
# Informationsintegration

## Anfrageplanung mit beschränkten Quellen

Ulf Leser

# Inhalt dieser Vorlesung

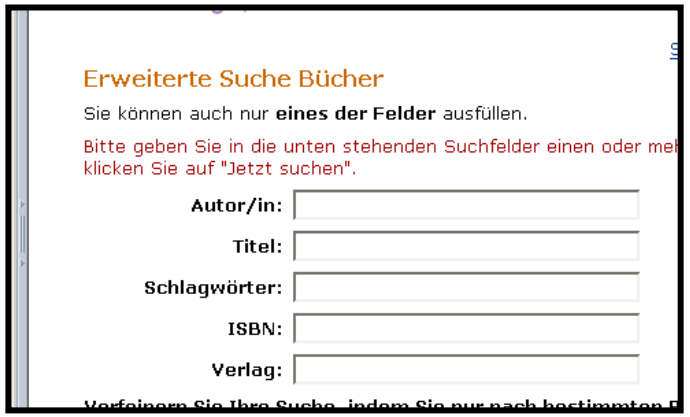
- Gebundene und freie Variablen
- Anfrageplanung
- Verfeinerungen

# Beschränkte Quellen

- Quellen, die nicht alle möglichen (relationalen) Anfragen beantworten können oder nicht wollen
  - Webquellen hinter Formularen
  - Quellen mit festen APIs: Web Services, CORBA, RPC, ...
- In der VL schon behandelt: Bestimmte Prädikate nicht ausführbar
  - Joins, Selektionen, Projektionen
- Annahme bisher stets: Quelle kann Obermenge liefern
  - Erlaubt Kompensation im Mediator
- Jetzt: Quellen, die bestimmte Prädikate verlangen

# Bindungen

- Beispiel
  - **`SELECT * FROM BOOK`**
    geht nicht
  - **`SELECT * FROM BOOK`**
    **`WHERE AUTHOR like …`** geht
  - **`SELECT * FROM BOOK`**
    **`WHERE TITLE like …`** geht




- Ein Attribut heißt ...
  - frei (f), wenn es in einer Anfrage nicht belegt sein muss (aber kann)
  - gebunden (b), wenn es in einer Anfrage belegt sein muss
  - unbindbar (n), wenn es in einer Anfrage nicht belegt sein darf
- „Belegt“ heißt: Selektion der Art „V=c“ oder „V in $[c_1,...c_n]$“

# Auswirkungen

- Viele Anfragepläne sind nicht ausführbar
- Pläne sind nicht mehr in jeder Reihenfolge ausführbar
  - Beispiel: **`order(OI,Title),info(OI,Schauspieler,aGe,rOlle)`**
    - **`order`** kann nach **`OI`** oder **`Titel`** oder nichts durchsucht werden
    - **`info`** kann nach **`OI`** oder **`Schauspieler`** durchsucht werden
  - Anfrage: **`q(OI):-order(OI,T),info(OI,S,G,O),G=90`**
  - Gute Reihenfolge: **`info(OI,S,90,O)`**,**`order(OI,T)`**
    - Eigentlich gut wegen hoher Selektivität der Bedingung
    - Aber nicht ausführbar: OI/S fehlt, G kann nicht gepushed werden
  - Mögliche Reihenfolge: **`order(OI,T),info(OI,S,G,O),G=90`**
    - Mit einem teuren **`SELECT * FROM order`**
    - Dann Bindungen von **`OI`** zu **`info`** pushen
    - Bedingung **`G=90`** erst im Mediator auswerten

# Binding Pattern

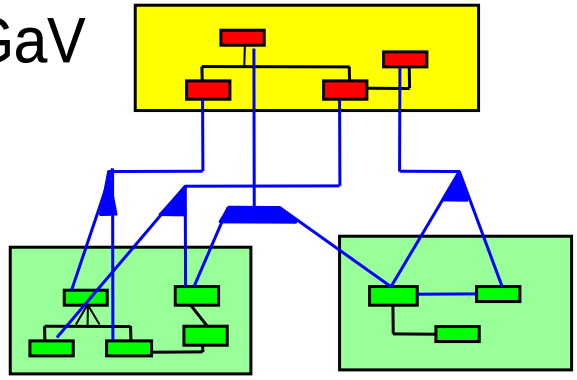
frei (f)
gebunden (b)
unbindbar (n)

- Einschränkungen auf den Bindungen von Attributen einer Anfrage nennt man Binding Pattern (BP)
  - Beispielschema: **`order(OI,T),info(OI,S,G,O)`**
    - **`order`** kann nach **`OI`** oder **`Titel`** oder nichts durchsucht werden
    - **`info`** kann nach **`OI`** oder **`Schauspieler`** durchsucht werden
  - Ergibt Binding Pattern: **$\texttt{order(OI}^f\texttt{,T}^f\texttt{),info(OI}^f\texttt{,S}^f\texttt{,G}^n\texttt{,O}^n\texttt{)}$**
- Ist das richtig? Hängt vom „oder“ ab
  - Korrekt, wenn „eines oder beides“ gemeint ist
  - Inkorrekt, wenn „entweder oder“ gemeint ist

# Spezifikation von Binding Pattern

- Binding Pattern werden mit Korrespondenzen angegeben
- Es kann notwendig werden, eine Korrespondenz mehrmals (mit unterschiedlichen BP) anzugeben
  - Beispiel: **`info`** kann entweder nach **`OI`** oder **`Schauspieler`** durchsucht werden, aber einer der Werte muss gegeben sein:

GaV



Korrespondenzen

**`global_info(OI`$^{b}$**`,S`$^{n}$`,G`$^{n}$`,O`$^{n}$`) :- info1(…), …`

**`global_info(OI`$^{n}$**`,S`$^{b}$`,G`$^{n}$`,O`$^{n}$`) :- info2(…), …`

# Anfrageplanung mit BP [RSU95]

- BP zunächst während der Planung ignorieren
- Sei p ein korrekter Anfrageplan für eine Anfrage q
- Für jede Ausführungsreihenfolge von p testen, ob sie ausführbar ist
  - Kostenbasierte Optimierung wählt unter allen ausführbaren Reihenfolgen denn die beste
- Nehmen wir einen Mediator an, der immer pushen will

# Algorithmus

- Gegeben: Anfrageplan p für Anfrage q
  - p besteht aus mit BP annotierte Korrespondenzen
- Initialisierung
  - Vektor $k_0$ mit einer Position für jedes Attribut in p (k[i] $\in$ {b,f,n})
  - Initialisiere $k_0$ mit „f“ an jeder Position
  - Wenn q eine Variable v bindet (Selektion), setze $k_0$[v]=„b“
- Beispiel: `q(OI,S):-order(OI,T),info(OI,S,G,O),T=,Marnie';`
  - Fünf Variable: `OI,T,S,G,O`
  - Nur `T` wird gebunden, alle anderen sind frei
  - Ergibt $k_0$ := [f,b,f,f,f]
    - Wir schreiben kürzer: fbfff

# Iteration

- Sei ein korrekter Plan für q: $<q_1,...q_n>$
  - $q_i$ sind rechte Seiten von Anfragekorrespondenzen
  - Jedes $q_i$ hat ein Binding Pattern $b_i$
  - Wir setzen $b_i[j]$=„_“, wenn Variable j in $q_i$ nicht auftaucht
- Wir iterieren über die $q_i$ und berechnen jeweils $k_{i+1}$
  - Wir sind in Schritt i und haben $k_{i-1}$ und $b_i$
  - Iteriere über alle Variablen j
    - Wenn $k_{i-1}[j]$=„b“ und $b_i[j]$=„n“, dann ist der Plan nicht ausführbar (*)
      - Weil die Bindung nicht weitergegeben werden kann
    - Wenn $k_{i-1}[j]$=„f“ und $b_i[j]$=„b“, dann ist der Plan nicht ausführbar
      - Weil eine notwendige Bindung nicht vorhanden ist
    - Wenn die Variable j in $q_i$ exportiert wird, dann $k_i[j]$=„b“
      - Weil die Bindungen weitergeführt werden müssen
    - Sonst $k_i[j]=k_{i-1}[j]$

# Beispiel

- Folgende Quellen

```
movie.order(OI^n, T^b);
```

```
movie.info1(OI^n, S^b, G^n, O^n);
movie.info2(OI^b, S^n, G^n, O^n);
```

  - Quellen exportieren alle Variable

- Benutzeranfrage
  - **q(OI,S):- order(OI,T), info(OI,S,G,O), T=‚Marnie‘;**

- Ergibt zwei semantisch korrekte Pläne

```
movie.order(OI^n, T^b), movie.info1(OI^n, S^b, G^n, O^n);
movie.order(OI^n, T^b), movie.info2(OI^b, S^n, G^n, O^n);
```

# Ausführbarkeit

movie.order(OI$^n$, T$^b$), movie.info1(OI$^n$, S$^b$, G$^n$, O$^n$);
movie.order(OI$^n$, T$^b$), movie.info2(OI$^b$, S$^n$, G$^n$, O$^n$);

- $k_0$=fbfff (für Variablen OI, T, S, G, N)
- Vier mögliche Reihenfolgen
  - **order, info1**
    - **fbfff** ◊ **nb___** → **bbfff**
    - **bbfff** ◊ **n_bnn** → Problem wegen b,n und wegen f,b
  - **info1, order**
    - **fbfff** ◊ **n_bnn** → Problem wegen f,b
  - **order, info2**
    - **fbfff** ◊ **nb___** → **bbfff**
    - **bbfff** ◊ **b_nnn** → **bbbbb** (Einzig mögliche Reihenfolge)
  - **info2, order**
    - **fbfff** ◊ **b_nnn** → Problem wegen f,b

# (*) Post-Processing

movie.order(OI$^n$, T$^b$), movie.info1(OI$^n$, S$^b$, G$^n$, O$^n$);
movie.order(OI$^n$, T$^b$), movie.info2(OI$^b$, S$^n$, G$^n$, O$^n$);

- Betrachten wir noch mal
  - **order, info1**: **fbfff◊nb___→bbfff, bbfff◊n_bnn:** Problem b,n
  - Ist das wirklich ein Problem?
- Bei „n“ kann man auch wie folgt vorgehen
  - Führe eine Teilanfrage ohne Bindung aus
  - Falls im Plan vorher schon eine Bindung entstanden ist, teste diese während eines Post-Processings
- Auswirkung: „n“ kann wie „f“ behandelt werden
- Es entstehen Pläne ohne Pushen von Selektionen
- Hier: Plan trotzdem nicht ausführbar
  - Wir haben keine Bindung für S in info1
  - Aber ein **info3(N_FFF)** wäre dann möglich

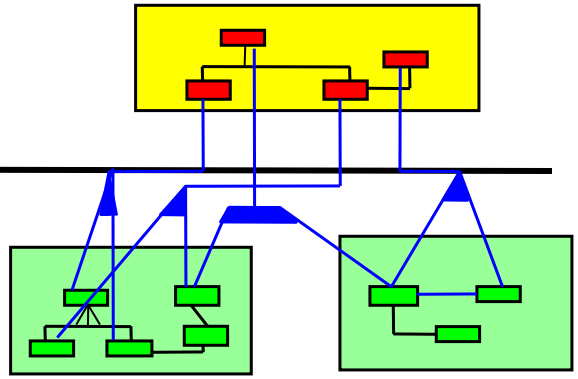
# Inhalt dieser Vorlesung

- Gebundene und freie Variablen
- Anfrageplanung
- Verfeinerungen
  - Mehr Adornments zur besseren Modellierung von Interfaces
  - Hierarchische Integration: Berechnung von Binding Pattern im Mediator
  - Binding Pattern und die Länge von Plänen

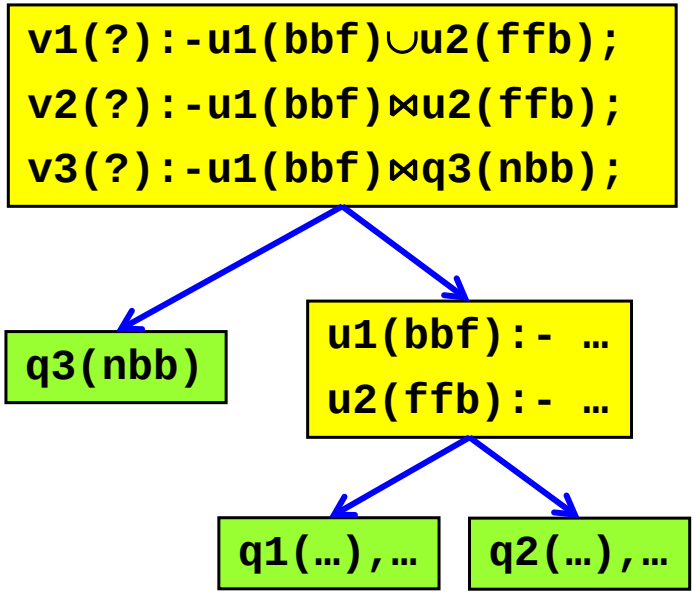
# Mehr Binding Pattern

- Anhänge (*adornments* = Verzierungen)
  - f (free): Kann in Anfrage spezifiziert werden, muss aber nicht
  - n (unspecifiable): Kann nicht spezifiziert werden
  - b (bound): Muss spezifiziert werden
  - c[s] (constant): Auswahl aus einer Menge s von Konstanten
    - Implizit bound: muss spezifiziert werden
    - Beispiel: Drop-Down Liste in Web-Formular ohne „keine Wahl“
  - o[s] (optional): Auswahl aus einer Menge s von Konstanten
    - Implizit free: Muss nicht spezifiziert werden
    - Beispiel: Drop-Down Liste in Web-Formular mit „keine Wahl“
- Behandlung
  - Im Prinzip: „c“ wie „b“ und „o“ wie „f“
  - Ggf. weitere Einschränkungen durch Wertemengen bei „c“ / „s“

# Geschachtelte Mediatoren [YLGU99]



- Annahme: GaV
  - Globale Relationen sind Sichten auf Quellen oder Mediatoren mit BP
  - Verknüpfungen: Union und Join
- Kann man die Binding Pattern der globalen Relationen (Sichten) aus den BP ihrer Komponenten (Source Queries) berechnen?
  - Wenn ein Mediator beschränkte Quellen integriert, welchen Beschränkungen unterliegt er dann selber?



# Beispiel

- Beispiel: 3 Sichten und insgesamt 5 Binding Pattern
  - **$u_1$(X,Y,Z): bff, ffb**
  - **$u_2$(X,Y,Z): fbf**
  - **$u_3$(X,Y,Z): ffc[$s_1$], c[$s_2$]ff**
- Welche BP ergeben sich für Views über diesen Views?
  - **v(?):-$u_1 \cup u_2$** : **bff∪fbf = bbf** oder
    **ffb∪fbf = fbb**
  - **v(?) :-($u_1 \cup u_2$)∪$u_3$**: **bbf∪ffc[$s_1$] = bbc[$s_1$]** oder
    **fbb∪ffc[$s_1$] = fbc[$s_1$]** oder ...

# Verknüpfungen mit Union

| | **f** | **o**$[s_3]$ | **b** | **c**$[s_4]$ | **n** |
|---|---|---|---|---|---|
| **f** | f | o$[s_3]$ | b | c$[s_4]$ | n |
| **o**$[s_1]$ | o$[s_1]$ | o$[s_1 \cap s_3]$ | c$[s_1]$ | c$[s_1 \cap s_4]$ | n |
| **b** | b | c$[s_3]$ | b | c$[s_4]$ | - |
| **c**$[s_2]$ | c$[s_2]$ | c$[s_2 \cap s_3]$ | c$[s_2]$ | c$[s_2 \cap s_4]$ | - |
| **n** | n | n | - | - | n |

Nicht ausführbar

Quelle: [YLGU99]

# Verknüpfung mit Join

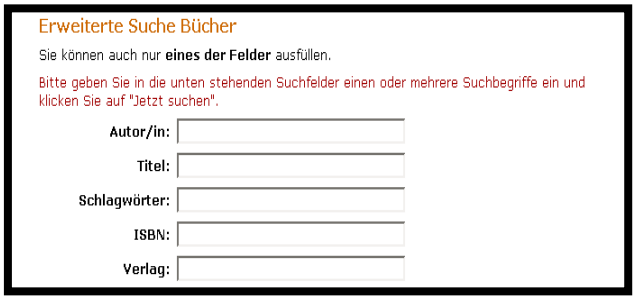
- Unterschied zu UNION
  - Nicht jedes Attribut der integrierten Sicht ist auch Attribut jeder beteiligten Quelle
  - Nur die Joinattribute haben zwei Adornments
- Berechnung des Binding Patterns der Sicht
  - Adornments der nicht-Join-Attribute werden kopiert
  - Adornments der Join-Attribute werden gemäß der UNION Tabelle vereint

# Passing Bindings

- Beispiel
  - **u1(X,Y,Z) : fbf**
  - **u4(Z,U) : bf**
  - **u=u1⋈u4 : fbbf**
- Anfrage 1: **<u,Y=foo,Z=bar>** ist beantwortbar
- Anfrage 2: **<u,Y=foo>** zunächst nicht beantwortbar
- Passing Bindings (siehe auch „BP ohne pushen“)
  - Ergebnisse einer Sicht werden vom Mediator in die gebundene Variable der nächsten Sicht eingetragen
  - Damit wird die Anfrage beantwortbar
  - Binding Pattern mit Passing Binding: u**=u1⋈u4 : fbff**
  - Funktioniert für Union (sehr eingeschränktes Ergebnis) und Join
  - Reihenfolge der Ausführung steht damit fest

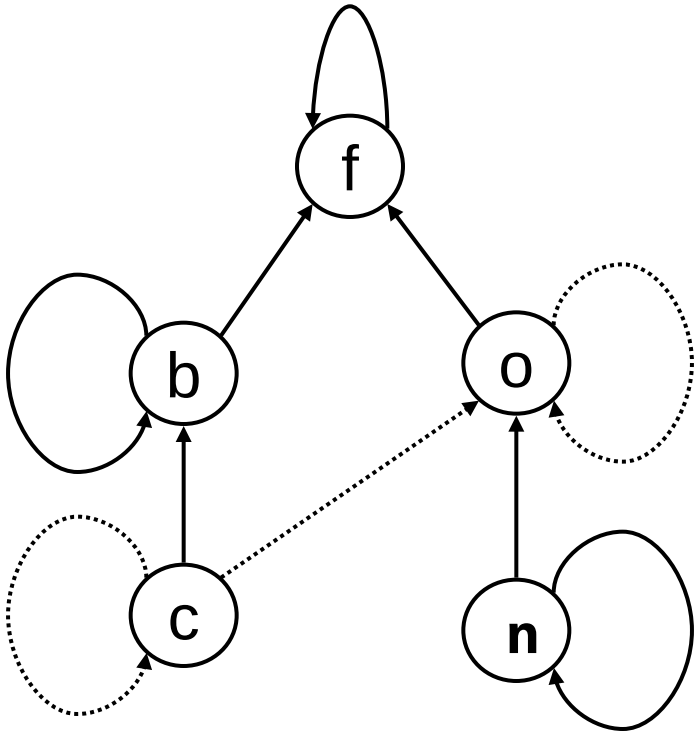
# Redundante Binding Pattern [YLGU99]

- Quellen benötigen oft mehrere BP für eine Relation
  - Beispiel: **`amazon(autor,titel,schlagwort,isbn,verlag)`**
    - bffff_, fbfff_, ffbff_, fffbf_, ffffb_
  - Beispiel: **`verlage(verlag,ort)`**
    - ____bf, ____fb
  - Sicht: **`v=amazon ⋈ verlage`**
  - Templates für v aus jeder Kombination
    - bfffff, fbffff, ffbfff, fffbff, ffffbf (über passing binding)
    - bffffb, fbfffb, ffbffb, fffbfb, ffffbb
    - fffffb (ffffb_⋈____fb mit passing binding von **`verlage`** nach **`ort`**)




- Anzahl berechneter Binding Pattern multipliziert sich
- Viele dieser Pattern sind redundant

# Subsumptionsbeziehungen




Weniger restriktiv

Weniger restriktiv falls Auswahllisten Teilmengen voneinander sind

# Fallbeispiel

- Amazon
  - Formular 1: Mindestens eine Bindung in author, title, subject, format (format aus Auswahlliste)
  - Formular 2: ISBN spezifizieren
  - Formular 3: Mindestens eine Bindung in keyword, publisher, date
  - Antwortrelation: author, title, ISBN, publisher, date, format, price, shipping info
- Barnes & Noble
  - Formular 1: Mindestens eine Bindung in author, title, keywords; optionale Bindung in format subject, price, age (alles aus Auswahllisten)
  - Formular 2: ISBN gebunden

# Mögliche BP des Mediators (n ist hier u)

Amazon

| author | title | format | subject | KW | ISBN | pub | date | price | ship |
|---|---|---|---|---|---|---|---|---|---|
| b | f | o | f' | u' | u | u | u | u | u |
| f | b | o | f' | u' | u | u | u | u | u |
| f | f | c | f' | u' | u | u | u | u | u |
| f | f | o | b' | u' | u | u | u | u | u |
| u | u | u | u' | u' | b | u | u | u | u |
| u | u | u | u' | f' | u | b | f | u | u |
| u | u | u | u' | b' | u | f | f | u | u |
| u | u | u | u' | f' | u | f | b | u | u |

Barnes & Noble

| author | title | format | subject | KW | ISBN | pub | date | price | ship | age |
|---|---|---|---|---|---|---|---|---|---|---|
| f | b | o | o' | f' | u | u | u | o | u | o' |
| b | f | o | o' | f' | u | u | u | o | u | o' |
| f | f | o | o' | b' | u | u | u | o | u | o' |
| u | u | u | u' | u' | b | u | u | u | u | u' |

Zusammen

| author | title | format | subject | KW | ISBN | pub | date | price | ship | age |
|---|---|---|---|---|---|---|---|---|---|---|
| f | f | f | u' | u' | b | f | f | f | f | u' |
| f | f | f | u' | b' | f | f | f | f | f | u' |
| b | f | f | o' | u' | f | f | f | f | f | u' |
| f | b | f | o' | u' | f | f | f | f | f | u' |
| b | f | f | u' | f' | f | b | f | f | f | u' |
| f | b | f | u' | f' | f | b | f | f | f | u' |
| f | b | f | u' | f' | f | f | b | f | f | u' |
| b | f | f | u' | f' | f | f | b | f | f | u' |

Quelle: [YLGU99]

# Redundanzfreie Binding Pattern

Zusammen

| author | title | format | subject | KW | ISBN | pub | date | price | ship | age |
|---|---|---|---|---|---|---|---|---|---|---|
| f | f | f | u' | u' | b | f | f | f | f | u' |
| f | f | f | u' | b' | f | f | f | f | f | u' |
| b | f | f | o' | u' | f | f | f | f | f | u' |
| f | b | f | o' | u' | f | f | f | f | f | u' |
| b | f | f | u' | f' | f | b | f | f | f | u' |
| f | b | f | u' | f' | f | b | f | f | f | u' |
| f | b | f | u' | f' | f | f | b | f | f | u' |
| b | f | f | u' | f' | f | f | b | f | f | u' |

Vier Formulare im Mediator:

- Spezifikation der ISBN (template 1)
- Spezifikation des keyword (template 2)
- Mindestens author oder title spezifizieren (templates 3 und 4)
- Mindestens author oder title und mindestens publisher oder date spezifizieren (templates 5-8)

# Inhalt dieser Vorlesung

- Gebundene und freie Variablen
- Anfrageplanung
- Verfeinerungen
  - Mehr Adornments zur besseren Modellierung von Interfaces
  - Hierarchische Integration: Berechnung von Binding Pattern im Mediator
  - Binding Pattern und die Länge von Plänen

# Planlänge [LC00]

- Binding Pattern machen weitere subtile Probleme bei der Anfrageplanung
- Beispiel

| Sources | View Schemas | Must Bind |
|---|---|---|
| 1 | v1(Song, CD) | Song |
| 2 | v2(CD, Artist, Price) | CD |
| 3 | v3(CD, Artist, Price) | Artist |

- Wir suchen die Preise von CDs mit dem Song „Friends“
- Zwei Anfragepläne
  - **v1⋈v2**: OK
  - **v1⋈v3**: Nicht OK, keine Bindung für Artist in **v3**

# Planung

- Nehmen wir folgenden Inhalt an

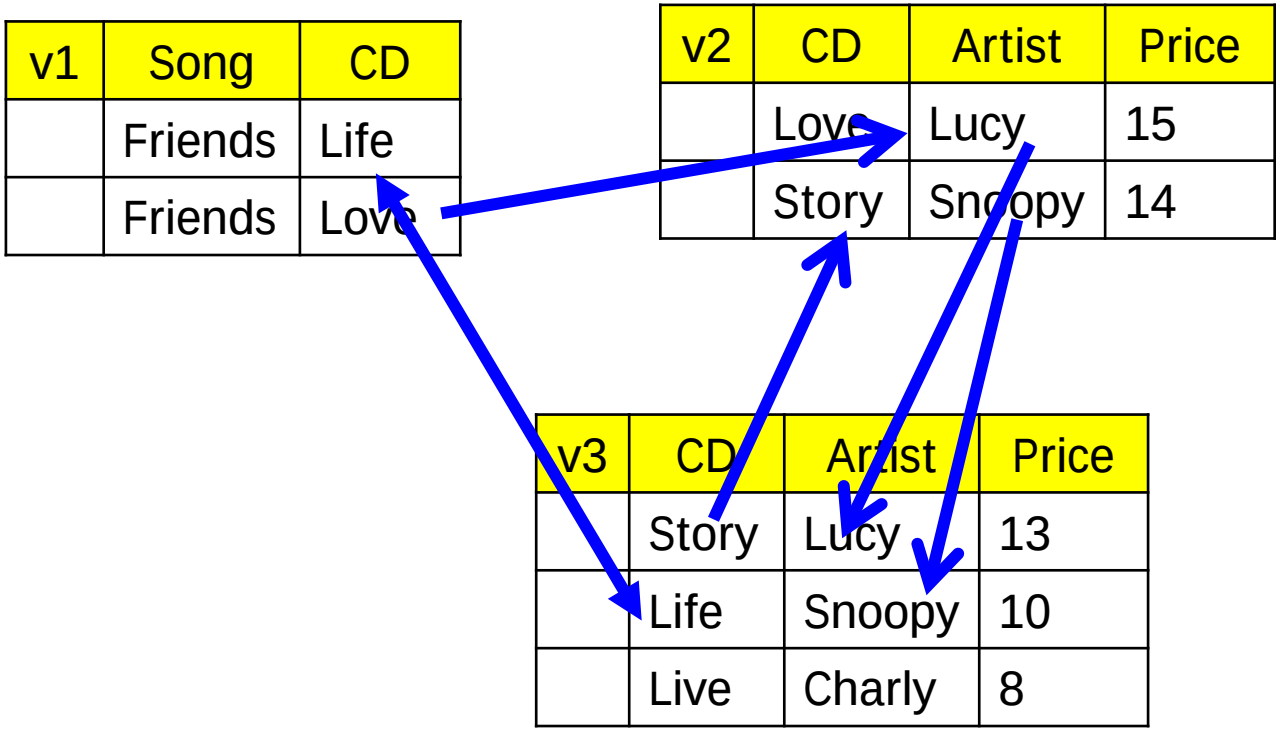
| v1 | Song | CD |
|---|---|---|
| | Friends | Life |
| | Friends | Love |

| v2 | CD | Artist | Price |
|---|---|---|---|
| | Love | Lucy | 15 |
| | Story | Snoopy | 14 |

| v3 | CD | Artist | Price |
|---|---|---|---|
| | Story | Lucy | 13 |
| | Life | Snoopy | 10 |
| | Live | Charly | 8 |

- Wir finden: **v1⋈v2: <friends, love, lucy, 15>**
- Aber: „Friends“ ist auch auf der CD „Life“ von „Snoopy“
- Der Plan **v1⋈v3** ist aber nicht ausführbar
- Können wir das Tupel trotzdem finden?

# Besser wäre




**(((v1⋈v2)⋈v3)⋈v2)⋈v3)⋈v1** produziert korrekte Tupel

# Unvollständigkeit

- Über längere Pläne kann man sich Bindungen für Attribute kreieren, die zu korrekten Tupeln führen, die durch kurze Pläne nicht generiert werden können
- Damit gilt die Längenbeschränkung für Anfragepläne nicht mehr
- Unsere Anfrageplanung ist unvollständig, wenn Korrespondenzen Binding Pattern haben
- Vollständige Anfrageplanung nur mit rekursiven Plänen möglich

# Literatur


- [RSU95] Anand Rajaraman, Yehoshua Sagiv und Jeffrey D. Ullman., „Answering Queries using Templates with Binding Patterns“. PODS 1995
- [YLGU99] Ramana Yerneni, Chen Li, Hector Garcia-Molina, Jeffrey D. Ullman, „Computing Capabilities of Mediators“, SIGMOD 1999
- [LC00] Chen Li, Edward Chang „Query Planning with Limited Source Capabilities“, ICDE 2000