



Einführung in die Anfragebearbeitung und - optimierung

Medieninformatik Bachelor
Modul 9:
Datenbanksysteme



- **Aufgabe 1 Anfragen & Modellierung“**

Denken Sie mal darüber nach, welche Anfragen Sie an die AOL Daten stellen möchten. Bitte Sie bitte ein logisches und physisches Schema zur Beantwortung dieser Anfragen.

- **Aufgabe 2 „SQL und Abfrageausführung“**

Bitte formulieren Sie für Ihre Analyseideen aus 1.) die SQL Anfragen. Sie verstehen auch Möglichkeiten der Abfrageausführung bzw. Optimierung.

- **Aufgabe 3 „Datenintegration“**

Zur Ausführung der Ausführung fehlen Ihnen noch externe Daten, z.B. aus dem Internet Archive, DMOZ oder Freebase.org. Bitte ergänzen Sie Ihr Schema und die Datenbasis.

- **Aufgabe 4 „Analyse, Erkenntnisgewinn und Wert“**

Stellen Sie in 5 Minuten die wichtigsten Erkenntnisse aus den Daten vor. Bewerten Sie den Erkenntnisgewinn, z.B. gegenüber Ihren Kommilitonen oder der Literatur! Welche Erkenntnisse hätten einen kommerziellen Wert?



- Was sind Datenbanken?
 - Motivation, Historie, Datenunabhängigkeit, Einsatzgebiete
- Datenbankentwurf im ER-Modell & Relationaler Datenbankentwurf
 - Entities, Relationships, Kardinalitäten, Diagramme
 - Relationales Modell, ER -> Relational, Normalformen, Transformationseigenschaften
- Relationale Algebra & SQL
 - Kriterien für Anfragesprachen, Operatoren, Transformationen
 - SQL DDL, SQL DML, SELECT ... FROM ... WHERE ...
- Datenintegration & Transaktionsverwaltung
 - JDBC, Cursor, ETL
 - Mehrbenutzerbetrieb, Serialisierbarkeit, Sperrprotokolle, Fehlerbehandlung, Isolationsebenen in SQL
- Ausblick
 - Map/Reduce, HDFS, Hive ...
 - Wert von Daten





- Einfache SQL Anfragen
 - Der SFW Block
 - Nullwerte
 - Mengen vs. Multimenge
- SQL Anfragen über mehrere Relationen
 - UNION, INTERSECT, EXCEPT
 - Joins und Outerjoins
- Geschachtelte SQL Anfragen
 - In FROM und WHERE
 - EXISTS, IN, ALL, ANY
- SQL Operationen auf einer Relation
 - GROUP BY, HAVING
 - ORDER BY
- Kapitel 6 und 10.6/10.7 des Lehrbuches



- **Einführung**
- Parsen der Anfrage
- Transformationsregeln der RA
- Optimierung
- Kostenmodelle



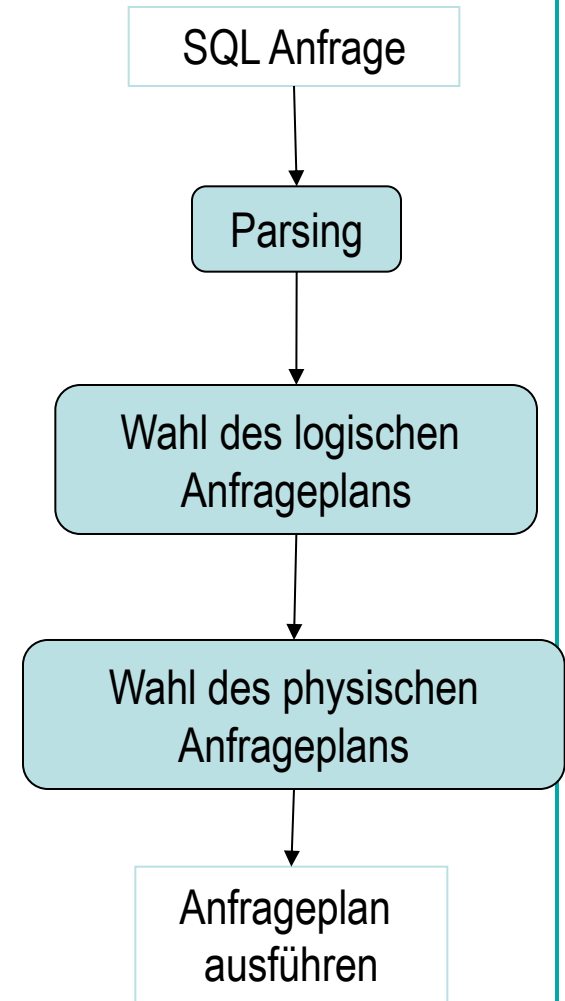


- Anfragen sind deklarativ.
 - SQL, Relationale Algebra
- Anfragen müssen in ausführbare (prozedurale) Form transformiert werden.
- Ziele
 - „QEP“ – prozeduraler Query Execution Plan
(auch „Zugriffspfad“ (access path) oder „Anfrageplan“ genannt)
 - Effizienz
 - Schnell
 - Wenig Ressourcenverbrauch (CPU, I/O, RAM, Bandbreite)





1. Parsing
 - Parsen der Anfrage (Syntax)
 - Überprüfen der Elemente („Semantik“)
 - Parsebaum
2. Wahl des logischen Anfrageplans
 - Baum mit logischen Operatoren
 - Potentiell exponentiell viele
 - Wahl des optimalen Plans
 - Logische Optimierung
 - Regelbasierter Optimierer
 - Kostenbasierter Optimierer
3. Wahl des physischen Anfrageplans
 - Ausführbar
 - Programm mit physischen Operatoren
 - Algorithmen
 - Scan Operatoren
 - Wahl des optimalen Plans
 - physische Optimierung



- Einführung
- **Parsen der Anfrage**
- Transformationsregeln der RA
- Optimierung
- Kostenmodelle

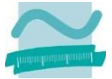
Kapitel 16 des Lehrbuchs





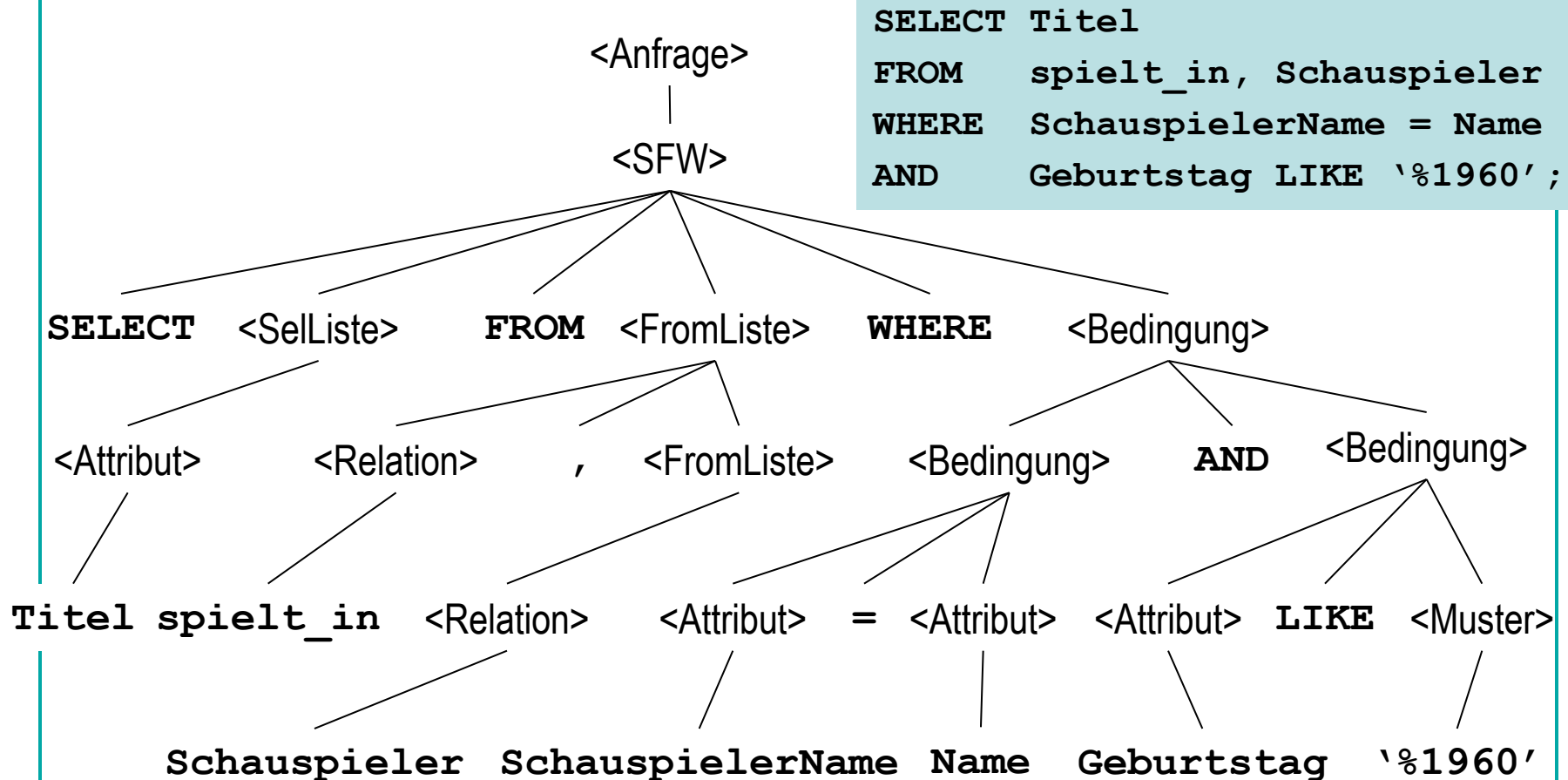
- Aufgabe: Umwandlung einer SQL Anfrage in einen Parsebaum.
 - Atome (Blätter)
 - Schlüsselworte
 - Konstanten
 - Namen (Relationen und Attribute)
 - Syntaxzeichen
 - Operatoren
 - Syntaktische Kategorien
 - Namen für Teilausdrücke einer Anfrage





- Anfragen
 - $\langle \text{Anfrage} \rangle ::= \langle \text{SFW} \rangle$
 - $\langle \text{Anfrage} \rangle ::= (\langle \text{SFW} \rangle)$
 - Mengenoperatoren fehlen
- SFWs
 - $\langle \text{SFW} \rangle ::= \text{SELECT } \langle \text{SelListe} \rangle \text{ FROM } \langle \text{FromListe} \rangle \text{ WHERE } \langle \text{Bedingung} \rangle$
 - Gruppierung, Sortierung etc. fehlen
- Listen
 - $\langle \text{SelListe} \rangle ::= \langle \text{Attribut} \rangle, \langle \text{SelListe} \rangle$
 - $\langle \text{SelListe} \rangle ::= \langle \text{Attribut} \rangle$
 - $\langle \text{FromListe} \rangle ::= \langle \text{Relation} \rangle, \langle \text{FromListe} \rangle$
 - $\langle \text{FromListe} \rangle ::= \langle \text{Relation} \rangle$
- Bedingungen (Beispiele)
 - $\langle \text{Bedingung} \rangle ::= \langle \text{Bedingung} \rangle \text{ AND } \langle \text{Bedingung} \rangle$
 - $\langle \text{Bedingung} \rangle ::= \langle \text{Tupel} \rangle \text{ IN } \langle \text{Anfrage} \rangle$
 - $\langle \text{Bedingung} \rangle ::= \langle \text{Attribut} \rangle = \langle \text{Attribut} \rangle$
 - $\langle \text{Bedingung} \rangle ::= \langle \text{Attribut} \rangle \text{ LIKE } \langle \text{Muster} \rangle$
- $\langle \text{Tupel} \rangle, \langle \text{Attribut} \rangle, \langle \text{Relation} \rangle, \langle \text{Muster} \rangle$ nicht durch grammatische Regeln definiert
- Vollständig z.B. hier: <http://docs.openlinksw.com/virtuoso/GRAMMAR.html>

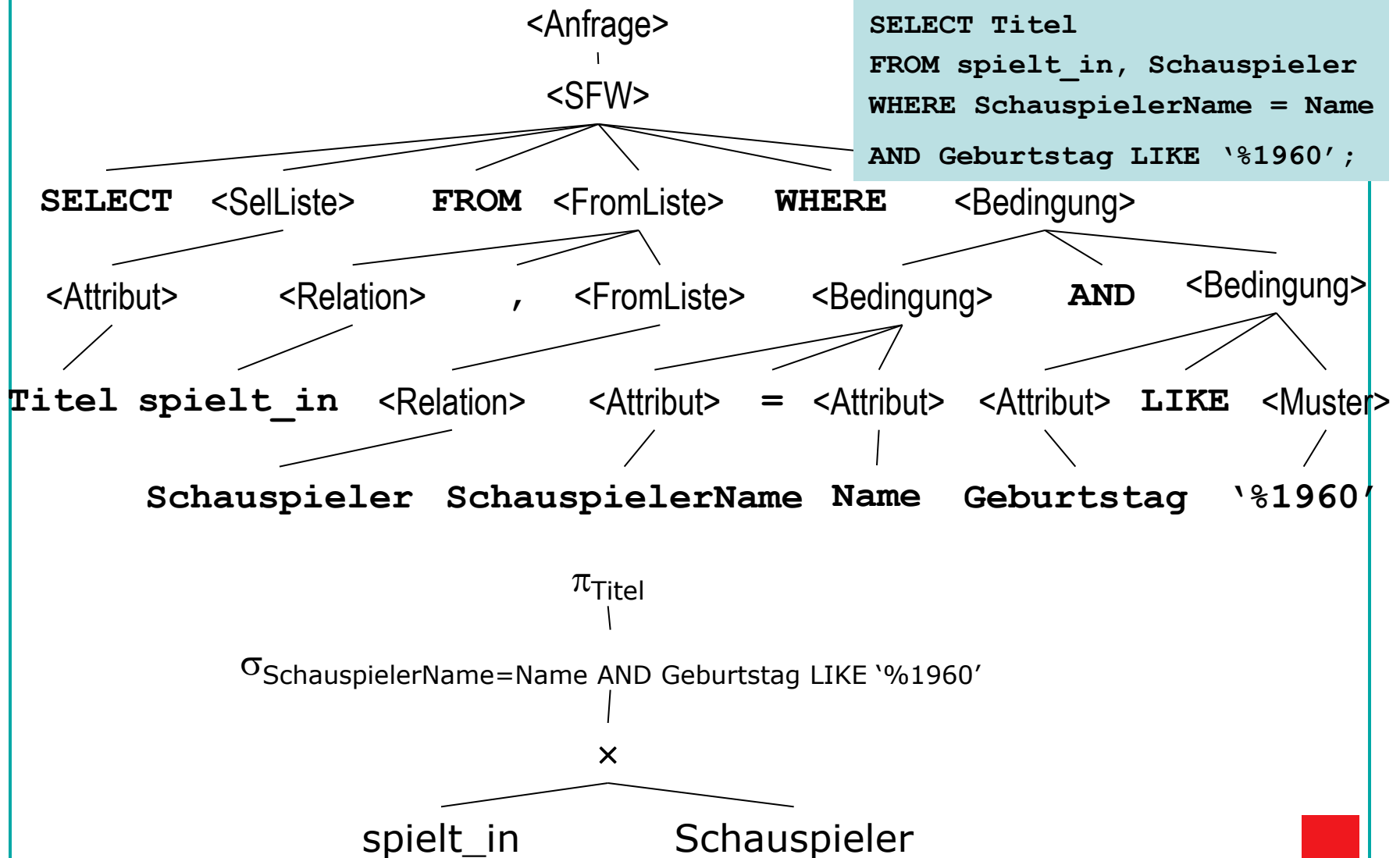






- Während der Übersetzung semantische Korrektheit prüfen
 - Existieren die Relationen und Sichten der FROM Klausel?
 - Existieren die Attribute in den genannten Relationen?
 - Sind sie eindeutig?
 - Korrekte Typen für Vergleiche?
 - Aggregation korrekt?
 - ...





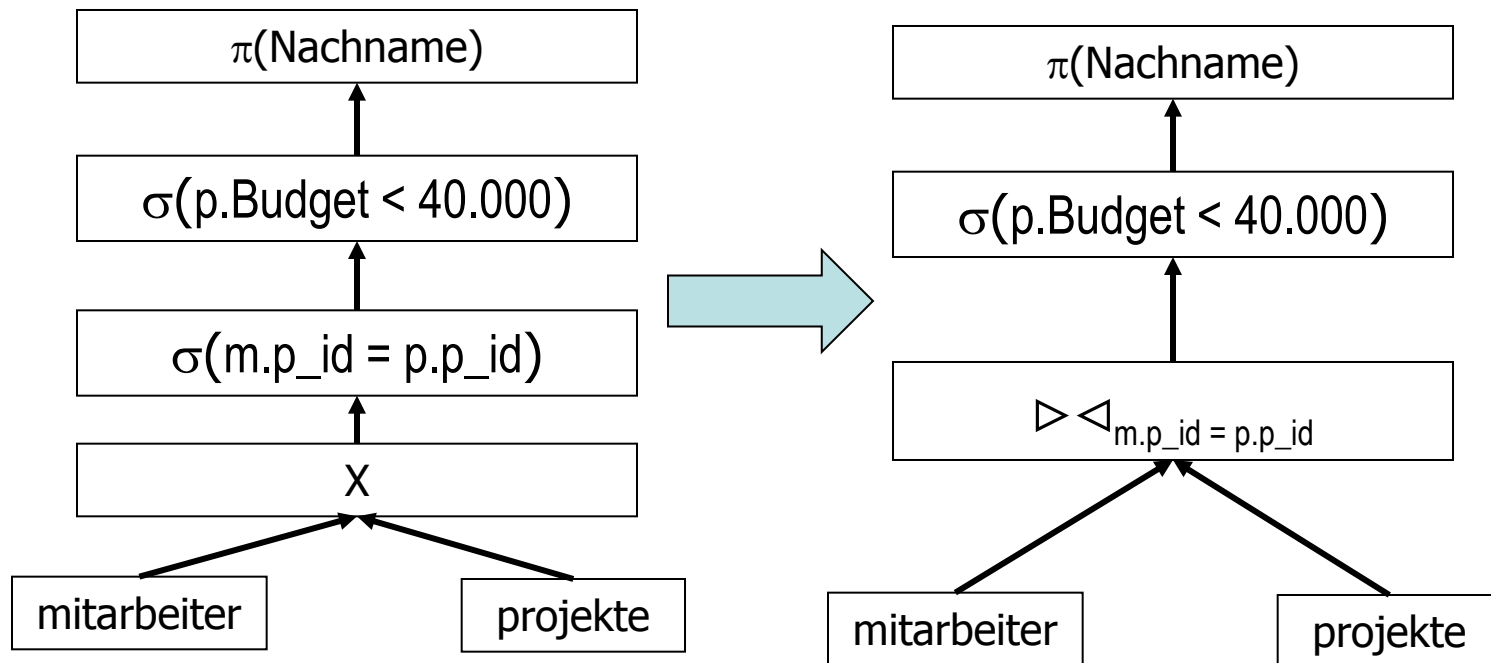
- Einführung
- Parsen der Anfrage
- **Transformationsregeln der RA**
- Optimierung
- Kostenmodelle

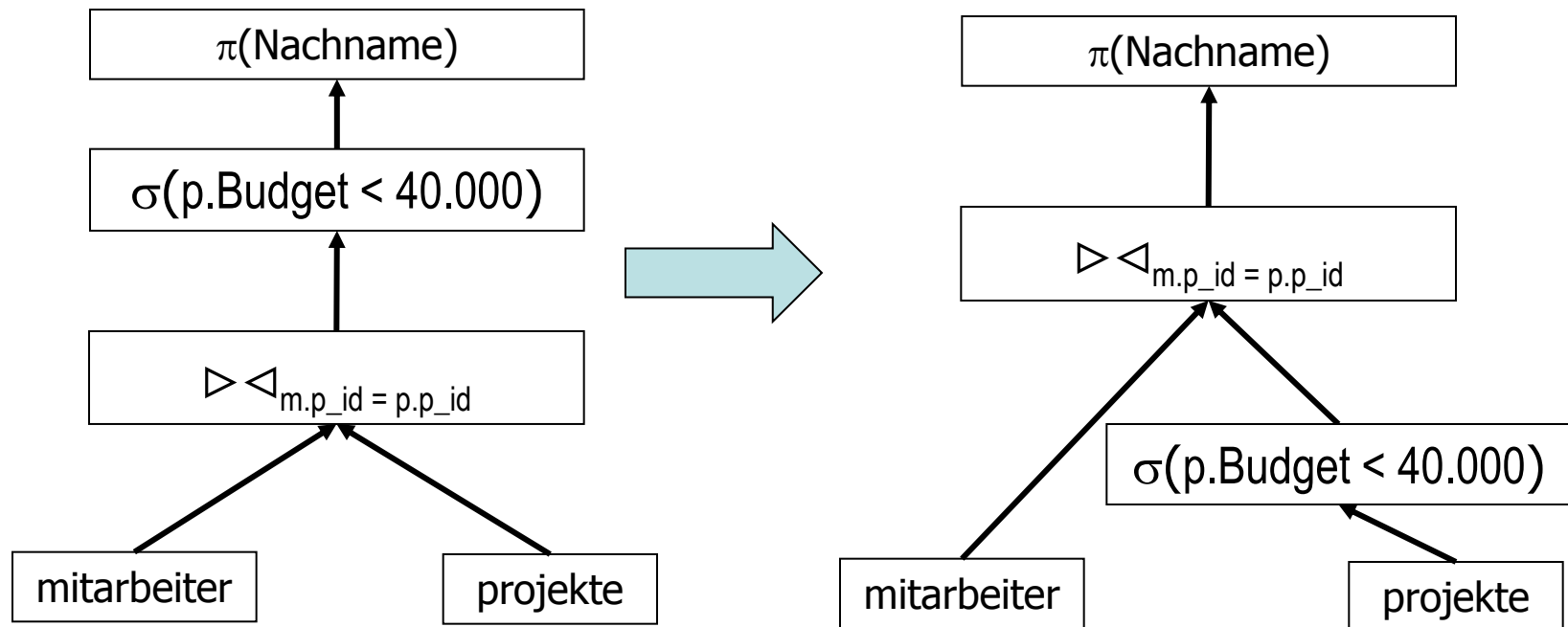




- Transformation der internen Darstellung
 - Ohne Semantik zu verändern
 - Zur effizienteren Ausführung
 - Insbesondere: Kleine Zwischenergebnisse
- Äquivalente Ausdrücke
 - Zwei Ausdrücke der relationalen Algebra heißen äquivalent, falls
 - Gleiche Operanden (= Relationen)
 - Stets gleiche Antwortrelation
 - Stets?

Für jede mögliche Instanz der Datenbank







- \times ist kommutativ und assoziativ
 - $R \times S = S \times R$
 - $(R \times S) \times T = R \times (S \times T)$
- \cup ist kommutativ und assoziativ
 - $R \cup S = S \cup R$
 - $(R \cup S) \cup T = R \cup (S \cup T)$
- \cap ist kommutativ und assoziativ
 - $R \cap S = S \cap R$
 - $(R \cap S) \cap T = R \cap (S \cap T)$
- \bowtie ist kommutativ und assoziativ
 - $R \bowtie S = S \bowtie R$
 - $(R \bowtie S) \bowtie T = R \bowtie (S \bowtie T)$

Gilt jeweils für Mengen
und Multimengen

Ausdrücke können in beide
Richtungen verwendet werden.

Welche ist besser?





Selektion

- $\sigma_{c_1 \text{ AND } c_2}(R) = \sigma_{c_1}(\sigma_{c_2}(R))$
- $\sigma_{c_1 \text{ OR } c_2}(R) = \sigma_{c_1}(R) \cup \sigma_{c_2}(R)$
 - Nicht bei Multimengen
- $\sigma_{c_1}(\sigma_{c_2}(R)) = \sigma_{c_2}(\sigma_{c_1}(R))$
- $\sigma_c(R \Phi S) \equiv (\sigma_c(R)) \Phi (\sigma_c(S))$
 - $\Phi \in \{\cup, \cap, -, \bowtie\}$
- $\sigma_c(R \Phi S) \equiv (\sigma_c(R)) \Phi S$
 - $\Phi \in \{\cup, \cap, -, \bowtie\}$
 - Falls sich c nur auf Attribute in R bezieht.

Projektion

- $\pi_L(R \bowtie S) = \pi_L(\pi_M(R) \bowtie \pi_N(S))$
- $\pi_L(R \bowtie_C S) = \pi_L(\pi_M(R) \bowtie_C \pi_N(S))$
- $\pi_L(R \times S) = \pi_L(\pi_M(R) \times \pi_N(S))$
- $\pi_L \sigma_C(R) = \pi_L(\sigma_C(\pi_M(R)))$

- Einführung
- Parsen der Anfrage
- Transformationsregeln der RA
- **Optimierung**
- Kostenmodelle





- High-level SQL (deklarativ nicht prozedural)
 - „was“, nicht „wie“.
- Das „wie“ bestimmt sich aus der Abbildung der mengen-orientierten Operatoren auf die Schnittstellen-Operatoren der internen Ebene.
 - Zugriff auf Datensätze in Dateien
 - Einfügen/Entfernen interner Datensätze
 - Modifizieren interner Datensätze
- Zu einem „was“ kann es zahlreiche „wie's“ geben.
 - Äquivalenzerhaltende Transformationen
- Im Allgemeinen wird nicht die optimale Auswertungsstrategie gesucht, sondern eine einigermaßen effiziente Variante.
 - Ziel: *Avoid the worst case.*



- Regelbasierte Optimierung
 - Fester Regelsatz schreibt Transformationen gemäß der genannten Regeln vor.
 - Prioritäten unter den Regeln
 - Heuristik
- Kostenbasierte Optimierung
 - Kostenmodell
 - Transformationen um Kosten zu verringern
 - Bestimmung des optimalen Plans
 - Bestimmung der optimalen Joinreihenfolge





- Logische Optimierung
 - Jeder Ausdruck kann in viele verschiedene, semantisch äquivalente Ausdrücke umgeschrieben werden.
 - Wähle den (hoffentlich) besten Ausdruck (=Plan, =QEP)
- Physische Optimierung
 - Für jede relationale Operation gibt es viele verschiedene Implementierungen.
 - Zugriff auf Tabellen
 - Scan, verschiedene Indizes, sortierter Zugriff, ...
 - Joins
 - Nested loop, sort-merge, hash, ...
 - Wähle für jede Operation die (hoffentlich) beste Implementierung
- Abhängigkeit beider Probleme!





- Grundsätze der logischen Optimierung
 - Selektionen so weit wie möglich im Baum nach unten schieben.
 - Selektionen mit `AND` können aufgeteilt und separat verschoben werden.
 - Projektionen so weit wie möglich im Baum nach unten schieben,
 - bzw. neue Projektionen können eingefügt werden.
 - Duplikateliminierung kann manchmal entfernt werden oder verschoben werden.
 - Kreuzprodukte mit geeigneten Selektionen zu einem Join zusammenfassen.

- Noch nicht hier: Suche nach der optimalen Joinreihenfolge



- Grundsätze der logischen Optimierung
 - Selektionen so weit wie möglich im Baum nach unten schieben.
 - Selektionen mit **AND** können aufgeteilt und separat verschoben werden.
 - Projektionen so weit wie möglich im Baum nach unten schieben,
 - bzw. neue Projektionen können eingefügt werden.
 - Duplikateliminierung kann manchmal entfernt werden oder verschoben werden.
 - Kreuzprodukte mit geeigneten Selektionen zu einem Join zusammenfassen.

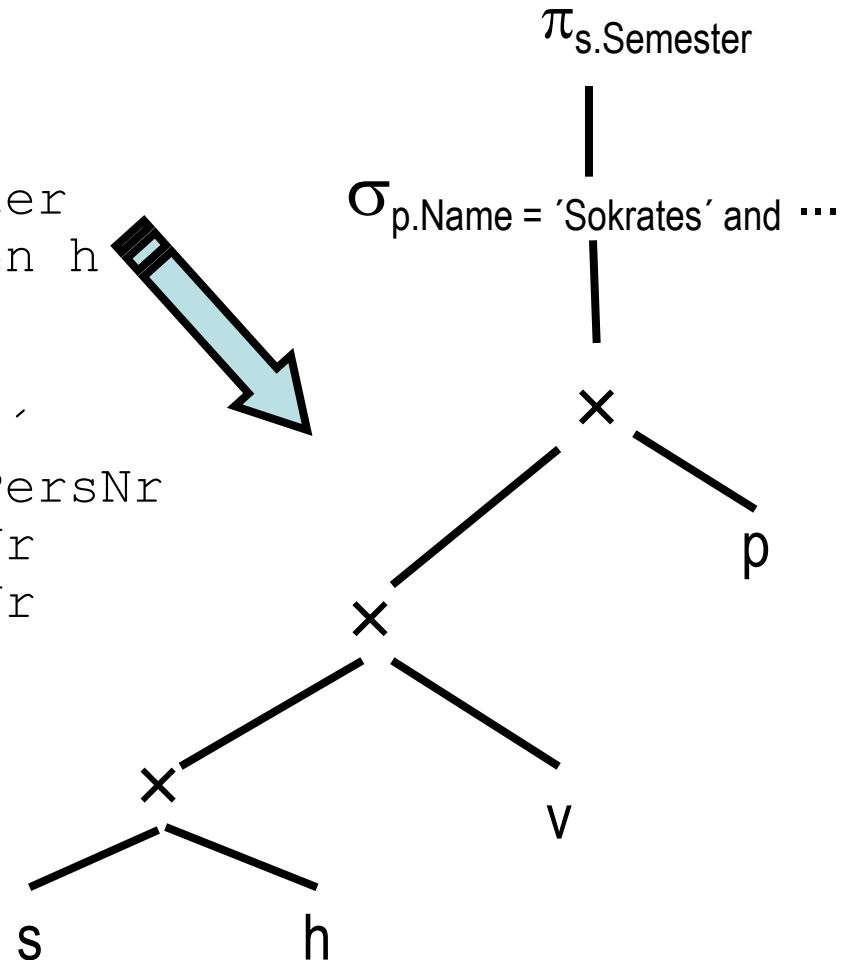
- Noch nicht hier: Suche nach der optimalen Joinreihenfolge



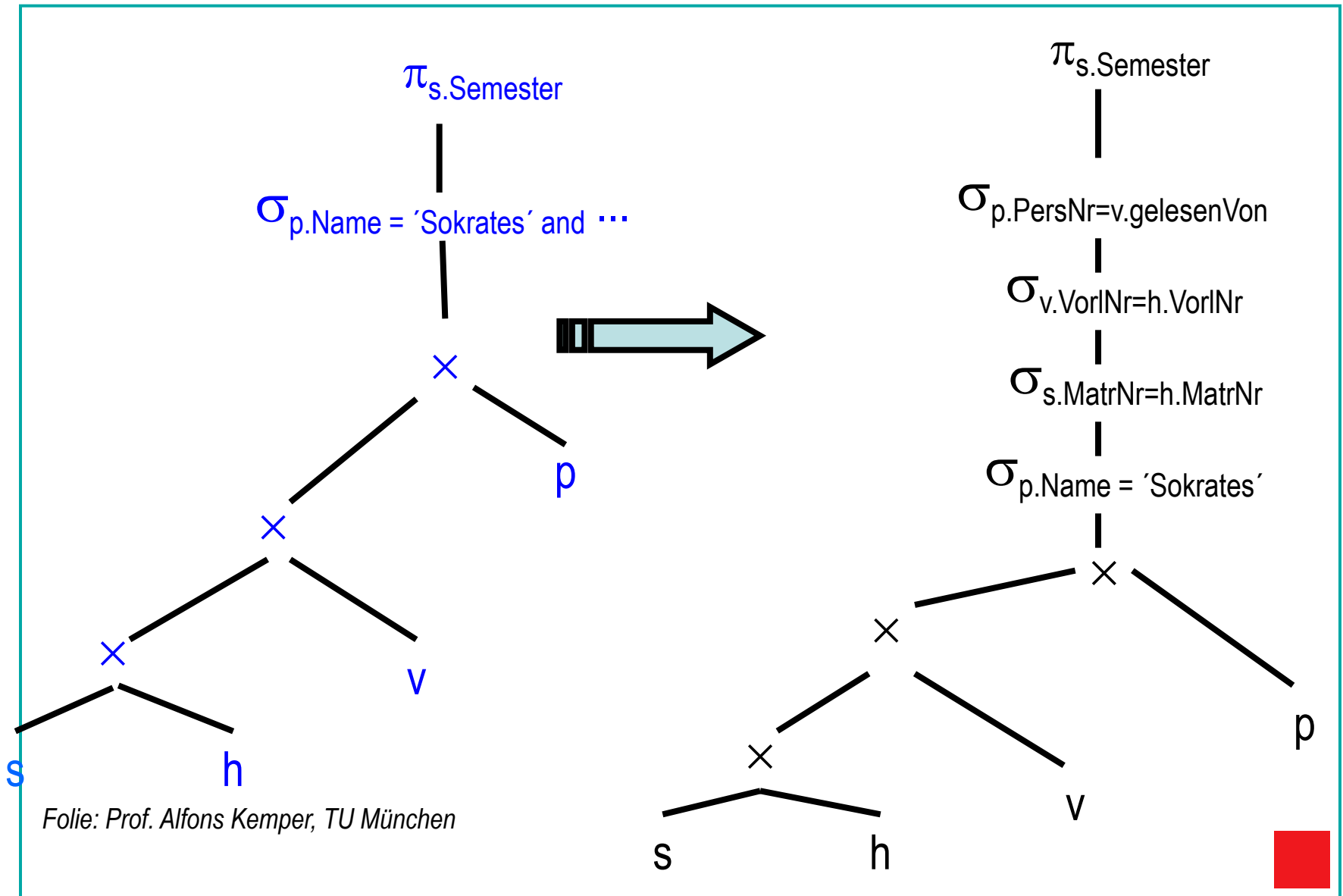


■ **select** distinct s.Semester
from Studenten s, hören h
Vorlesungen v,
Professoren p
where p.Name = 'Sokrates'
and v.gelesenVon = p.PersNr
and v.VorlNr = h.VorlNr
and h.MatrNr = s.MatrNr

*In welchen Semestern sind
die Studenten, die VLen
bei Sokrates hören?*

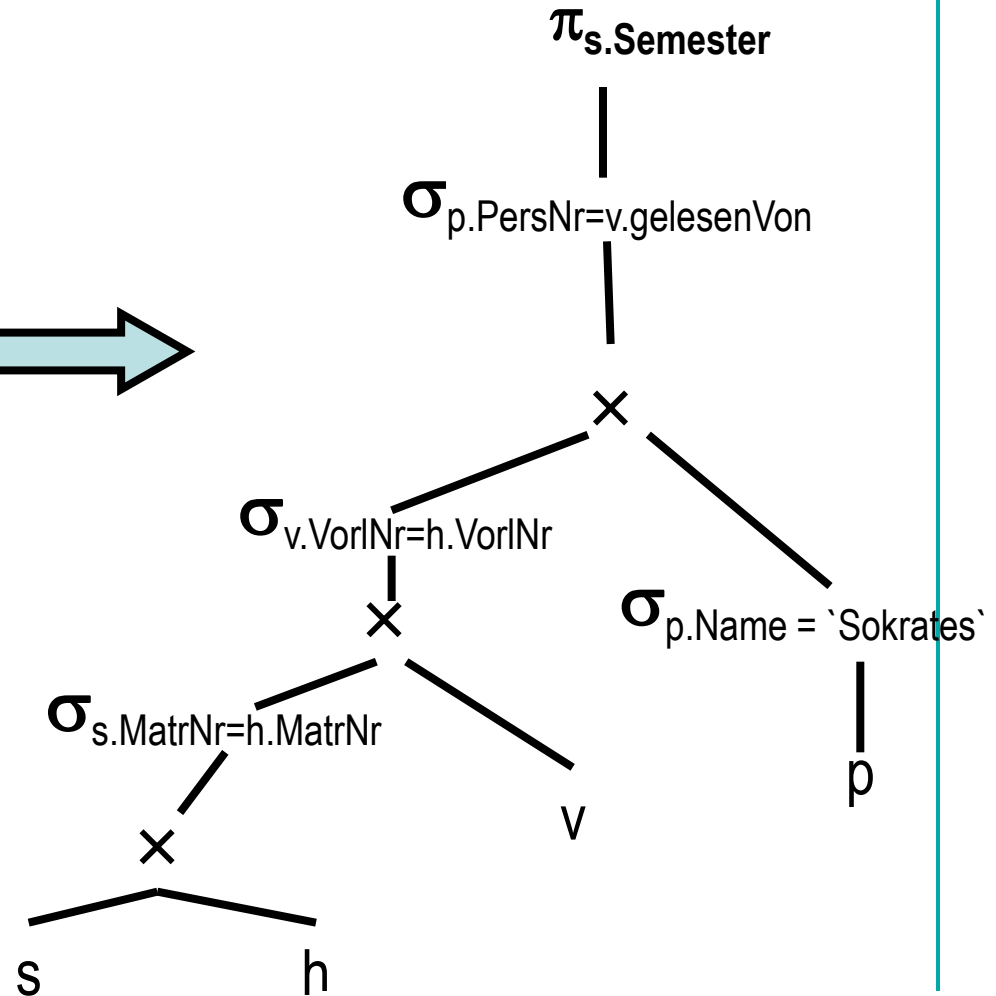
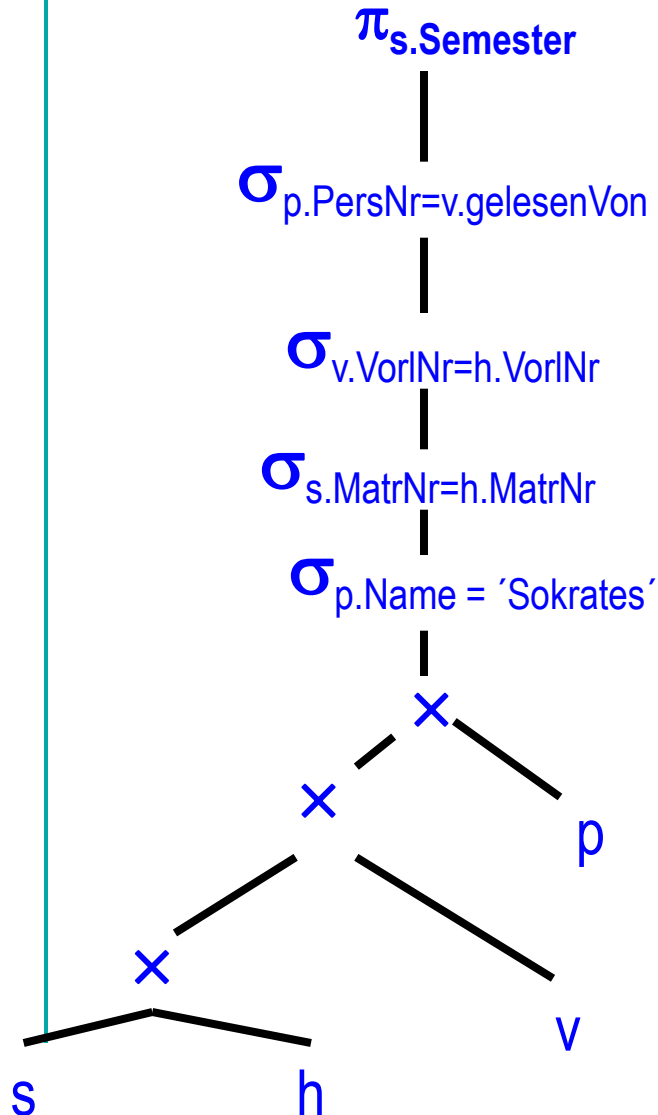


Aufspalten der Selektionsprädikate





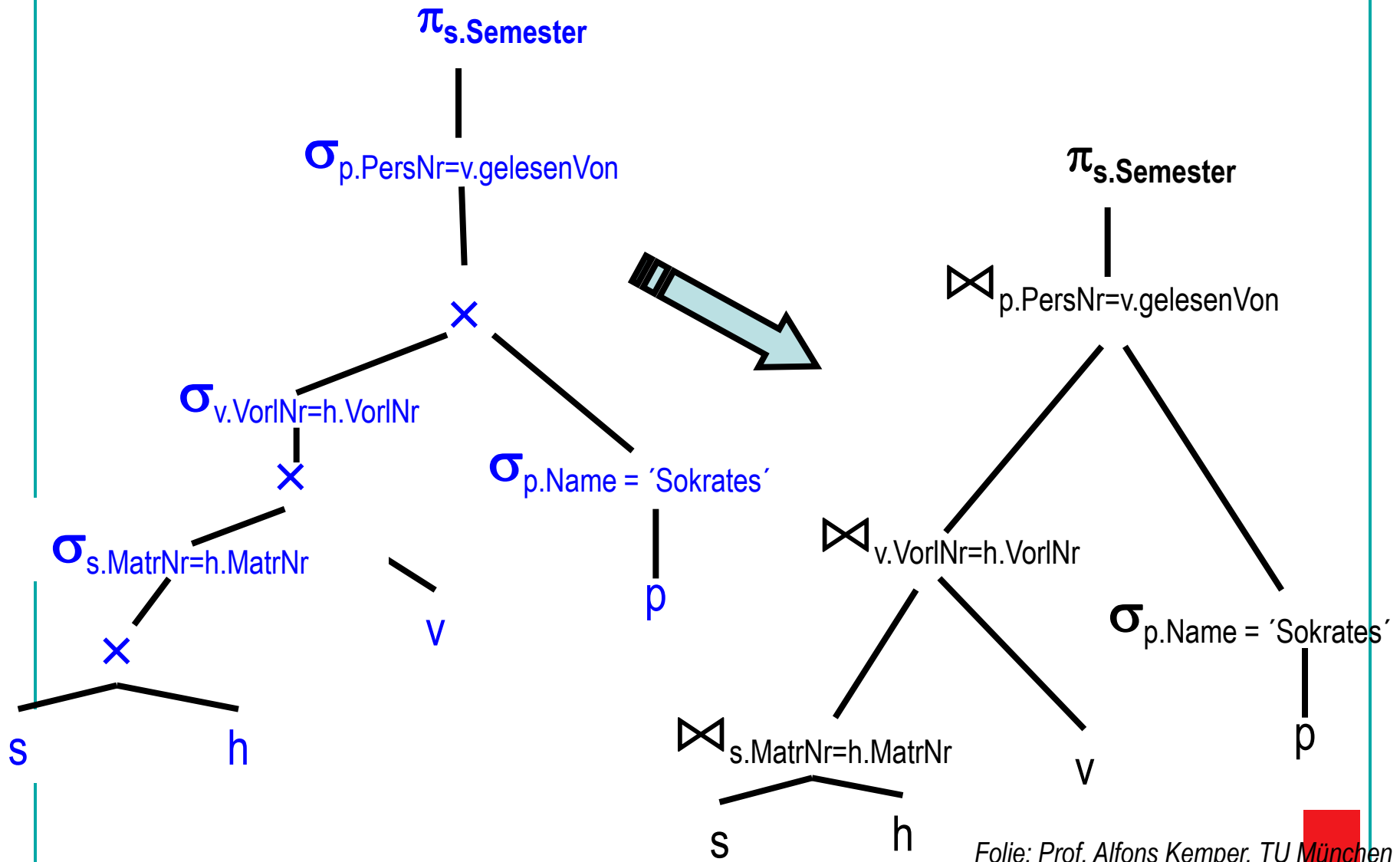
Verschieben der Selektionsprädikate „Pushing Selections“



Folie: Prof. Alfons Kemper, TU München



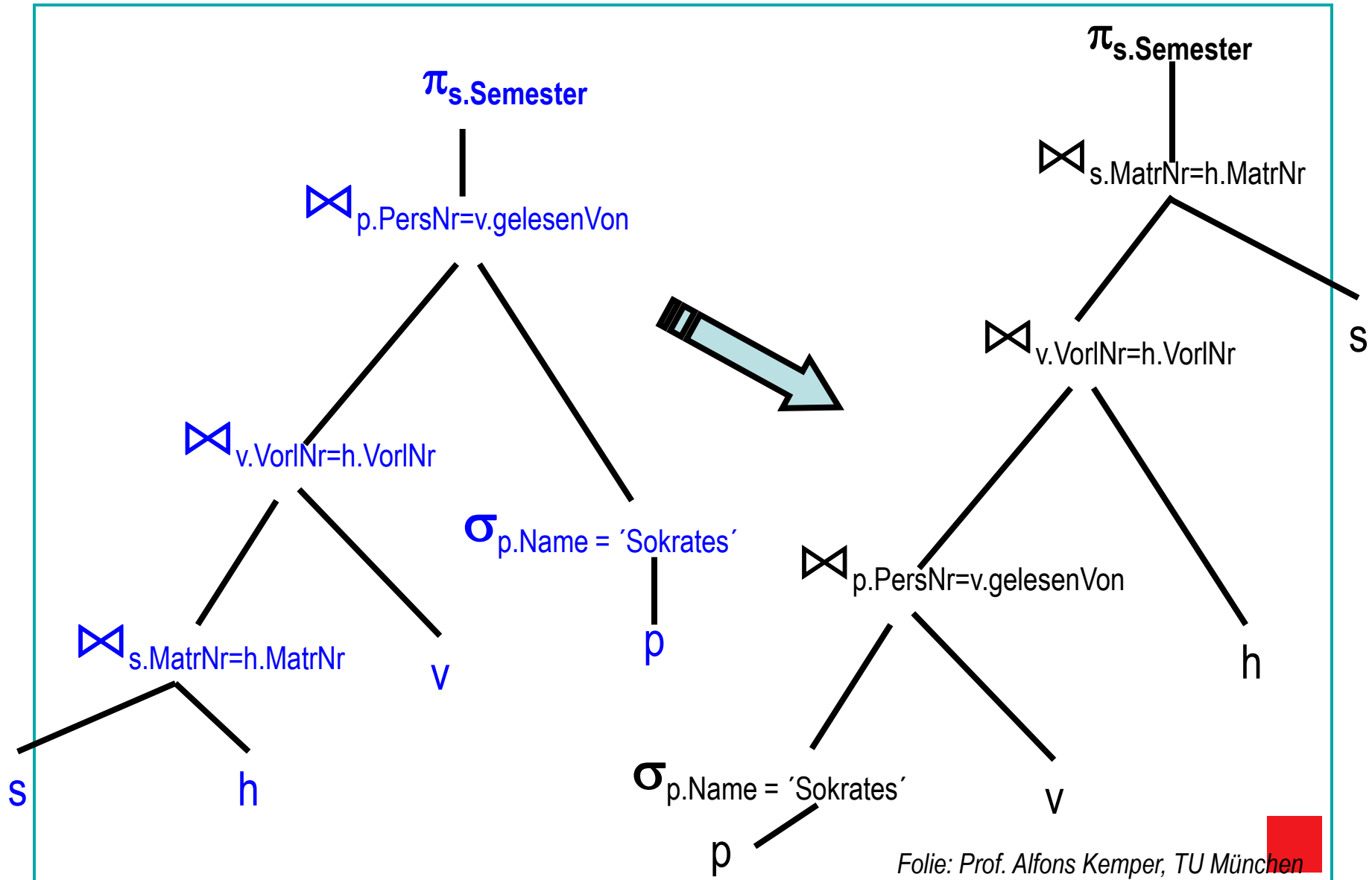
Zusammenfassung von Selektionen und Kreuzprodukten zu Joins



Folie: Prof. Alfons Kemper, TU München

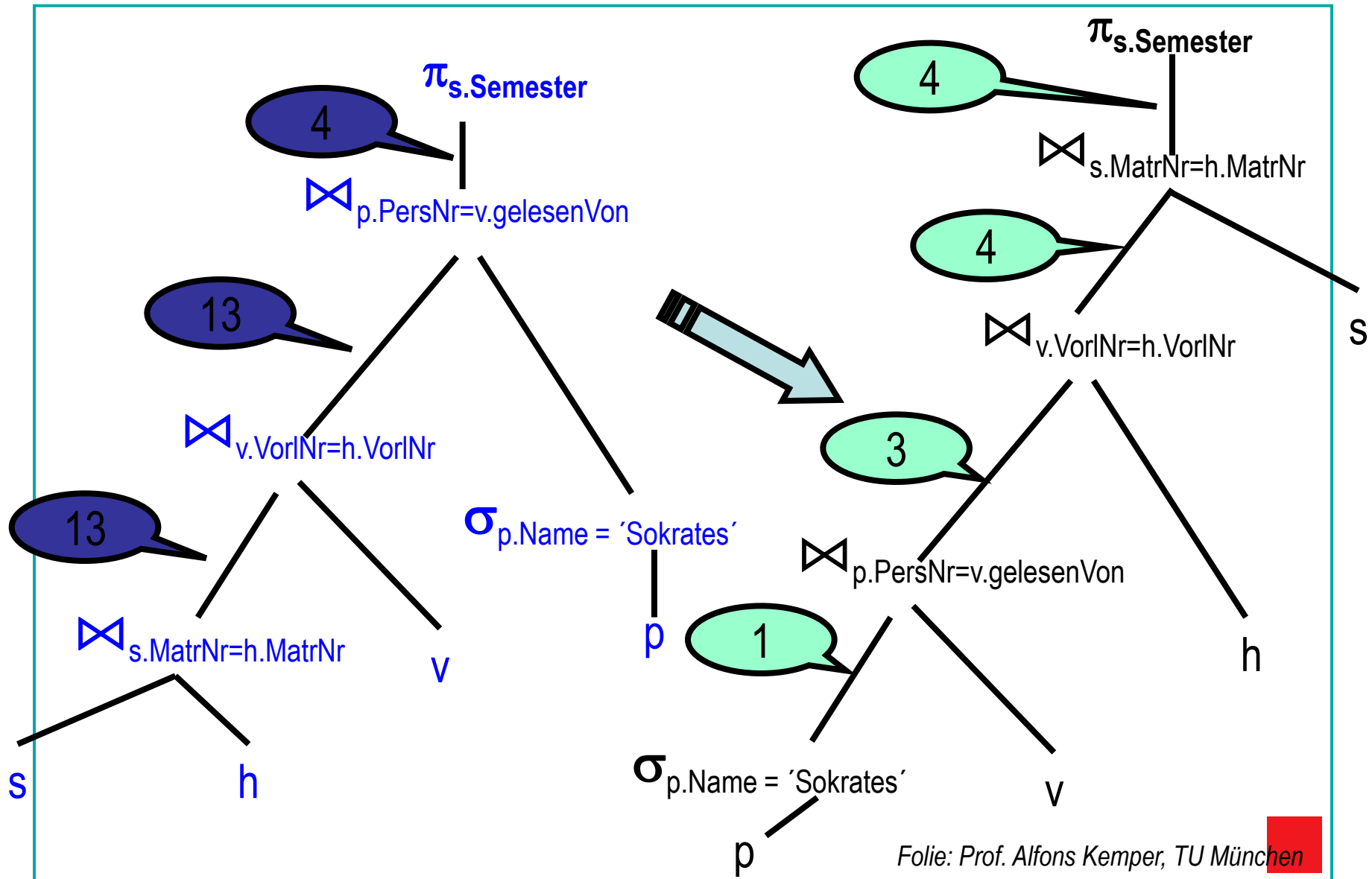


Optimierung der Joinreihenfolge: Kommutativität und Assoziativität ausnutzen

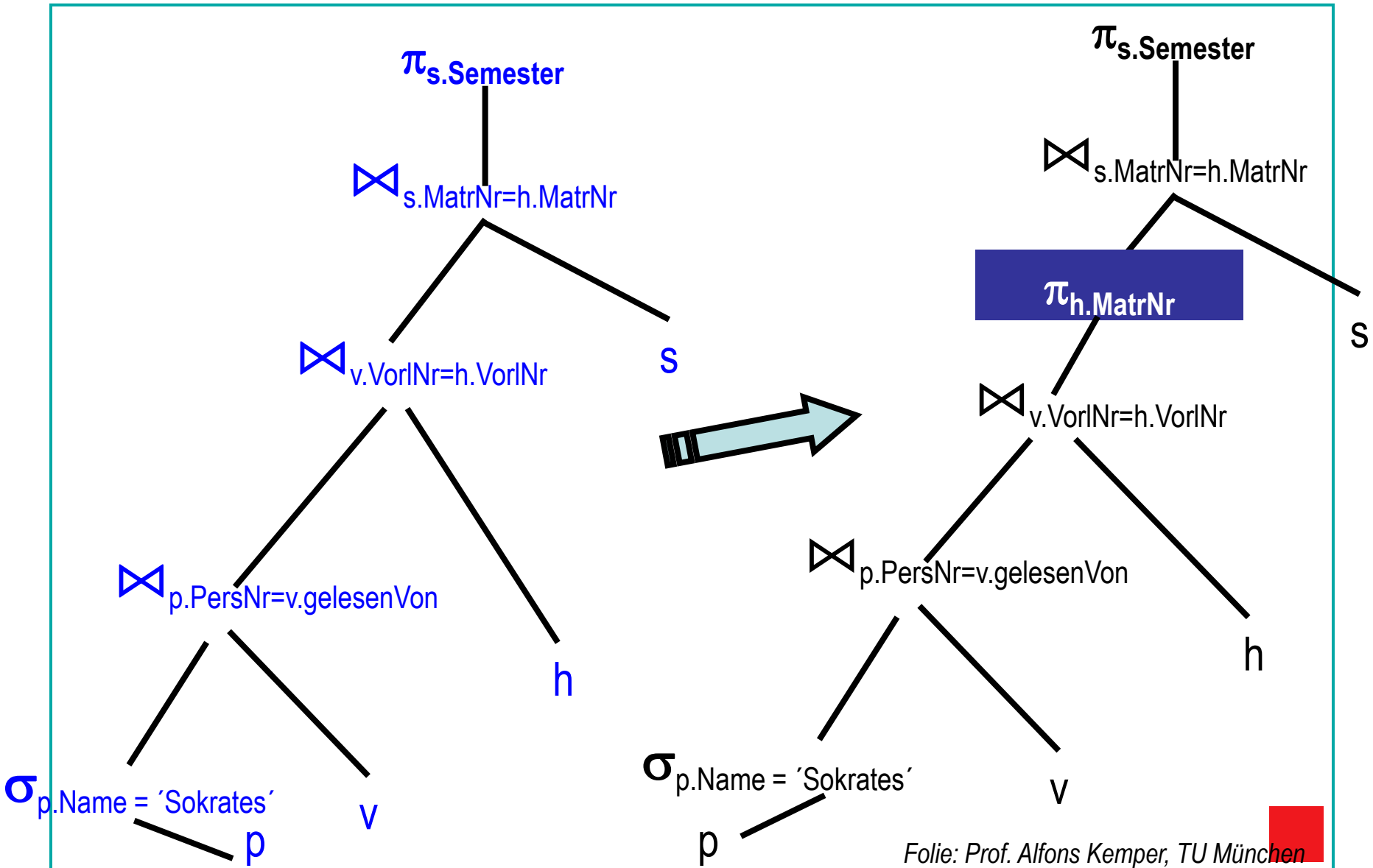


Folie: Prof. Alfons Kemper, TU München

Was hat's gebracht?



Folie: Prof. Alfons Kemper, TU München



- Bitte erstellen Sie eine Multiple Choice Aufgabe zum Thema Query Optimierung
 - Formulieren Sie eine Frage und 3 Antworten (A, B, C)
 - Davon sollte mindestens eine Antwort richtig und mindestens eine Antwort falsch sein
- Geben Sie die Aufgabe an Ihren rechten Nachbarn. Diskutieren Sie gemeinsam und markieren Sie die richtigen Lösungen
- Geben Sie am Ende der Vorlesung Ihre Aufgabe bei mir ab

5 min



- Einführung
- Parsen der Anfrage
- Transformationsregeln der RA
- Optimierung
- **Kostenmodelle**





- Konzeptionell: Generiere alle denkbaren Anfrageausführungspläne.
- Bewerte deren Kosten anhand eines Kostenmodells
 - Statistiken und Histogramme
 - Kalibrierung gemäß verwendeter Rechner
 - Abhängig vom verfügbaren Speicher
 - Aufwands-Kostenmodell
 - Durchsatz-maximierend
 - Nicht Antwortzeit-minimierend
- Führe billigsten Plan aus

Achtung: Nicht zu lange optimieren!



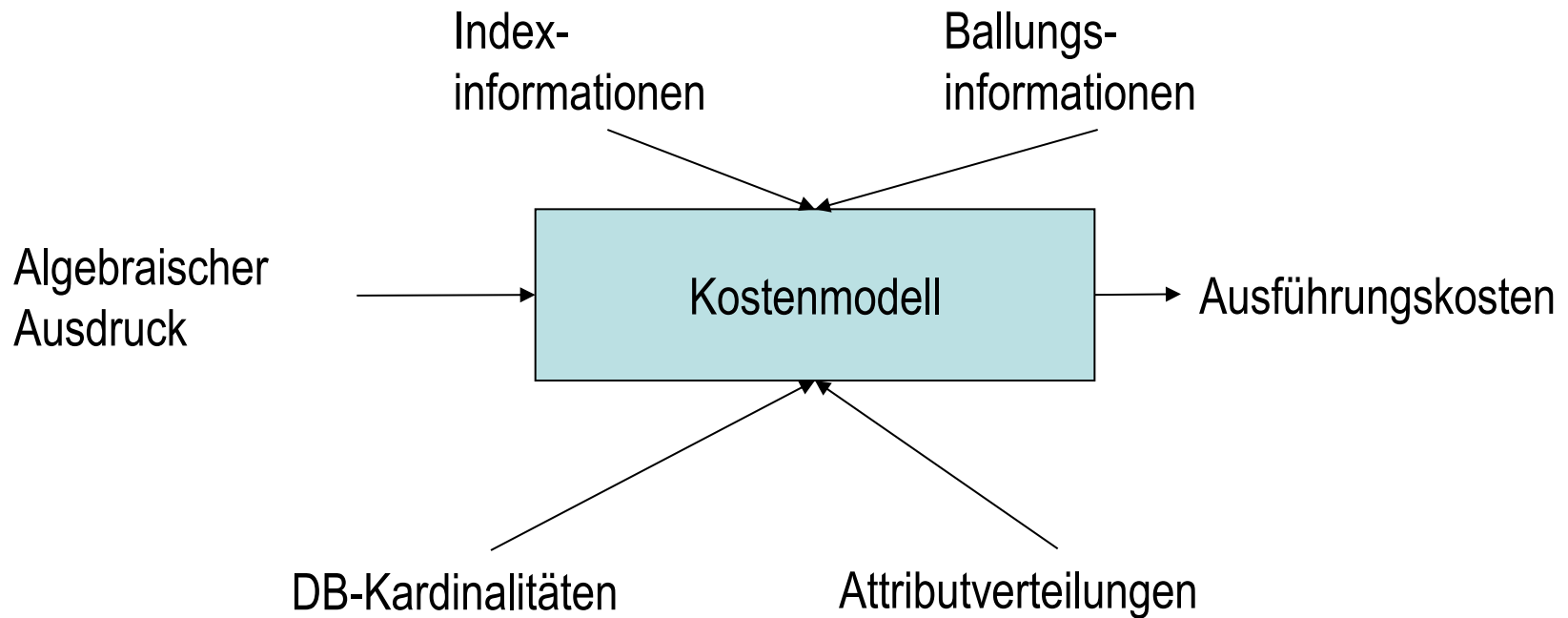


- Konzeptionell: Generiere alle denkbaren Anfrageausführungspläne
- Anzahl Bushy-Pläne mit n Tabellen

$$\frac{(2(n-1))!}{(n-1)!}$$

n	2^n	$(2(n-1))!/(n-1)!$
2	4	2
5	32	1.680
10	1.024	$1,76 \cdot 10^{10}$
20	1.048.576	$4,3 \cdot 10^{27}$

- Plankosten unterscheiden sich um viele Größenordnungen.
- Optimierungsproblem ist NP-hart



Folie nach Prof. Alfons Kemper, TU München



- Zu jeder Basisrelation
 - Anzahl der Tupel (Kardinalität)
 - Tupelgröße
- Zu (jedem) Attribut
 - Min / Max
 - Werteverteilung (Histogramm)
 - Anzahl der unterschiedlichen Werte (distinct values)
- Zum System
 - Speichergröße
 - Bandbreite
 - I/O Zeiten
 - CPU Zeiten
- Problem: Erstellung und Update der Statistiken
 - Deshalb meist nur explizit/manuell zu initiieren
 - runstats()





- Projektion:
 - Keine Kosten falls mit anderem Operator kombiniert
- Selektion
 - Ohne Index: Gesamte Relation von Festplatte lesen
 - Mit Baum-Index: Teil des Index von Platte lesen (Baumtiefe) und gesuchte Seite von Platte lesen
 - Bei Pipelining: (Fast) keine Kosten
- Join
 - Je nach Joinalgorithmus
 - Nested Loops, Hash-Join, Sort-Merge Join
- Sortierung: Nicht hier





- Wesentliches Kostenmerkmal: Anzahl der Tupel im Input
 - Insbesondere: Passt die Relation in den Hauptspeicher?
 - Selektion, Projektion, Sortierung, Join
- Output ist Input des nächsten Operators.
- Deshalb: Ein Kostenmodell schätzt u.a. für jede Operation die Anzahl der Ausgabetupel.
 - „Selektivität“ in Bezug auf Inputgröße
 - $\text{\#Ausgabetupel} = \text{\#Eingabetupel} \times \text{Selektivität}$
 - Auch „Selektivitätsfaktor“ (*selectivity factor*, sf)



- Selektivität schätzt Anzahl der qualifizierenden Tupel relativ zur Gesamtanzahl der Tupel in der Relation.
 - Vgl. rel. Häufigkeit, Wahrscheinlichkeit
- Projektion:
 - $sf = |R|/|R| = 1$
- Selektion:
 - $sf = |\sigma_C(R)| / |R|$
- Join:
 - $sf = |R \bowtie S| / |R \times S| = |R \bowtie S| / (|R| \cdot |S|)$





- Selektion:
 - Selektion auf einen Schlüssel:
 - $sf = 1 / |R|$
 - Selektion auf einen Attribut A mit m verschiedenen Werten:
 - $sf = (|R| / m) / |R| = 1/m$
 - Dies ist nur geschätzt!
- Join
 - Equijoin zwischen R und S über Fremdschlüssel in S
 - $sf = 1 / |R|$
 - „Beweis“: $sf = |R \bowtie S| / (|R| \times |S|) = |S| / (|R| \cdot |S|)$





- Gleichverteilung der Werte
 - Platzsparend (count, max, min), einfach
 - Schlechte Abschätzung bei “skew” (ungleiche Verteilung)
- Histogramme (Beispiel gleich)
 - Parametrisierte Größe, einfach
 - Güte der Abschätzung hängt von Histogrammtyp und -größe ab.
 - Außerdem: Aktualität
- Sampling
 - Repräsentative Teilmenge der Relation
 - Parametrisierte Größe, schwierig zu finden
 - Güte hängt von Samplingmethode und Samplegröße ab
 - Außerdem: Aktualität

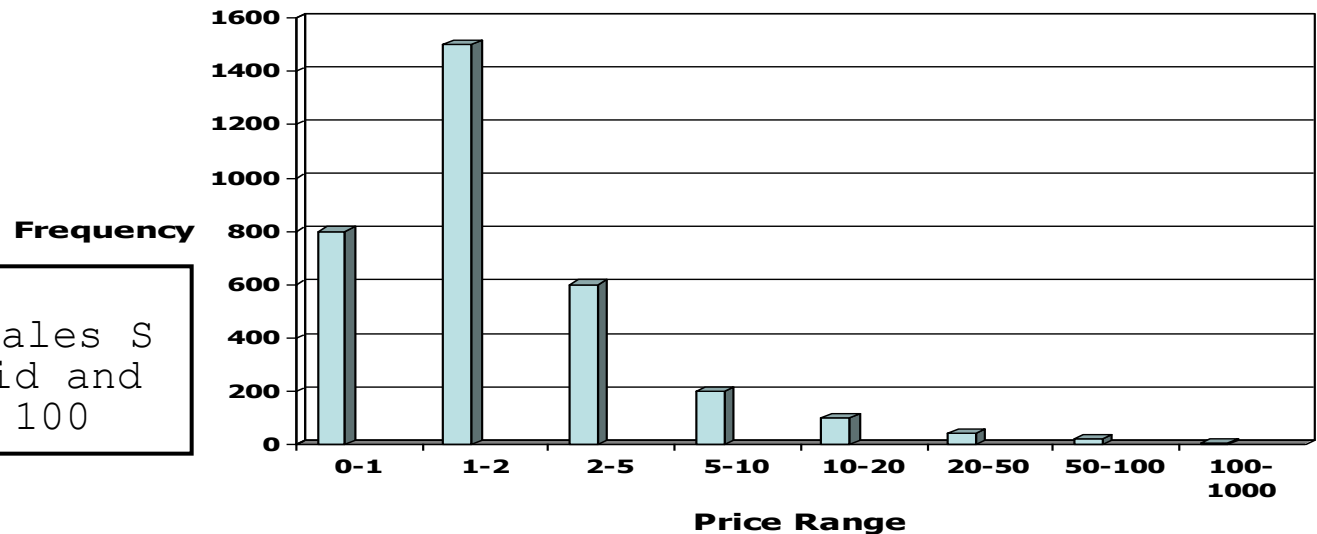




Beispiel zu Histogrammen



```
SELECT *  
FROM product p, sales S  
WHERE p.id=s.p_id and  
p.price > 100
```



- Gegeben 3300 products, 1M sales
- Gleichverteilung
 - Preisspanne ist 0-1000 => Selektivität der Bedingung ist 9/10
 - Erwartet: $9/10 \cdot 3300 \approx 3000$ Produkte
- Histogramm-basiert
 - Angenommen 10 equi-width buckets
 - Selektivität der Bedingung ist $5/3300 \approx 0,0015$ also 5 Produkte





- Parallelität / Pipelining
 - Kosten aller Operatoren können nicht addiert werden.
- Hauptspeichergröße
 - Pufferung und Caching
- I/O Kosten (Lesen einer Seite) vs. CPU Kosten
- Multiuser: Durchsatz statt Antwortzeit
- => Kostenmodelle sind hochkomplex





- Diverse Algorithmen für einzelne Operatoren
 - Insbesondere Join und Sortierung
- Kostenmodelle/Kostenschätzung genauer
- Optimale Joinreihenfolge: Dynamische Programmierung
- Physische Anfragepläne / Pipelining





- **Aufgabe 1 Anfragen & Modellierung“**

Denken Sie mal darüber nach, welche Anfragen Sie an die AOL Daten stellen möchten. Bitte Sie bitte ein logisches und physisches Schema zur Beantwortung dieser Anfragen.

- **Aufgabe 2 „SQL und Abfrageausführung“**

Bitte formulieren Sie für Ihre Analyseideen aus 1.) die SQL Anfragen. Sie verstehen auch Möglichkeiten der Abfrageausführung bzw. Optimierung.

- **Aufgabe 3 „Datenintegration“**

Zur Ausführung der Ausführung fehlen Ihnen noch externe Daten, z.B. aus dem Internet Archive, DMOZ oder Freebase.org. Bitte ergänzen Sie Ihr Schema und die Datenbasis.

- **Aufgabe 4 „Analyse, Erkenntnisgewinn und Wert“**

Stellen Sie in 5 Minuten die wichtigsten Erkenntnisse aus den Daten vor. Bewerten Sie den Erkenntnisgewinn, z.B. gegenüber Ihren Kommilitonen oder der Literatur! Welche Erkenntnisse hätten einen kommerziellen Wert?



- Was sind Datenbanken?
 - Motivation, Historie, Datenunabhängigkeit, Einsatzgebiete
- Datenbankentwurf im ER-Modell & Relationaler Datenbankentwurf
 - Entities, Relationships, Kardinalitäten, Diagramme
 - Relationales Modell, ER -> Relational, Normalformen, Transformationseigenschaften
- Relationale Algebra & SQL
 - Kriterien für Anfragesprachen, Operatoren, Transformationen
 - SQL DDL, SQL DML, SELECT ... FROM ... WHERE ...
- Datenintegration & Transaktionsverwaltung
 - JDBC, Cursor, ETL
 - Mehrbenutzerbetrieb, Serialisierbarkeit, Sperrprotokolle, Fehlerbehandlung, Isolationsebenen in SQL
- Ausblick
 - Map/Reduce, HDFS, Hive ...
 - Wert von Daten



- Parsen der Anfrage
- Transformationsregeln der RA
- Optimierung
 - Logische Optimierung
 - Physische Optimierung
- Kostenmodelle
 - Statistiken
 - Selektivität
 - Histogramme



In der nächsten Veranstaltung:

- Datenintegration im Data Warehouse

