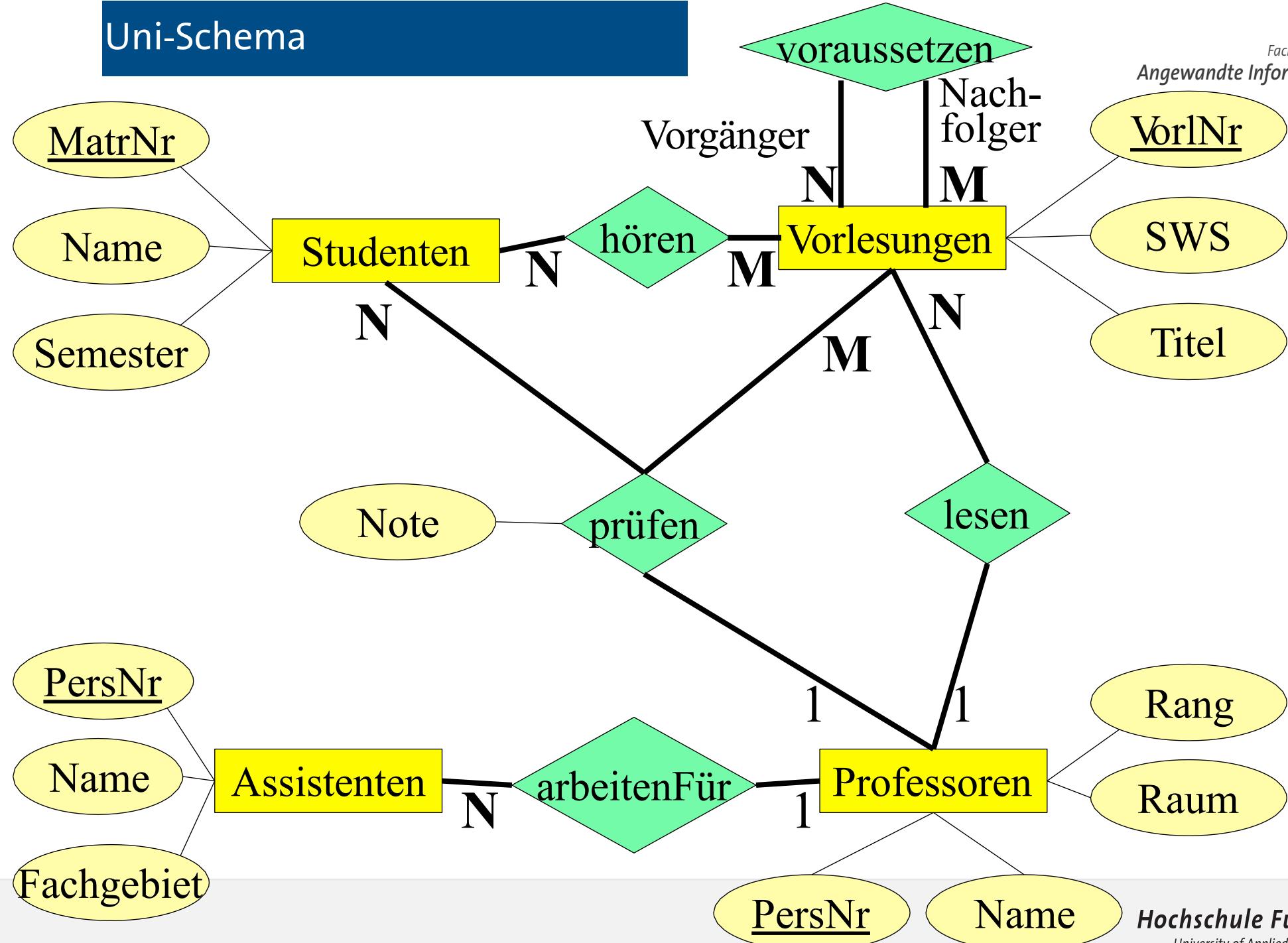
- Standardisierte
 - Datendefinitions (DDL)-
 - Datenmanipulations (DML)-
 - Anfrage (Query)-Sprache
- Derzeit aktueller Standard ist SQL 99 und SQL3 (2003)
 - objektrelationale Erweiterung
- Für praktische Übungen steht eine Web-Seite zur Verfügung:
 - <http://www-db.in.tum.de/research/publications/books/DBMSseinf>
- Man kann eigene Relationen anlegen und/oder die Uni-DB verwenden
- Oracle „liegt dahinter“



SQL Übungen



Uni-Schema



Professoren				Studenten			Vorlesungen			
PersNr	Name	Rang	Raum	MatrNr	Name	Semester	VorlNr	Titel	SWS	gelesen von
2125	Sokrates	C4	226	24002	Xenokrates	18	5001	Grundzüge	4	2137
2126	Russel	C4	232	25403	Jonas	12	5041	Ethik	4	2125
2127	Kopernikus	C3	310	26120	Fichte	10	5043	Erkenntnistheorie	3	2126
2133	Popper	C3	52	26830	Aristoxenos	8	5049	Mäeutik	2	2125
2134	Augustinus	C3	309	27550	Schopenhauer	6	4052	Logik	4	2125
2136	Curie	C4	36	28106	Carnap	3	5052	Wissenschaftstheorie	3	2126
2137	Kant	C4	7	29120	Theophrastos	2	5216	Bioethik	2	2126
voraussetzen				29555	Feuerbach	2	5259	Der Wiener Kreis	2	2133
Vorgänger		Nachfolger		hören			5022	Glaube und Wissen	2	2134
5001		5041		26120	5001		4630	Die 3 Kritiken	4	2137
5001		5043		27550	5001					
5001		5049		27550	4052					
5041		5216		28106	5041					
5043		5052		28106	5052					
5041		5052		28106	5216					
5052		5259		28106	5259					
prüfen				29120	5001					
MatrNr	VorlNr	PersNr	Note	29120	5041					
28106	5001	2126	1	29120	5049					
25403	5041	2125	2	29555	5022					
27550	4630	2137	2	25403	5022					

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

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

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

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

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

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

Assistenten			
PersNr	Name	Fachgebiet	Boss
3002	Platon	Ideenlehre	2125
3003	Aristoteles	Syllogistik	2125
3004	Wittgenstein	Sprachtheorie	2126
3005	Rhetikus	Planetenebewegung	2127
3006	Newton	Keplersche Gesetze	2127
3007	Spinoza	Gott und Natur	2126

Datentypen

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

Anlegen von Tabelle

- **create table** Professoren
 - (PersNr **integer not null,**
 - Name **varchar (30) not null,**
 - Rang **character (2) ;**



Einfügen von Tupeln

insert into hoeren

select MatrNr, VorlNr

from Studenten, Vorlesungen

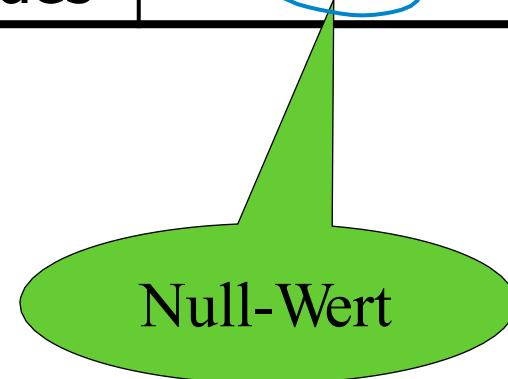
where Titel= 'Logik' ;

insert into Studenten (MatrNr, Name)

values (28121, 'Archimedes');



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



Null-Wert

Löschen von Tupeln

delete Studenten

where Semester > 13;

Verändern von Tupeln

update Studenten

set Semester= Semester + 1;



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

PersNr	Name
2125	Sokrates
2126	Russel
2136	Curie
2137	Kant



Sortierung

```
select PersNr, Name, Rang  
from Professoren  
order by Rang desc, Name asc;
```

PersNr	Name	Rang
2136	Curie	C4
2137	Kant	C4
2126	Russel	C4
2125	Sokrates	C4
2134	Augustinus	C3
2127	Kopernikus	C3
2133	Popper	C3



Duplikateliminierung

```
select distinct Rang  
from Professoren
```

Rang
C3
C4



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

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

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

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

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

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

Assistenten			
PersNr	Name	Fachgebiet	Boss
3002	Platon	Ideenlehre	2125
3003	Aristoteles	Syllogistik	2125
3004	Wittgenstein	Sprachtheorie	2126
3005	Rhetikus	Planetenebewegung	2127
3006	Newton	Keplersche Gesetze	2127
3007	Spinoza	Gott und Natur	2126

Anfragen über mehrere Relationen

Welcher Professor liest "Mäeutik"?

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

Als SQL ...

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

In relationer Algebra ...



Anfragen über mehrere Relationen

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	

Verknüpfung X



PersN	Name	Rang	Raum	VorlNr	Titel	SWS	gelesen Von
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
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

Allgemein hat eine
(ungeschachtelte) SQL-
Anfrage die Form:

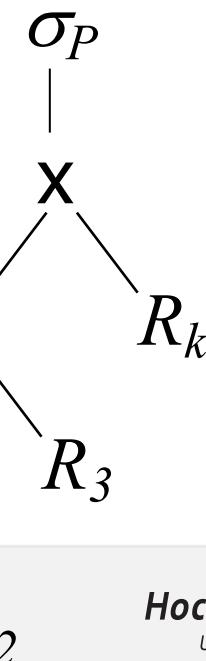
select A_1, \dots, A_n
from R_1, \dots, R_k
where $P;$

Übersetzung in die relationale
Algebra:

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

$$\Pi_{A_1, \dots, A_n}$$

|



Welche Studenten hören welche Vorlesungen?

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

Alternativ:

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



Welche Studenten kennen sich aus Vorlesungen

```
select s1.Name, s2.Name  
from Studenten s1, hoeren h1, hoeren h2, Studenten s2  
where h1.VorlNr = h2.VorlNr and h1.MatrNr = s1.MatrNr and  
h2.MatrNr = s2.MatrNr
```

Wie sieht das Ergebnis
aus? Besonderheiten?



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

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

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

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

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

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

Assistenten			
PersNr	Name	Fachgebiet	Boss
3002	Platon	Ideenlehre	2125
3003	Aristoteles	Syllogistik	2125
3004	Wittgenstein	Sprachtheorie	2126
3005	Rhetikus	Planetenebewegung	2127
3006	Newton	Keplersche Gesetze	2127
3007	Spinoza	Gott und Natur	2126

Mengenoperationen

Mengenoperationen **union, intersect, minus**

```
( select Name  
      from Assistenten )
```

union

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



```
select *  
from pruefen  
where Note = ( select avg(Note)  
                from pruefen );
```

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



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

Bedeutung der
Anfrage?



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



Korrelation

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

Unkorrelierte
Unteranfrage: meist
effizienter, wird nur
einmal ausgewertet



Der Vergleich mit "all"

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



Aggregatfunktion und Gruppierung

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

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

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

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



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

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

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

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

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

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

Assistenten			
PersNr	Name	Fachgebiet	Boss
3002	Platon	Ideenlehre	2125
3003	Aristoteles	Syllogistik	2125
3004	Wittgenstein	Sprachtheorie	2126
3005	Rhetikus	Planetenebewegung	2127
3006	Newton	Keplersche Gesetze	2127
3007	Spinoza	Gott und Natur	2126

- 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



Ausführen einer Anfrage mit group by

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

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
...
4630	Die 3 Kritiken	4	2137	2137	Kant	C4	7

↓ **where**-Bedingung



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

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

↓ Aggregation (**sum** und **avg**)

gelesenVon	Name	Sum(SWS)	Avg(SWS)
2125	Sokrates	10	3.3
2126	Russel	8	2.7
2137	Kant	8	4

↓ **Having-clause** und
Projektion



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



Geschachtelte Anfrage (Forts. - Nochmal)

- Unteranfrage in der **where**-Klausel
- Welche Prüfungen sind besser als durchschnittlich verlaufen?

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



Geschachtelte Anfrage (Forts. - Nochmal)

- Unteranfrage in der **select**-Klausel
- Für **jedes Ergebnistupel wird die Unteranfrage ausgeführt**
- Man beachte, dass die Unteranfrage korreliert ist (greift auf Attribute der umschließenden Anfrage zu)

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



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

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

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

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

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

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

Assistenten			
PersNr	Name	Fachgebiet	Boss
3002	Platon	Ideenlehre	2125
3003	Aristoteles	Syllogistik	2125
3004	Wittgenstein	Sprachtheorie	2126
3005	Rhetikus	Planetenebewegung	2127
3006	Newton	Keplersche Gesetze	2127
3007	Spinoza	Gott und Natur	2126

Unkorrelierte versus korrelierte Unteraufgaben

- korrelierte Formulierung

select s.*

from Studenten s

where exists

(**select** p.*

from Professoren p

where p.GebDatum > s.GebDatum);



Unkorrelierte versus korrelierte Untermanagen

- Äquivalente unkorrelierte Formulierung

select s.*

from Studenten s

where s.GebDatum <

(**select max** (p.GebDatum)

from Professoren p);

- Vorteil: Untermanageergebnis kann materialisiert werden
- Untermanage braucht nur einmal ausgewertet zu werden



```
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;
```



```
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

