

Relationale Algebra

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

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

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



Assistenten			
PerslNr	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

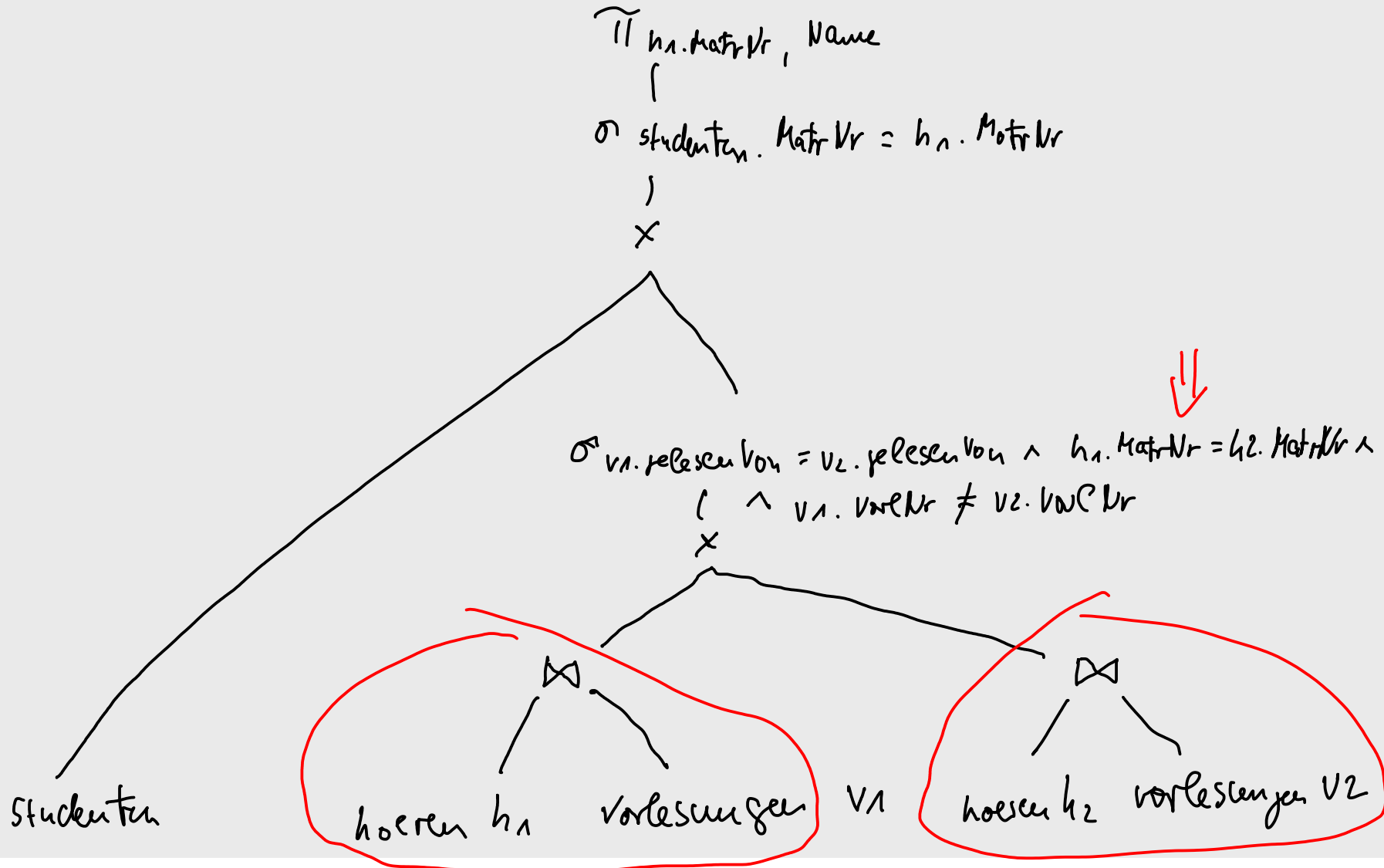
prüfen			
MatrNr	VorlNr	PersNr	Note
28106	5001	2126	1
25403	5041	2125	2
27550	4630	2137	2

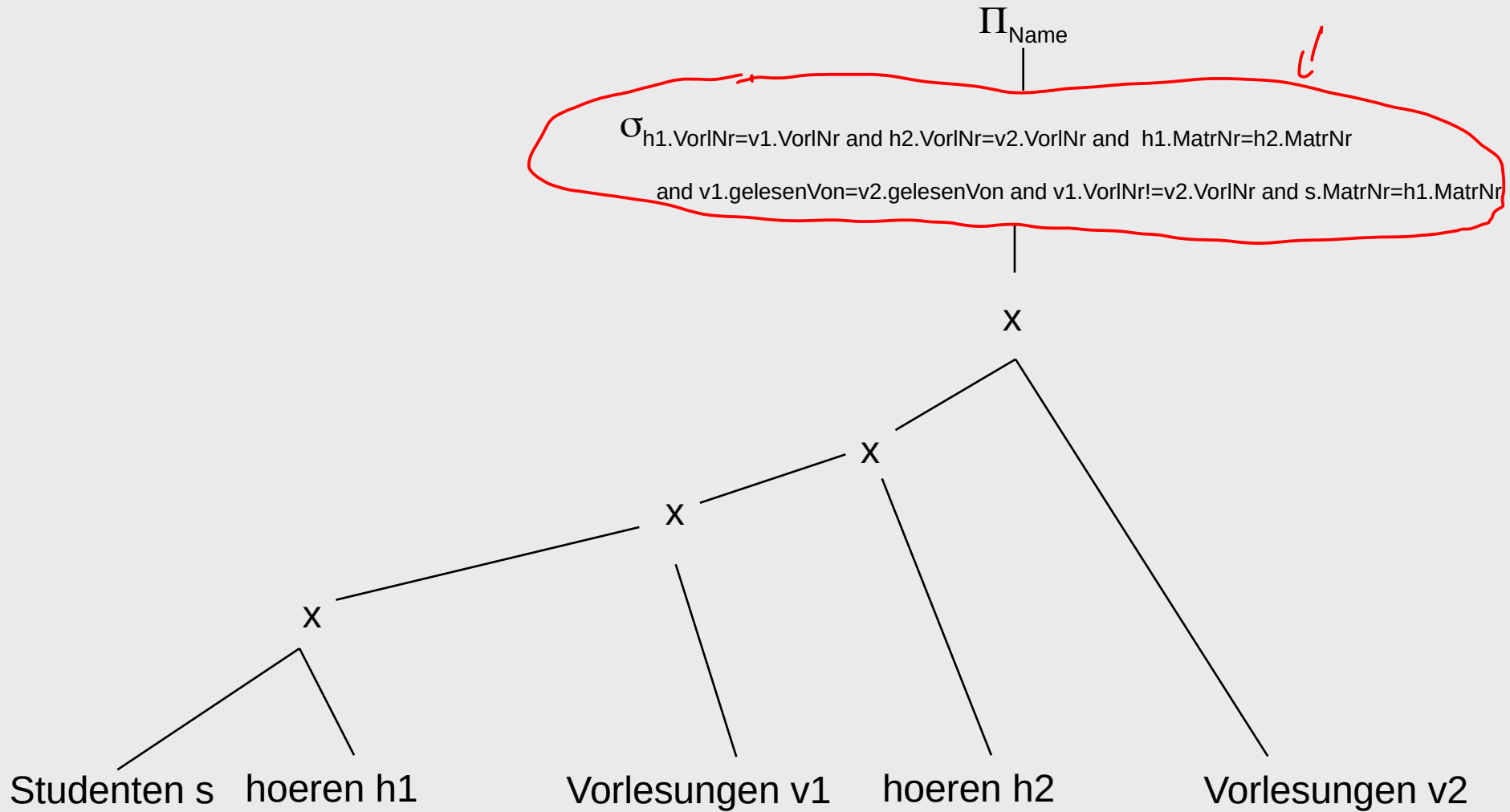
Operatoren der Relationalen Algebra

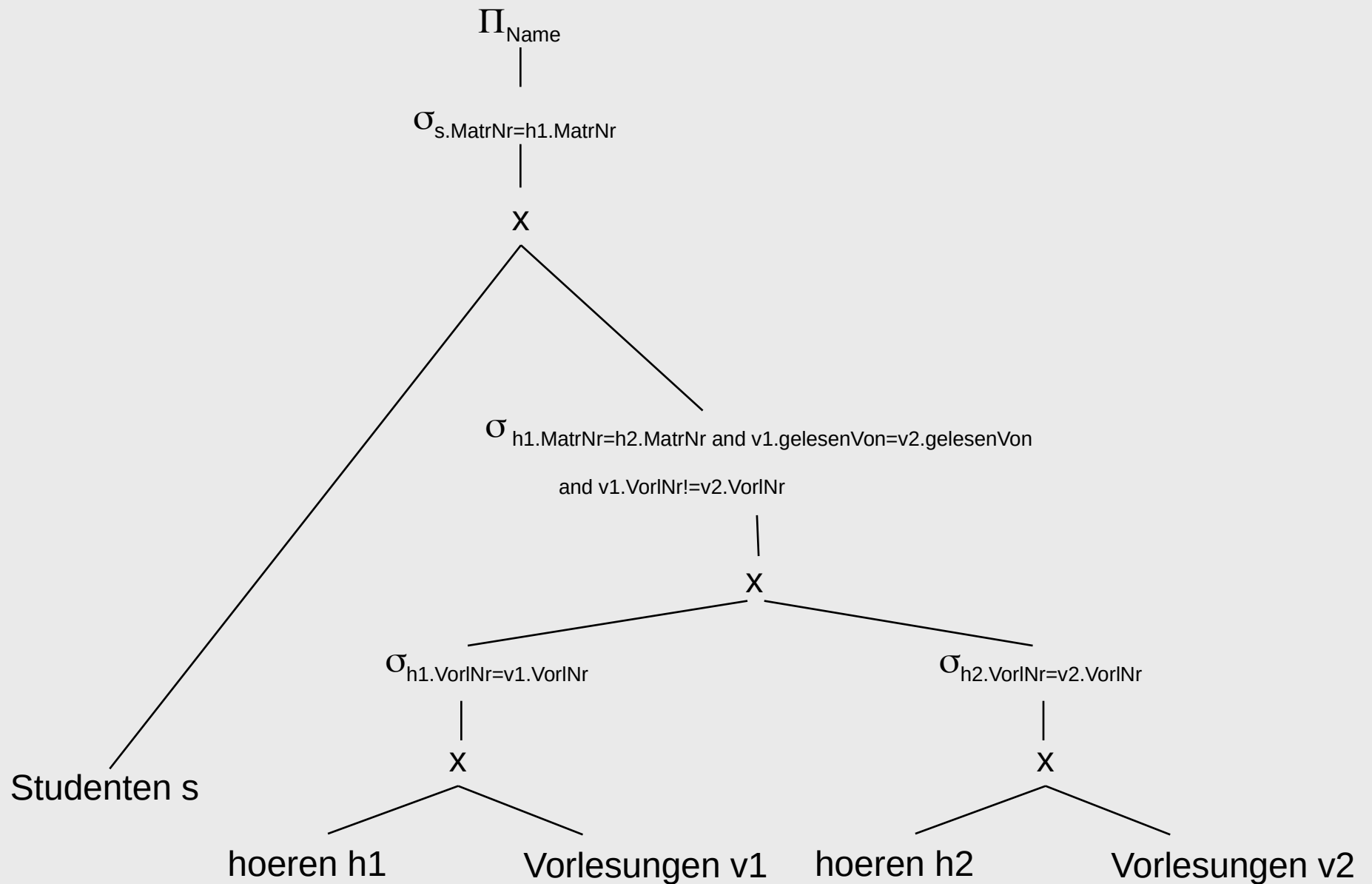
- σ Selektion
- π Projektion
- \times Kreuzprodukt
- \bowtie Join (Verbund)
- ρ Umbenennung
- $-$ Mengendifferenz
- \div Division
- \cup Vereinigung
- \cap Mengendurchschnitt
- \ltimes Semi-Join (linker)
- \rtimes Semi-Join (rechter)
- \ltimes linker äußerer Join
- \rtimes rechter äußerer Join

Operatoren stammen aus der Mengenlehre

Welche Studenten hören zwei Veranstaltungen, die vom selben Professor gelesen werden!







Anfrageabarbeitung

Welche Studenten hören welche Vorlesungen?

Superative Programmierung

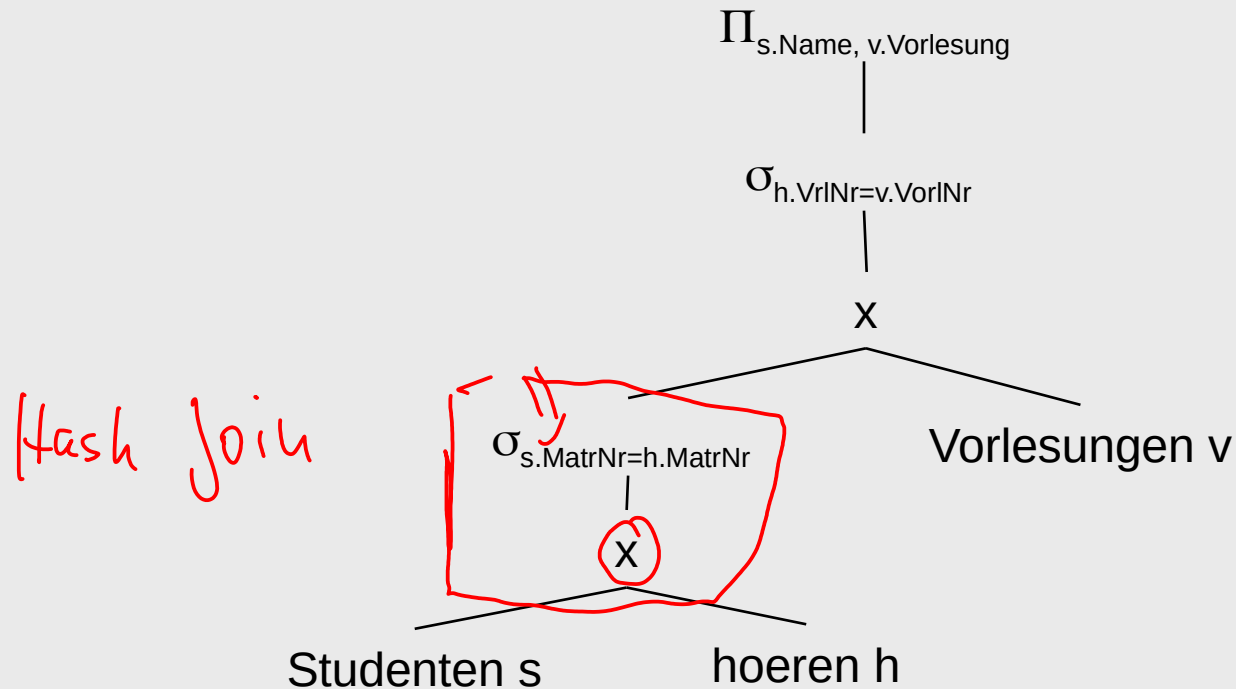
Deklarative Sprache: Wie sieht mein Ergebnis aus? Welche Eigenschaften erfüllt es?

Anfrage: SELECT name, titel FROM studenten s, hoeren h, vorlesungen v WHERE s.matrnr=h.matrnr and v.vorlnr=h.vorlnr

Datenbanksystem/MR-Framework

- Wie komme ich zu meinem Ergebnis?
- Welche Kriterien spielen bei der Performance eine Rolle?
- Wie wird mit Datenmengen umgegangen, die nicht in den Speicher passen?

Operatorbaum



- Operatorbaum legt die Operationen fest, die für die Auswertung der Anfrage erforderlich sind
- Operatorbaum bestimmt ebenfalls die Reihenfolge der Operatoren

Äquivalente Anfragen

Variante 1:

$\Pi_{s.Name, v.Vorlesung}(\sigma_{h.VorlNr=v.VorlNr} ($
Vorlesungen v x $\sigma_{s.MatrNr=h.MatrNr}(\text{Studenten s x hoeren h})))$

Variante 2:

$\Pi_{s.Name, v.Vorlesung}(\sigma_{s.MatrNr=h.MatrNr} ($
Studenten s x $\sigma_{v.VorlNr=h.VorlNr}(\text{Vorlesungen v x hoeren h})))$

Variante 3:

$\Pi_{s.Name, v.Vorlesung}(\sigma_{h.VorlNr=v.VorlNr \wedge s.MatrNr=h.MatrNr} (\text{Studenten s x$
 hoeren h x Volesungen v))

Algebraische Vereinfachungen

Term: $N = ((z*2)+((z*3)+0))/1$ ↵

Algebraische Regeln:

1. (+) Identität: $x+0 = x$ ↵
2. (/) Indentität: $x/1 = x$
3. (*) Distributivgesetz: $(n*x+n*y) = n*(x+y)$
4. (*) Kommutativgesetz: $x*y = y*x$

Anwendung der Regeln 1, 3, 4, 2

$N = (2+3)*z$ ↵

Das selbe kann auch mit der Relationalen Algebra angewendet werden.

Äquivalenzen in der Relationalen Algebra

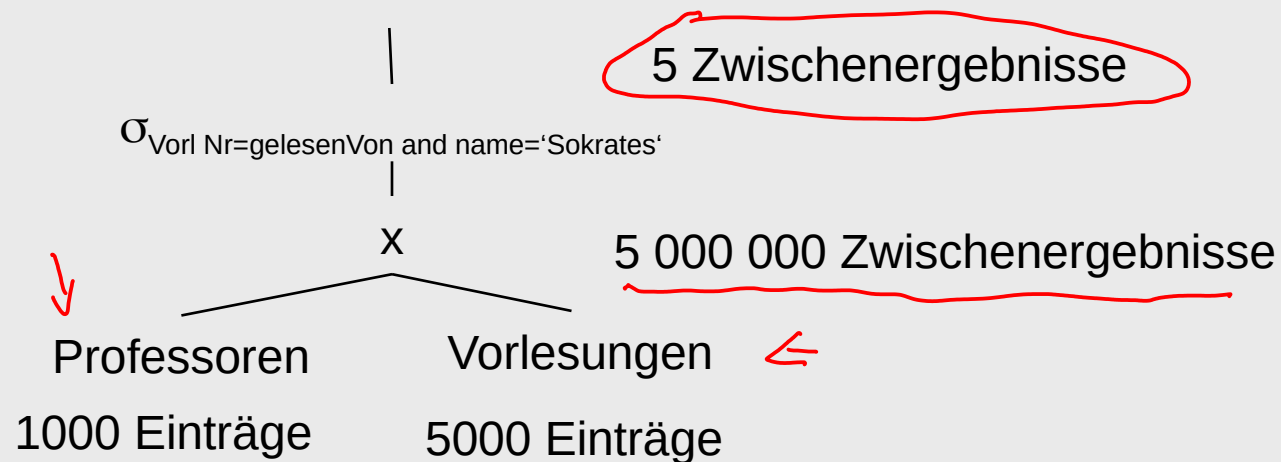
1. Join, Vereinigung, Schnitt und Kreuzprodukt sind kommutativ, also z.B.: $R_1 \times R_2 = R_2 \times R_1$ ↵
2. Selektionen sind untereinander vertauschbar:
 $\sigma_p(\sigma_q(R)) = \sigma_q(\sigma_p(R))$ ↵
3. Join, Vereinigung, Schnitt und Kreuzprodukt sind assoziativ, also z.B.: $R_1 \times (R_2 \times R_3) = (R_1 \times R_2) \times R_3$ ↵
4. Konjunktionen in einer Selektionsbedingung können in mehrere Selektionen aufgebrochen, bzw. nacheinander ausgeführte Selektionen können durch Konjunktionen zusammengefügt werden:
 $\sigma_{p_1 \wedge p_2 \wedge \dots \wedge p_n} = \sigma_{p_1}(\sigma_{p_2}(\dots \sigma_{p_n}(R)\dots))$ ↵

Und noch viele weitere Regeln: Kemper, et al: Datenbanksysteme, S. 240/241

Optimierungsbeispiel

Welche VL liest Professor Sokrates? ↵

SELECT * FROM vorlesungen, professoren WHERE
persnr=gelesenVon AND name='Sokrates'

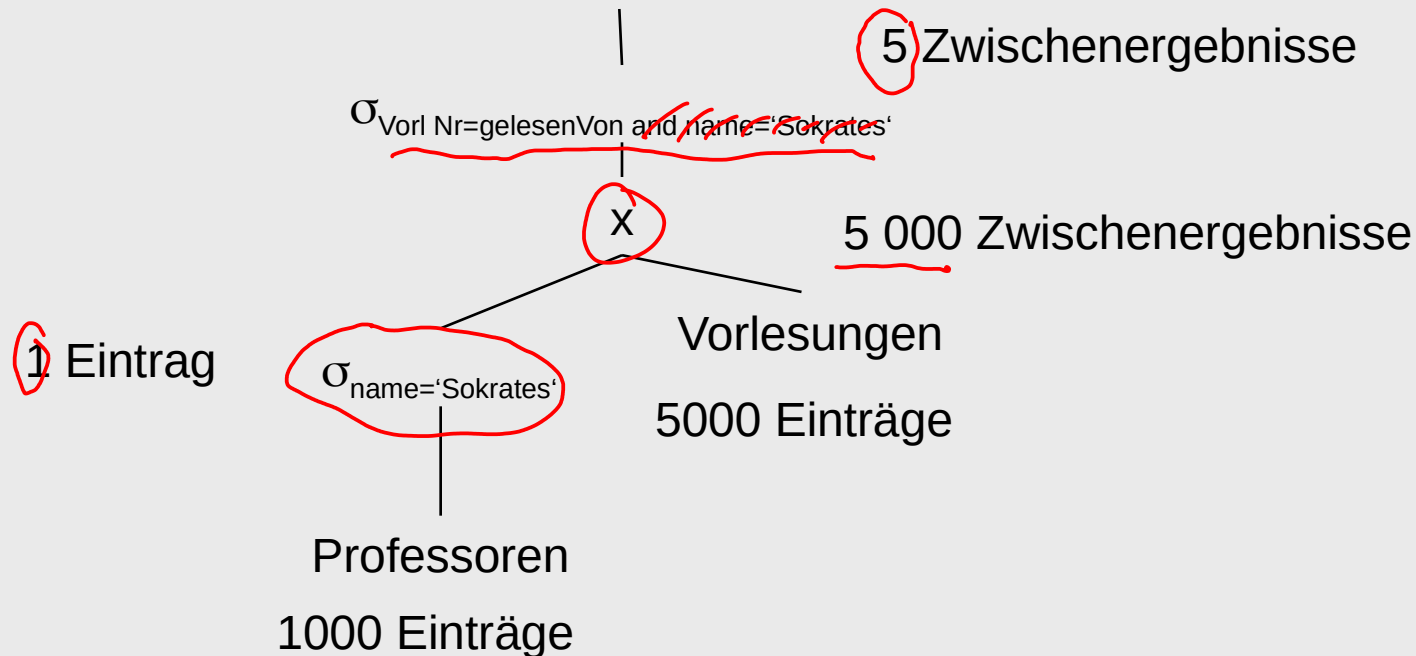


Wie würde eine Optimierung aussehen?

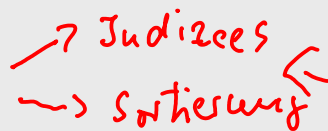


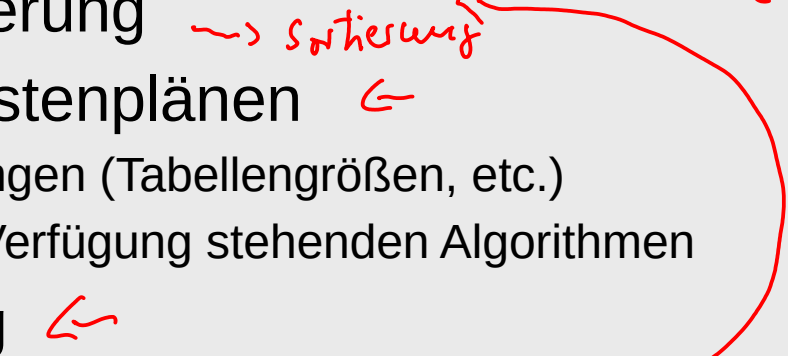
Optimierungsbeispiel

Welche VL liest Professor Sokrates?

SELECT * FROM vorlesungen, professoren WHERE
persnr=gelesenVon AND name='Sokrates'



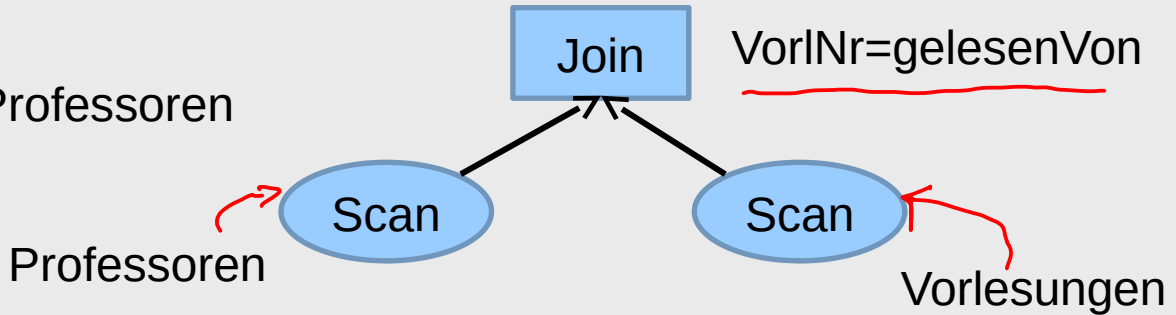
Abarbeitung von Anfragen

1. Algebraische Vereinfachungen des Terms¹ (logische Optimierung)
 2. Physische Optimierung 
 3. Erstellung von Kostenplänen 
 - 1. Statische Auswertungen (Tabellengrößen, etc.)
 - 2. Abwägung der zur Verfügung stehenden Algorithmen
 4. Datenbank Tuning 
- 
- Handwritten red notes:*
→ Indizes
→ Sortierung
create index

¹Die Art der Formulierung der SQL-Anfrage (z.B. werden Joins oder das Kartesische verwendet) ist meist zweitrangig. Nach der Optimierung kommen dieselben Anfragepläne heraus.

Algorithmenauswahl

select * from Vorlesungen, Professoren
where VorlNr=gelesenVon



Iskellen prüfen?

Welche der
beiden
Varianten ist
performanter?

Variante 1:

For each record p in Professoren:
For each record v in Vorlesungen
if (p.persnr=v.gelesenVon) return (p,v)

Variante 2:

For each record p in Professoren:
insert into HashTable

For each record v in Vorlesungen
lookup corresponding records in HashTable
return matching pairs

Anzeigen der Anfragepläne

Query - uni on postgres@localhost:5432 *

SQL Editor Graphical Query Builder

Previous queries Delete Delete All

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

Output pane

Data Output Explain Messages History

studenten hoeren Hash Hash Join Hash Join vorlesungen Hash

OK. Unix Ln 1, Col 106, Ch 106 9 rows. 14 ms

Explain Query

explain SELECT.....

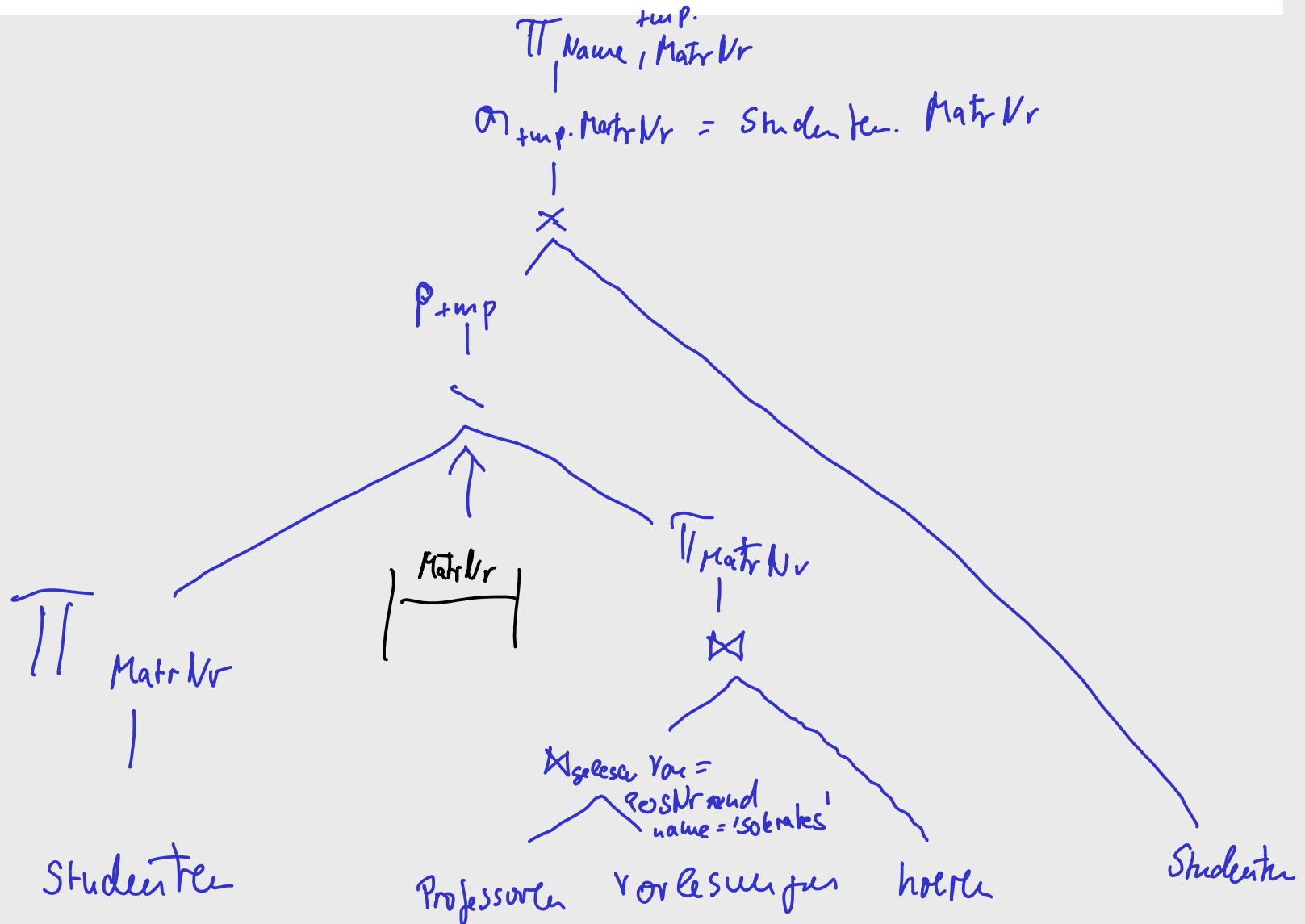
*Erstellung von
Indizes*

```
hendrik@ruebe: ~  
(9 rows)  
  
uni=#  
uni=#  
uni=#  
uni=#  
uni=#  
uni=# explain select Name, Titel from Studenten s, hoeren h, Vorlesungen v where  
s.MatrNr=h.MatrNr and v.VorlNr= h.VorlNr  
;  
  
                        QUERY PLAN  
-----  
Hash Join (cost=2.41..3.89 rows=13 width=22)  
  Hash Cond: (h.vorlNr = v.vorlNr)  
    -> Hash Join (cost=1.18..2.49 rows=13 width=13)  
      Hash Cond: (h.matrNr = s.matrNr)  
        -> Seq Scan on hoeren h (cost=0.00..1.13 rows=13 width=8)  
        -> Hash (cost=1.08..1.08 rows=8 width=13)  
          -> Seq Scan on studenten s (cost=0.00..1.08 rows=8 width=13)  
        -> Hash (cost=1.10..1.10 rows=10 width=17)  
          -> Seq Scan on vorlesungen v (cost=0.00..1.10 rows=10 width=17)  
(9 rows)  
  
uni=#
```

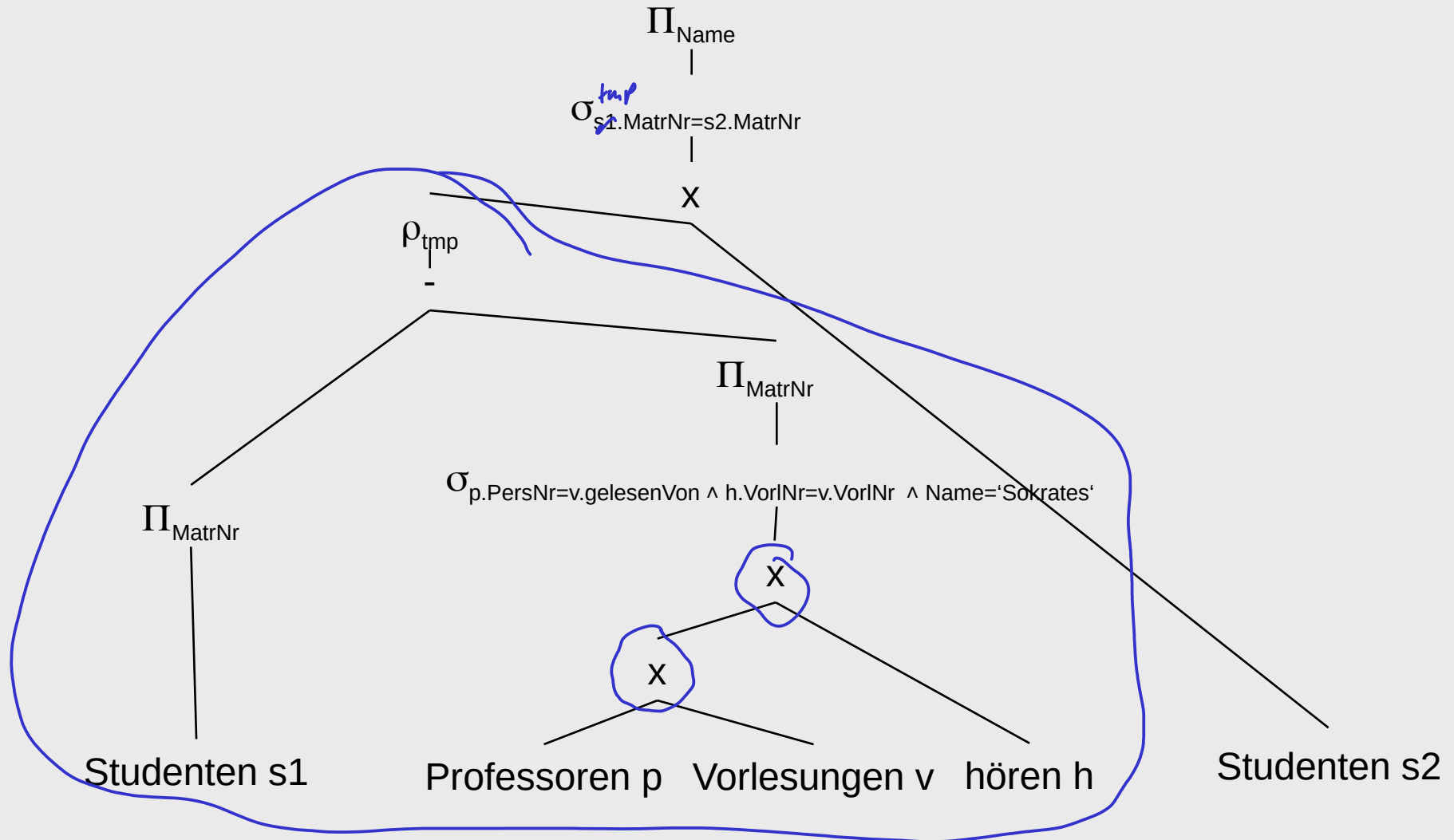
Entscheidend für die Gestaltung des Query-Plans ist der Physische Entwurf der Datenbank!

Beispiele: Operatorbäume

Welche Studenten haben noch keine VL bei Sokrates gehört?

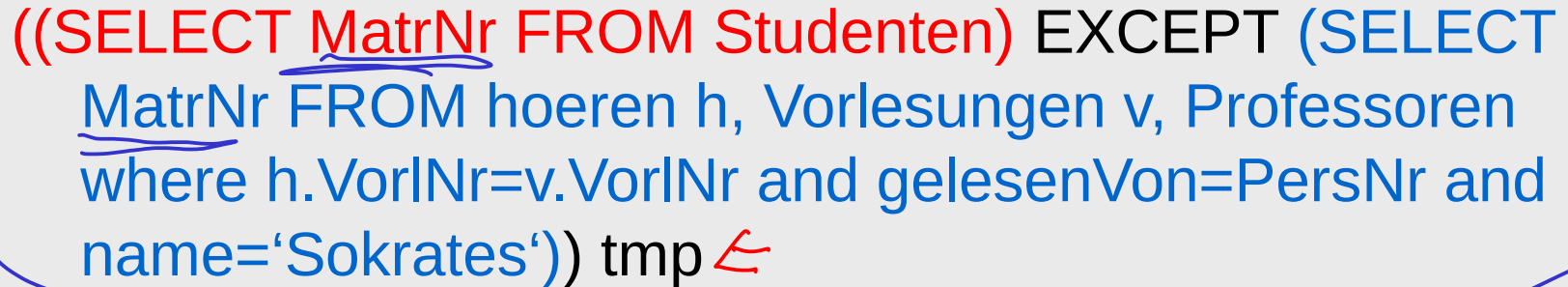


Welche Studenten haben noch keine VL bei Sokrates gehört?



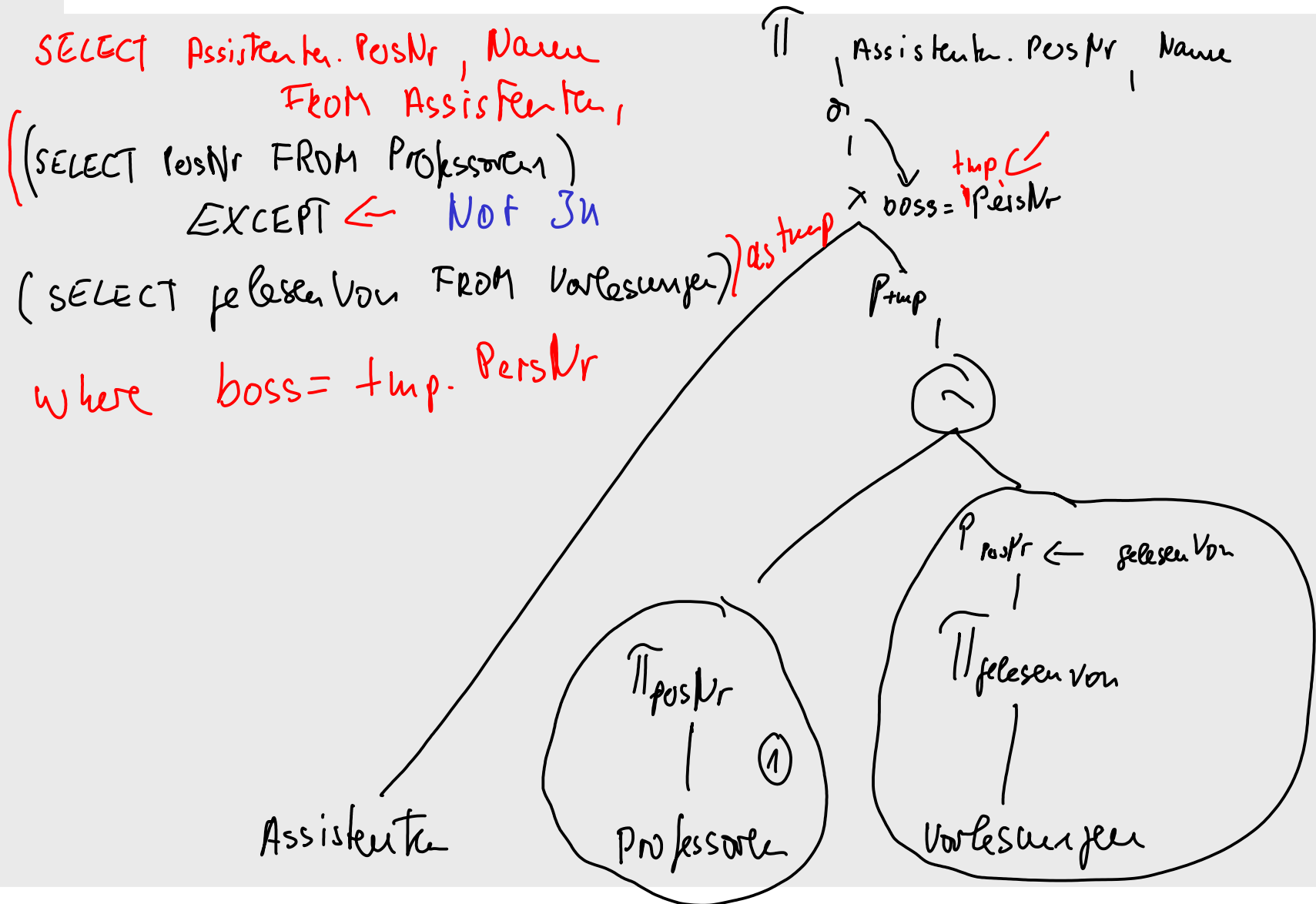
Welche Studenten haben noch keine VL bei Sokrates gehört?

SELECT Studenten.MatrNr, Name FROM Studenten,

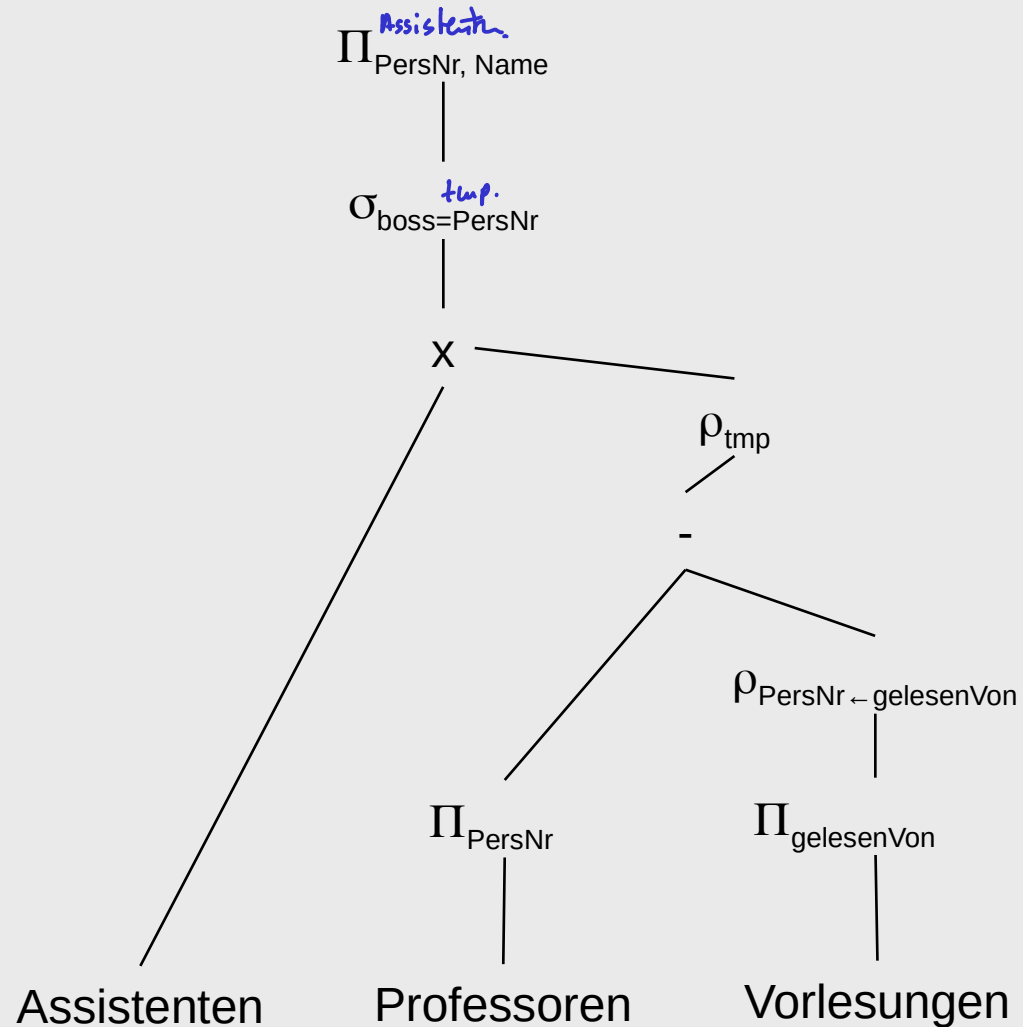

((SELECT MatrNr FROM Studenten) EXCEPT (SELECT
MatrNr FROM hoeren h, Vorlesungen v, Professoren
where h.VorlNr=v.VorlNr and gelesenVon=PersNr and
name='Sokrates')) tmp

WHERE tmp.MatrNr=Studenten.MatrNr

Welche Assistenten arbeiten für Professoren, die keine Veranstaltung halten?



Welche Assistenten arbeiten für Professoren, die keine Veranstaltung halten?



Welche Assistenten arbeiten für Professoren, die keine Veranstaltung halten?

SELECT Assistenten.PersNr, Name FROM Assistenten,

((SELECT PersNr FROM Professoren) EXCEPT (SELECT
gelesenVon as PersNr FROM Vorlesungen)) tmp

WHERE tmp.PersNr=Assistenten.PersNr

Welche Professoren lesen Vorlesungen, die keine Voraussetzungen benötigen?

```
SELECT PersNr, Name FROM
```

(Professoren JOIN Vorlesungen ON (lexikon = posn,))

Join

$$\prod posN_i, Name$$
$$\Delta_{+wp} \text{ VorlNr} = \text{Vorlesungs. VolNr}$$

↓
(SELECT VorNr FROM Vorlesungen)
EXCEPT
(select Nachfolger FROM voraussetzen)) temp

ON (temp.VorNr = Vorlesungen.VorNr)

$$\chi_{\text{sektion}} = \text{PosNr}$$

Professoren

Vorlesungen

$$\Pi_{\text{vorl Nr}}$$

Vorlesungen

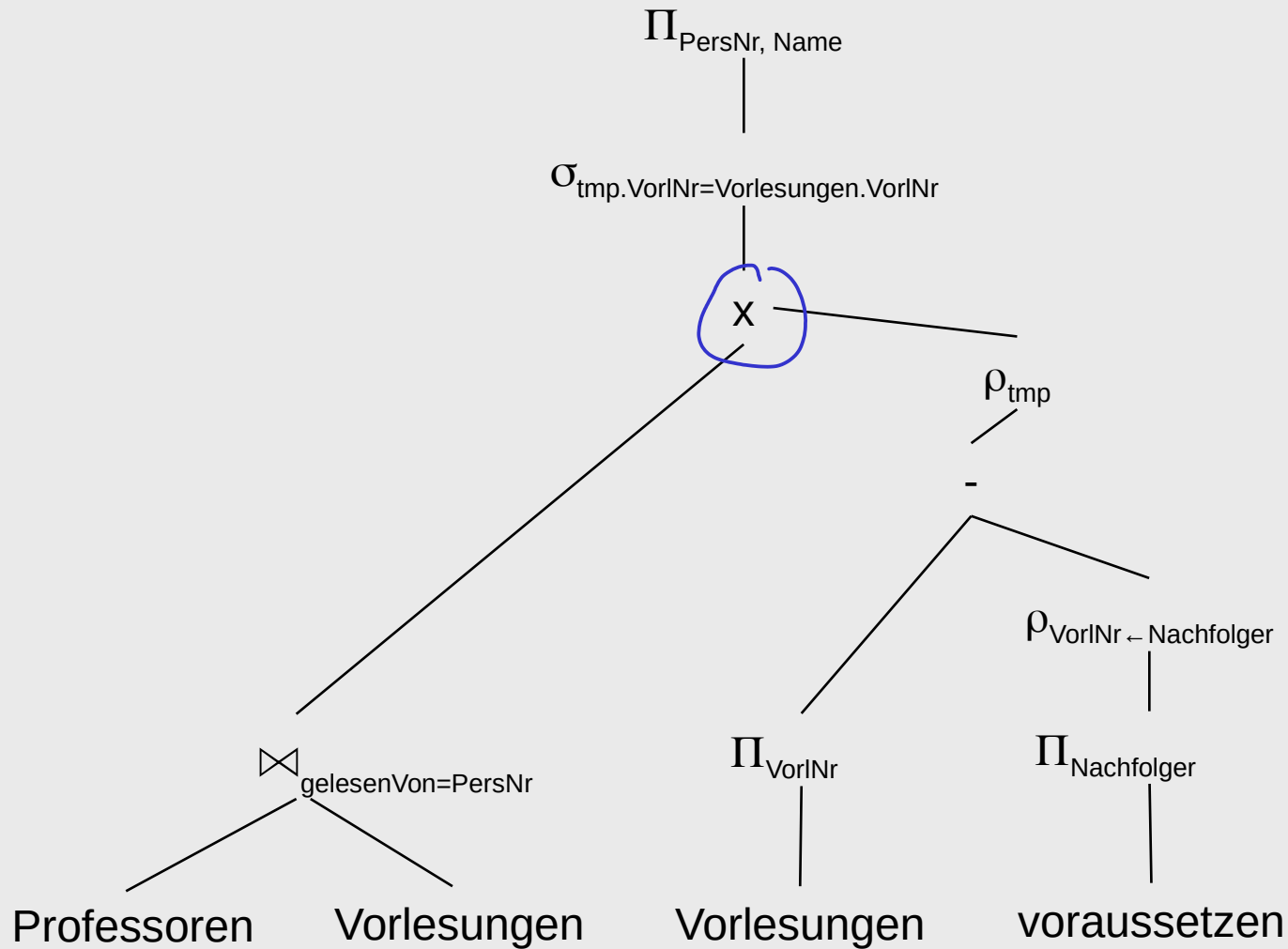
 P_{imp}

$P_{\text{vorr}} \leftarrow \text{Nachfolge}$

1) Nachfolger

voraussetzen

Welche Professoren lesen Vorlesungen, die keine Voraussetzungen benötigen?



Welche Professoren lesen Vorlesungen, die keine Voraussetzungen benötigen?

```
SELECT Professoren.PersNr, Name FROM Professoren  
  JOIN Vorlesungen ON(gelesenVon=PersNr),  
  
  ((SELECT VorlNr FROM Vorlesungen) EXCEPT (SELECT  
    nachfolger as VorlNr FROM voraussetzen)) tmp  
  
WHERE tmp.VorlNr=Vorlesungen.VorlNr
```

Vielen Dank für

Ihre Aufmerksamkeit

Prof. Dr.-Ing. Hendrik Gärtner