



**Hasso
Plattner
Institut**

IT Systems Engineering | Universität Potsdam

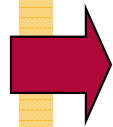
Informationsintegration
Der Bucket-Algorithmus

19.6.2012

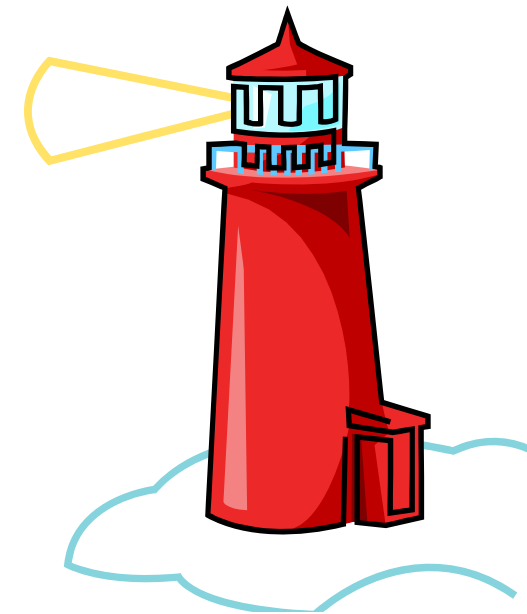
Felix Naumann

Überblick

2



- Nutzbarkeit und Nützlichkeit von Views
- Bucket Algorithmus am Beispiel
- Bucket Algorithmus en detail



Anfrageumschreibung (WdH)

3

- Prinzipiell:
 - Prüfe jede Kombination an Sichten auf Containment
 - Unendlich viele Kombinationen, da eine Sicht auch mehrfach in eine Umschreibung eingehen kann.
- Verbesserungen:
 - Satz: Umschreibung mit maximal so vielen Sichten wie Relationen in Anfrage (ohne range-Prädikate).
 - Geschickte Vorauswahl der Sichten: Nutzbarkeit

LaV – Nutzen der Sichten

4

- Frage 1: Wann sind Sichten nutzbar?
 - Informell:
 - Mindestens eine Relationen mit Anfrage gemeinsam
 - Mindestens einige Attribute der Anfrage
 - Prädikate sind schwächer oder gleich (äquivalente Umschreibung)
 - Prädikate sind stärker (contained Umschreibung)
- Frage 2: Wann sind Sichten nützlich?
 - Bei Optimierung mit MVs: Schneller Ausführung
 - Bei Integration mit LaV:
 - Zusätzliche Tupel
 - Zusätzliche Attribute

Beispiel – Globale Anfrage

5

■ Ausschnitt Globales Schema

- `Lehrt (prof, kurs_id, sem, eval, univ)` L
- `Eingeschrieben (stud, kurs_id, sem)` E
- `Betreut (prof, stud)` B

■ Anfrage

- ```
SELECT B.prof, B.stud, E.sem
FROM Eingeschrieben E, Lehrt L, Betreut B
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
AND B.prof = L.prof
AND B.stud = E.stud
AND E.sem ≥ „WS98“
```

## ■ In Worten?

- Finde Professoren mit betreuten Studenten und Semester, die seit WS98 auch einen Kurs dieses Professors besucht haben.

## ■ Anmerkung: kurs\_id sei global konsistent

# LaV – Beispiel

6

## Globale Anfrage

```
SELECT B.prof, B.stud, E.sem
FROM Eingeschrieben E, Lehrt L, Betreut B
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
AND B.prof = L.prof
AND B.stud = E.stud
AND E.sem ≥ „WS98“
```

## Quelle 7:

```
CREATE VIEW StudProf AS
SELECT L.prof, E.stud, E.sem
FROM Eingeschrieben E, Lehrt L
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
AND E.sem ≥ „WS98“
```

Frage:

- nutzbar?
- nützlich?

## Umschreibung

```
SELECT B.prof, B.stud, S.sem
FROM StudProf S, Betreut B
WHERE B.prof = S.prof
AND B.stud = S.stud
```

Vorsicht: Umschreibung nutzt  
noch globale Relation B!

Annahme hier: Triviale Sicht

# LaV – Beispiel

7

## Globale Anfrage

```
SELECT B.prof, B.stud, E.sem
FROM Eingeschrieben E, Lehrt L, Betreut B
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
AND B.prof = L.prof
AND B.stud = E.stud
AND E.sem ≥ „WS98“
```

## Quelle 8:

```
CREATE VIEW StudProf2 AS
SELECT L.prof, E.stud, E.sem
FROM Eingeschrieben E, Lehrt L
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
AND E.sem ≥ „WS97“
```

Schwächere  
Bedingung

Frage:

- nutzbar?
- nützlich?

## Umschreibung

```
SELECT B.prof, B.stud, S.sem
FROM StudProf2 S, Betreut B
WHERE B.prof = S.prof
AND B.stud = S.stud
AND S.sem ≥ „WS98“
```

Umschreibung ist contained  
bzw. äquivalent.

# LaV – Beispiel

8

## Globale Anfrage

```
SELECT B.prof, B.stud, E.sem
FROM Eingeschrieben E, Lehrt L, Betreut B
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
AND B.prof = L.prof
AND B.stud = E.stud
AND E.sem ≥ „WS98“
```

## Quelle 9:

```
CREATE VIEW StudProf3 AS
SELECT L.prof, E.stud, E.sem
FROM Eingeschrieben E, Lehrt L
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
AND E.sem ≥ „WS99“
```

Stärkere  
Bedingung

Frage:

- nutzbar?
- nützlich?

## Umschreibung

```
SELECT B.prof, B.stud, S.sem
FROM StudProf3 S, Betreut B
WHERE B.prof = S.prof
AND B.stud = S.stud
```

Umschreibung ist contained  
aber nicht äquivalent  
⇒ OK für Datenintegration,  
aber ungünstig für MVs



# LaV – Beispiel

9

## Globale Anfrage

```
SELECT B.prof, B.stud, E.sem
FROM Eingeschrieben E, Lehrt L, Betreut B
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
AND B.prof = L.prof
AND B.stud = E.stud
AND E.sem ≥ „WS98“
```

Frage:

- nutzbar?
- nützlich?

L.prof fehlt

## Quelle 10:

```
CREATE VIEW StudProf4 AS
SELECT E.stud, E.sem
FROM Eingeschrieben E, Lehrt L
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
AND E.sem ≥ „WS98“
```

## Umschreibung

```
SELECT B.prof, B.stud, S.sem
FROM StudProf4 S, Betreut B, Lehrt L
WHERE B.prof = L.prof
AND B.stud = S.stud
```

Mühselig: L muss erneut  
gejoint werden

# LaV – Beispiel

10

## Globale Anfrage

```
SELECT B.prof, B.stud, E.sem
FROM Eingeschrieben E, Lehrt L, Betreut B
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
AND B.prof = L.prof
AND B.stud = E.stud
AND E.sem ≥ „WS98“
```

## Quelle 11:

```
CREATE VIEW StudProf5 AS
SELECT E.stud, E.sem, L.prof
FROM Eingeschrieben E, Lehrt L
WHERE E.kurs_id = L.kurs_id
AND E.sem ≥ „WS98“
```

E.sem =  
L.sem fehlt

Frage:

- nutzbar?
- nützlich?

## Umschreibung

```
SELECT B.prof, B.stud, S.sem
FROM StudProf5 S, Lehrt L, Betreut B
WHERE S.sem = L.sem
AND B.prof = L.prof
AND B.stud = S.stud
```

Mühselig: L muss erneut  
gejoint werden, weil L.sem  
von Quelle 11 nicht  
exportiert wird.

# LaV – Nutzbarkeit von Sichten

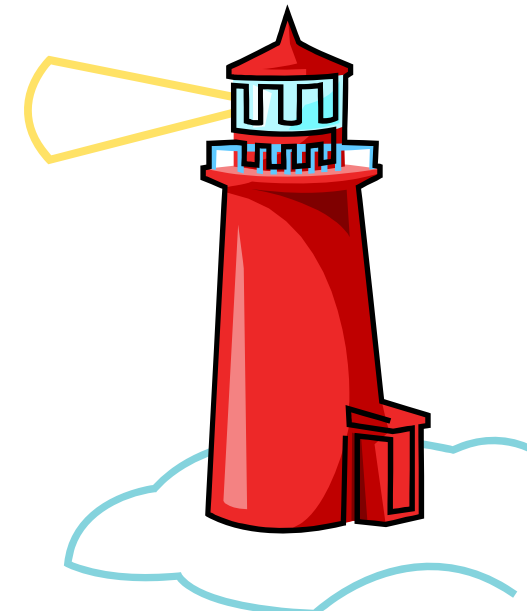
11

- Eine Sicht  $V$  ist nutzbar für eine äquivalente Umschreibung der Anfrage  $Q$ , falls
  - Jede Relation in  $V$  muss einer Relation in  $Q$  entsprechen.
  - Bedingungen in  $Q$  (Joins und Selektionen) auf den Relationen in  $V$  müssen entweder direkt in  $V$  angewendet werden  
oder  
Schwächere oder keine Bedingungen werden angewandt und die entsprechenden Attribute werden exportiert.
  - $V$  projiziert keine Attribute heraus, die noch in  $Q$  gebraucht werden und nicht anderweitig (andere Sicht oder Basisrelation) verfügbar sind.
- Nützlichkeit hängt von DBMS und Extension ab.
  - Indizes, etc.

# Überblick

12

- Nutzbarkeit und Nützlichkeit von Views
- Bucket Algorithmus am Beispiel
- Bucket Algorithmus en detail
- (Pruning im Bucket Algorithmus)



# LaV – Bucket Algorithm (BA)

13

- Problem:
  - Man muss exponentiell viele Kombinationen auf containment prüfen.
  - Schon die Prüfung selbst ist ein NP-vollständiges Problem
    - es gibt jedoch effiziente Algorithmen
    - idR sind die Kombinationen nicht groß (Anzahl der Relationen)
- Idee zur weiteren Verbesserung: Reduktion der Anzahl der Kombinationen durch **geschickte Vorauswahl**
  - Jede Relation der Anfrage erhält einen bucket (Eimer).
  - Schritt 1: Füge in jeden bucket alle Sichten, die für die Relation nutzbar sind.
  - Schritt 2: Prüfe alle Anfrageumschreibungs-Kombinationen, die aus jedem bucket genau eine Sicht enthalten.

# LaV – BA – Beispiel

14

## Quelle 1:

```
CREATE VIEW V1 AS
SELECT E.stud, K.titel, K.sem, K.kurs_id
FROM Eingeschrieben E, Kurs K
WHERE E.kurs_id = K.kurs_id
AND K.kurs_id ≥ 500
AND E.sem ≥ WS98
```

## Quelle 2:

```
CREATE VIEW V2 AS
SELECT E.stud, L.prof, E.sem, L.kurs_id
FROM Eingeschrieben E, Lehrt L
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
```

## Quelle 3:

```
CREATE VIEW V3 AS
SELECT E.stud, E.kurs_id
FROM Eingeschrieben E
WHERE E.sem ≤ WS94
```

## Quelle 4:

```
CREATE VIEW V4 AS
SELECT L.prof, K.kurs_id, K.titel, E.sem
FROM Eingeschrieben E, Kurs K, Lehrt L
WHERE E.kurs_id = K.kurs_id
AND E.kurs_id = L.kurs_id
AND L.sem = E.sem
AND E.sem ≤ WS97
```

V1 (stud, titel, sem, kurs\_id) :- E(stud,kurs\_id,sem), K(kurs\_id,titel), kurs\_id ≥ 500, sem ≥ WS98

V2 (stud, prof, sem, kurs\_id) :- E(stud, kurs\_id, sem), L(prof, kurs\_id, sem)

V3 (stud, kurs\_id) :- E(stud, kurs\_id, sem), sem ≤ WS94

V4 (prof, kurs\_id, titel, sem) :- L(prof, kurs\_id, sem), K(kurs\_id, titel), E(stud, kurs\_id, sem), sem ≤ WS97

# LaV – BA – Beispiel

15

V1 (stud, titel, sem, kurs\_id) :- E(stud,kurs\_id,sem), K(kurs\_id,titel), kurs\_id  $\geq$  500, sem  $\geq$  WS98

V2 (stud, prof, sem, kurs\_id) :- E(stud, kurs\_id, sem), L(prof, kurs\_id, sem)

V3 (stud, kurs\_id) :- E(stud, kurs\_id, sem), sem  $\leq$  WS94

V4 (prof, kurs\_id, titel, sem) :- L(prof, kurs\_id, sem), K(kurs\_id, titel), E(stud, kurs\_id, sem), sem  $\leq$  WS97

## Anfrage Q:

```
SELECT E.stud, K.kurs_id, L.prof
FROM Lehrt L, Eingeschrieben E, Kurs K
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
AND K.kurs_id = E.kurs_id
AND E.sem \geq WS95 AND kurs_id \geq 300
```

## Anfrage Q:

Q (stud, kurs\_id, prof) :- L(prof, kurs\_id, sem), E(stud,kurs\_id,sem), K(kurs\_id, titel),  
sem  $\geq$  WS95, kurs\_id  $\geq$  300

# LaV – BA – Beispiel

16

V1 (stud, titel, sem, kurs\_id) :- E(stud,kurs\_id,sem), K(kurs\_id,titel), kurs\_id ≥ 500, sem ≥ WS98

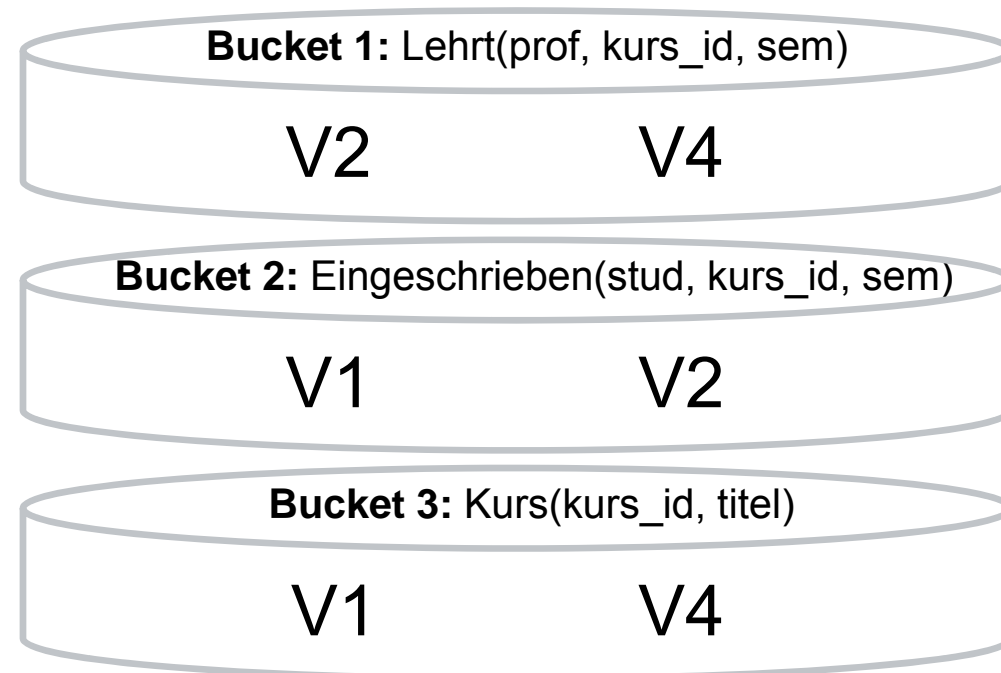
V2 (stud, prof, sem, kurs\_id) :- E(stud, kurs\_id, sem), L(prof, kurs\_id, sem)

V3 (stud, kurs\_id) :- E(stud, kurs\_id, sem), sem ≤ WS94

V4 (prof, kurs\_id, titel, sem) :- L(prof, kurs\_id, sem), K(kurs\_id, titel), E(stud, kurs\_id, sem), sem ≤ WS97

## Anfrage:

Q (stud, kurs\_id, prof) :-  
 L(prof, kurs\_id, sem),  
 E(stud,kurs\_id,sem),  
 K(kurs\_id, titel),  
 sem ≥ WS95,  
 kurs\_id ≥ 300





# LaV – BA – Beispiel

17

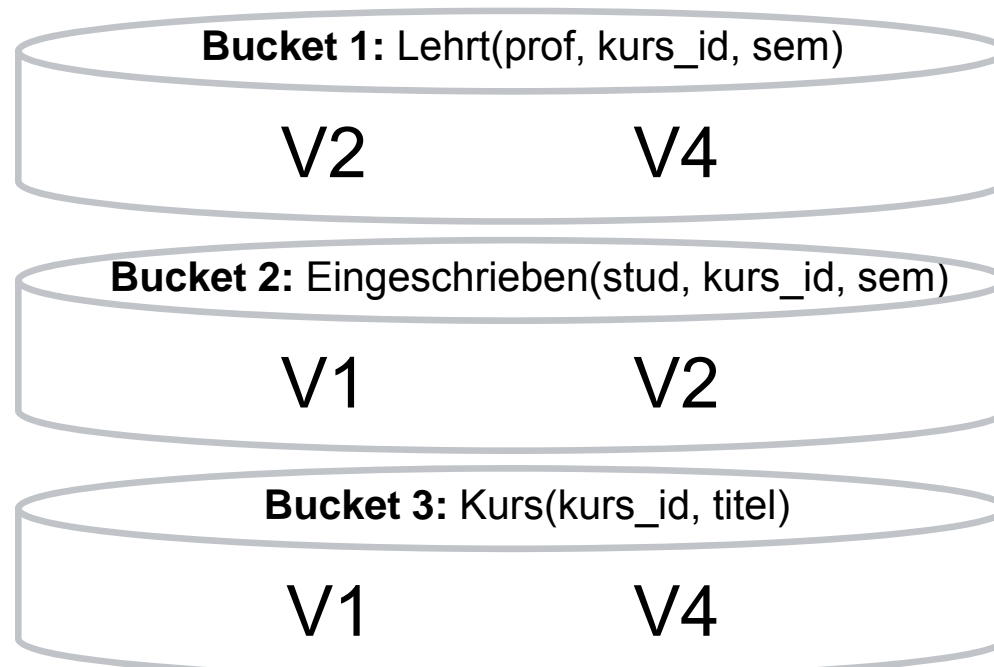
V1 (stud, titel, sem, kurs\_id) :- E(stud,kurs\_id,sem), K(kurs\_id,titel), kurs\_id ≥ 500, sem ≥ WS98

V2 (stud, prof, sem, kurs\_id) :- E(stud, kurs\_id, sem), L(prof, kurs\_id, sem)

V3 (stud, kurs\_id) :- E(stud, kurs\_id, sem), sem ≤ WS94

V4 (prof, kurs\_id, titel, sem) :- L(prof, kurs\_id, sem), K(kurs\_id, titel), E(stud, kurs\_id, sem), sem ≤ WS97

**Anfrage:** Q (stud, kurs\_id, prof) :- L(prof, kurs\_id, sem), E(stud,kurs\_id,sem), K(kurs\_id, titel),  
sem ≥ WS95, kurs\_id ≥ 300



## Kombinationen

V2, V1, V1 → V1, V2

V4, V1, V4 → ∅

V2, V2, V4 → V2, V4

V2, V2, V1 → V1, V2

V2, V1, V4 → ∅

V4, V1, V1 → ∅

V4, V2, V4 → V2, V4

V4, V2, V1 → ∅

# LaV – BA – Beispiel

18

V1 (stud, titel, sem, kurs\_id) :- E(stud,kurs\_id,sem), K(kurs\_id,titel), kurs\_id  $\geq$  500, sem  $\geq$  WS98

V2 (stud, prof, sem, kurs\_id) :- E(stud, kurs\_id, sem), L(prof, kurs\_id, sem)

V3 (stud, kurs\_id) :- E(stud, kurs\_id, sem), sem  $\leq$  WS94

V4 (prof, kurs\_id, titel, sem) :- L(prof, kurs\_id, sem), K(kurs\_id, titel), E(stud, kurs\_id, sem), sem  $\leq$  WS97

**Anfrage:** Q (stud, kurs\_id, prof) :- L(prof, kurs\_id, sem), E(stud,kurs\_id,sem), K(kurs\_id, titel),  
sem  $\geq$  WS95, kurs\_id  $\geq$  300

## Kombinationen

V2, V1, V1  $\rightarrow$  V1,V2

V4, V1, V4  $\rightarrow$   $\emptyset$

V2, V2, V4  $\rightarrow$  V2,V4

V2, V2, V1  $\rightarrow$  V1,V2

V2, V1, V4  $\rightarrow$   $\emptyset$

V4, V1, V1  $\rightarrow$   $\emptyset$

V4, V2, V4  $\rightarrow$  V2,V4

V4, V2, V1  $\rightarrow$   $\emptyset$

Ergebnis des Algorithmus:

V1,V2  $\cup$  V2,V4

# LaV – BA – Beispiel

19

## Quelle 1:

```
CREATE VIEW V1 AS
SELECT E.stud, K.titel, K.sem, K.kurs_id
FROM Eingeschrieben E, Kurs K
WHERE E.kurs_id = K.kurs_id
AND K.kurs_id LIKE „%VL_“
AND E.sem ≥ WS98
```

## Quelle 2:

```
CREATE VIEW V2 AS
SELECT E.stud, L.prof, E.sem, L.kurs_id
FROM Eingeschrieben E, Lehrt L
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
```

## Quelle 4:

```
CREATE VIEW V4 AS
SELECT L.prof, K.kurs_id, K.titel, E.sem
FROM Eingeschrieben E, Kurs K, Lehrt L
WHERE E.kurs_id = K.kurs_id
AND E.kurs_id = L.kurs_id
AND L.sem = E.sem
AND E.sem ≤ WS97
```

## Anfrage Q:

```
SELECT E.stud, K.kurs_id, L.prof
FROM Lehrt L, Eingeschrieben E, Kurs K
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
AND K.kurs_id = E.kurs_id
AND E.sem ≥ WS95
```

## Umgeschriebene Anfrage Q'

V1, V2 ∪ V2, V4:

```
SELECT V1.stud, V1.kurs_id, V2.prof
FROM V1, V2
WHERE V1.sem = V2.sem
AND V1.kurs_id = V2.kurs_id
```

UNION

```
SELECT V2.stud, V2.kurs_id, V2.prof
FROM V2, V4
WHERE V2.sem = V4.sem
AND V2.kurs_id = V4.kurs_id
AND V2.prof = V4.prof
AND V2.sem ≥ WS95
```

# LaV – BA – Beispiel

20

## Anfrage Q:

```
SELECT E.stud, K.kurs_id, L.prof
FROM Lehrt L, Eingeschrieben E, Kurs K
WHERE E.kurs_id = L.kurs_id
AND E.sem = L.sem
AND K.kurs_id = E.kurs_id
AND E.sem ≥ WS95
```

## Umgeschriebene Anfrage Q':

```
SELECT V1.stud, V1.kurs_id, V2.prof
FROM V1, V2
WHERE V1.sem = V2.sem
AND V1.kurs_id = V2.kurs_id
```

UNION

```
SELECT V2.stud, V2.kurs_id, V2.prof
FROM V2, V4
WHERE V2.sem = V4.sem
AND V2.kurs_id = V4.kurs_id
AND V2.prof = V4.prof
AND V2.sem ≥ WS95
```

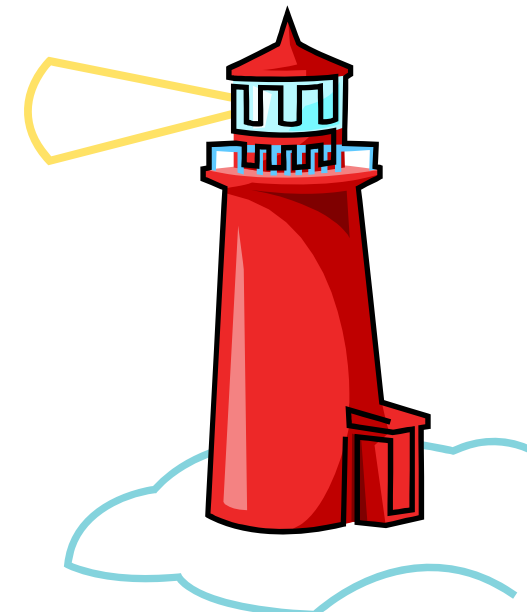
Jetzt: BA genauer

- Q' verwendet ausschließlich Sichten.
  - Jede Sicht entspricht einer lokalen Relation
- $Q' \subseteq Q$
- Q' ist maximally contained in Q.

# Überblick

21

- Nutzbarkeit und Nützlichkeit von Views
- Bucket Algorithmus am Beispiel
- Bucket Algorithmus en detail
- (Pruning im Bucket Algorithmus)



# BA – en detail

22

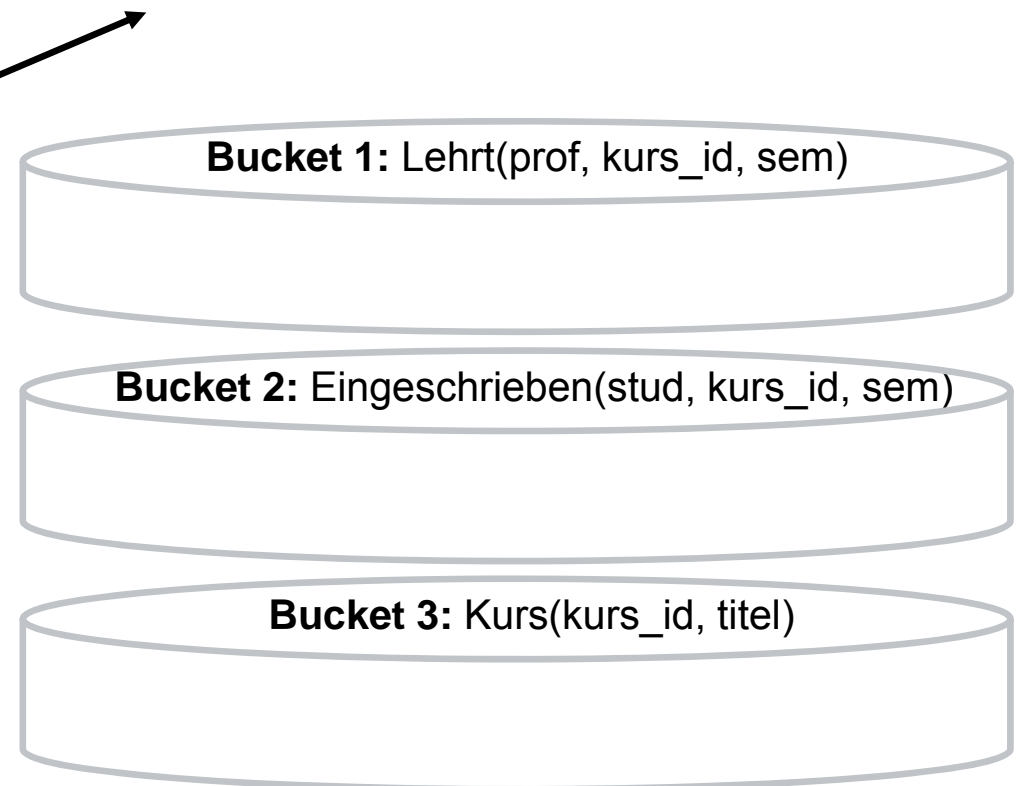
## Anfrage:

Q (stud, kurs\_id, prof) :-  
L(prof, kurs\_id, sem), E(stud,kurs\_id,sem), K(kurs\_id, titel), sem ≥ WS95, kurs\_id ≥ 300

Teilziele (*subgoals*) von Q

Erzeuge für jedes  
Teilziel einen *bucket*.

Fülle Sichten in  
die *buckets*.



## BA – en detail

23

Eine Sicht wird in ein bucket gestellt, wenn mindestens ein Teilziel der Sicht „geeignet“ ist. Also:

Für jeden bucket

Für jede Sicht:

Für jedes Teilziel der Sicht

Prüfung auf „Eignung“:

- 1. Unifier: Mapping von allen Attributen im Teilziel von Q auf Attribute im Teilziel von V.  
D.h.: Alle Anfrageattribute müssen vorkommen.
- 2. Kompatibilität: Prädikate sind passend.

# BA – en detail



```
procedure generate-relevant-sources(\mathcal{I}, Q)
/* \mathcal{I} is a set of information sources, and Q is a conjunctive
query of the form $q(\bar{X}) : (\exists \bar{X}) g_1(\bar{X}_1) \wedge \dots \wedge g_m(\bar{X}_m) \wedge C_q$,
where C_q is the conjunction of order atoms in Q . */
for every subgoal $g_i(\bar{X}_i)$, $1 \leq i \leq m$ do:
 $relevantSources_i = \emptyset$
 for every non-order conjunct $u(\bar{Y})$ in a formula
 r_v in the pair (v, r_v) in a description of a source in
 \mathcal{I} do:
 if $g_i = u$ or g_i and u are non-disjoint classes then:
 Let ψ be the mapping defined on the variables of
 r_v as follows:
 if Y is the j 'th variable in \bar{Y} and is not
 existentially quantified in r_v
 then $\psi(Y) = X_j$, where X_j is the j 'th
 variable in \bar{X}_i .
 else $\psi(Y)$ is a new variable that does not
 appear in Q or r_v .
 Let $C(Q)$ and $C(v)$ be the conjunction of
 constraint subgoals in Q and $\psi(r_v)$, respectively.
 if $C(Q) \wedge C(v)$ is satisfiable, then add $\psi(r_v)$
 to $relevantSources_i$.
 return $\{relevantSources_1, \dots, relevantSources_m\}$.
end generate-relevant-sources.
```

- Für jedes Teilziel in  $Q$
- erzeuge einen bucket
- Für jedes Teilziel in allen  $V$
- Finde Unifier (mapping)
- Prüfe Kompatibilität der Prädikate
- Füge  $V$  in bucket ein

Aus [LRO96a]



# BA – en detail Beispiel

25

Für jede Sicht: Betrachte deren Teilziele.

Falls mindestens ein Teilziel geeignet ist, füge Sicht in bucket ein.

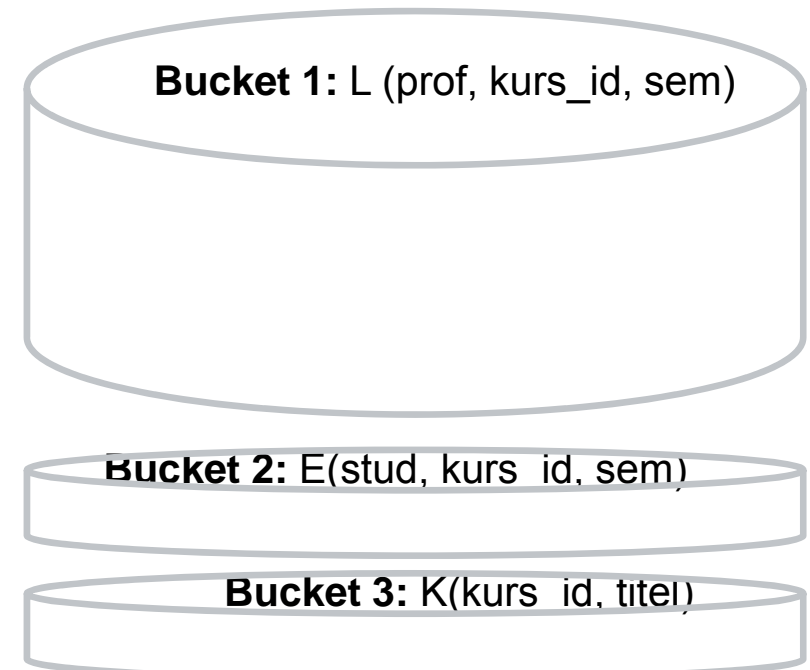
Prüfung auf „Eignung“:

1. Unifier: Alle Anfrageattribute müssen vorkommen.
2. Kompatibilität: Prädikate sind passend.

Q (stud, kurs\_id, prof) :-  
**L(prof, kurs\_id, sem),**  
 E(stud,kurs\_id,sem),  
 K(kurs\_id, titel),  
 sem ≥ WS95,  
 kurs\_id ≥ 300

V1 (stud, titel, sem, kurs\_id) :-  
 E(stud,kurs\_id,sem),  
 K(kurs\_id,titel),  
 kurs\_id ≥ 500,  
 sem ≥ WS98

prof wird nicht exportiert!



# BA – en detail Beispiel

26

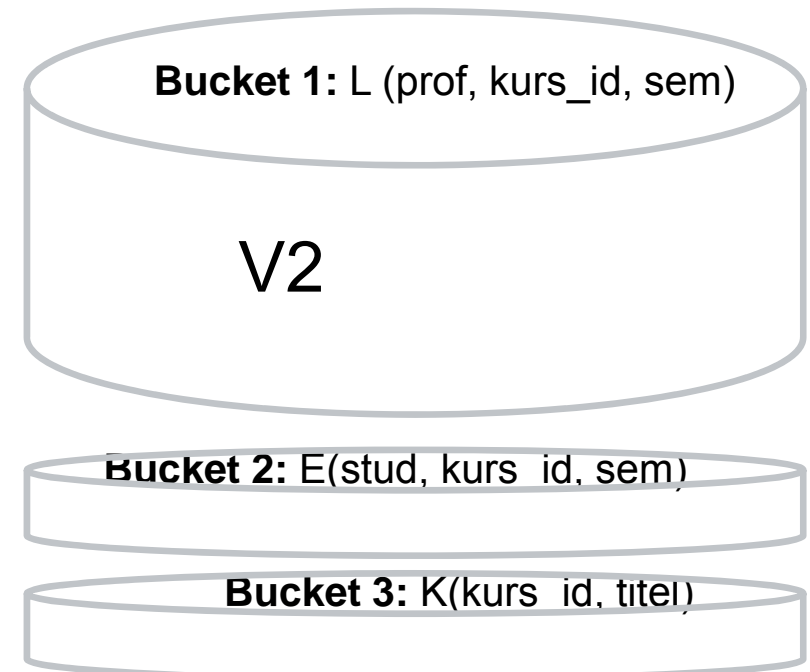
Für jede Sicht: Betrachte deren Teilziele.

Falls mindestens ein Teilziel geeignet ist, füge Sicht in bucket ein.

Prüfung auf „Eignung“:

1. Unifier: Alle Anfrageattribute müssen vorkommen.
2. Kompatibilität: Prädikate sind passend.

|                                                                                                                                                                                            |                                                                                                                |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| <p>Q (stud, kurs_id, prof) :-<br/> <b>L(prof, kurs_id, sem),</b><br/>         E(stud,kurs_id,sem),<br/>         K(kurs_id, titel),<br/>         sem ≥ WS95,<br/>         kurs_id ≥ 300</p> | <p>V2 (stud, prof, sem, kurs_id) :-<br/>         E(stud, kurs_id, sem),<br/>         L(prof, kurs_id, sem)</p> |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|



# BA – en detail Beispiel

27

Für jede Sicht: Betrachte deren Teilziele.

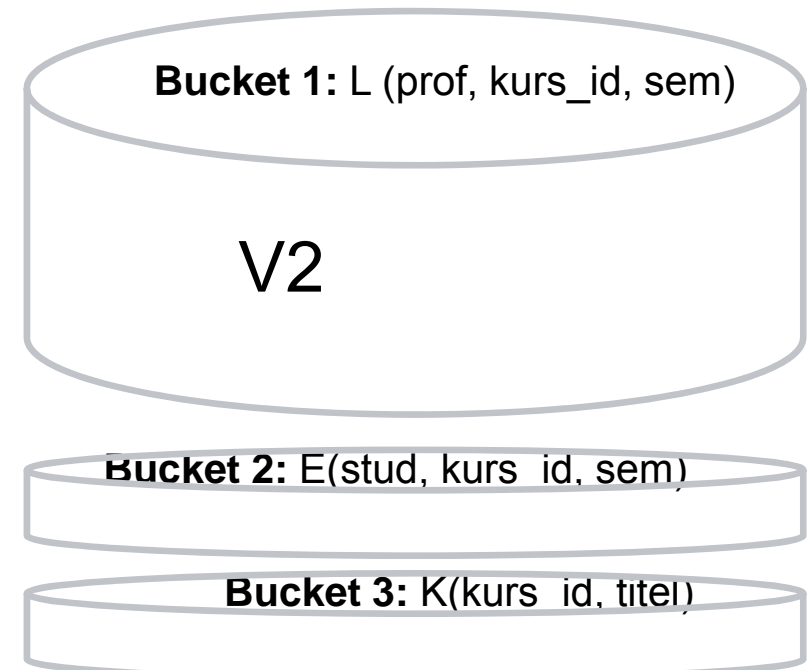
Falls mindestens ein Teilziel geeignet ist, füge Sicht in bucket ein.

Prüfung auf „Eignung“:

1. Unifier: Alle Anfrageattribute müssen vorkommen.
2. Kompatibilität: Prädikate sind passend.

|                                                                                                                                                                                            |                                                                                          |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|
| <p>Q (stud, kurs_id, prof) :-<br/> <b>L(prof, kurs_id, sem),</b><br/>         E(stud,kurs_id,sem),<br/>         K(kurs_id, titel),<br/>         sem ≥ WS95,<br/>         kurs_id ≥ 300</p> | <p>V3 (stud, kurs_id) :-<br/>         E(stud, kurs_id, sem),<br/>         sem ≤ WS94</p> |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|

prof wird nicht exportiert!



# BA – en detail Beispiel

28

Für jede Sicht: Betrachte deren Teilziele.

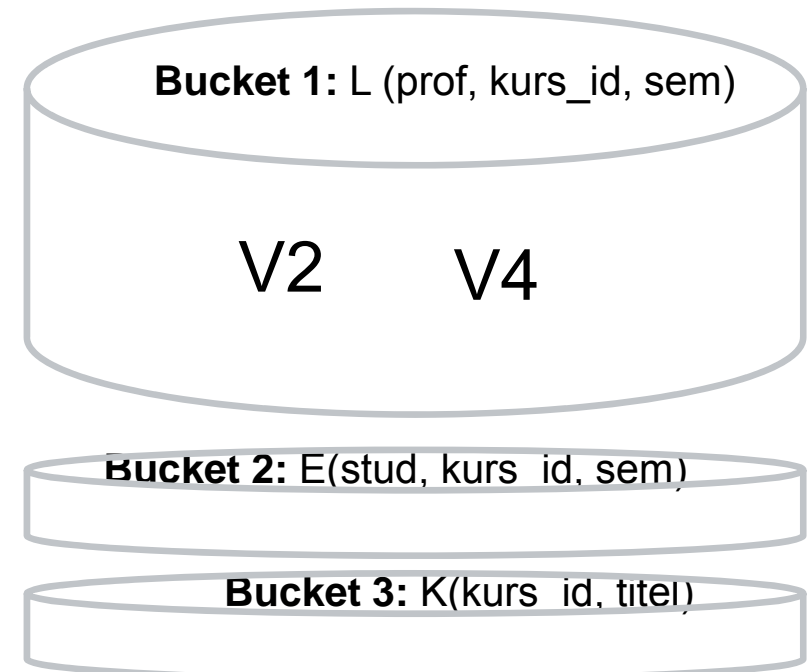
Falls mindestens ein Teilziel geeignet ist, füge Sicht in bucket ein.

Prüfung auf „Eignung“:

1. Unifier: Alle Anfrageattribute müssen vorkommen.
2. Kompatibilität: Prädikate sind passend.

Q (stud, kurs\_id, prof) :-  
**L(prof, kurs\_id, sem),**  
 E(stud,kurs\_id,sem),  
 K(kurs\_id, titel),  
 sem ≥ WS95,  
 kurs\_id ≥ 300

V4 (prof, kurs\_id, titel, sem) :-  
 L(prof, kurs\_id, sem),  
 K(kurs\_id, titel),  
 E(stud, kurs\_id, sem),  
 sem ≤ WS97



# BA – en detail Beispiel

29

Für jede Sicht: Betrachte deren Teilziele.

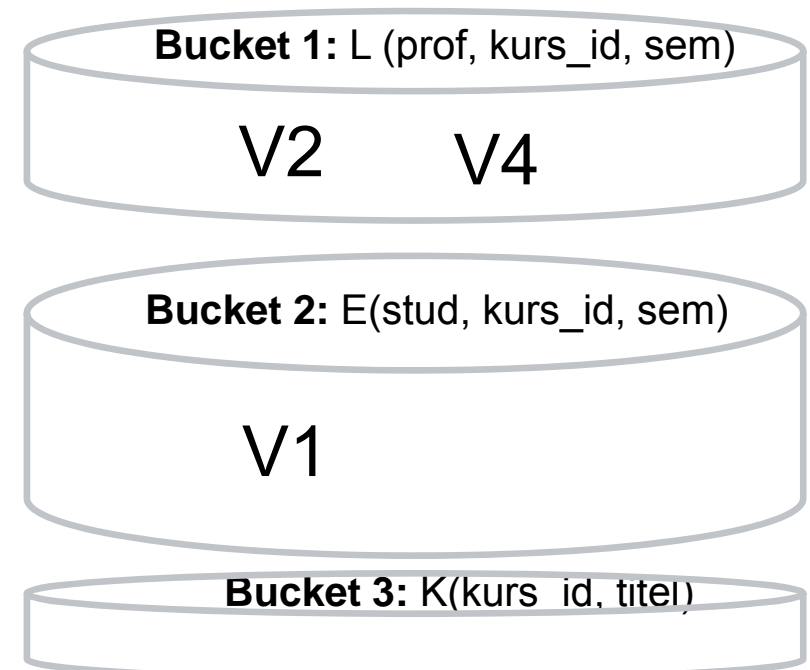
Falls mindestens ein Teilziel geeignet ist, füge Sicht in bucket ein.

Prüfung auf „Eignung“:

1. Unifier: Alle Anfrageattribute müssen vorkommen.
2. Kompatibilität: Prädikate sind passend.

Q (stud, kurs\_id, prof) :-  
 L(prof, kurs\_id, sem),  
**E(stud,kurs\_id,sem),**  
 K(kurs\_id, titel),  
 sem ≥ WS95,  
 kurs\_id ≥ 300

V1 (stud, titel, sem, kurs\_id) :-  
 E(stud,kurs\_id,sem),  
 K(kurs\_id,titel),  
 kurs\_id ≥ 500,  
 sem ≥ WS98



# BA – en detail Beispiel

30

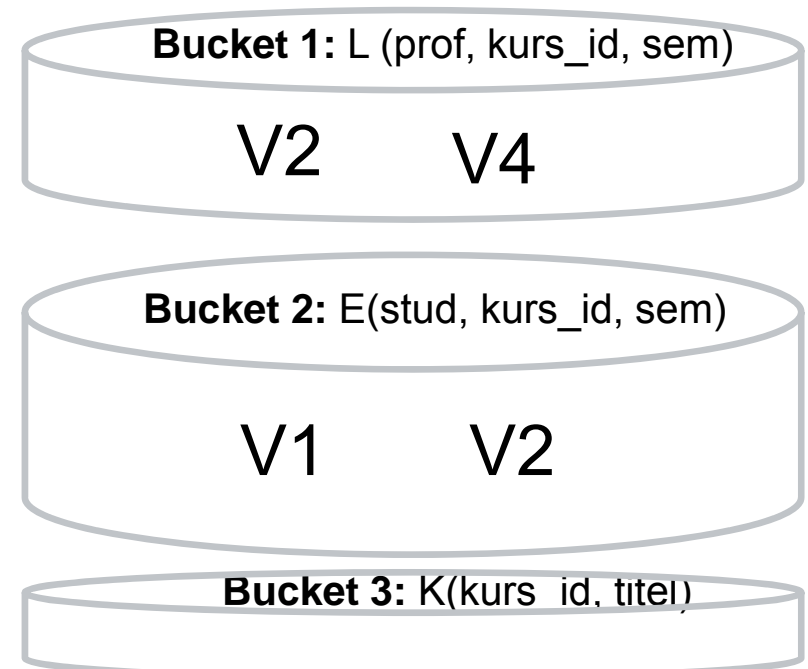
Für jede Sicht: Betrachte deren Teilziele.

Falls mindestens ein Teilziel geeignet ist, füge Sicht in bucket ein.

Prüfung auf „Eignung“:

1. Unifier: Alle Anfrageattribute müssen vorkommen.
2. Kompatibilität: Prädikate sind passend.

|                                                                                                                                                                                            |                                                                                                                |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| <p>Q (stud, kurs_id, prof) :-<br/>         L(prof, kurs_id, sem),<br/> <b>E(stud,kurs_id,sem)</b>,<br/>         K(kurs_id, titel),<br/>         sem ≥ WS95,<br/>         kurs_id ≥ 300</p> | <p>V2 (stud, prof, sem, kurs_id) :-<br/>         E(stud, kurs_id, sem),<br/>         L(prof, kurs_id, sem)</p> |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|



# BA – en detail Beispiel

31

Für jede Sicht: Betrachte deren Teilziele.

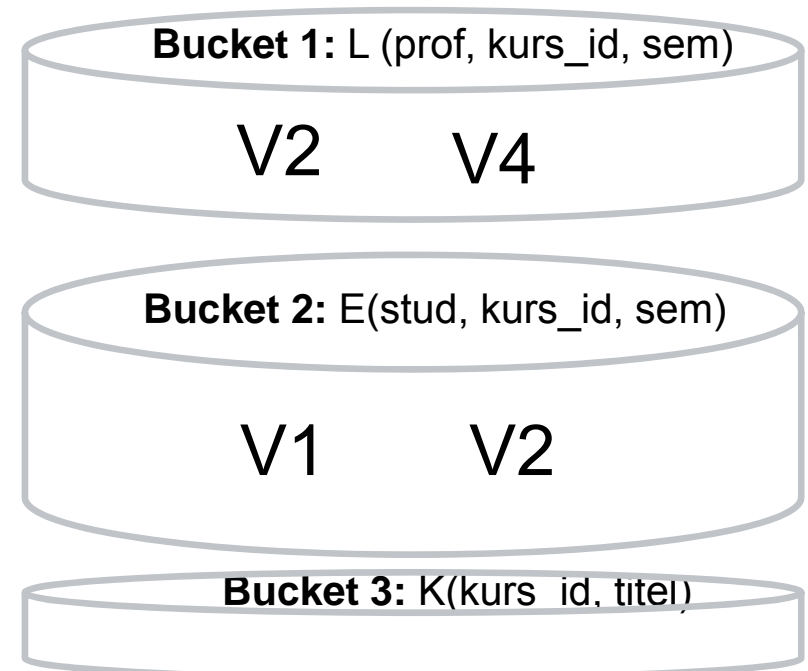
Falls mindestens ein Teilziel geeignet ist, füge Sicht in bucket ein.

Prüfung auf „Eignung“:

1. Unifier: Alle Anfrageattribute müssen vorkommen.
2. Kompatibilität: Prädikate sind passend.

|                                                                                                                                                                                            |                                                                                          |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|
| <p>Q (stud, kurs_id, prof) :-<br/>         L(prof, kurs_id, sem),<br/> <b>E(stud,kurs_id,sem),</b><br/>         K(kurs_id, titel),<br/>         sem ≥ WS95,<br/>         kurs_id ≥ 300</p> | <p>V3 (stud, kurs_id) :-<br/>         E(stud, kurs_id, sem),<br/>         sem ≤ WS94</p> |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|

sem ≥ WS95 vs. sem ≤ WS94



# BA – en detail Beispiel

32

Für jede Sicht: Betrachte deren Teilziele.

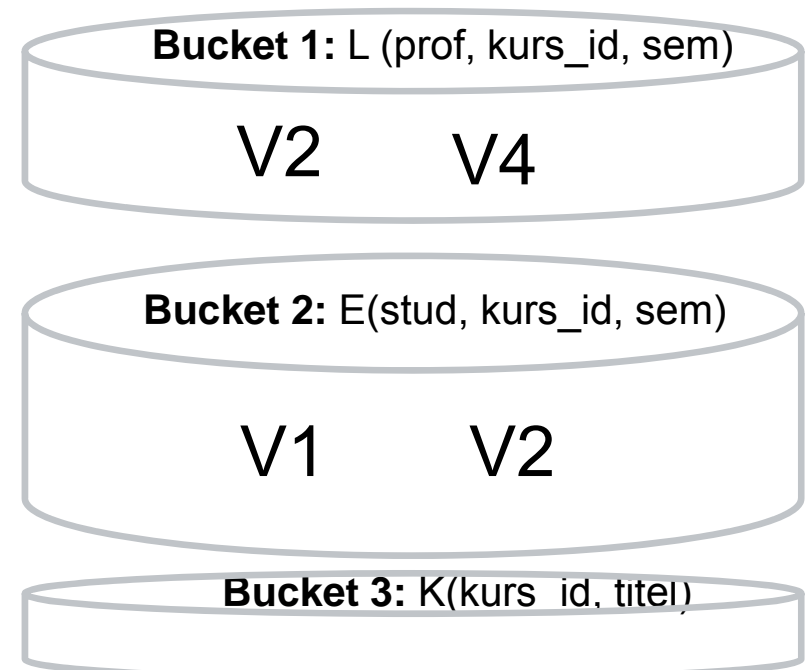
Falls mindestens ein Teilziel geeignet ist, füge Sicht in bucket ein.

Prüfung auf „Eignung“:

1. Unifier: Alle Anfrageattribute müssen vorkommen.
2. Kompatibilität: Prädikate sind passend.

|                                                                                                                                                                                            |                                                                                                                                                                          |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p>Q (stud, kurs_id, prof) :-<br/>         L(prof, kurs_id, sem),<br/> <b>E(stud,kurs_id,sem),</b><br/>         K(kurs_id, titel),<br/>         sem ≥ WS95,<br/>         kurs_id ≥ 300</p> | <p>V4 (prof, kurs_id, titel, sem) :-<br/>         L(prof, kurs_id, sem),<br/>         K(kurs_id, titel),<br/>         E(stud, kurs_id, sem),<br/>         sem ≤ WS97</p> |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**stud wird nicht exportiert!**





# BA – en detail Beispiel

33

Für jede Sicht: Betrachte deren Teilziele.

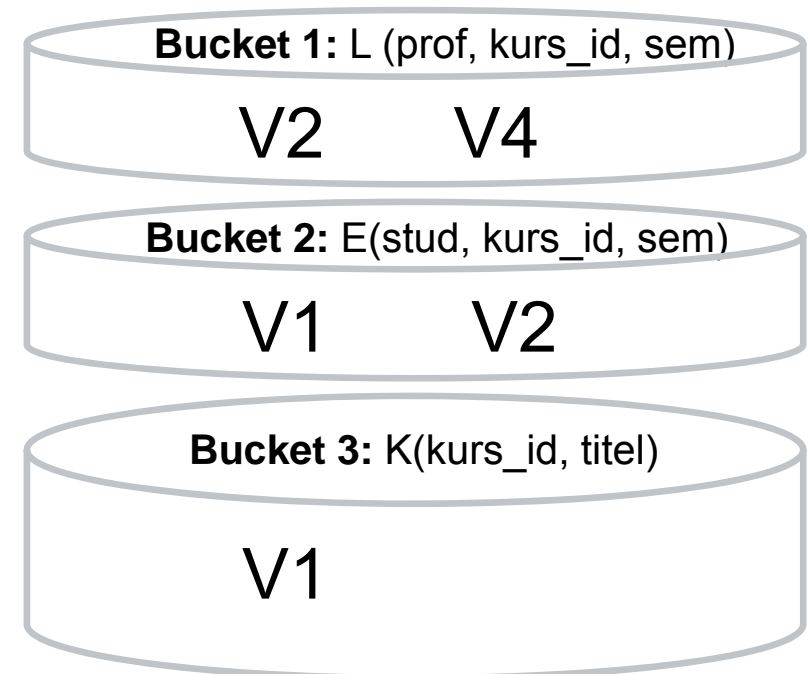
Falls mindestens ein Teilziel geeignet ist, füge Sicht in bucket ein.

Prüfung auf „Eignung“:

1. Unifier: Alle Anfrageattribute müssen vorkommen.
2. Kompatibilität: Prädikate sind passend.

Q (stud, kurs\_id, prof) :-  
 L(prof, kurs\_id, sem),  
 E(stud,kurs\_id,sem),  
**K(kurs\_id, titel),**  
 sem ≥ WS95,  
 kurs\_id ≥ 300

V1 (stud, titel, sem, kurs\_id) :-  
 E(stud,kurs\_id,sem),  
 K(kurs\_id,titel),  
 kurs\_id ≥ 500,  
 sem ≥ WS98



# BA – en detail Beispiel

34

Für jede Sicht: Betrachte deren Teilziele.

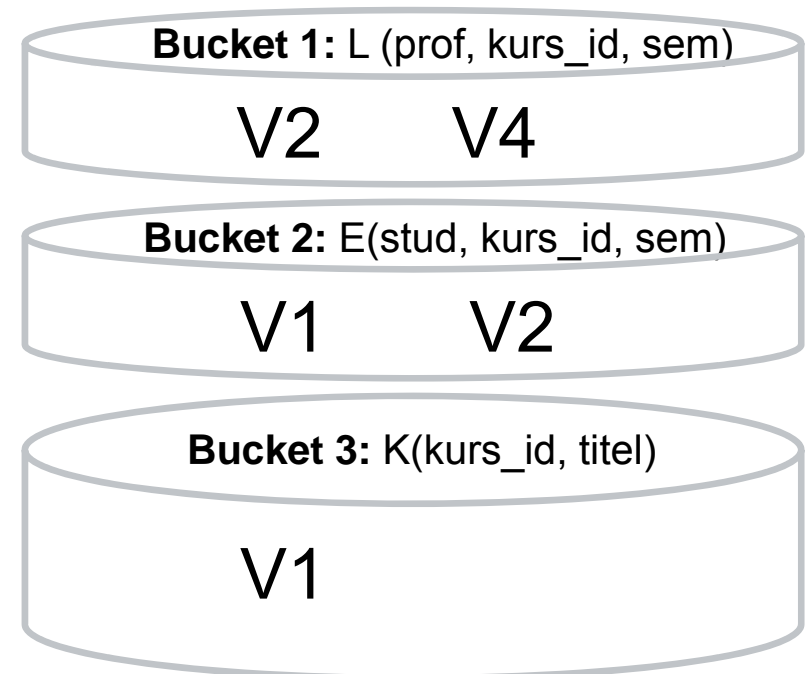
Falls mindestens ein Teilziel geeignet ist, füge Sicht in bucket ein.

Prüfung auf „Eignung“:

1. Unifier: Alle Anfrageattribute müssen vorkommen.
2. Kompatibilität: Prädikate sind passend.

|                                                                                                                                                                                            |                                                                                                                |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| <p>Q (stud, kurs_id, prof) :-<br/>         L(prof, kurs_id, sem),<br/>         E(stud,kurs_id,sem),<br/> <b>K(kurs_id, titel),</b><br/>         sem ≥ WS95,<br/>         kurs_id ≥ 300</p> | <p>V2 (stud, prof, sem, kurs_id) :-<br/>         E(stud, kurs_id, sem),<br/>         L(prof, kurs_id, sem)</p> |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|

titel wird nicht exportiert!



# BA – en detail Beispiel

35

Für jede Sicht: Betrachte deren Teilziele.

Falls mindestens ein Teilziel geeignet ist, füge Sicht in bucket ein.

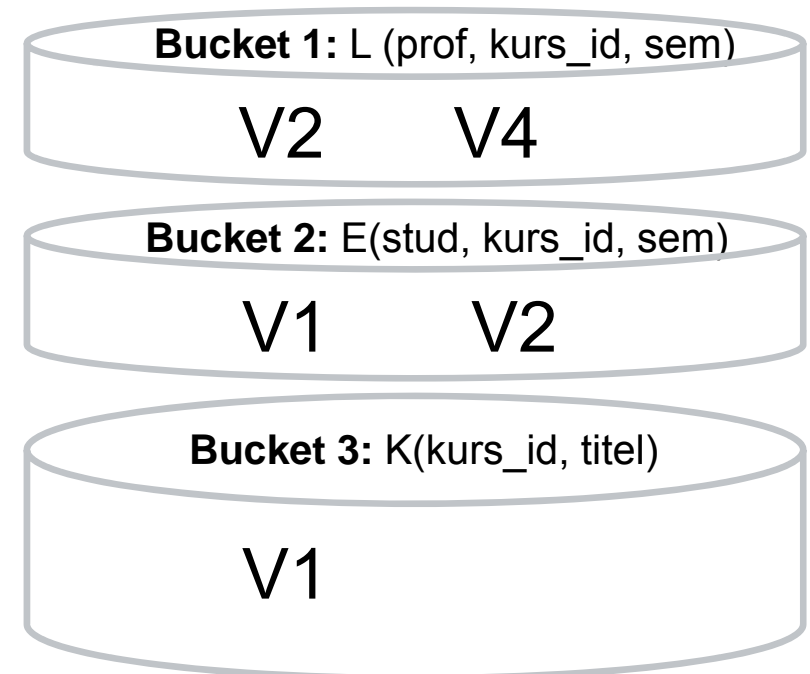
Prüfung auf „Eignung“:

1. Unifier: Alle Anfrageattribute müssen vorkommen.
2. Kompatibilität: Prädikate sind passend.

Q (stud, kurs\_id, prof) :-  
 L(prof, kurs\_id, sem),  
 E(stud, kurs\_id, sem),  
**K(kurs\_id, titel),**  
 sem ≥ WS95,  
 kurs\_id ≥ 300

V3 (stud, kurs\_id) :-  
 E(stud, kurs\_id, sem),  
 sem ≤ WS94

sem ≤ WS94



# BA – en detail Beispiel

36

Für jede Sicht: Betrachte deren Teilziele.

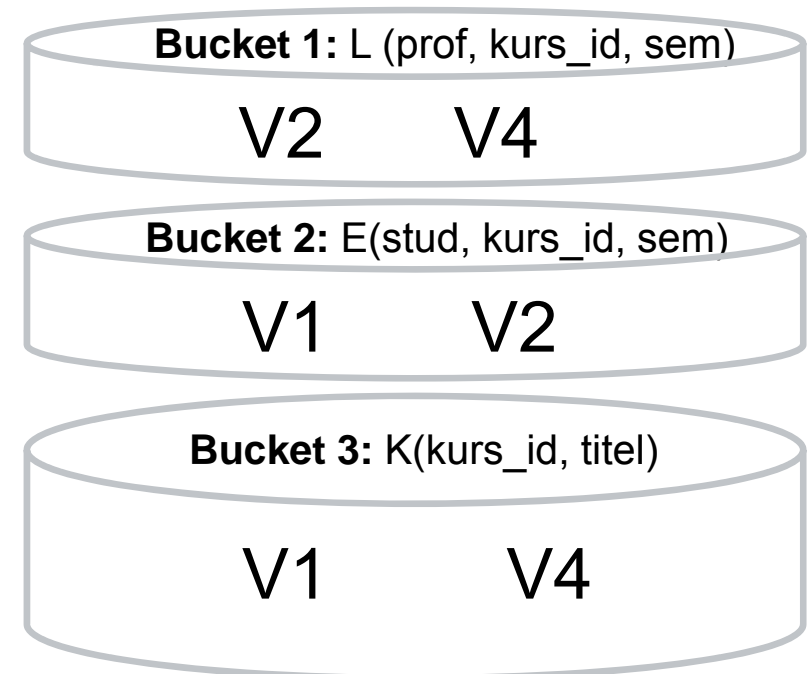
Falls mindestens ein Teilziel geeignet ist, füge Sicht in bucket ein.

Prüfung auf „Eignung“:

1. Unifier: Alle Anfrageattribute müssen vorkommen.
2. Kompatibilität: Prädikate sind passend.

Q (stud, kurs\_id, prof) :-  
 L(prof, kurs\_id, sem),  
 E(stud,kurs\_id,sem),  
**K(kurs\_id, titel),**  
 sem ≥ WS95,  
 kurs\_id ≥ 300

V4 (prof, kurs\_id, titel, sem) :-  
 L(prof, kurs\_id, sem),  
 K(kurs\_id, titel),  
 E(stud, kurs\_id, sem),  
 sem ≤ WS97



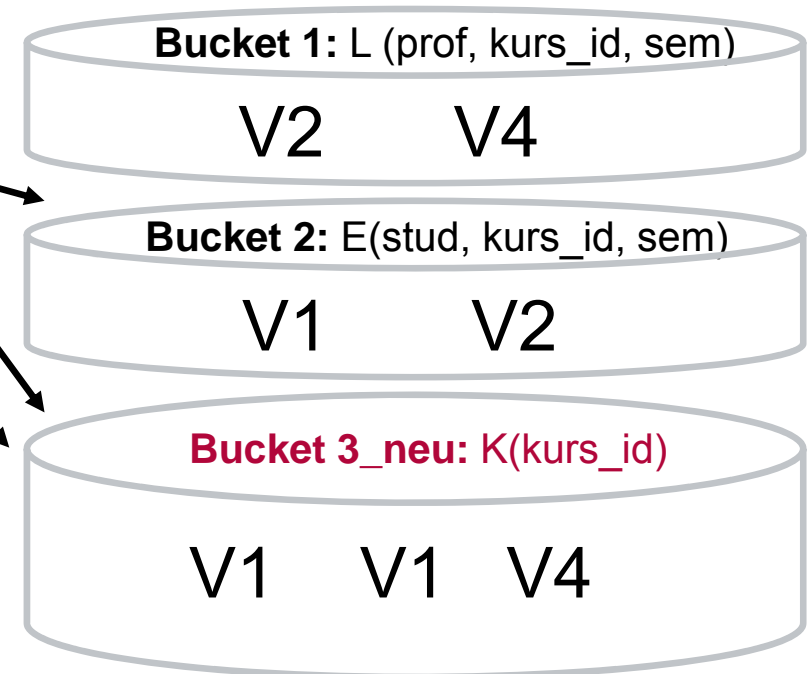
# BA – Besonderheiten

37

- Eine Sicht kann in verschiedenen Buckets auftauchen.
  - Verschiedene Rollen
- Eine Sicht kann mehrmals in einem Bucket auftauchen.
  - Wenn mehrere Teilziele passen

```
Q (stud, kurs_id, prof) :-
 L(prof, kurs_id, sem),
 E(stud, kurs_id, sem),
 K(kurs_id),
 sem ≥ WS95,
 kurs_id ≥ 300
```

```
V1 (stud, titel, sem, kurs_id) :-
 E(stud, kurs_id, sem),
 K(kurs_id, titel),
 kurs_id ≥ 500,
 sem ≥ WS98
```



# LaV – BA – Beispiel

38

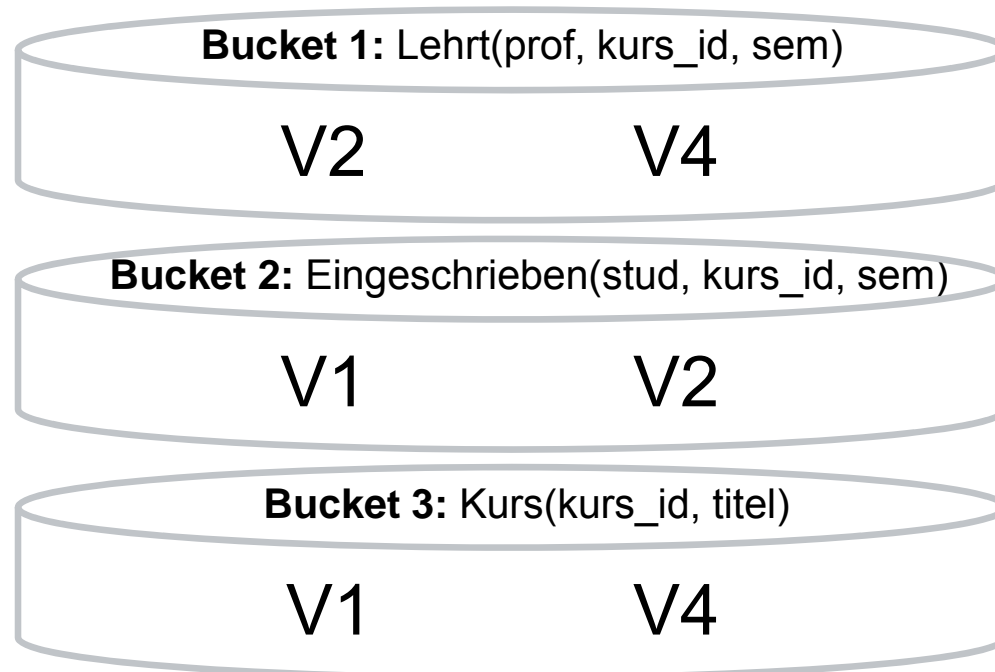
V1 (stud, titel, sem, kurs\_id) :- E(stud,kurs\_id,sem), K(kurs\_id,titel), kurs\_id ≥ 500, sem ≥ WS98

V2 (stud, prof, sem, kurs\_id) :- E(stud, kurs\_id, sem), L(prof, kurs\_id, sem)

V3 (stud, kurs\_id) :- E(stud, kurs\_id, sem), sem ≤ WS94

V4 (prof, kurs\_id, titel, sem) :- L(prof, kurs\_id, sem), K(kurs\_id, titel), E(stud, kurs\_id, sem), sem ≤ WS97

**Anfrage:** Q (stud, kurs\_id, prof) :- L(prof, kurs\_id, sem), E(stud,kurs\_id,sem), K(kurs\_id, titel), sem ≥ WS95, kurs\_id ≥ 300



## Kombinationen

V2, V1, V1

V4, V1, V4

V2, V2, V4

V2, V2, V1

V2, V1, V4

V4, V1, V1

V4, V2, V4

V4, V2, V1

Frage: Welche Kombinationen (Pläne) sind contained? Welche sind nützlich?

# LaV – BA – Kombinationen

39

- Wieviele Kombinationen?
  - $|B_1| \times |B_2| \times \dots \times |B_n|$  ( $n = |Q|$ )
  - Falls  $m =$  Anzahl Sichten:  $m^n$
  - Wichtig: Jede der exponentiell vielen Kombinationen liefert potentiell einen Teil des Ergebnisses.
- Eine Kombination  $Q' = V_1, \dots, V_k$  ist eine Anfrageumschreibung von  $Q$ , falls
  - $Q' \subseteq Q$
  - (oder  $Q', \{\text{zusätzliche Prädikate}\} \subseteq Q$ )

# LaV – BA – Kombinationen

40

V1 (stud, titel, sem, kurs\_id) :- E(stud,kurs\_id,sem), K(kurs\_id,titel), kurs\_id ≥ 500, sem ≥ WS98

V2 (stud, prof, sem, kurs\_id) :- E(stud, kurs\_id, sem), L(prof, kurs\_id, sem)

V3 (stud, kurs\_id) :- E(stud, kurs\_id, sem), sem ≤ WS94

V4 (prof, kurs\_id, titel, sem) :- L(prof, kurs\_id, sem), K(kurs\_id, titel), E(stud, kurs\_id, sem), sem ≤ WS97

**Anfrage:** Q (stud, kurs\_id, prof) :- L(prof, kurs\_id, sem), E(stud,kurs\_id,sem), K(kurs\_id, titel),  
sem ≥ WS95, kurs\_id ≥ 300

## Kombinationen

V2, V1, V1

V4, V1, V4

V2, V2, V4

V2, V2, V1

V2, V1, V4

V4, V1, V1

V4, V2, V4

V4, V2, V1

Einzelne Sichten nützlich (in Bezug auf Anfrage), aber zusammen Widerspruch:  
sem ≥ WS98 vs. sem ≤ WS97



# LaV – BA – Kombinationen

41

V1 (stud, titel, sem, kurs\_id) :- E(stud,kurs\_id,sem), K(kurs\_id,titel), kurs\_id ≥ 500, sem ≥ WS98

V2 (stud, prof, sem, kurs\_id) :- E(stud, kurs\_id, sem), L(prof, kurs\_id, sem)

V3 (stud, kurs\_id) :- E(stud, kurs\_id, sem), sem ≤ WS94

V4 (prof, kurs\_id, titel, sem) :- L(prof, kurs\_id, sem), K(kurs\_id, titel), E(stud, kurs\_id, sem), sem ≤ WS97

**Anfrage:** Q (stud, kurs\_id, prof) :- L(prof, kurs\_id, sem), E(stud,kurs\_id,sem), K(kurs\_id, titel),  
sem ≥ WS95, kurs\_id ≥ 300

## Kombinationen

V2, V1, V1

~~V4, V1, V4~~

V2, V2, V4

V2, V2, V1

~~V2, V1, V4~~

~~V4, V1, V1~~

V4, V2, V4

~~V4, V2, V1~~

Q'(stud, kurs\_id, prof):-

V2(stud', prof, sem, kurs\_id),

V1 (stud, titel', sem, kurs\_id),

V1 (stud', titel, sem', kurs\_id)

x' markiert nicht gebrauchte Attribute.

# LaV – BA – Kombinationen

42

- Prüfe  $Q \supseteq Q'$ (stud, kurs\_id, prof):-
  - V2(stud`, prof, sem, kurs\_id),
  - V1 (stud, titel`, sem, kurs\_id),
  - ~~V1 (stud`, titel, sem`, kurs\_id)~~
- Wie?
  - Q verwendet globale Relationen, Q' verwendet Sichten
- Durch „unfolding“:
  - Q' = Q''(stud, kurs\_id, prof):-
    - E(stud`, kurs\_id, sem), L(prof, kurs\_id, sem),
    - E(stud, kurs\_id, sem), K(kurs\_id, titel`), kurs\_id ≥ 500, sem ≥ WS98
- und Finden eines containment mappings.

# LaV – BA – Kombinationen

43

Endgültiges Ergebnis: Kombination aller Kombinationen durch UNION:

$V1, V2 \cup V2, V4$

**Umgeschriebene Anfrage Q':**

```
SELECT V1.stud, V1.kurs_id, V2.prof
FROM V1, V2
WHERE V1.sem = V2.sem
AND V1.kurs_id = V2.kurs_id
```

UNION

```
SELECT V2.stud, V2.kurs_id, V2.prof
FROM V2, V4
WHERE V2.sem = V4.sem
AND V2.kurs_id = V4.kurs_id
AND V2.prof = V4.prof
```

# LaV – BA – Analyse

44

- Vollständigkeit: Gegeben Anfrage  $Q$  und Sichten  $V_1, \dots, V_m$ , findet BA eine Anfrageumschreibung  $Q'$  aus den Sichten, falls eine Umschreibung existiert?
- Beweis:
  - Nach [LMSS95] gibt es nur eine Umschreibung, falls es eine Umschreibung der Länge  $|Q|$  gibt.
    - Einschränkung: Die Anfrage darf keine  $\geq, \leq, <, >, \neq$  Prädikate enthalten.
  - D.h. es reicht, alle Kombinationen zu prüfen.
  - BA verwirft nur Kombinationen, die unzulässig sind.
- Maximally contained
  - BA produziert nach [LMSS95] maximal contained Umschreibungen.

# Zusammenfassung

45

- Query Rewriting
- Query Containment
- Naiver Algorithmus
- Bucket Algorithmus
  - Beispiel
  - Analyse

**Globale Anfrage**  
 SELECT prof  
 FROM Lehrt L, Kurs K  
 WHERE L.kurs\_id = K.kurs\_id  
 AND K.titel = „VL\_Datenbanken“  
 AND L.univ = „Humboldt“

→

**Umgeschriebene Anfrage**  
 SELECT prof  
 FROM DB-kurs D  
 WHERE D.univ = „Humboldt“

} Vollständige Antwort

**Globale Anfrage**  
 SELECT titel, kurs\_id  
 FROM Kurs K  
 WHERE L.univ = „Humboldt“

→

**Umgeschriebene Anfrage**  
 SELECT titel, kurs\_id  
 FROM DB-kurs D  
 WHERE D.univ = „Humboldt“  
 UNION  
 SELECT titel, kurs\_id  
 FROM Hum-VL

} Maximale Antwort



- Gute Zusammenfassung für LaV und weiterführende Literatur:
  - [Hal01] Alon Y. Halevy: Answering queries using views: A survey, in VLDB Journal 10: 270-294, 2001.
- Erweiterung des BA für Information Quality
  - [NLF99] Felix Naumann, Ulf Leser, and Johann-Christoph Freytag: Quality-driven Integration of Heterogenous Information Systems, VLDB 1999
  - [LN00] Ulf Leser and Felix Naumann: Query Planning with Information Quality Bounds, FQAS 2000.
- Weitere Literatur
  - [LRO96b] Alon Y. Levy , Anand Rajaraman , Joann J. Ordille , [Querying Heterogeneous Information Sources Using Source Descriptions](#) Proceedings of the 22nd VLDB Conference, Bombay, India. 1996.
  - [LRO96a] Alon Y. Levy , Anand Rajaraman , Joann J. Ordille , [Query answering algorithms for information agents](#) Proceedings of the AAAI Thirteenth National Conference on Artificial Intelligence 1996 Click [here](#) for abstract.
  - [LMSS95] Alon Y. Levy, [Alberto O. Mendelzon](#), [Yehoshua Sagiv](#), [Divesh Srivastava](#): Answering Queries Using Views. [PODS 1995](#): 95-104