Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

Informationsintegration

Schema Mapping

2.6.2008

Felix Naumann

Erste Hälfte

Einführung in die Informationsintegration
Szenarien der Informationsintegration
Verteilung und Autonomie
Heterogenität

Problemstellung

Materialisierte und virtuelle Integration
5-Schichten-Architektur
Mediator/Wrapper-Architektur / PDMS

Architekturen

Schema Mapping
Schema Matching
SchemaSQL

Mapping

Global-as-View und Lokal-as-View Modellierung
Global-as-View Anfragebearbeitung

Modellierung

Zweite Hälfte

Containment & Local-as-View Anfragebearbeitung
Bucket Algorithmus
Verteilte Anfragebearbeitung

Anfragen

Duplikaterkennung
Datenfusion – Union & Co.
DWH, ETL & Data Lineage
Informationsqualität

Datenintegration

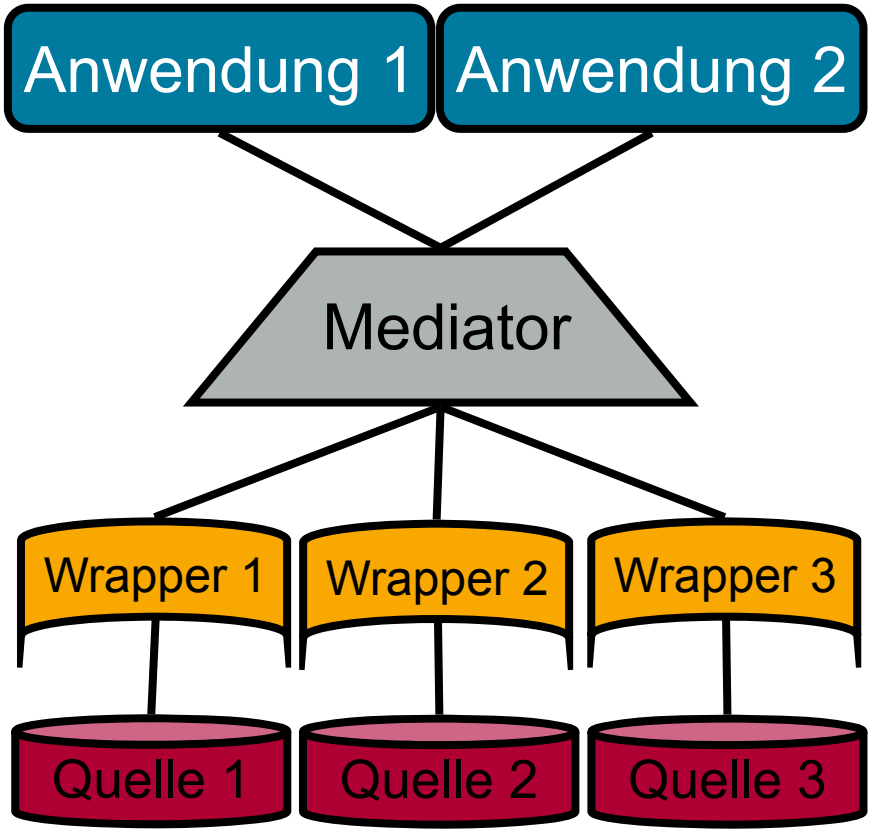
Hidden Web
Semantic Web

Anwendungen

Überblick

- Motivation
- Schema Mapping
- Schema Matching
- Mapping Interpretation
- Mapping Werkzeuge

Wdh: Virtuelle Integration



- Datenfluss
- Anfragebearbeitung
- Entwicklung
  - Top-down
- Schema

Wdh: Bottom-up oder Top-down Entwurf

- Beim Entwurf des integrierten Systems
- Bottom-up
  - Ausgelöst durch den Bedarf, mehrere Quellen integriert anzufragen
  - Schemaintegration ist wichtig.
  - Änderungen schwierig, da neu integriert werden muss.
  - Typisches Szenario: Data Warehouse
- Top-down
  - Ausgelöst durch globalen Informationsbedarf
  - Vorteilhaft bei labilen Quellen
  - Schemaintegration nicht nötig, bzw. leichter

Wdh: Bottom-up Entwicklung

Wesentliche Merkmale

- Vorhandene lokale Schemata
- ALLE Daten sollen integriert werden

Deshalb

- Schema-integratic

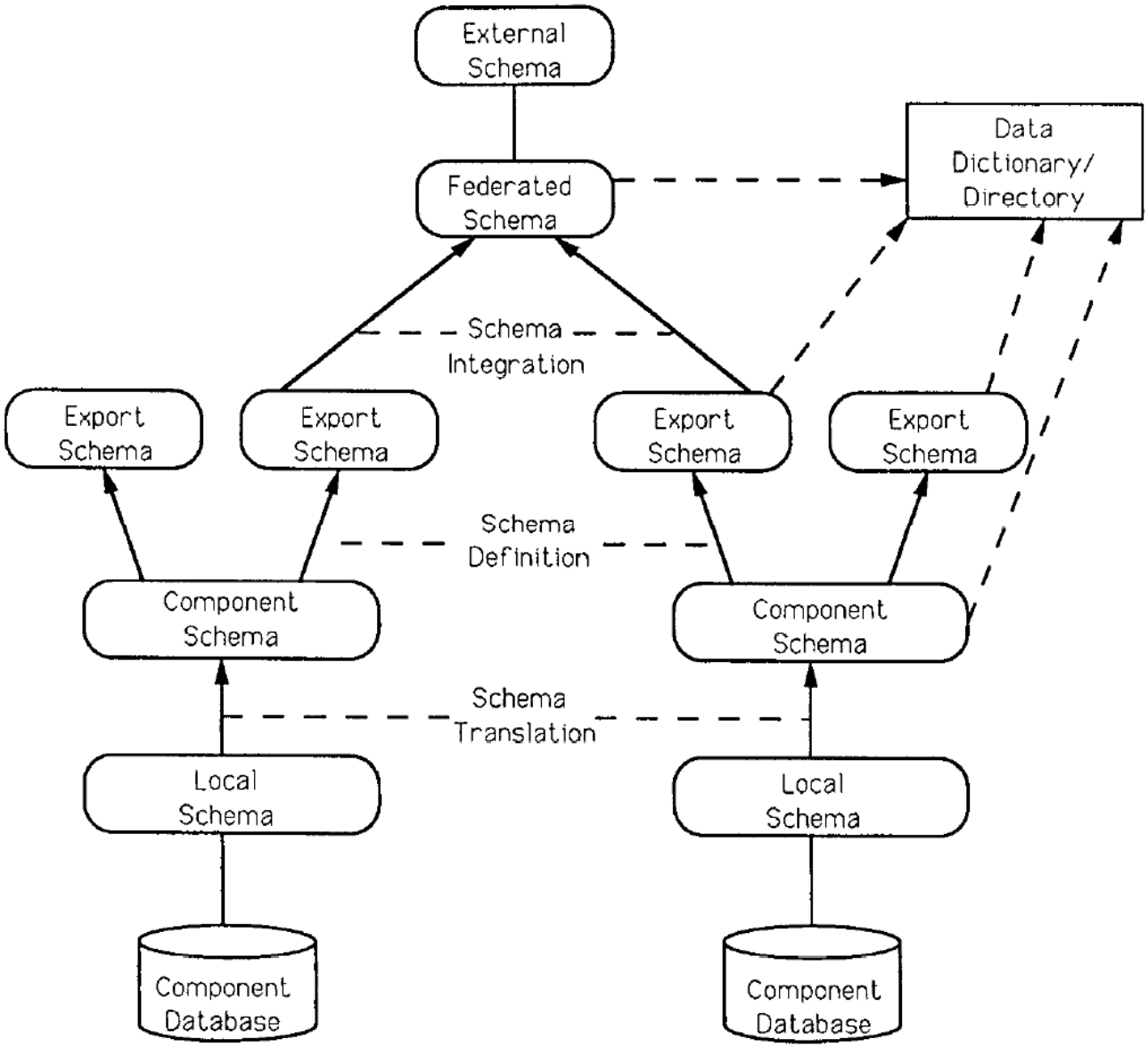


Figure 21. Bottom-up FDBS developing process.

Wdh: Top-down Entwicklung

Wesentliche Merkmale

- Vorhandenes globales
- Passende lokale Syste werden gesucht

Deshalb

- LaV
- und Schema Mapping

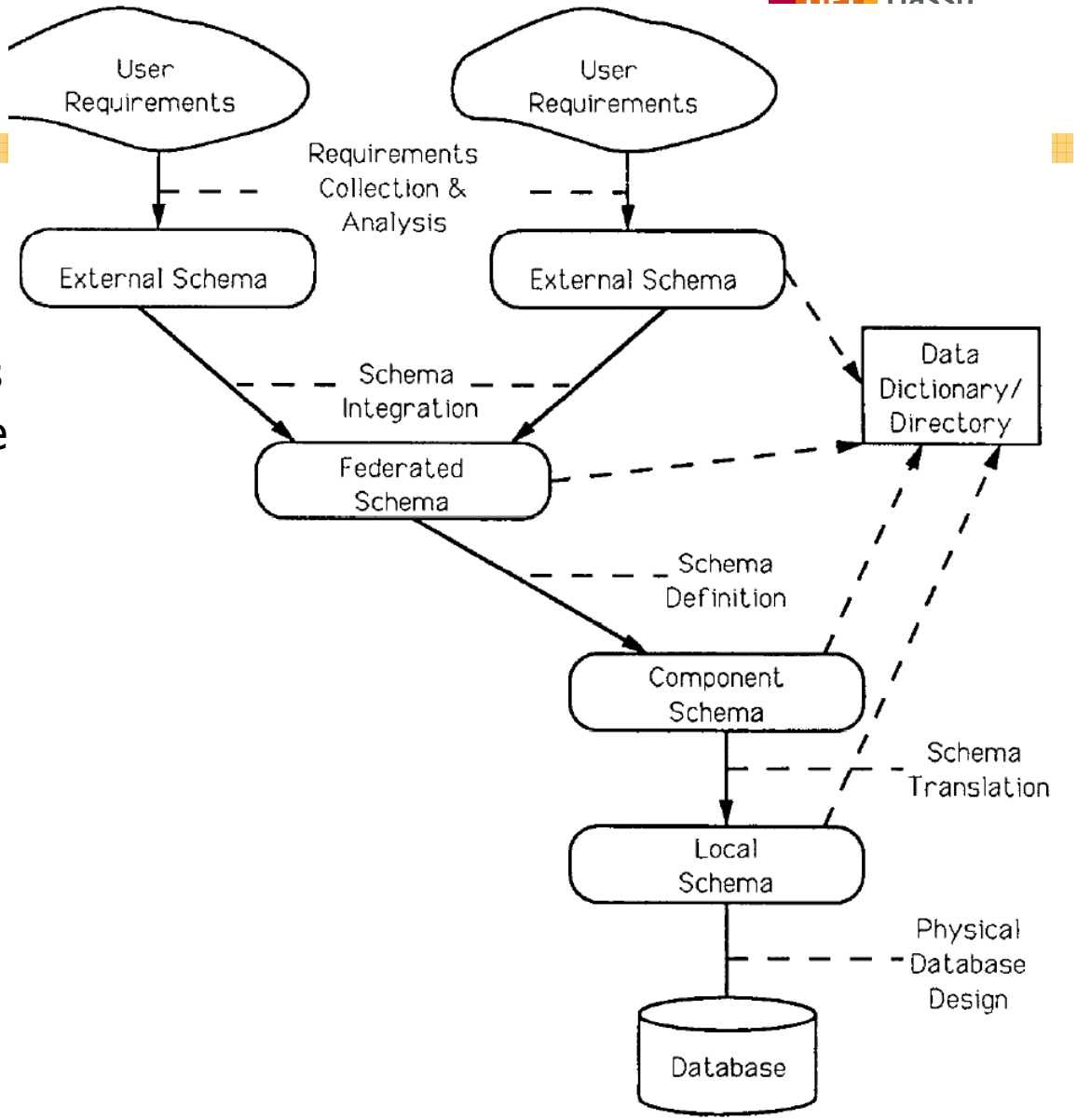


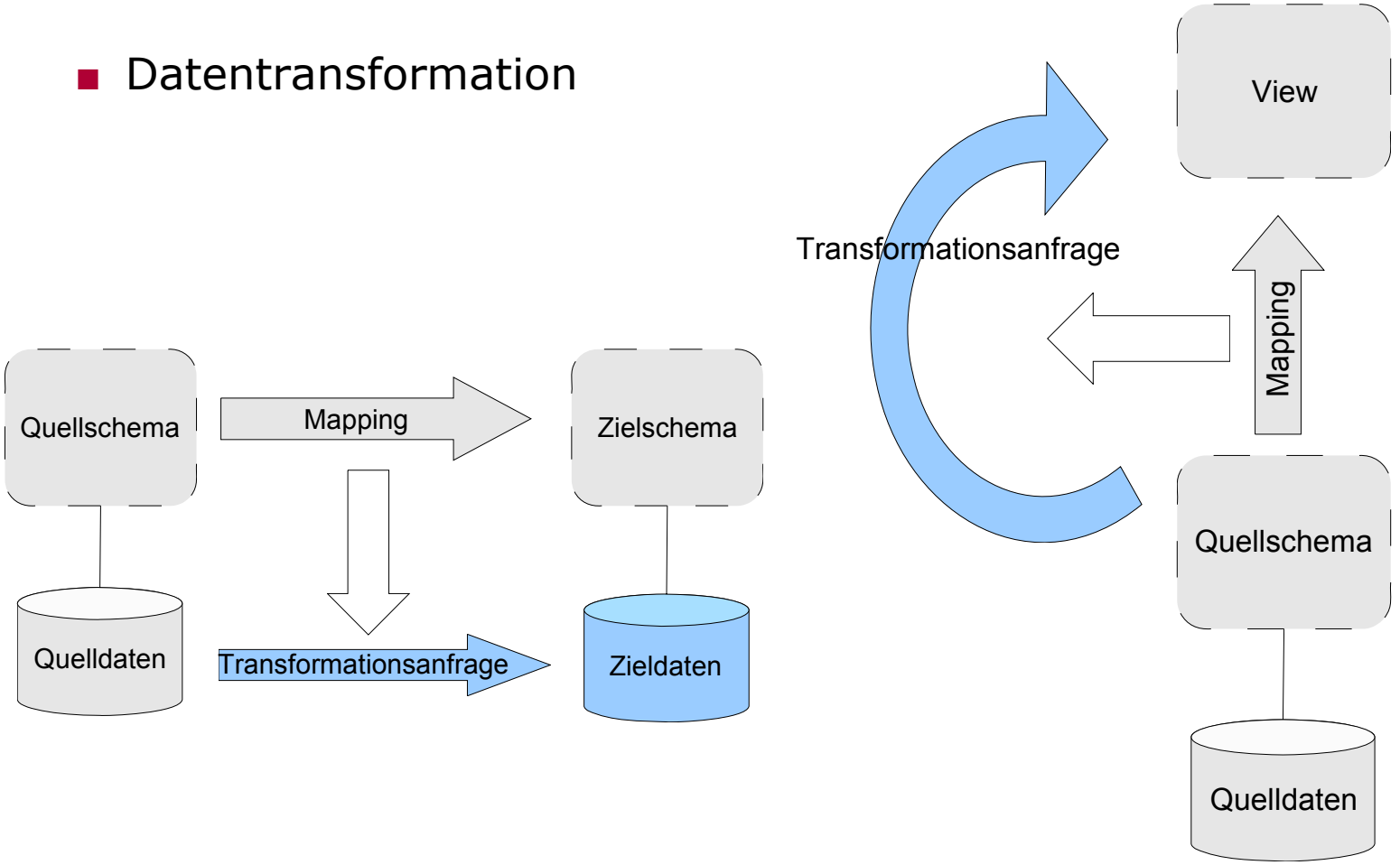
Figure 22. Top-down FDBS developing process.

Schemaintegration vs. Schema Mapping

- Beide Probleme müssen strukturelle und semantische Heterogenität überwinden.
- Aber:
  - Schemaintegration liefert Schema Mapping „frei Haus“.
  - Zielschema hat keine eigene Semantik.
  - Schemaintegration ist unflexibel.
- Deshalb nun: Schema Mapping

Verwendung von Schema Mappings

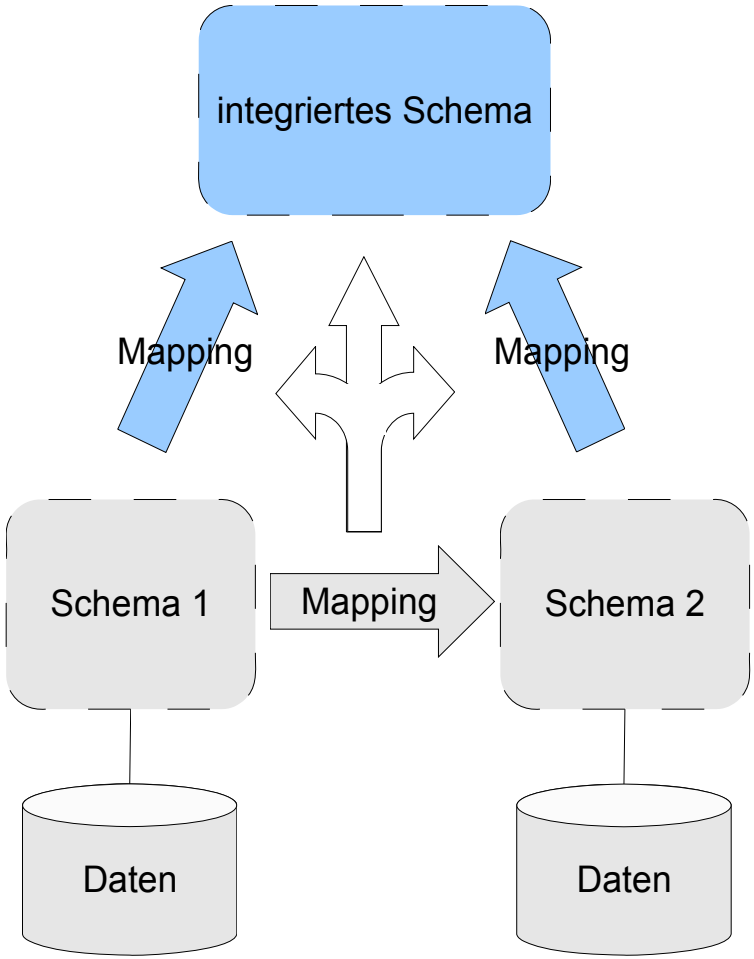
- Datentransformation



Folie: Frank Legler

Verwendung von Schema Mappings

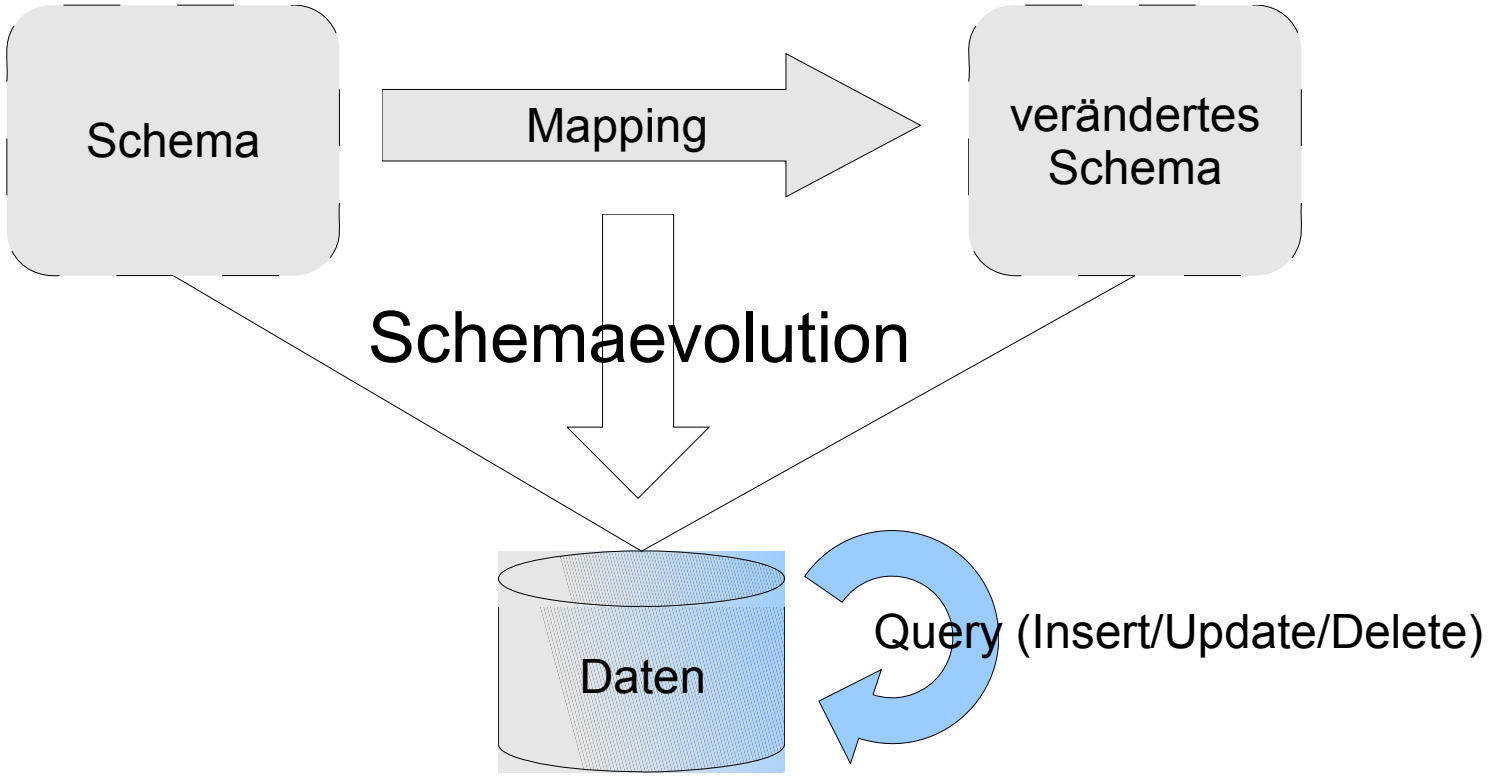
- Schemaintegration



Folie: Frank Legler

Verwendung von Schema Mappings

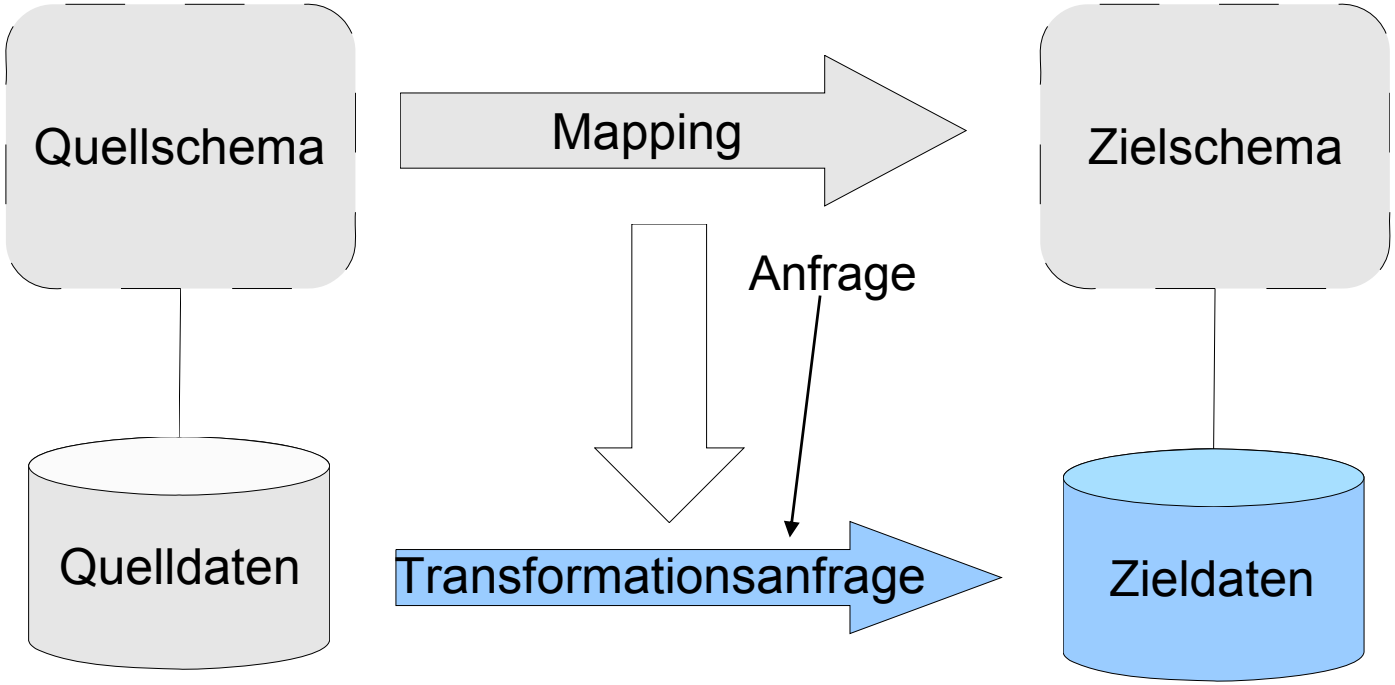
- Schemaevolution



Folie: Frank Legler

Verwendung von Schema Mappings

- Im weiteren: Datentransformation
  - Materialisierte Integration
  - Virtuelle Integration



Überblick

- Motivation
- Schema Mapping
- Schema Matching
- Mapping Interpretation
- Mapping Werkzeuge

Schema Mapping - Definitionen

- (Inter-Schema) **Korrespondenz**
  - Eine Zuordnung eines oder mehrerer Elemente eines (Quell-) Schemas zu einem oder mehreren Elementen eines anderen (Ziel-) Schemas
  - Auch: Value-correspondence
- (High-level) **Mapping**
  - Eine Menge von Korrespondenzen zwischen zwei Schemas.
- (Low-Level) **Logisches Mapping**
  - Logische Übersetzung eines oder mehrerer Mappings, die
    - den Integritätsbedingungen beider Schemas gehorcht und
    - die Intention des Nutzers wiederspiegelt.
- **Interpretation**
  - Übersetzung eines Mappings in ein oder mehrere logische Mappings
  - Übersetzung eines logischen Mappings in eine Transformationsanfrage
- **Transformationsanfrage**
  - Anfrage in einer Anfragesprache (z.B. SQL), die Daten des Quellschemas in die Struktur des Zielschemas überführt

Wdh: Schematische Heterogenität

- Struktur
  - Modellierung
    - Relation vs. Attribut
    - Attribut vs. Wert
    - Relation vs. Wert
  - Benennung
    - Relationen
    - Attribute
  - Normalisiert vs. Denormalisiert
  - Geschachtelt vs. Fremdschlüssel

High-order Mappings

hier

Motivation

- Datentransformation zwischen heterogenen Schemas
  - Altes aber immer wiederkehrendes Problem
  - Üblicherweise schreiben Experten komplexe Anfragen oder Programme
    - Zeitintensiv
    - Experte für die Domäne, für Schemata und für Anfrage
    - XML macht alles noch schwieriger
      - XML Schema, XQuery
- Idee: Automatisierung
  - Gegeben: Zwei Schemata und ein high-level Mapping dazwischen.
  - Gesucht: Anfrage zur Datentransformation

Motivation – Probleme

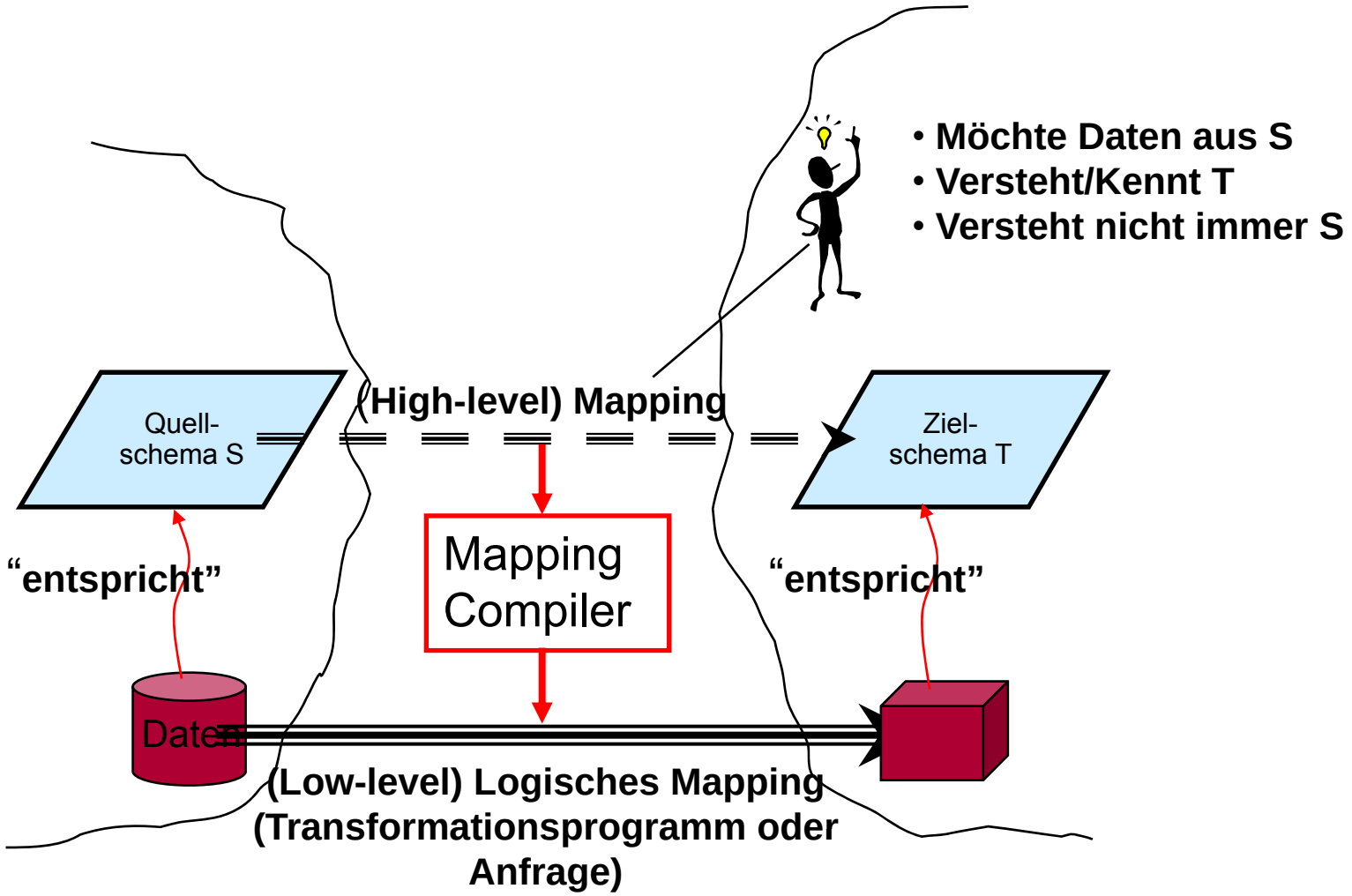
- Generierung der „richtigen“ Anfrage unter Berücksichtigung der Schemata und des Mappings
- Garantie, dass die transformierten Daten dem Zielschema entsprechen
- Effiziente Datentransformation
  - Für Materialisierung (Ausführung, inkrementell)
  - Für virtuelle Integration (query-unfolding)

Hier: Nur Effektivität, nicht Effizienz

Motivation – Weitere Probleme

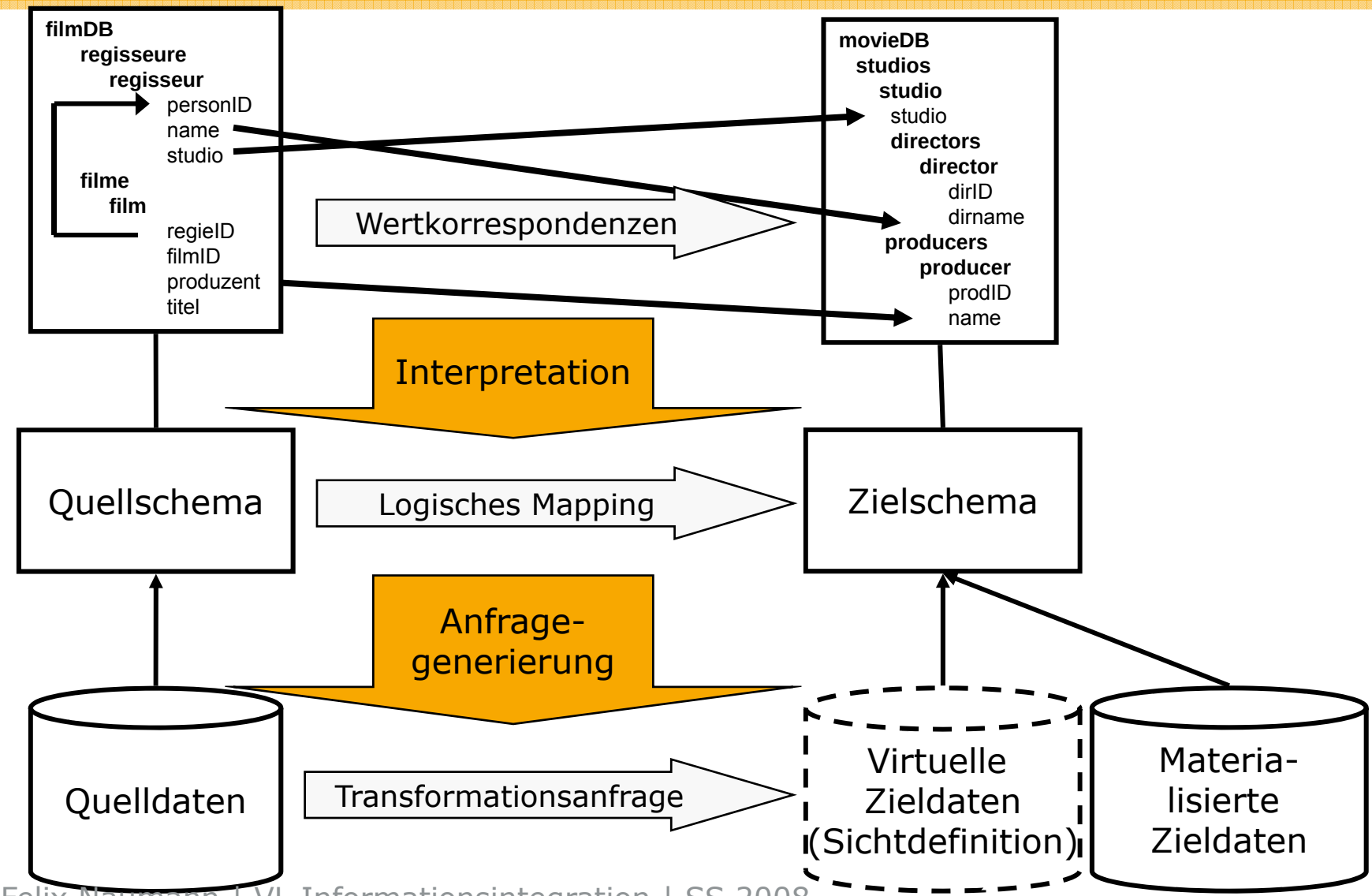
- Geschachtelte Strukturen unterstützen
  - Geschachteltes, relationales Modell
  - XML
  - Geschachtelte Integritätsbedingungen
- Korrespondenzen
  - Nutzerfreundlich
  - Automatische Entdeckung (Schema Matching)
- Intention des Nutzers erkennen und repräsentieren
- Semantik der Daten erhalten
  - Assoziationen entdecken & erhalten
  - Schemata und deren Integritätsbedingungen nutzen
- Neue Datenwerte erzeugen
- Korrekte Gruppierungen erzeugen
- …

Schema Mapping im Kontext



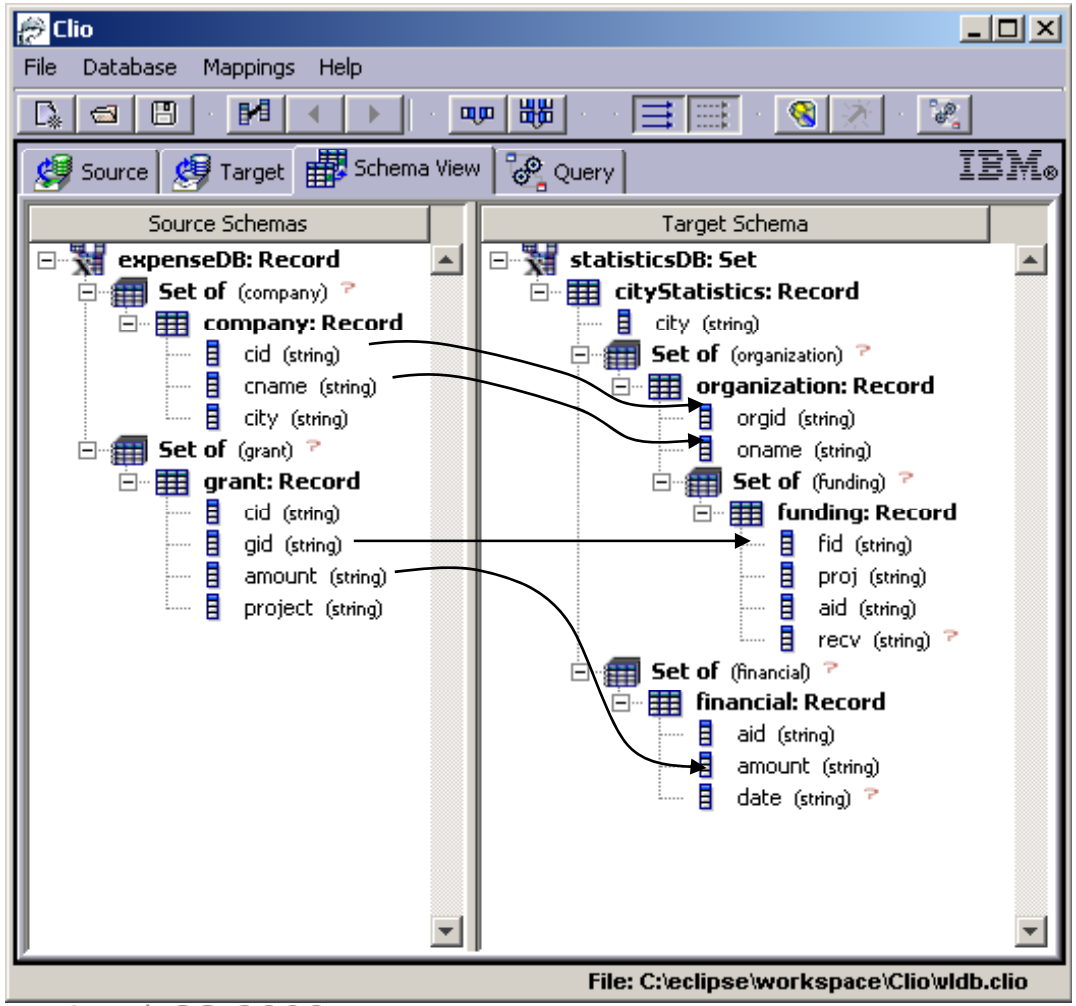
Quelle: [FHP+02]

Schema Mapping im Kontext



Schema Mapping im Kontext

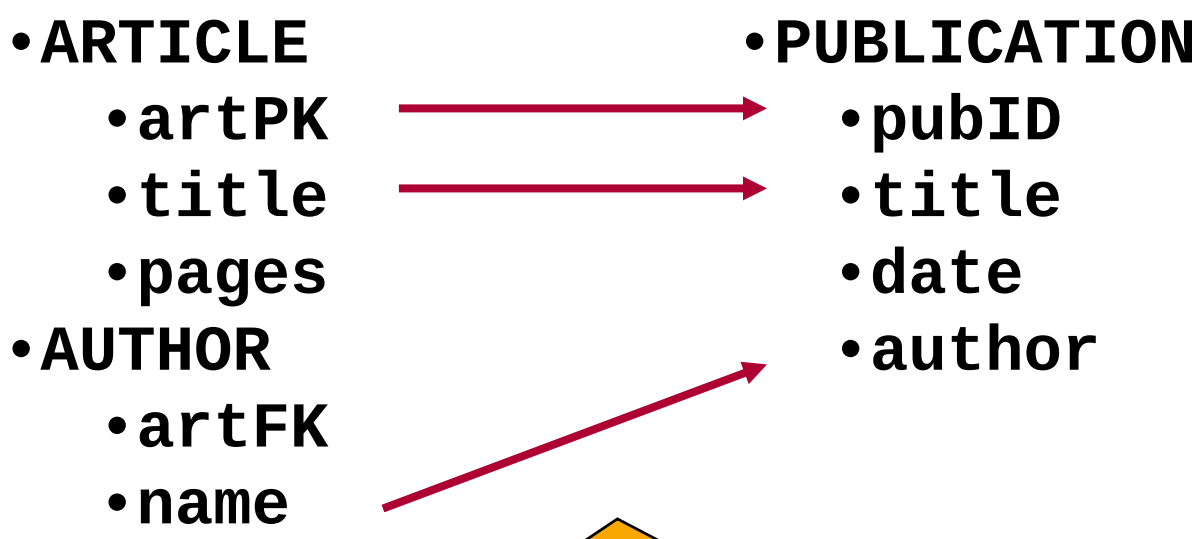
- Schema Matching & Korrespondenzen
- Schema Mapping
- Mapping Interpretation
- Daten-transformation



Wdh: Schematische Heterogenität – Beispiel

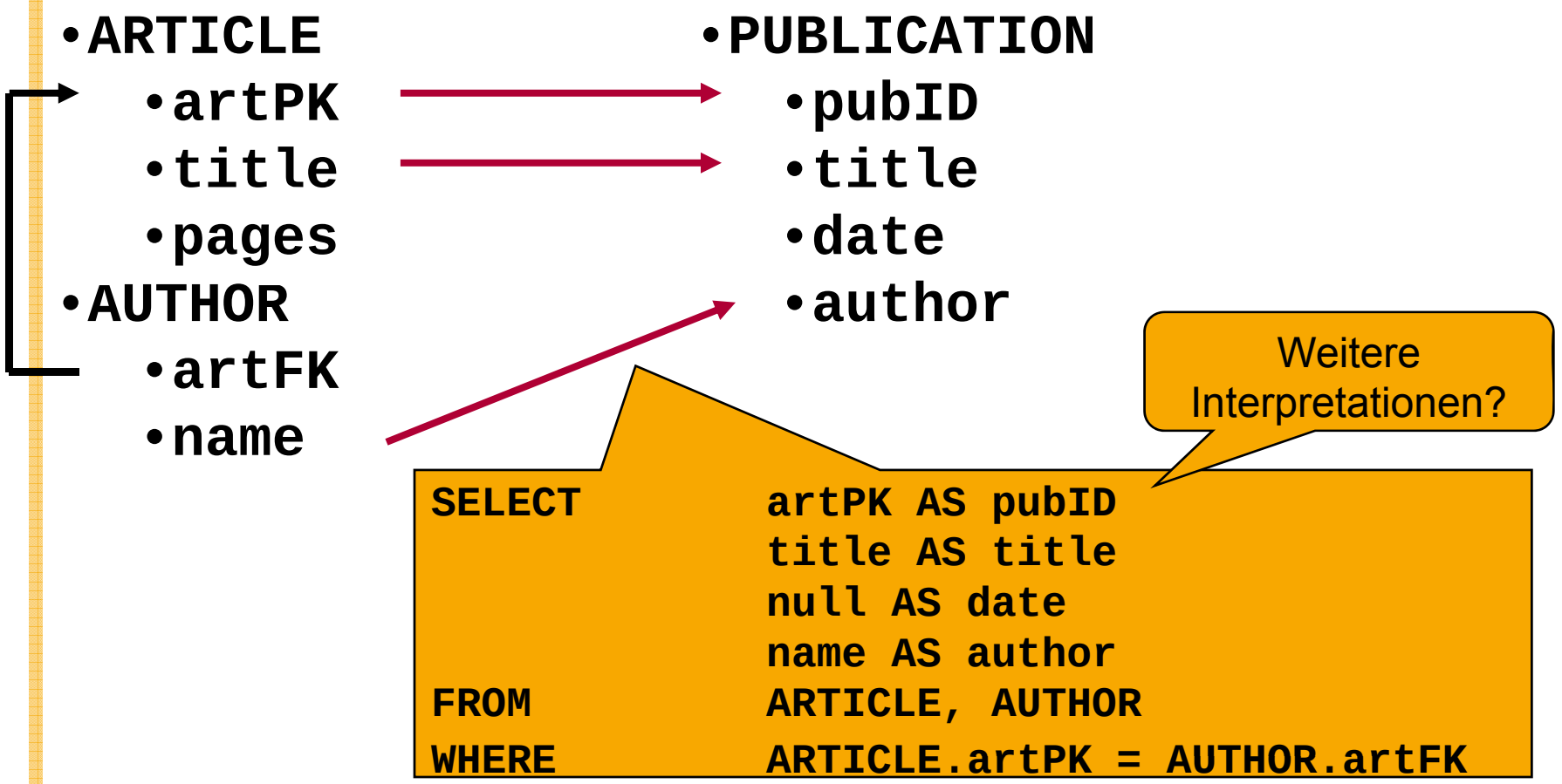
- Normalisiert vs. Denormalisiert
  - 1:n Assoziationen zwischen Werten wird unterschiedlich dargestellt
    - Durch Vorkommen im gleichen Tupel
    - Durch Schlüssel-Fremdschlüssel Beziehung
- Lösung: Schema Mapping

Schema Mapping Beispiel

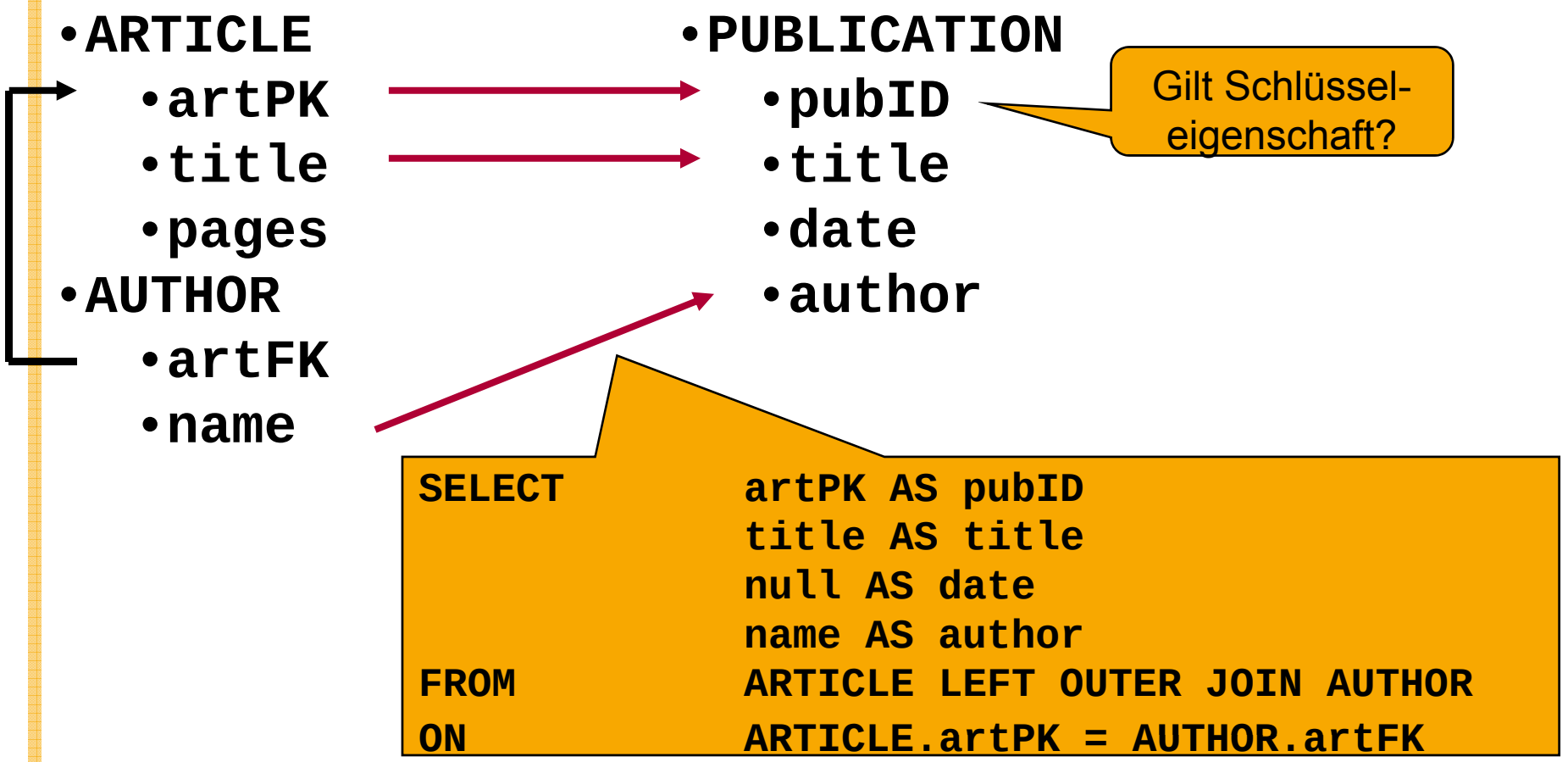



```
SELECT artPK AS pubID      UNION  SELECT null AS pubID
       title AS title                    null AS title
       null AS date                      null AS date
       null AS author                    name AS author
FROM   ARTICLE                   FROM   AUTHOR
```

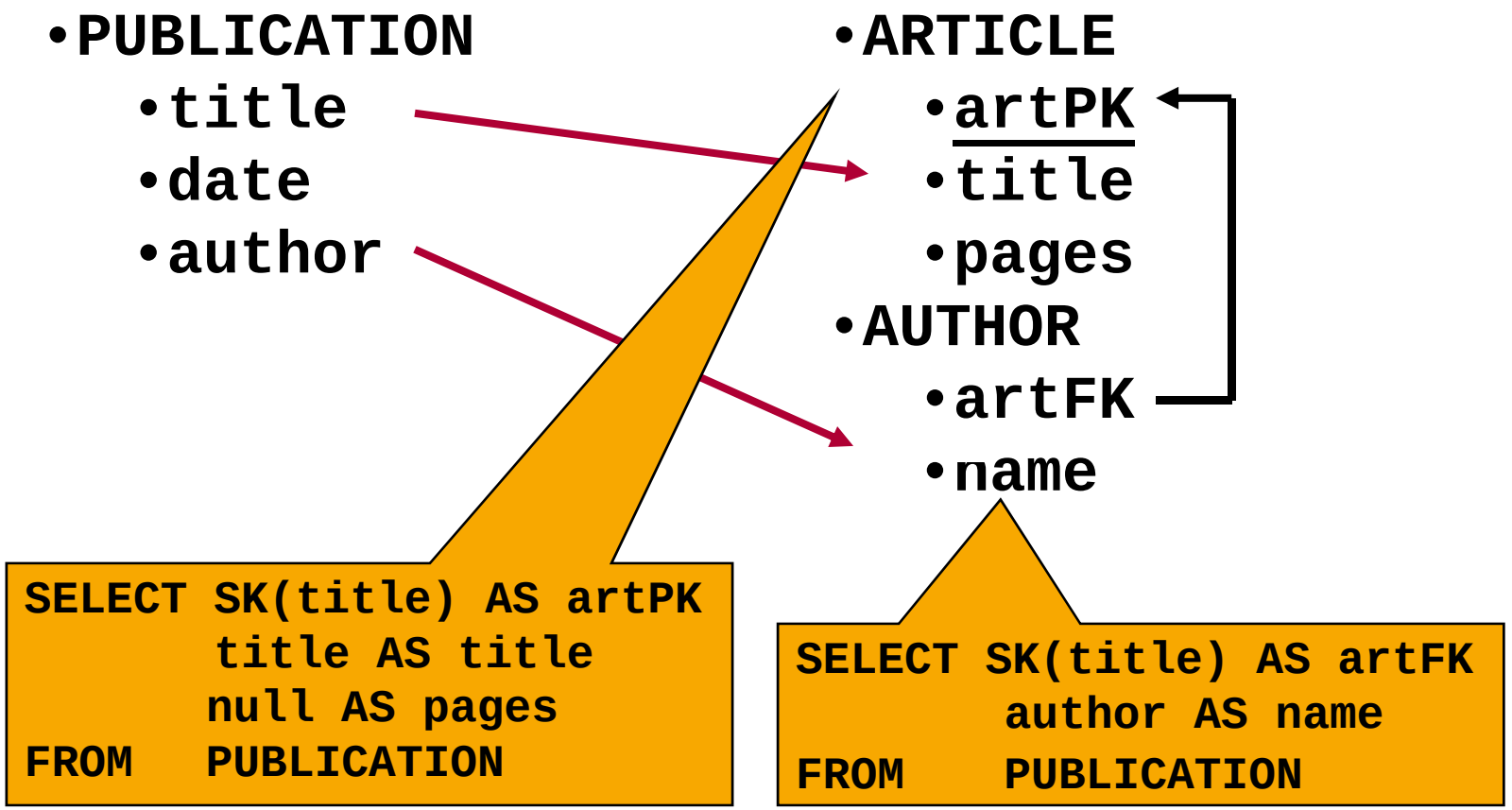
Schematische Heterogenität – Lösungen



Schematische Heterogenität – Lösungen



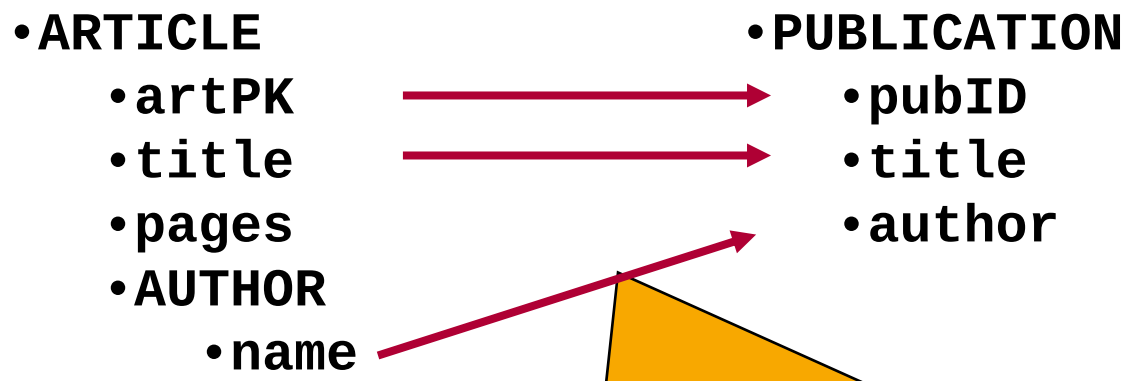
Wdh: Schematische Heterogenität – Lösungen



Wdh: Schematische Heterogenität – Beispiel

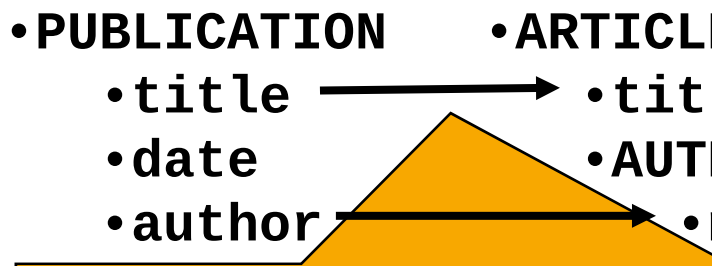
- Geschachtelt vs. Flach
  - 1:n Assoziationen werden unterschiedlich dargestellt
    - Als geschachtelte Elemente
    - Als Schlüssel-Fremdschlüssel Beziehung
- Lösung: Schema Mapping

Wdh: Schematische Heterogenität – Lösungen

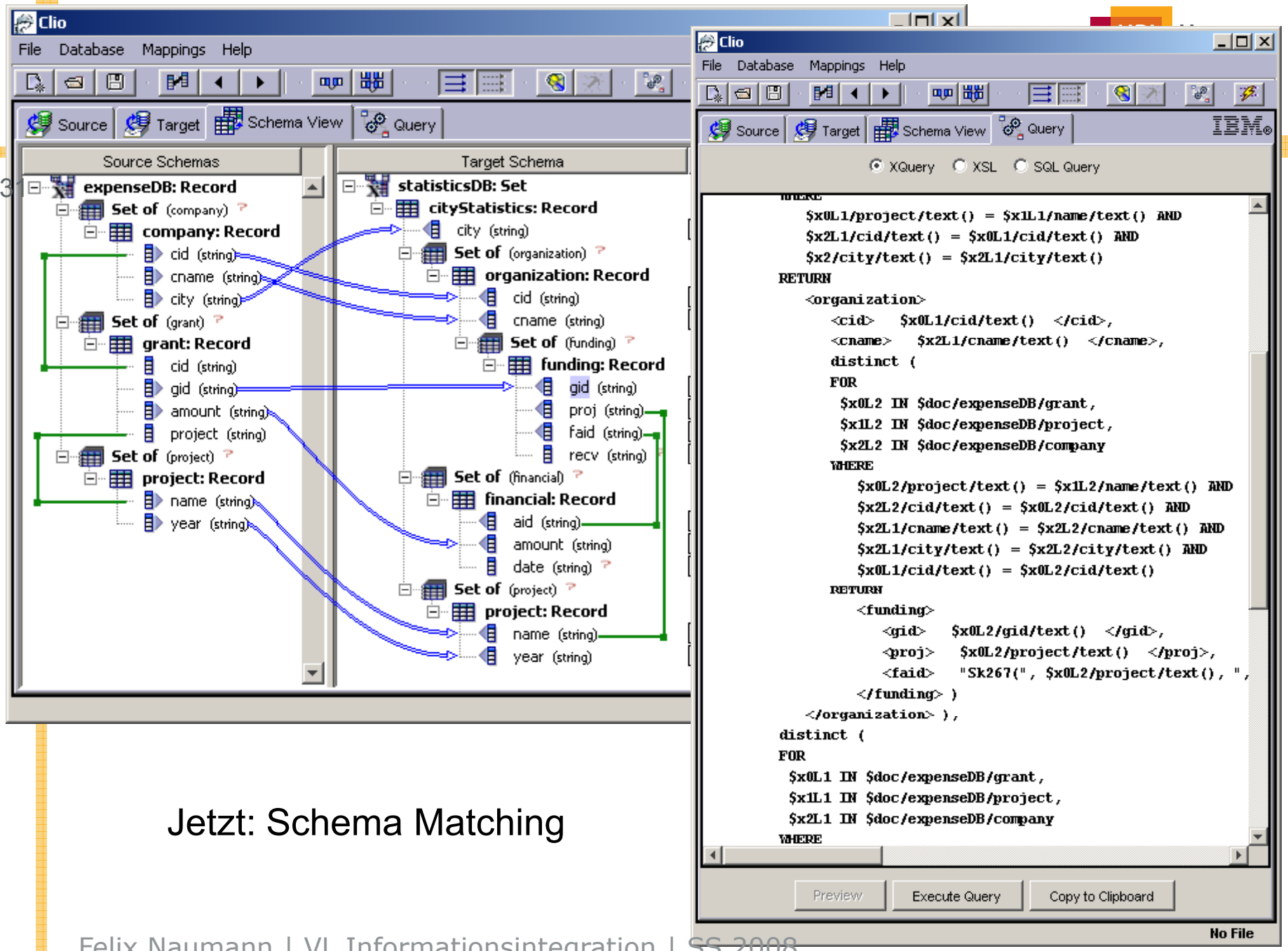



```
LET $doc0 := document("articles.xml") RETURN
<root> { distinct-values (
   FOR $x0 IN $doc0/authorDB/ARTICLE, $x1 IN
$x0/AUTHOR
   RETURN
        <publication>
        <pubID>  {  $x0/artPK/text() }  </pubID>
        <title>  {  $x0/title/text() }  </title>
        <author>  {  $x1/name/text() }  </author>
        </publication> )
   } </root>
```

Wdh: Schematische Heterogenität – Lösungen




```
LET $doc0 := document("publication.xml")
RETURN
<articles> { distinct-values (
   FOR $x0 IN $doc0/dblp/publication  RETURN
      <ARTICLE>
         <title>  {  $x0/title/text() }  </title>
         { distinct-values (
         FOR $x0L1 IN $doc0/dblp/publication
         WHERE $x0/title/text() = $x0L1/title/text()
         RETURN
            <AUTHOR>
               <name>  {  $x0L1/author/text() }  </name>
            </AUTHOR> )}
      </ARTICLE> ) } </articles>
```



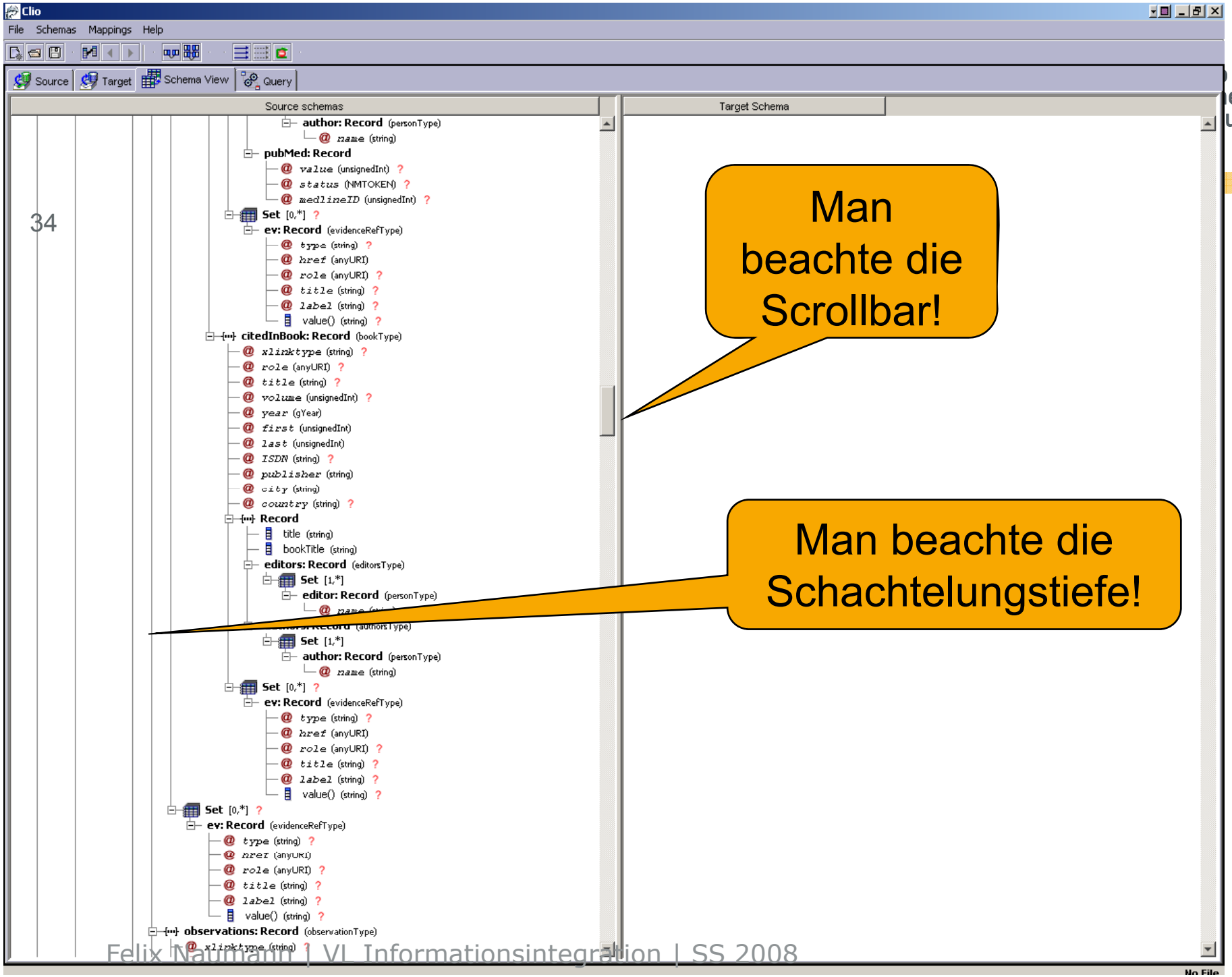
Jetzt: Schema Matching

Überblick

- Motivation
- Schema Mapping
- Schema Matching
  - Klassifikation von Schema Matching Methoden
  - Erweiterungen
  - Globales Matching
- Mapping Interpretation
- Mapping Werkzeuge

Schema Matching – Motivation

- Große Schemas
  - > 100 Tabellen, viele Attribute
  - Bildschirm nicht lang genug
- Unübersichtliche Schemas
  - Tiefe Schachtelungen
  - Fremdschlüssel
  - Bildschirm nicht breit genug
  - XML Schema
- Fremde Schemas
  - Unbekannte Synonyme
- Irreführende Schemas
  - Unbekannte Homonyme
- Fremdsprachliche Schemas
- Kryptische Schemas
  - |Attributnamen| ≤ 8 Zeichen
  - |Tabellennamen| ≤ 8 Zeichen

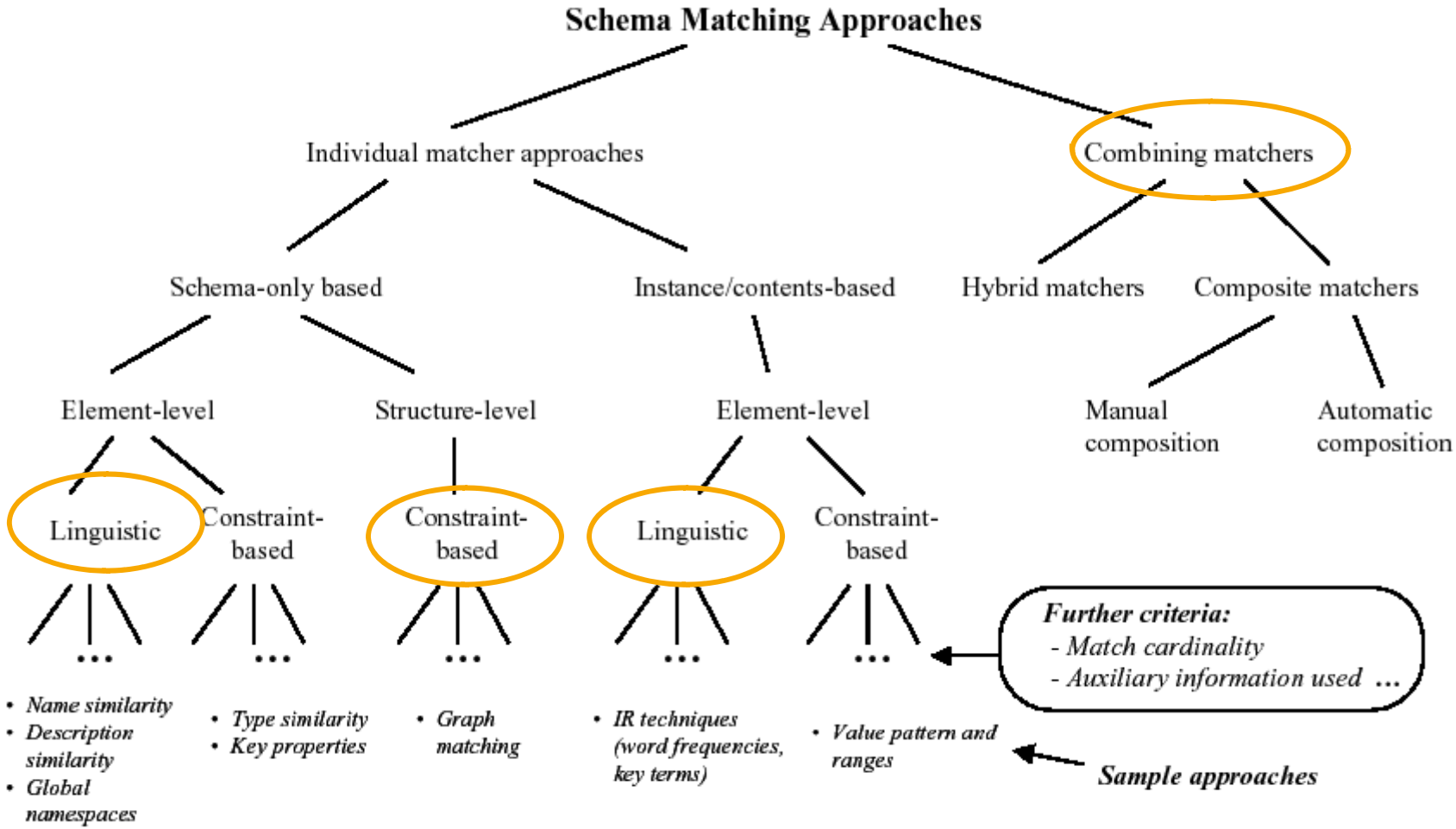

Clio
File Schemas Mappings Help
Source
Target
Schema View
Query
Source schemas
Target Schema
author: Record (personType)
@ name (string)
pubMed: Record
@ value (unsignedInt) ?
@ status (NMTOKEN) ?
@ medlineID (unsignedInt) ?
Set [0,*] ?
ev: Record (evidenceRefType)
@ type (string) ?
@ href (anyURI)
@ role (anyURI) ?
@ title (string) ?
@ label (string) ?
value() (string) ?
citedInBook: Record (bookType)
@ xlinktype (string) ?
@ role (anyURI) ?
@ title (string) ?
@ volume (unsignedInt) ?
@ year (gYear)
@ first (unsignedInt)
@ last (unsignedInt)
@ ISDN (string) ?
@ publisher (string)
@ city (string)
@ country (string) ?
Record
title (string)
bookTitle (string)
editors: Record (editorsType)
Set [1,*]
editor: Record (personType)
Set [1,*]
author: Record (personType)
@ name (string)
observations: Record (observationType)
Man beachte die Scrollbar!
Man beachte die Schachtelungstiefe!

Schema Matching – Motivation

Die Folgen

- Falsche Korrespondenzen (false positives)
- Fehlende Korrespondenzen (false negatives)
- Frustration
  - User verlieren sich im Schema
  - User verstehen Semantik der Schemas nicht

Schema Matching Klassifikation nach [RB01]



Schema Matching Klassifikation

Schema Matching basierend auf

- Namen der Schemaelemente (*label-based*)
- Darunterliegende Daten (*instance-based*)
- Struktur des Schemas (*structure-based*)
- Mischformen

Schema Matching – Label-based

Gegeben zwei Schemata mit Attributmengen A und B

Kernidee:

- Bilde Kreuzprodukt aller Attribute aus A und B.
- Für jedes Paar vergleiche Ähnlichkeit bezgl. Attributnamen (Label).
  - Z.B. Edit-distance
- Ähnlichste Paare sind Matches

Probleme:

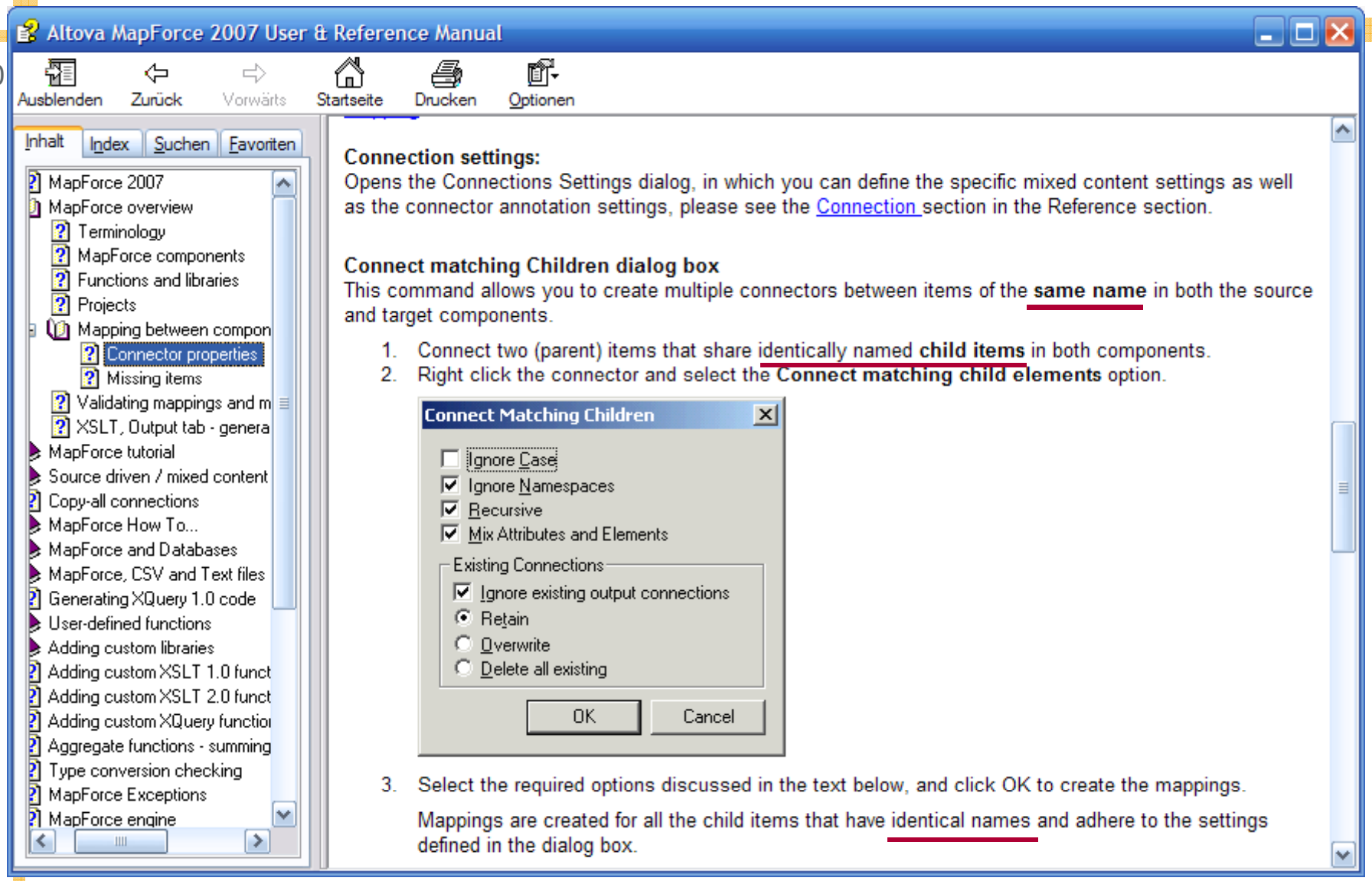
- Effizienz
- Auswahl der besten Matches (globales Matching)
  - Iterativ?
  - Stable Marriage?
- Synonyme und Homonyme werden nicht erkannt

Schema Matching – Label-based

Stand der Technik in kommerziellen Produkten

- Label-based
- Namensgleichheit
- Kein globales Matching
- Keine Ähnlichkeitsmaße
- Kein Instanz-basiertes Matching

Altova MapForce 2007

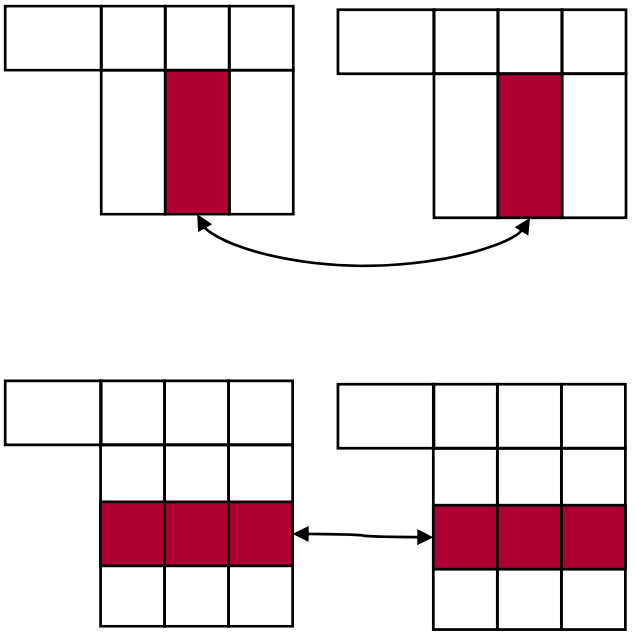


Schema Matching – Instance-based

- Gegeben zwei Schemata mit Attributmengen A und B, jeweils mit darunterliegenden Daten.
- Kernidee
  - Für jedes Attribute extrahiere interessante Eigenschaften der Daten
    - Buchstabenverteilung, Länge, etc.
  - Bilde Kreuzprodukt aller Attribute aus A und B.
  - Für jedes Paar vergleiche Ähnlichkeit bzgl. der Eigenschaften
- Probleme
  - Auswahl der Eigenschaften
  - Datenmenge: Sampling
  - Vergleichsmethode, z.B. Naive Bayes
  - Gewichtung (Maschinelles Lernen)

Instanz-basiertes Schema Matching

- Herkömmliche Lösung: Vertikal
  - Vergleich von Spalten
  - = Attribut-Klassifikation
  - [ICDE'02] u.v.a.m.
- Neue Lösung: Horizontal
  - Vergleich von Zeilen
  - = Duplikaterkennung
    - trotz fehlender Attribut-korrespondenzen
  - [ICDE'05]



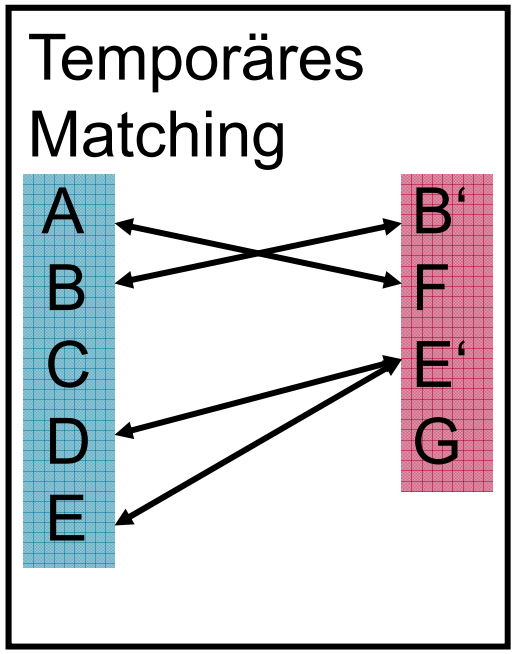
Duplicate-driven Schema Matching

A	B	C	D	E
Max	Michel	m	601- 4839204	601- 4839204
...	...	...	...	...

?

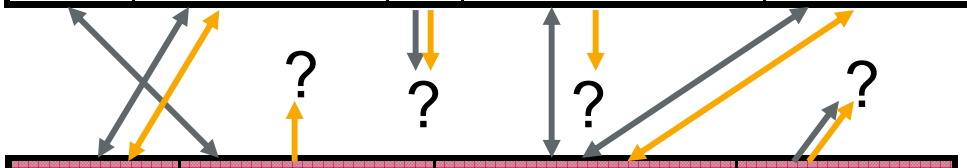
?

B‘	F	E‘	G
Michel	maxm	601- 4839204	UNIX
...	...	...	...

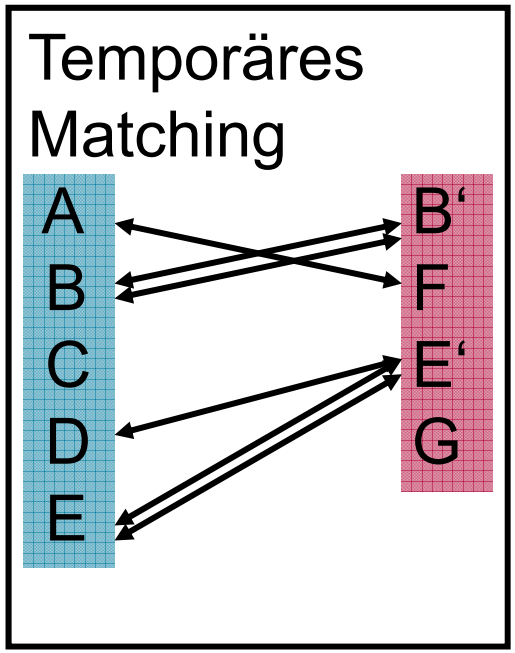


Duplicate-driven Schema Matching

A	B	C	D	E
Max	Michel	m	601- 4839204	601- 4839204
Sam	Adams	m	541- 8127100	541- 8121164



B`	F	E`	G
Michel	maxm	601- 4839204	UNIX
Adams	beer	541- 8127164	WinXP



Annahmen

- Es existieren Daten in beiden DBs.
- Es existieren (wenigstens ein paar) Duplikate in beiden DBs.

Schema Matching – Structure-based

- Gegeben zwei Schemata mit Elementmengen A und B.
- Kernidee
  - Nutze (komplexe) Struktur des Schemas aus.
  - Hierarchieebene
  - Elementtyp (Attribut, Relation, ...)
  - Nachbarschaftsbeziehungen

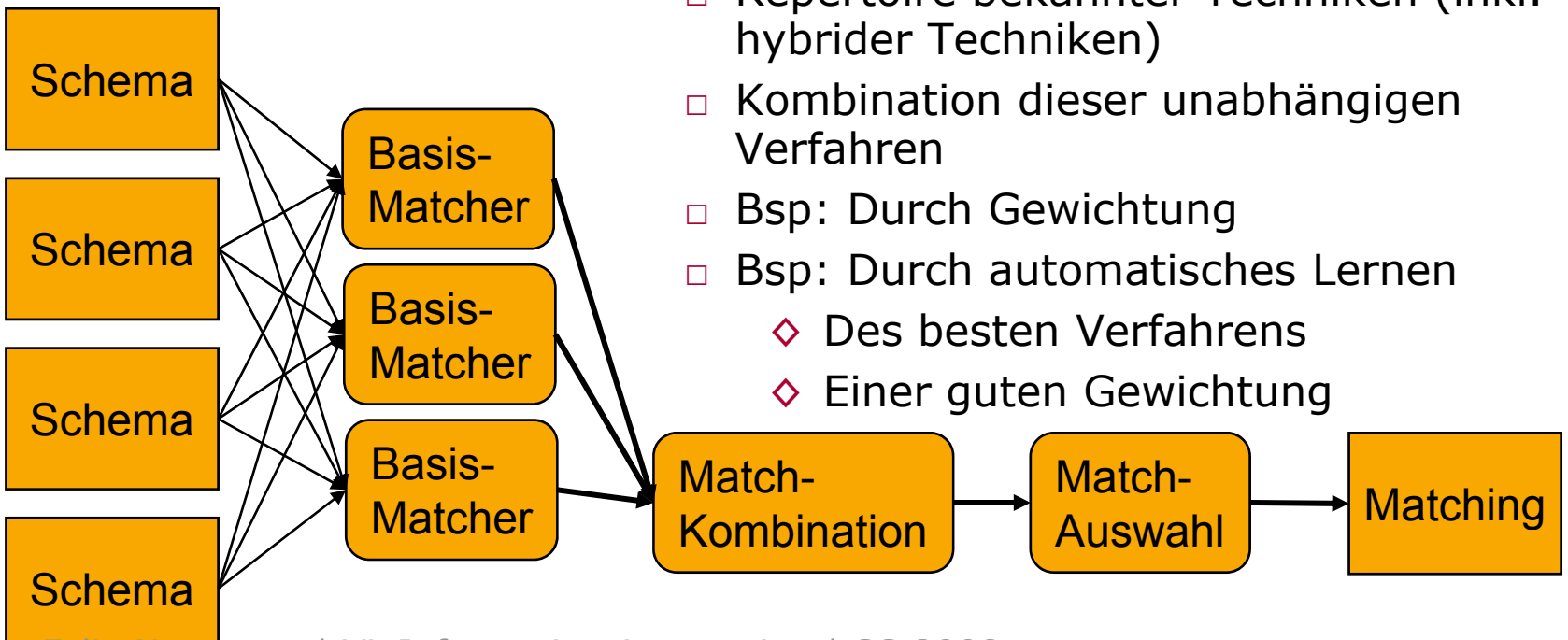
Schema Matching – Structure-based

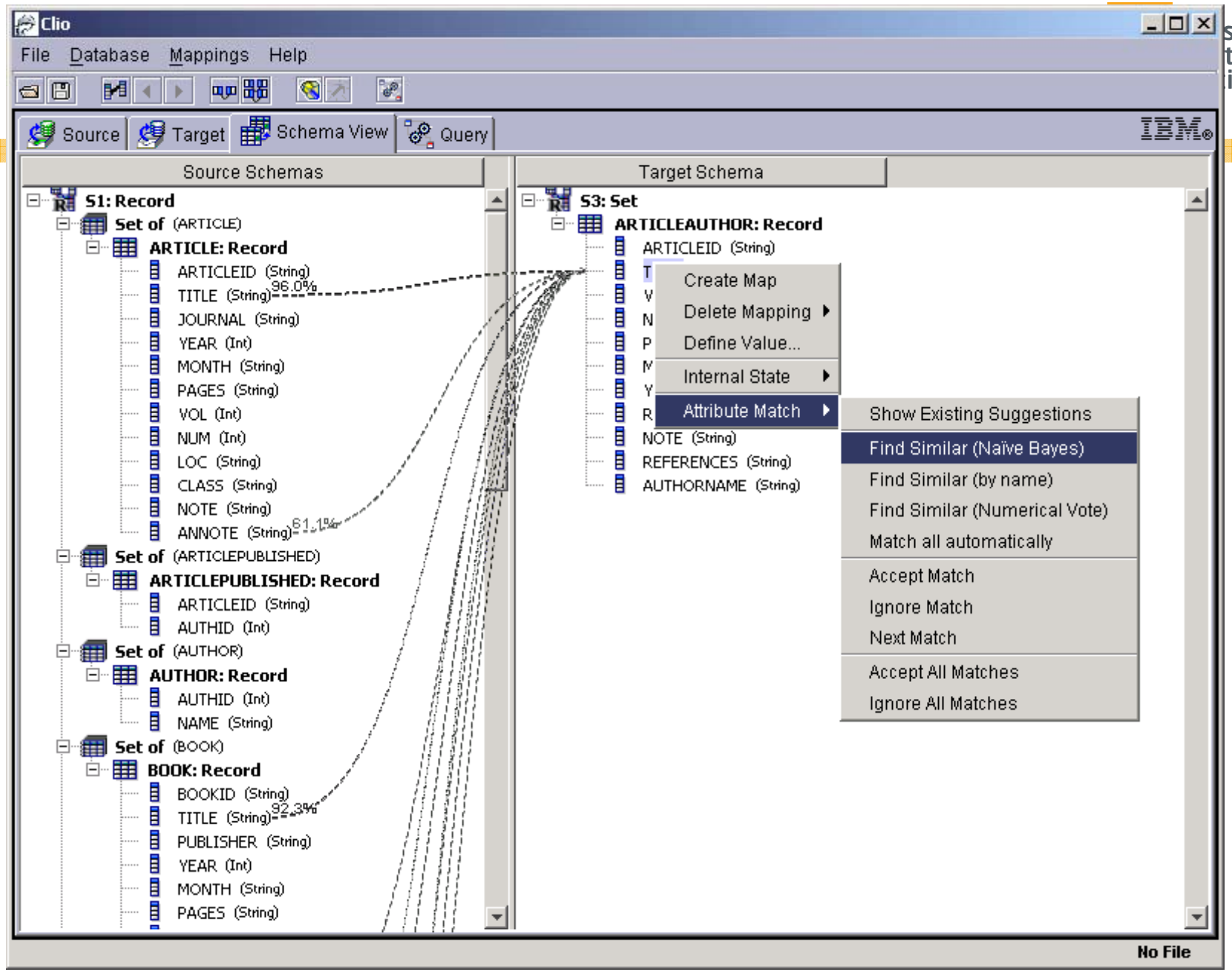
Beispiel: Similarity Flooding nach [MGMR02]

- Gegeben initiale Ähnlichkeit zwischen Schemaelementen (z.B. durch edit-distance oder durch Analyse der darunterliegenden Daten)
- Lasse Ähnlichkeiten „abfärben" auf die Nachbarn
  - Nachbarn sind durch Struktur definiert
  - Sind alle Nachbarn von x und y ähnlich zueinander, sind (vielleicht) auch x und y ein match.
- Analogie: Man „flutet" das Netzwerk der Ähnlichkeiten bis ein Gleichgewicht erreicht ist.

Schema Matching – Mischformen

- Hybrid
  - Gleichzeitige Anwendung mehrerer Techniken
  - Bsp: Instance-based + Datentypvergleich
- Composite
  - Repertoire bekannter Techniken (inkl. hybrider Techniken)
  - Kombination dieser unabhängigen Verfahren
  - Bsp: Durch Gewichtung
  - Bsp: Durch automatisches Lernen
    - Des besten Verfahrens
    - Einer guten Gewichtung




Clio
File Database Mappings Help
Source
Target
Schema View
Query
IBM
Source Schemas
S1: Record
Set of (ARTICLE)
ARTICLE: Record
ARTICLEID (String)
TITLE (String)
96.0%
JOURNAL (String)
YEAR (Int)
MONTH (String)
PAGES (String)
VOL (Int)
NUM (Int)
LOC (String)
CLASS (String)
NOTE (String)
ANNOTE (String)
61.1%
Set of (ARTICLEPUBLISHED)
ARTICLEPUBLISHED: Record
ARTICLEID (String)
AUTHID (Int)
Set of (AUTHOR)
AUTHOR: Record
AUTHID (Int)
NAME (String)
Set of (BOOK)
BOOK: Record
BOOKID (String)
TITLE (String)
92.3%
PUBLISHER (String)
YEAR (Int)
MONTH (String)
PAGES (String)
Target Schema
S3: Set
ARTICLEAUTHOR: Record
ARTICLEID (String)
NOTE (String)
REFERENCES (String)
AUTHORNAME (String)
Create Map
Delete Mapping
Define Value...
Internal State
Attribute Match
Show Existing Suggestions
Find Similar (Naïve Bayes)
Find Similar (by name)
Find Similar (Numerical Vote)
Match all automatically
Accept Match
Ignore Match
Next Match
Accept All Matches
Ignore All Matches
No File

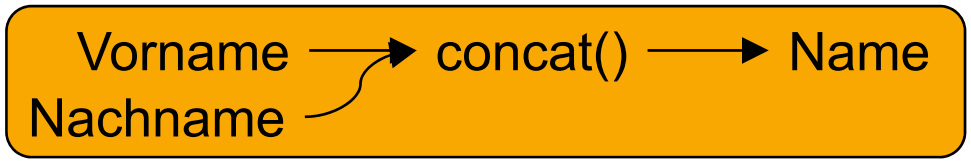
Überblick

- Motivation
- Schema Mapping
- Schema Matching
  - Klassifikation von Schema Matching Methoden
  - Erweiterungen
  - Globales Matching
- Mapping Interpretation
- Mapping Werkzeuge

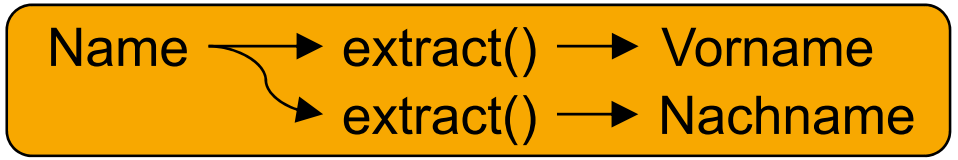
Schema Matching – Erweiterungen

- n:1 und 1:n Matches
  - Viele Kombinations-möglichkeiten
  - Viele Funktionen denkbar
    - Mathematische Operatoren, Konkatenation, etc.
  - Parsingregeln
- n:m Matching?
- Matching in komplexen Schemata
  - Ziel: Finde Mapping, nicht Korrespondenzen

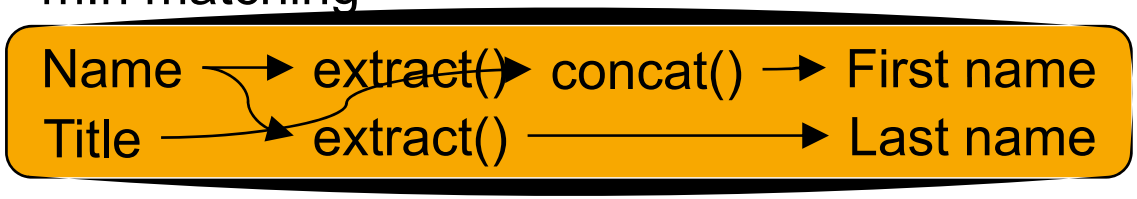
n:1 Matching



1:n Matching

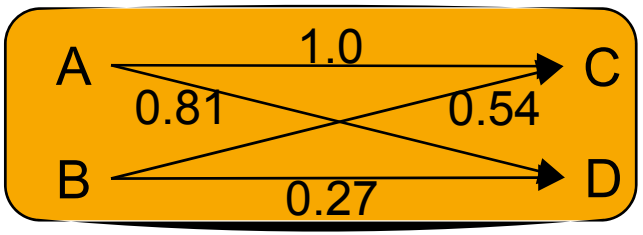


m:n matching



Schema Matching – Erweiterungen

- Global matching (gleich)
  - Matche nicht nur einzelne Attribute (oder Attributmengen)
  - Sondern komplette Tabellen oder komplette Schemata
  - Stable Marriage Problem



Schema Matching – Weitere Anwendungen

- Herkömmlich: Korrespondenzen finden
- Schlüssel – Fremdschlüssel finden
  - Ähnliche Attribute innerhalb eines Schemas sind gute Kandidaten
- Höher-stufige Korrespondenzen finden
  - Ähnlichkeiten von Tabellen durch Aggregation der Matches ihrer Attribute

Überblick

- Motivation
- Schema Mapping
- Schema Matching
  - Klassifikation von Schema Matching Methoden
  - Erweiterungen
  - Globales Matching
- Mapping Interpretation
- Mapping Werkzeuge

Schema Matching – Stable Marriage

- Gegeben
  - *n* Frauen (Attribute in Schema A) und *m* Männer (Attribute in Schema B)
- Monogamie
  - Je eine Frau kann nur mit je einem Mann verheiratet sein (nur 1:1 matches)
- Jede Frau hat eine Rangliste der Männer und umgekehrt
  - Bei Schema Matching
    - Attribut-Ähnlichkeit gemäß eines der vorigen Verfahren
    - Rangliste ist (normalerweise) symmetrisch
- Gesucht: Paarung (globales Matching), so dass niemals gilt
  - f_1 heiratet m_1, f_2 heiratet m_2,
  - aber f_1 bevorzugt m_2 und m_2 bevorzugt f_1 (Instabil!)

Stable Marriage – Beispiel

Männer (1-4)	Frauen (A-D)
1: B, D, A, C	A: 2, 1, 4, 3
2: C, A, D, B	B: 4, 3, 1, 2
3: B, C, A, D	C: 1, 4, 3, 2
4: D, A, C, B	D: 2, 1, 4, 3

Beispiel aus: David Toth,
"The Stable Marriage Problem: More Marital Happiness than Reality TV"
April 25, 2003, Connecticut College, New London, CT, USA,

Stable Marriage – Beispiel

Männer (1-4)	Frauen (A-D)
1: B, D, A, C	A: 2, 1, 4, 3
2: C, A, D, B	B: 4, 3, 1, 2
3: B, C, A, D	C: 1, 4, 3, 2
4: D, A, C, B	D: 2, 1, 4, 3

1 stellt Antrag an B, sie willigt ein: (1, B)

Stable Marriage – Beispiel

Männer (1-4)	Frauen (A-D)
1: B, D, A, C	A: 2, 1, 4, 3
2: C, A, D, B	B: 4, 3, 1, 2
3: B, C, A, D	C: 1, 4, 3, 2
4: D, A, C, B	D: 2, 1, 4, 3

1 stellt Antrag an B, sie willigt ein: (1, B)

2 stellt Antrag an C, sie willigt ein: (1, B) (2, C)

Stable Marriage – Beispiel

Männer (1-4)	Frauen (A-D)
1: ~~B~~, D, A, C	A: 2, 1, 4, 3
2: C, A, D, B	B: 4, 3, 1, 2
3: B, C, A, D	C: 1, 4, 3, 2
4: D, A, C, B	D: 2, 1, 4, 3

1 stellt Antrag an B, sie willigt ein: (1, B)

2 stellt Antrag an C, sie willigt ein: (1, B) (2, C)

3 stellt Antrag an B, sie willigt ein & verlässt 1: (2, C) (3, B)

Stable Marriage – Beispiel

Männer (1-4)	Frauen (A-D)
1: ~~B~~, D, A, C	A: 2, 1, 4, 3
2: C, A, D, B	B: 4, 3, 1, 2
3: B, C, A, D	C: 1, 4, 3, 2
4: D, A, C, B	D: 2, 1, 4, 3

1 stellt Antrag an B, sie willigt ein : (1, B)

2 stellt Antrag an C, sie willigt ein : (1, B) (2, C)

3 stellt Antrag an B, sie willigt ein & verlässt 1: (2, C) (3, B)

1 stellt Antrag an D, sie willigt ein : (1, D) (2, C) (3, B)

Stable Marriage – Beispiel

Männer (1-4)
1: ~~B~~, D, A, C
2: C, A, D, B
3: B, C, A, D
4: ~~D~~, A, C, B

Frauen (A-D)
A: 2, 1, 4, 3
B: 4, 3, 1, 2
C: 1, 4, 3, 2
D: 2, 1, 4, 3

1 stellt Antrag an B, sie willigt ein : (1, B)
2 stellt Antrag an C, sie willigt ein : (1, B) (2, C)
3 stellt Antrag an B, sie willigt ein & verlässt 1: (2, C) (3, B)
1 stellt Antrag an D, sie willigt ein : (1, D) (2, C) (3, B)
4 stellt Antrag an D, sie lehnt ab : (1, D) (2, C) (3, B)

Stable Marriage – Beispiel

Männer (1-4)
1: B, D, A, C
2: C, A, D, B
3: B, C, A, D
4: D, A, C, B

Frauen (A-D)
A: 2, 1, 4, 3
B: 4, 3, 1, 2
C: 1, 4, 3, 2
D: 2, 1, 4, 3

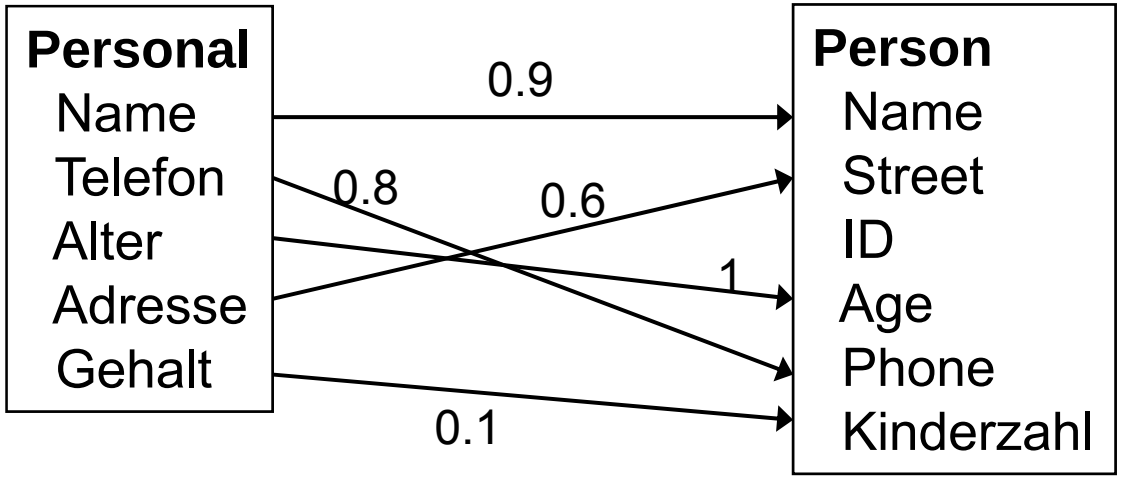
1 stellt Antrag an B, sie willigt ein : (1, B)
2 stellt Antrag an C, sie willigt ein : (1, B) (2, C)
3 stellt Antrag an B, sie willigt ein & verlässt 1: (2, C) (3, B)
1 stellt Antrag an D, sie willigt ein : (1, D) (2, C) (3, B)
4 stellt Antrag an D, sie lehnt ab: (1, D) (2, C) (3, B)
4 stellt Antrag an A, sie willigt ein : (1, D) (2, C) (3, B) (4, A)

Vier stabile Paare!

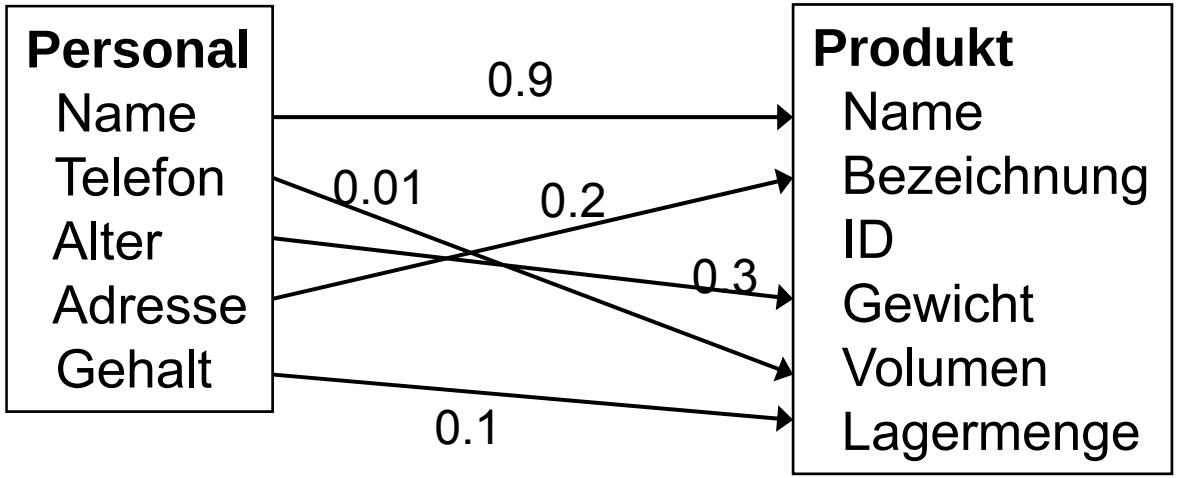
Maximum Weighted Matching

- Alternative zu Stable Marriage
- Suche Matching mit maximalem Gewicht in bipartiten Graphen
  - Bipartit:
    - Knoten in zwei Klassen (Quelle & Ziel)
    - Kanten nur zwischen Knoten verschiedener Klassen (Korrespondenzen)
  - Maximiere Summe der einzelnen Gewichte/Ähnlichkeiten
- $O(n^3)$ („Ungarische Methode“)

Diskussion: Globales Matching



Diskussion: Globales Matching



Zusammenfassung – Schema Matching

- Schema Matching basierend auf
  - Namen der Schemaelemente (label-based)
  - Darunterliegende Daten (instance-based)
  - Struktur des Schemas (structure-based)
  - Mischformen, Meta-Matcher
- High-order Matching
- n:m Matching
- Globales Matching

Überblick

- Motivation
- Schema Mapping
- Schema Matching
- Mapping Interpretation
  - Aus [FHP+02] und VLDB 2002 Vortragsfolien
- Mapping Werkzeuge

Mapping – Das Problem

- Gegeben: Zwei Schemata
  - Unabhängig voneinander erzeugt
  - Relational
  - Geschachtelt
  - Mit Integritätsbedingungen (Schlüssel/Fremdschlüssel)
  - Stellen teilweise unterschiedliche Daten dar
- Gegeben: Eine Menge von Korrespondenzen zwischen den Schemata
- Gesucht: Anfrage, die Daten des einen in Daten des anderen Schemas transformiert, wobei
  - Semantik des Quellschemas erhalten bleibt,
  - Integritätsbedingungen des Zielschemas berücksichtigt werden,
  - und möglichst alle Korrespondenzen berücksichtigt werden.

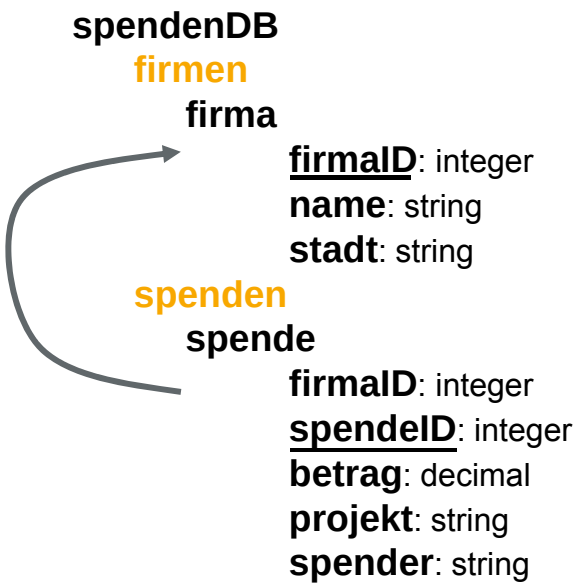
Relationale vs. XML Schemata

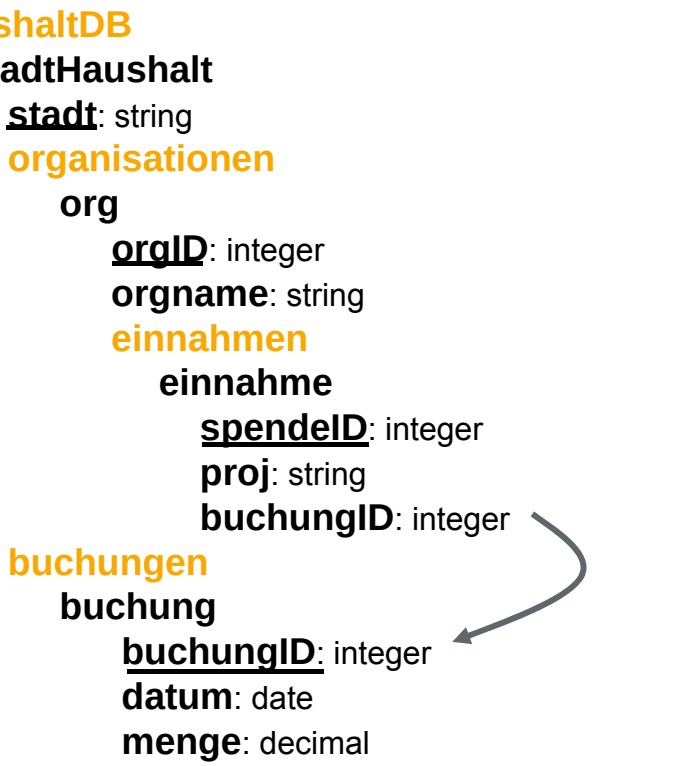
- Relationale Schemata
  - Flach
- XML Schemata
  - Flach oder geschachtelt
- NF2

Employee

ename	Children		Skills		
	name	dob	type	Exams	
				year	city
Smith	Sam	2/10/84	typing	1984	Atlanta
	Sue	1/20/85		1985	Dallas
			dictation	1984	Atlanta
Watson	Sam	3/12/78	filing	1984	Atlanta
				1975	Austin
				1971	Austin
			typing	1962	Waco

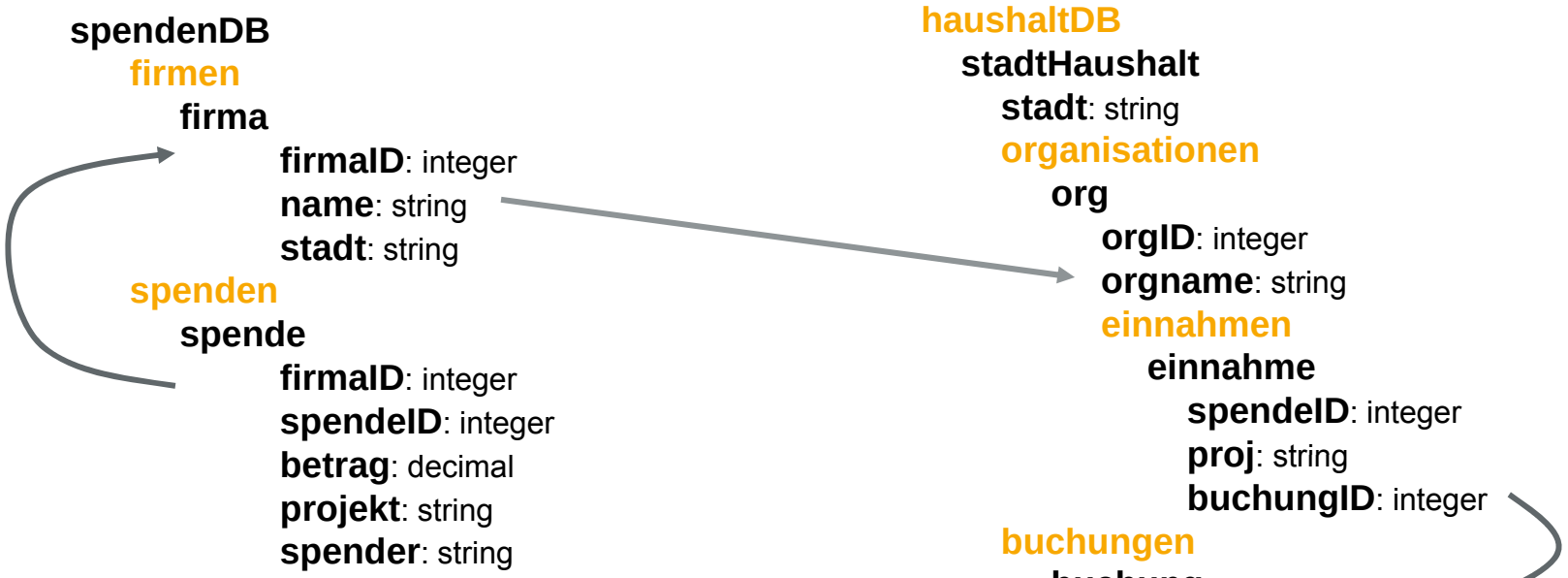
Mapping – Beispiel





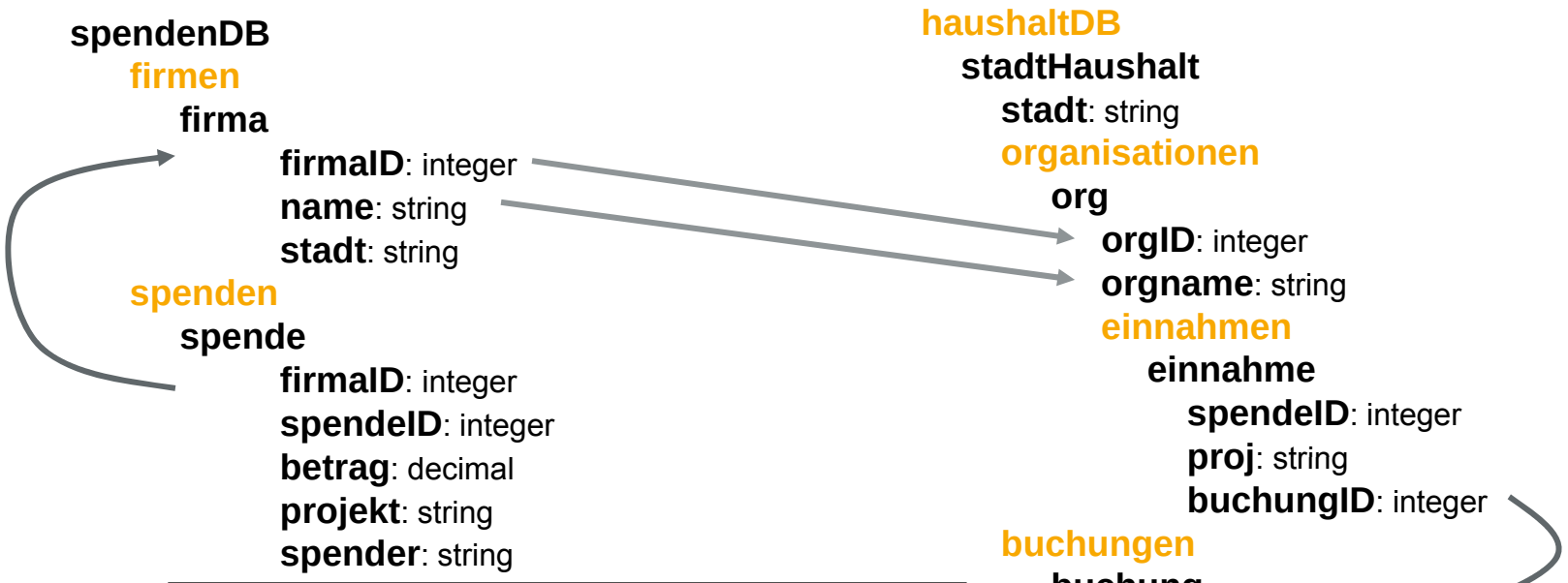
Quelle für Beispiel: [FHP+02]

Mapping – Beispiel



Interpretation:
- Erzeuge für jede **firma** in **spendenDB**. eine **org** in **haushaltDB**.
- **orgID** muss „erfunden“ werden
- **stadt** muss „erfunden“ werden

Mapping – Beispiel



Interpretation:
- Erzeuge für jede **firma** in **spendenDB**. eine **org** in **haushaltDB**.
- **stadt** muss „erfunden“ werden

„Erfinden“ von Werten

Zwei Gründe zum Erfinden: „non-null“ und Identität

„non-null“ Werte

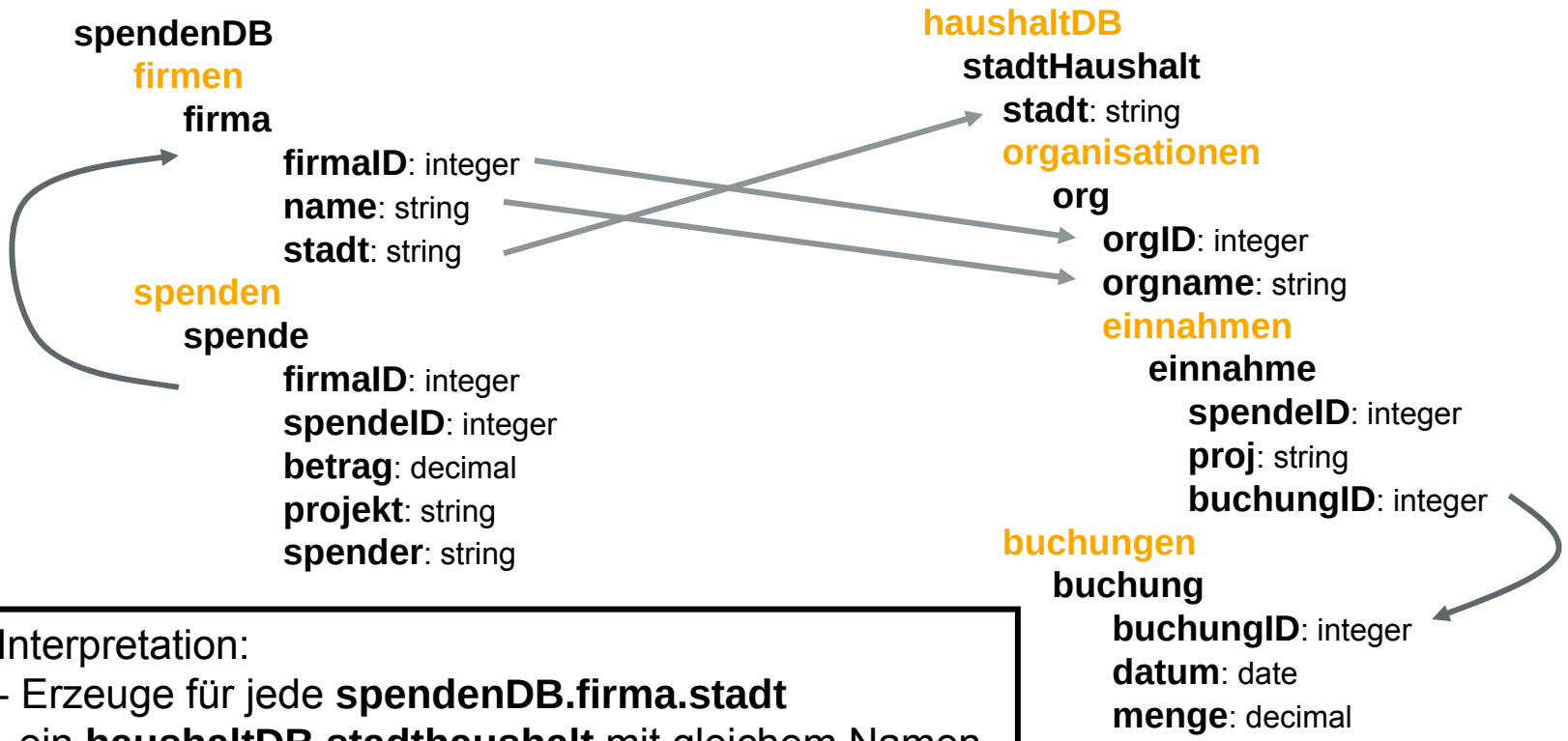
- Erfundener Wert egal
- Z.B. „unbekannt“ oder „null“ (oder „Berlin“)

ID Werte

- Skolemfunktion:
  - Input: n Werte (beliebige Domäne)
  - Output: bezgl. Input eindeutiger Wert (beliebiger Domäne)
  - Beispiel: Konkatenation aller Inputwerte als String

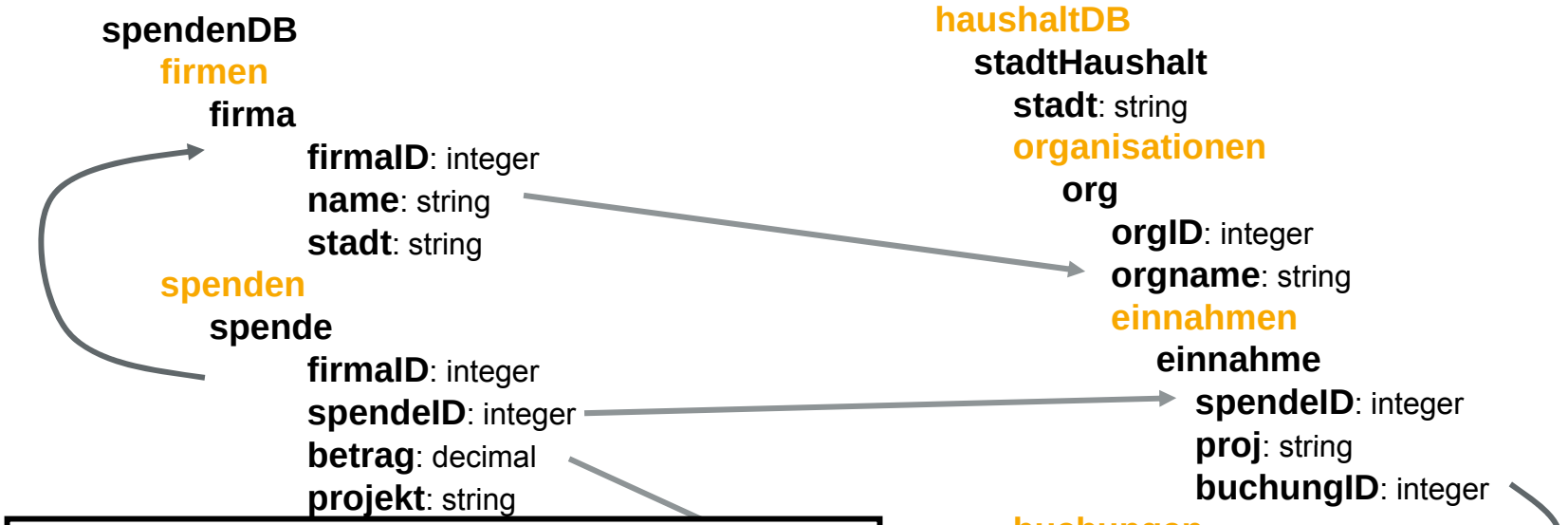
Wert für org.orgID nicht egal, sondern je nach firma.name eindeutig!

Mapping – Beispiel



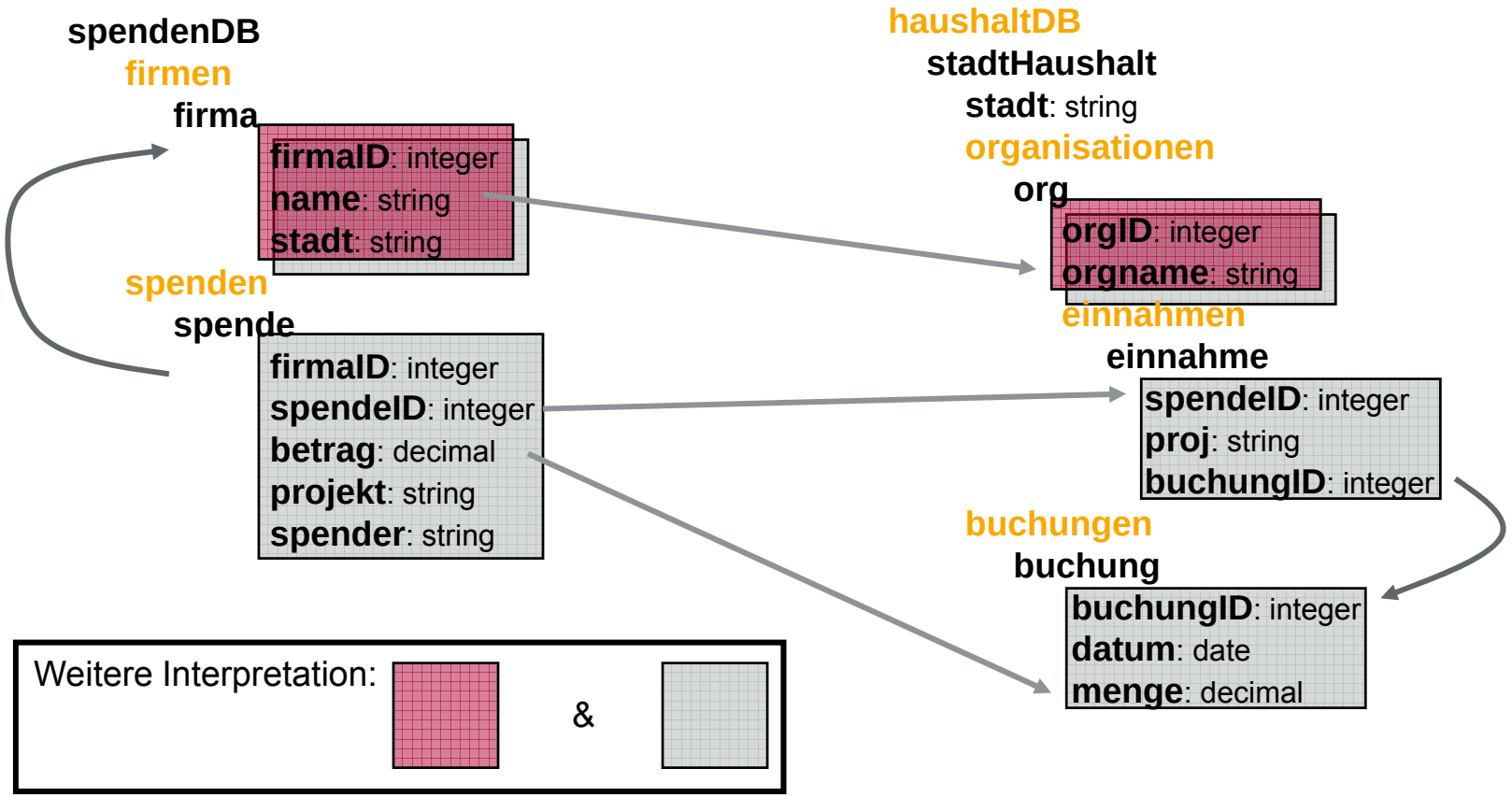
Interpretation:
- Erzeuge für jede **spendenDB.firma.stadt** ein **haushaltDB.stadthaushalt** mit gleichem Namen.
- Gruppiere jede **firma** unter den entsprechenden **stadtHaushalt**.

Mapping – Beispiel

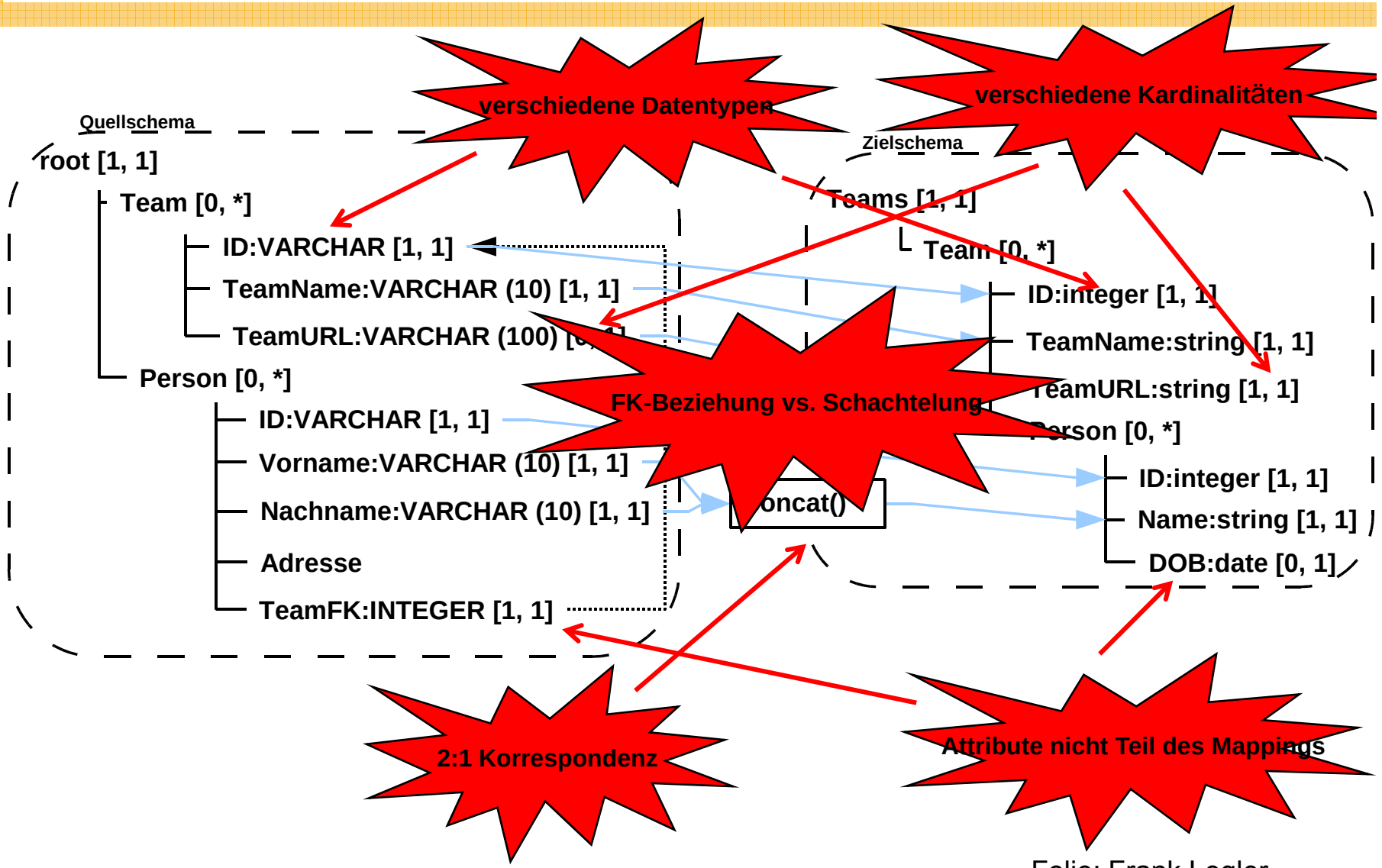


Interpretation:
- Erzeuge für jede **firma** in **spendenDB**. eine **org** in **haushaltDB**.
- Erzeuge für jede **spende** in **spendenDB** eine **einnahme** in **haushaltDB**
- Erzeuge für jede **spende** in **spendenDB** eine **buchung** in **haushaltDB**
- Gruppiere korrekt: Schachtelung & Fremdschlüssel!

Mapping – Beispiel

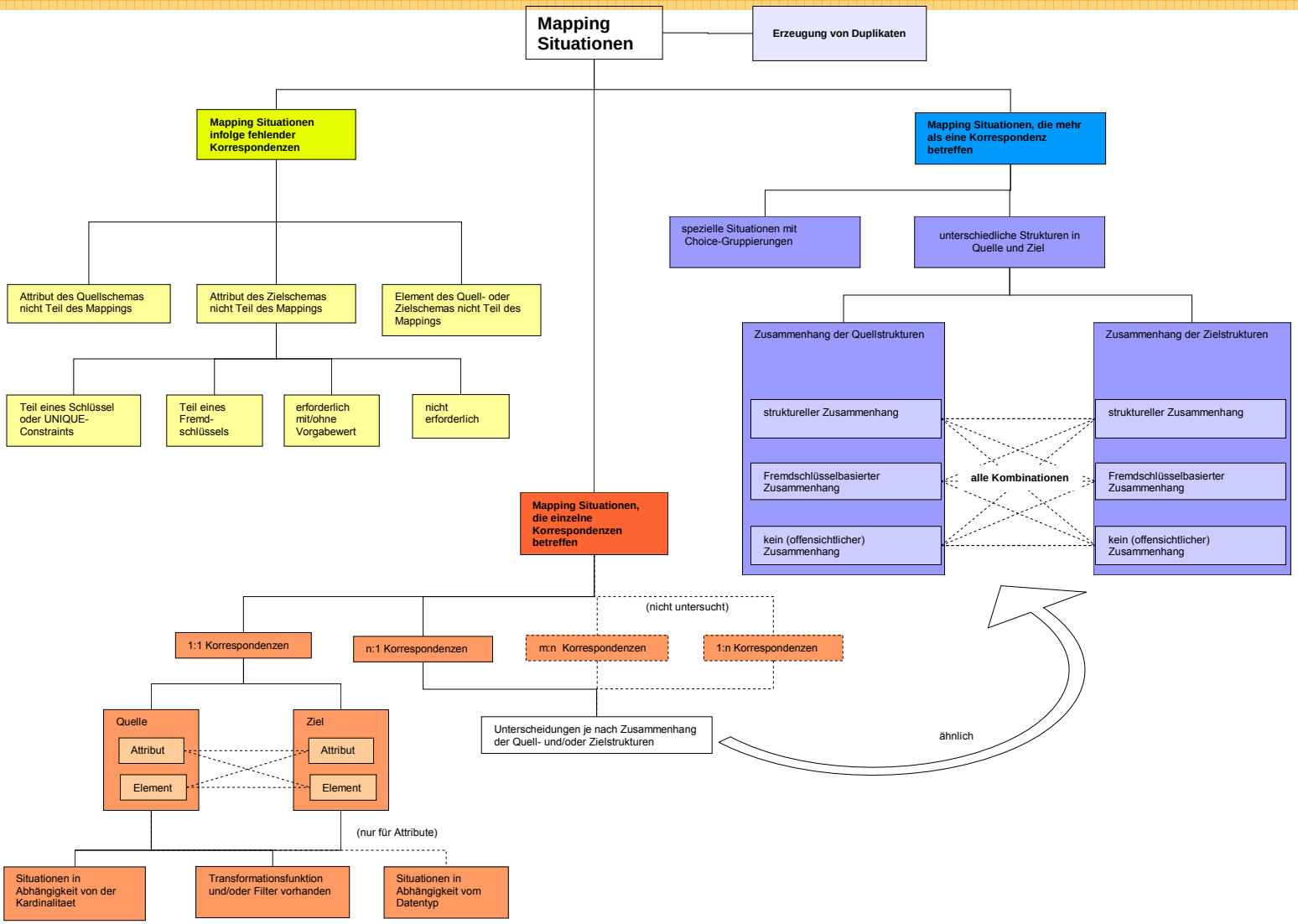


Verschiedene Mapping Situationen



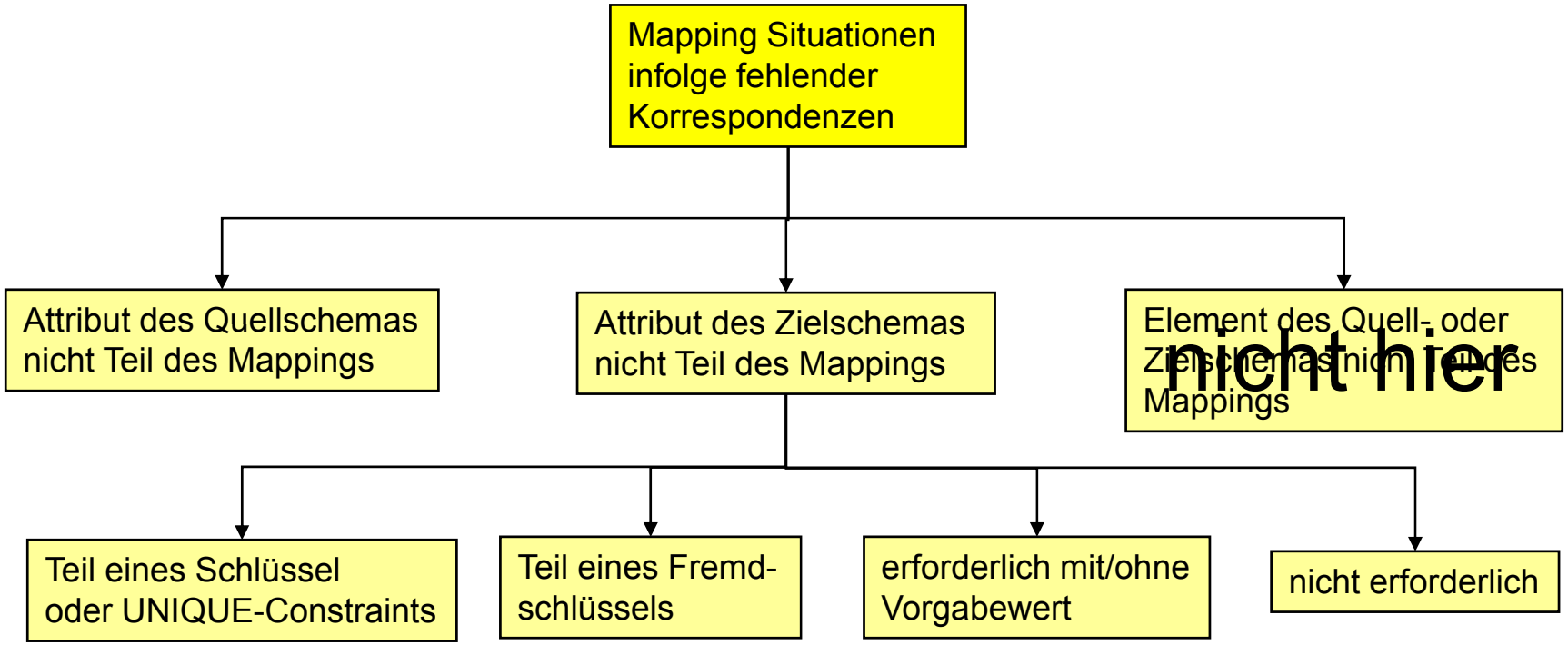
Folie: Frank Legler

Klassifikation von Mapping Situationen



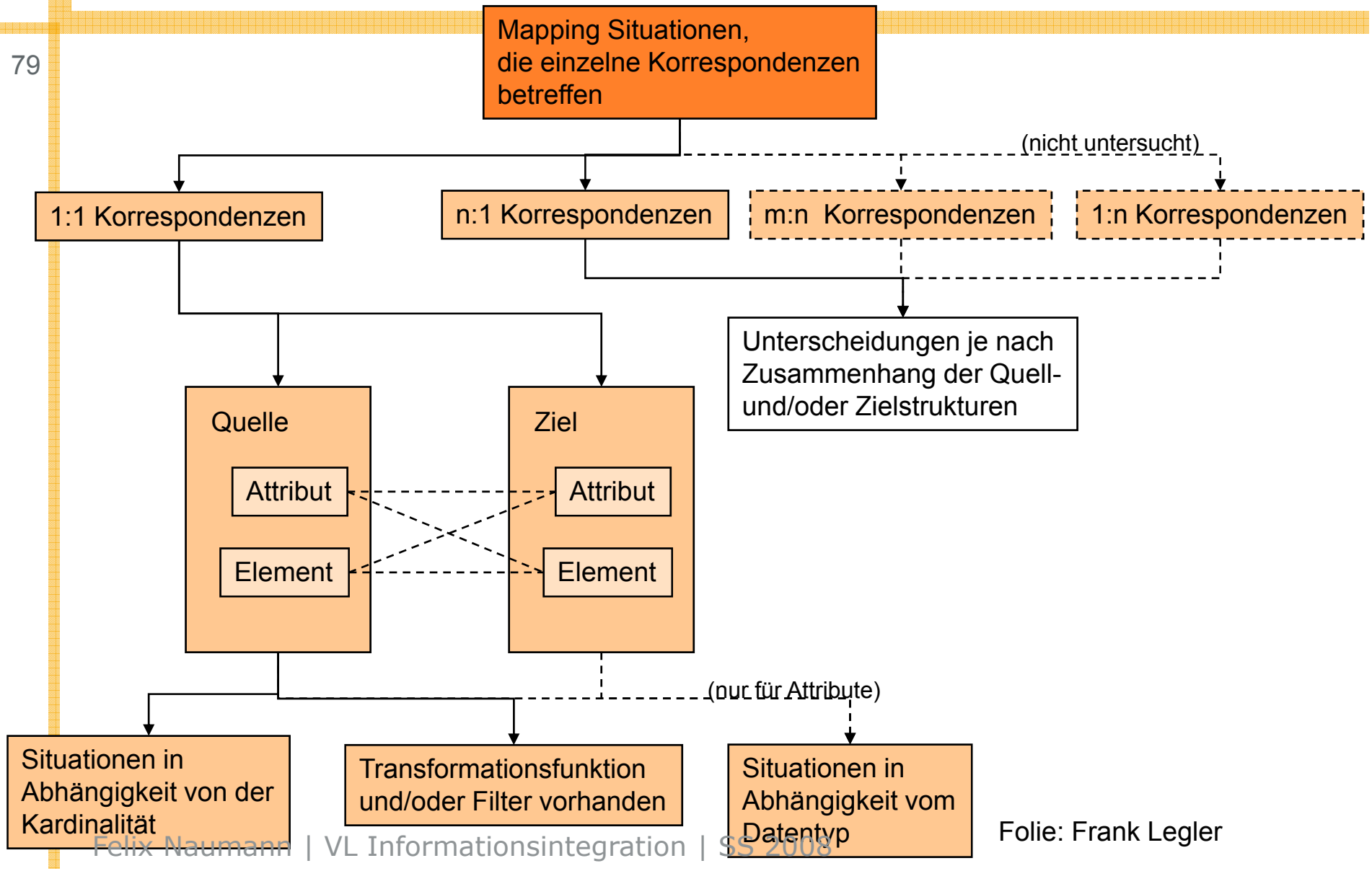
Folie: Frank Legler

Klassifikation von Mapping Situationen



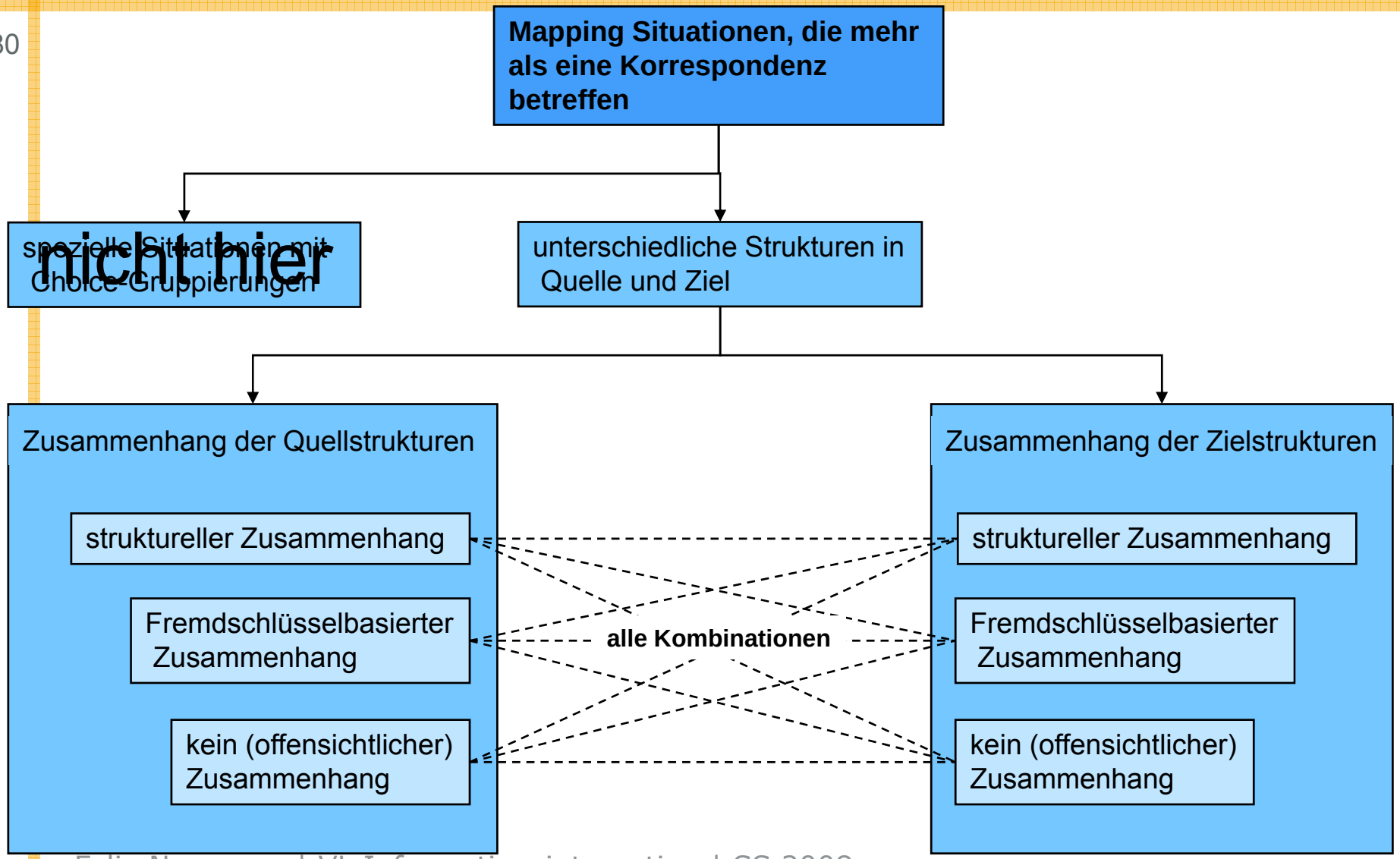
Folie: Frank Legler

Klassifikation von Mapping Situationen



Folie: Frank Legler

Klassifikation von Mapping Situationen

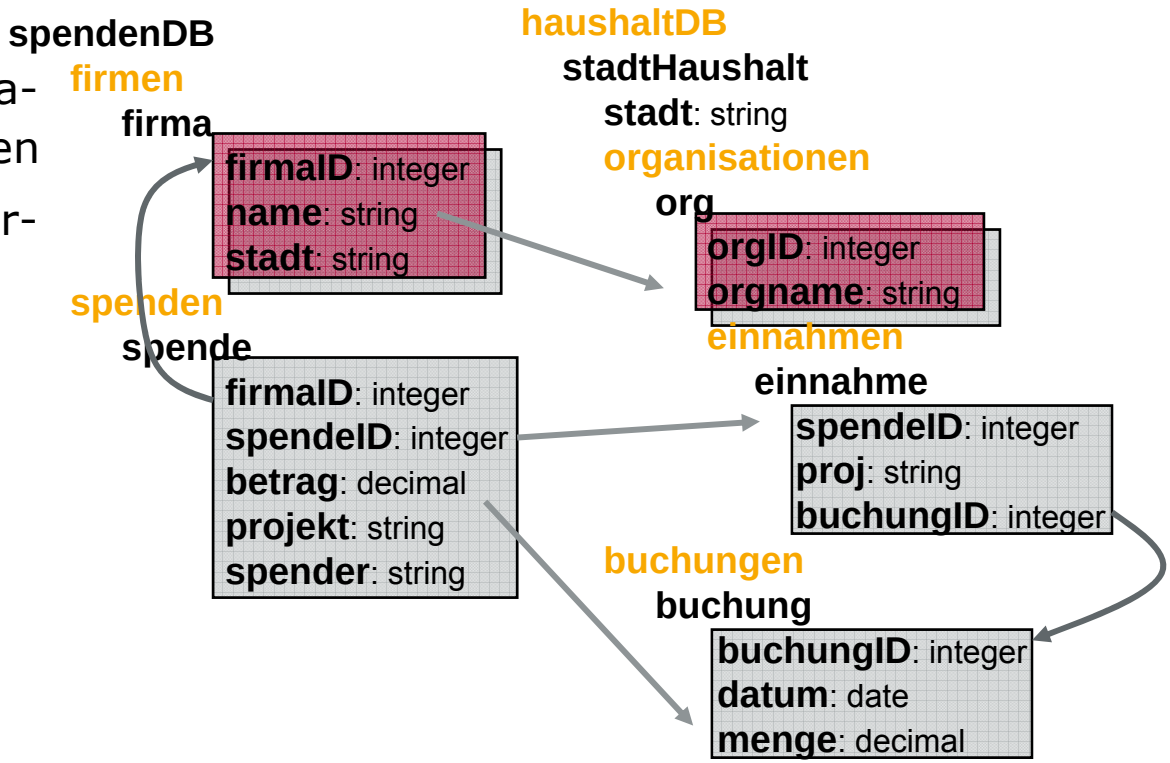


Folie: Frank Legler

Mapping – Algorithmus

Drei Schritte

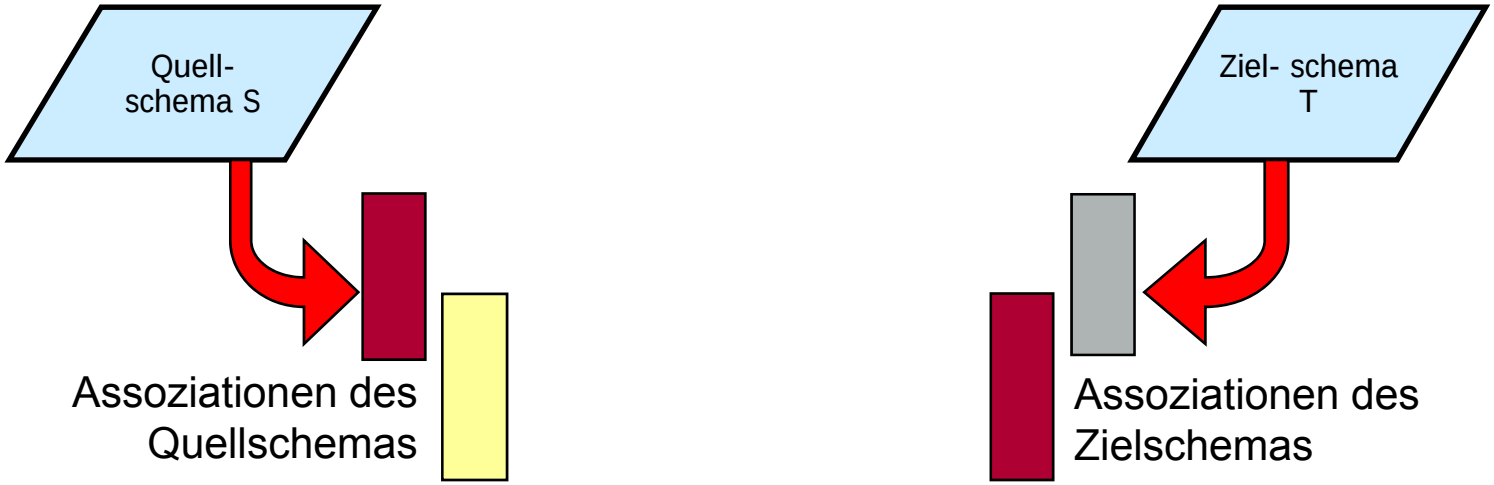
1. Entdeckung von intra-Schema Assoziationen
2. Entdeckung von inter-Schema logischen Mappings
3. Anfrage-erzeugung



Entdeckung von Assoziationen

Schritt 1

- Intra-schema Assoziationen zwischen Schemaelementen
- Relationale Sichten enthalten maximale Gruppen assoziierter Elemente
- Jede Sicht repräsentiert eine eigene „Kategorie" an Daten der Datenquelle
- Unabhängig vom Mapping (aber beschränkt auf „gemappte" Elemente)



Quelle: [FHP+02]

Entdeckung von Assoziationen

Start: Alle „primären" Pfade (*primary paths*)

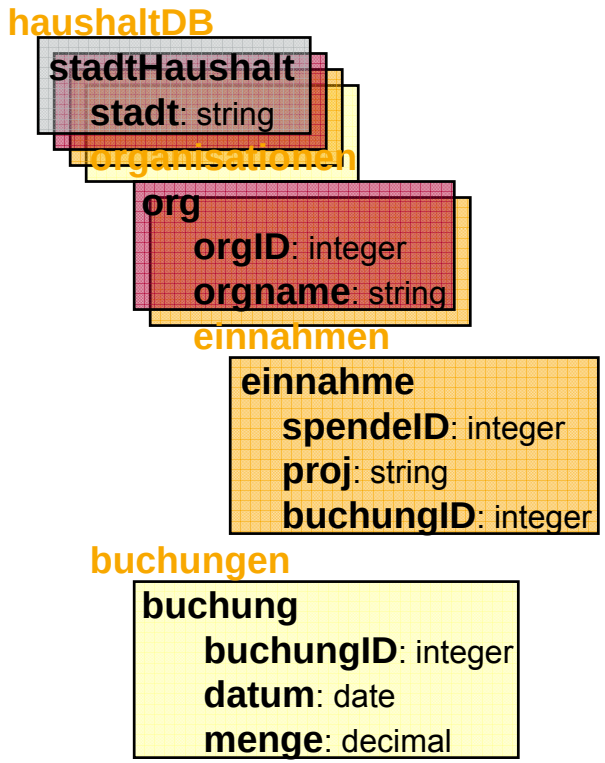
- Assoziationen im Schema ohne Integritätsbedingungen

Relationale Schemas

- Jede Relation entspricht einem primären Pfad

Geschachtelte Schemas

- Attribute einer Ebene
- Attribute geschachtelter Ebenen

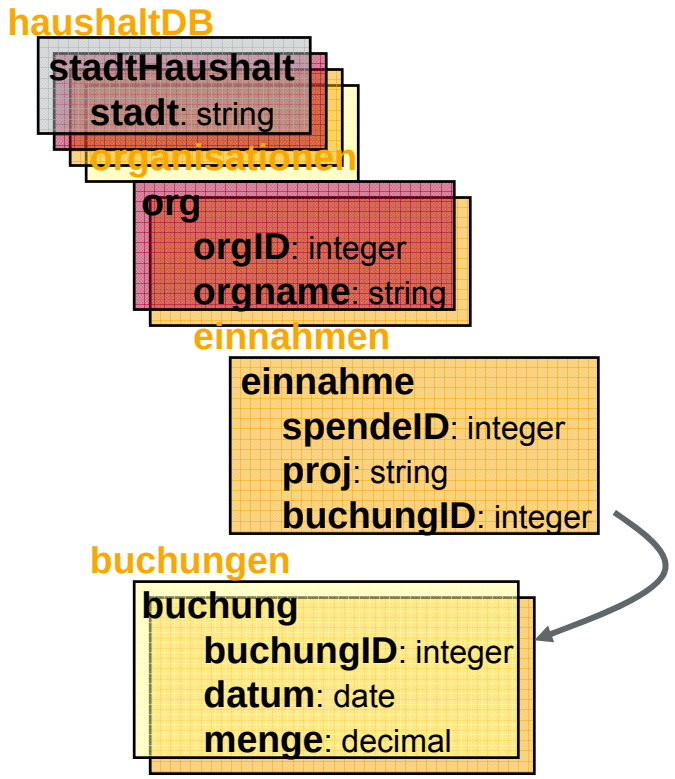


Entdeckung von Assoziationen

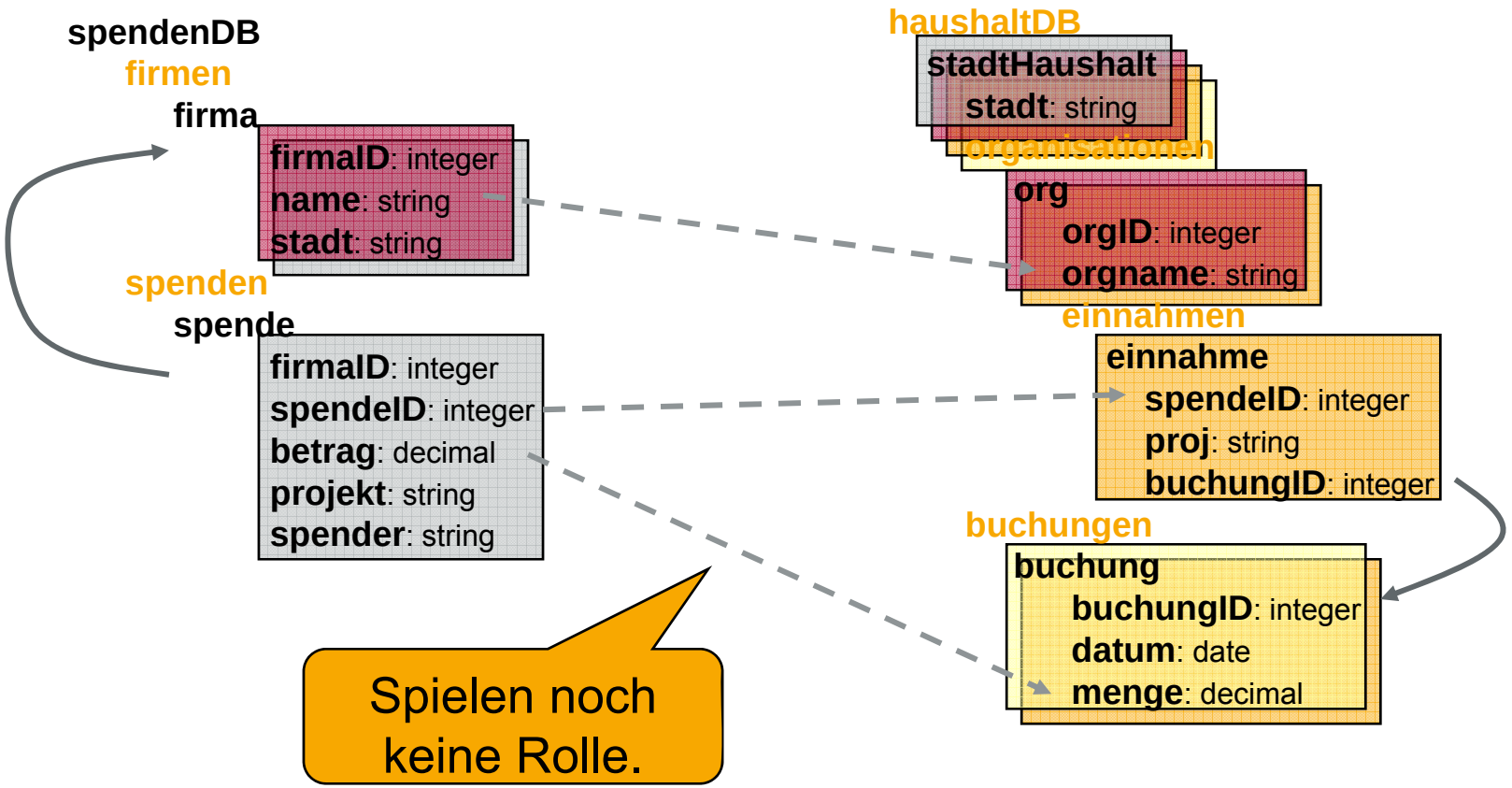
Betrachte nun Schlüssel / Fremdschlüssel (ICs)

Logische Relation

- Erweitere jeden primären Pfad durch „Verfolgen“ der ICs (*chase*)



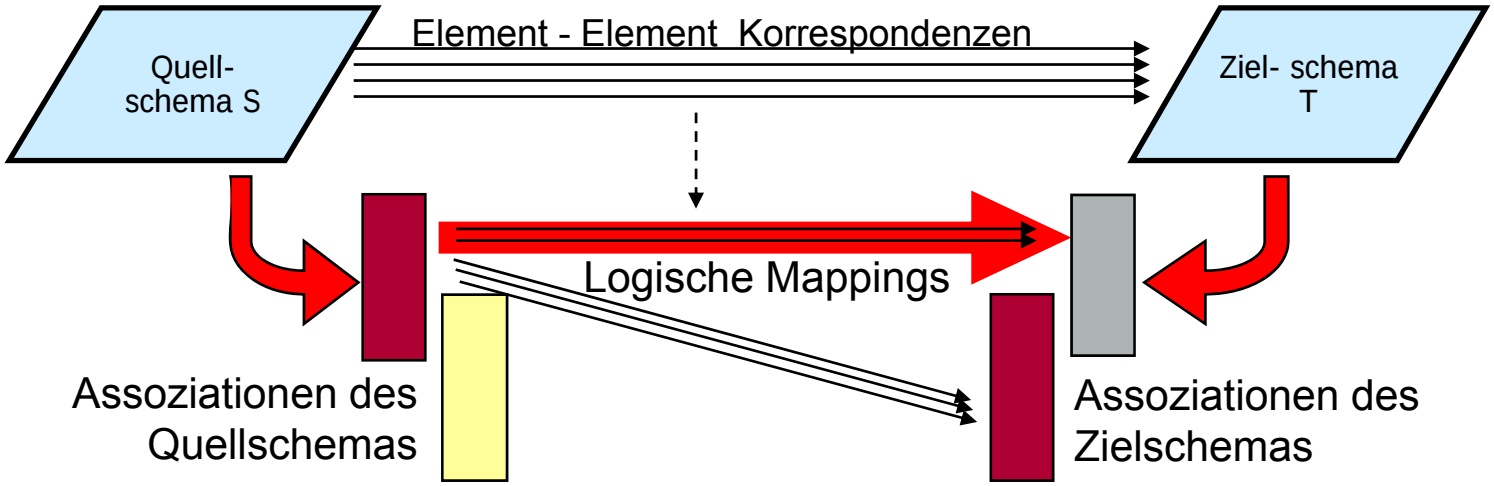
Entdeckung von Assoziationen



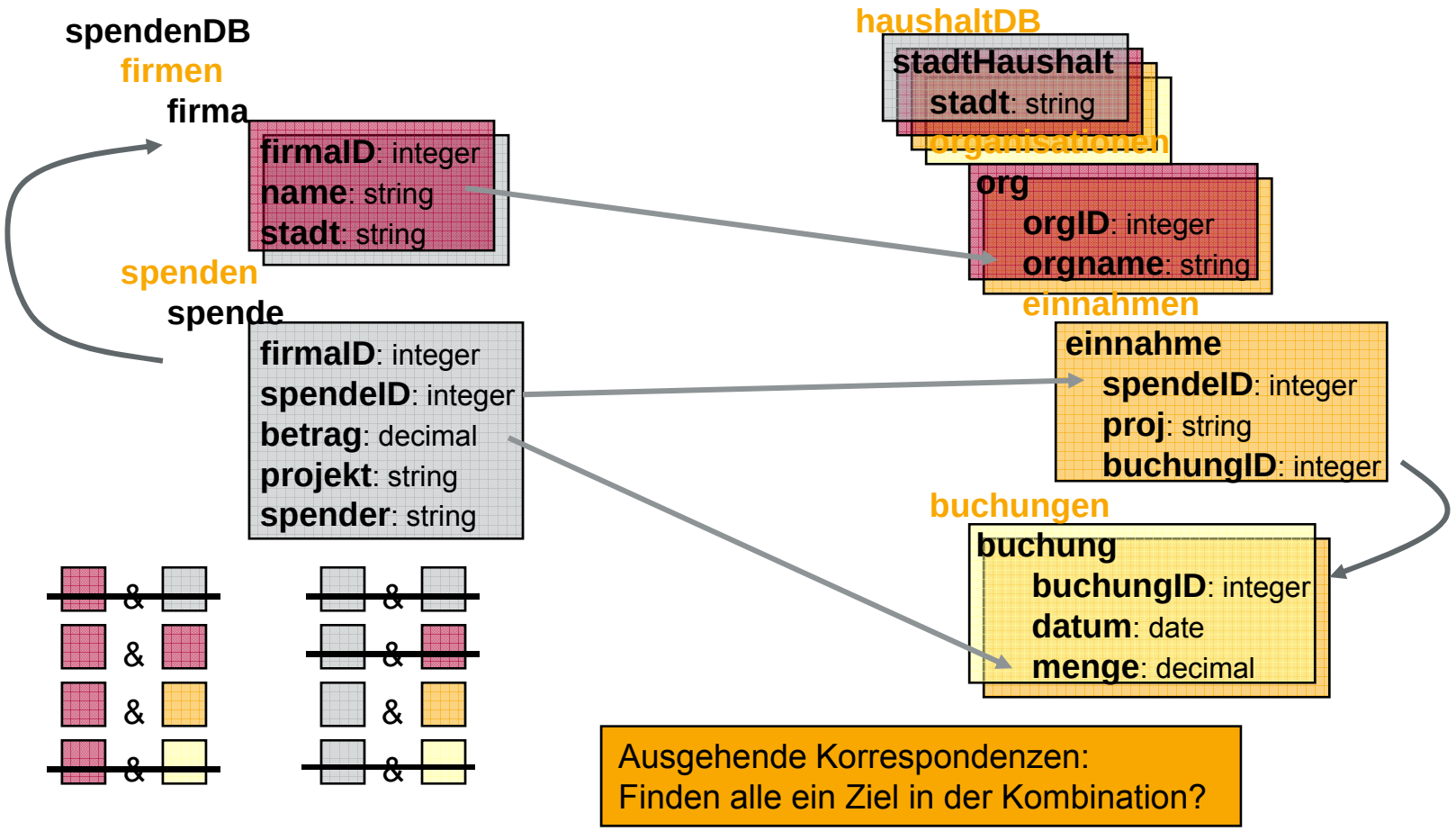
Entdeckung von logischen Mappings

Schritt 2

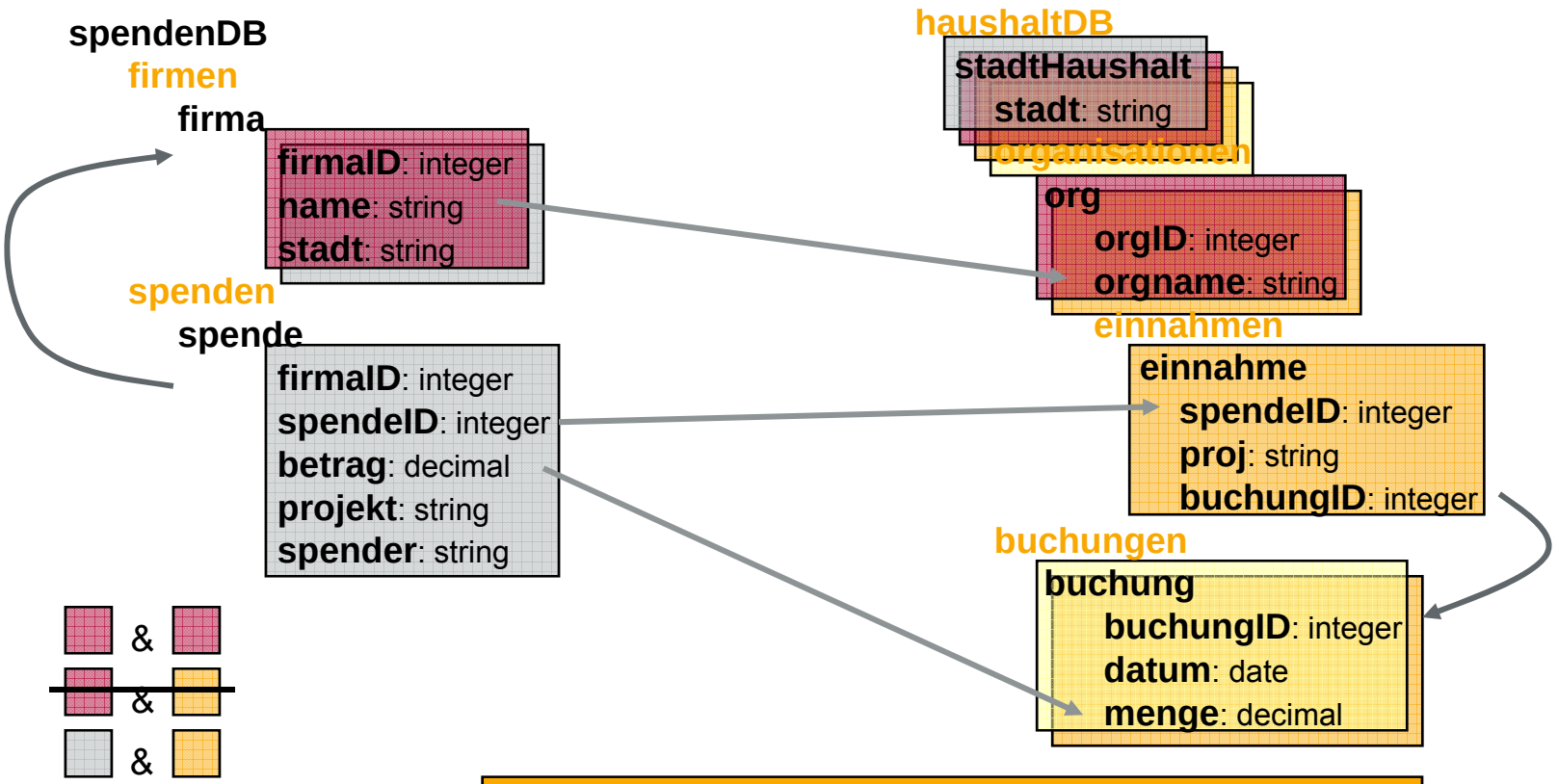
- Entdecke logische Mappings zwischen Quell- und Zielschema
- Betrachte alle Kombinationen aus Assoziationen des Quellschemas und Assoziationen des Zielschemas
  - Unter Berücksichtigung aller Korrespondenzen (sofern Korrespondenzen zwischen ihnen überhaupt existieren)



Entdeckung von logischen Mappings

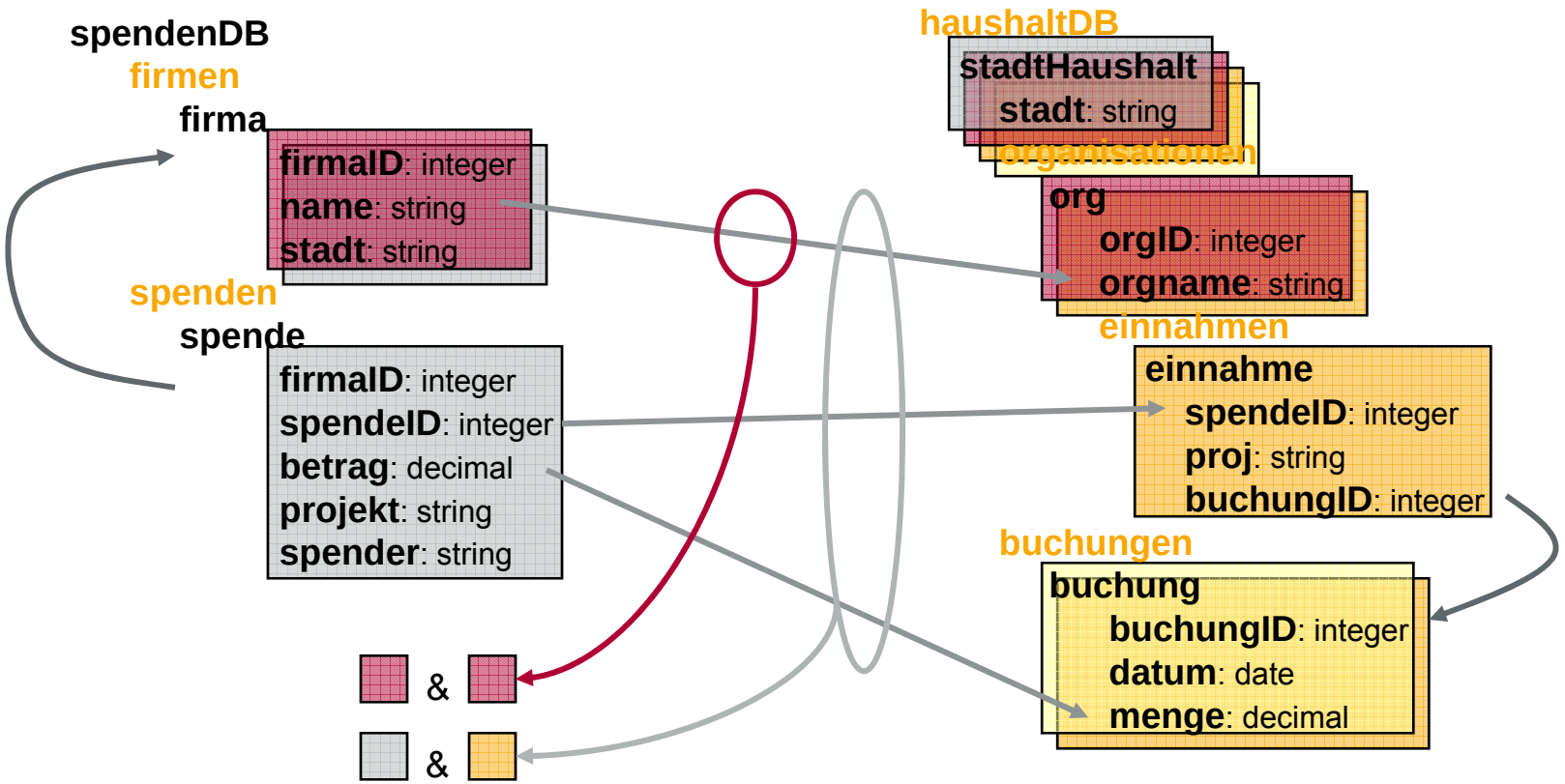


Entdeckung von logischen Mappings

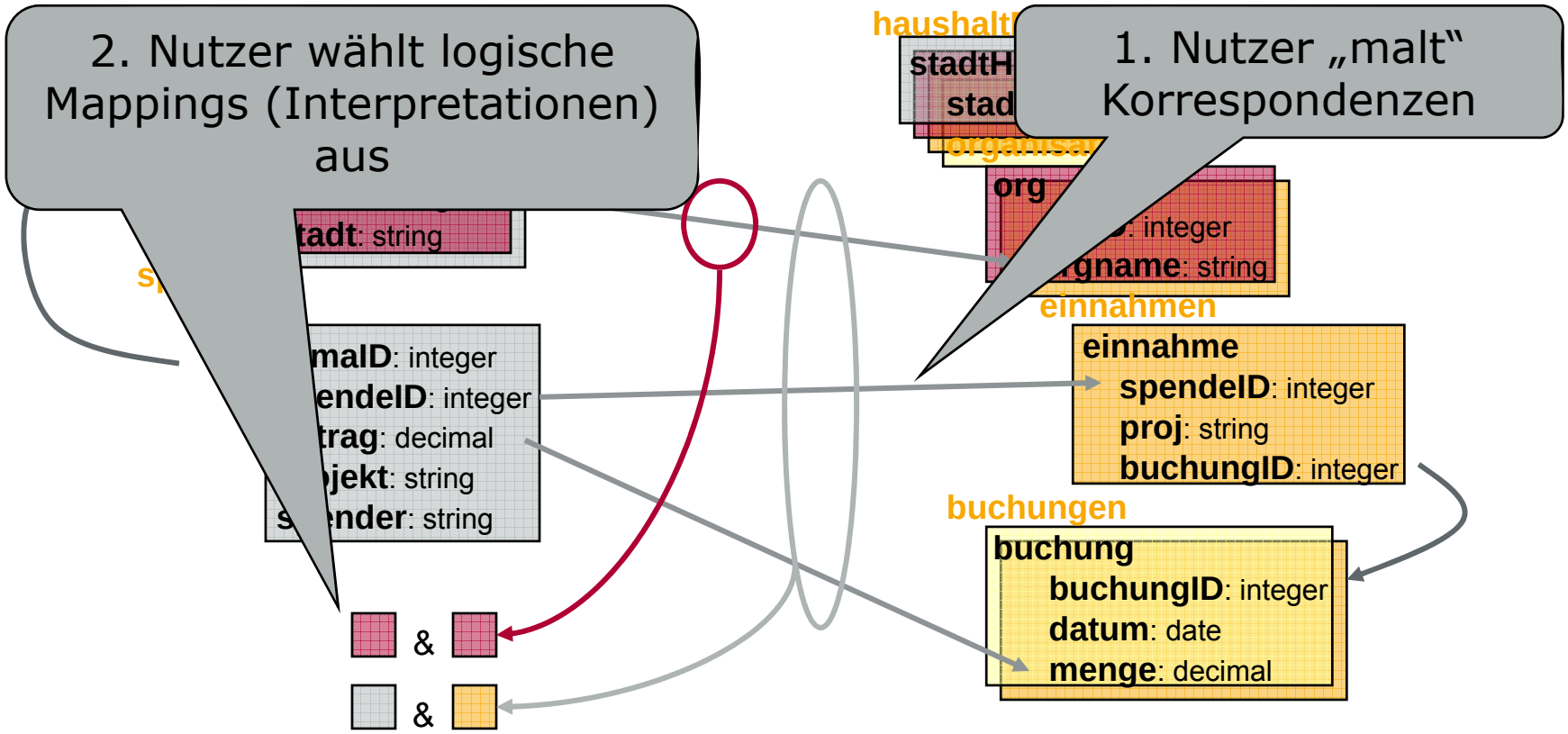


Eingehende Korrespondenzen:
Stammen sie alle aus Assoziationen der Kombination?

Entdeckung von logischen Mappings



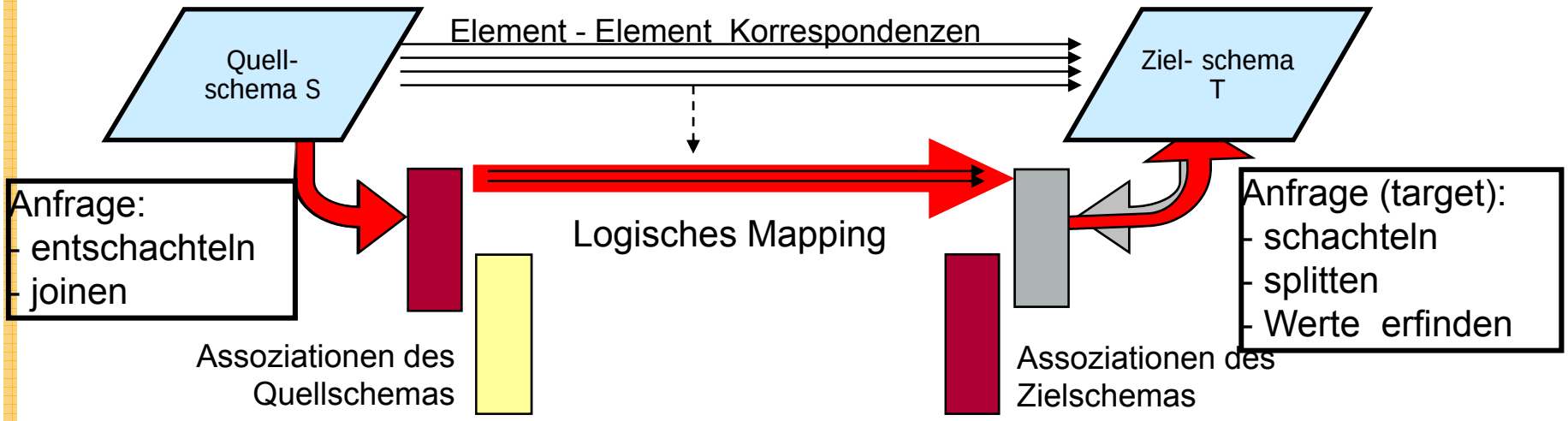
Input des Nutzers bzw. Domänenexperten



Erzeugung der Anfragen

Schritt 3

- Erzeuge für jedes (ausgewählte) logische Mapping eine Anfrage
  - Auswahl und verknüpfen der entsprechenden Quelldaten
  - Generierung der entsprechenden Zieldaten



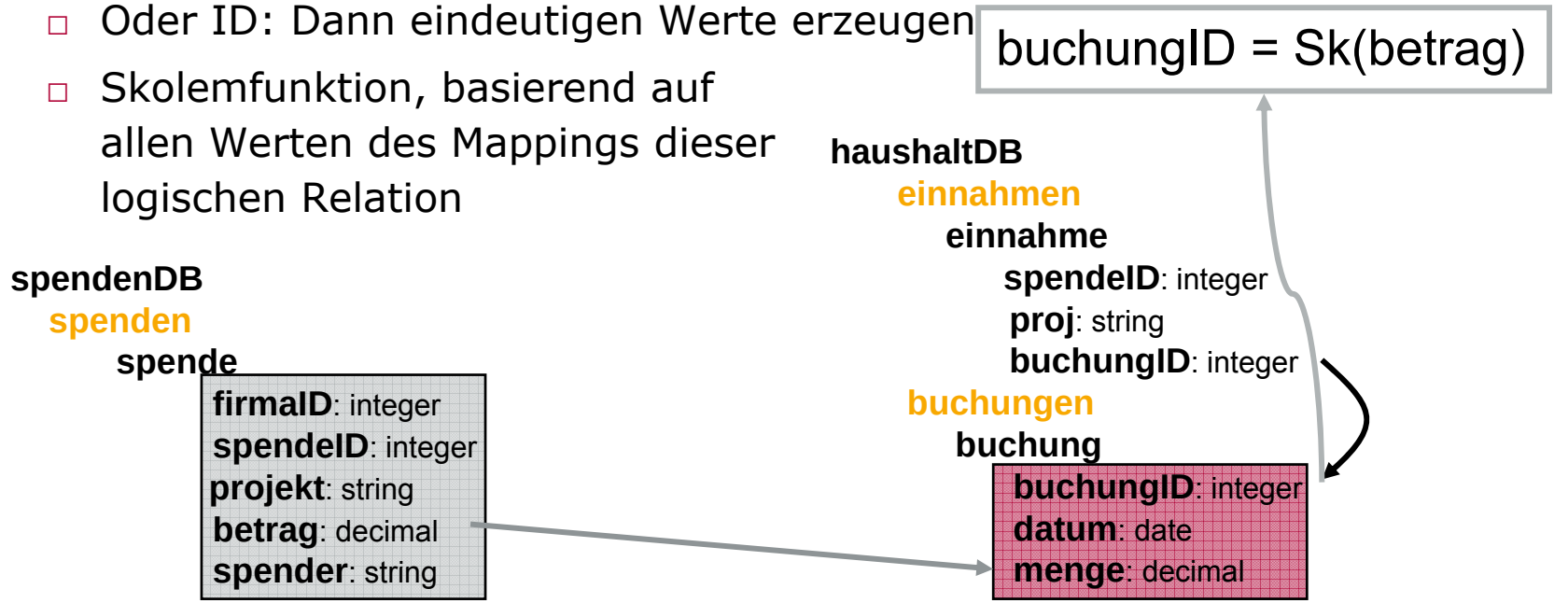
Erzeugung der Anfragen

2 Probleme

- Erfinden von Werten
  - NULL-Werte nicht immer ausreichend.
  - Schlüssel und passende Fremdschlüssel müssen erzeugt werden.
- Schachtelung
  - Es soll nicht eine logische Relation (flach) materialisiert werden, sondern geschachtelte Strukturen.
  - Es muss entsprechend gruppiert werden.

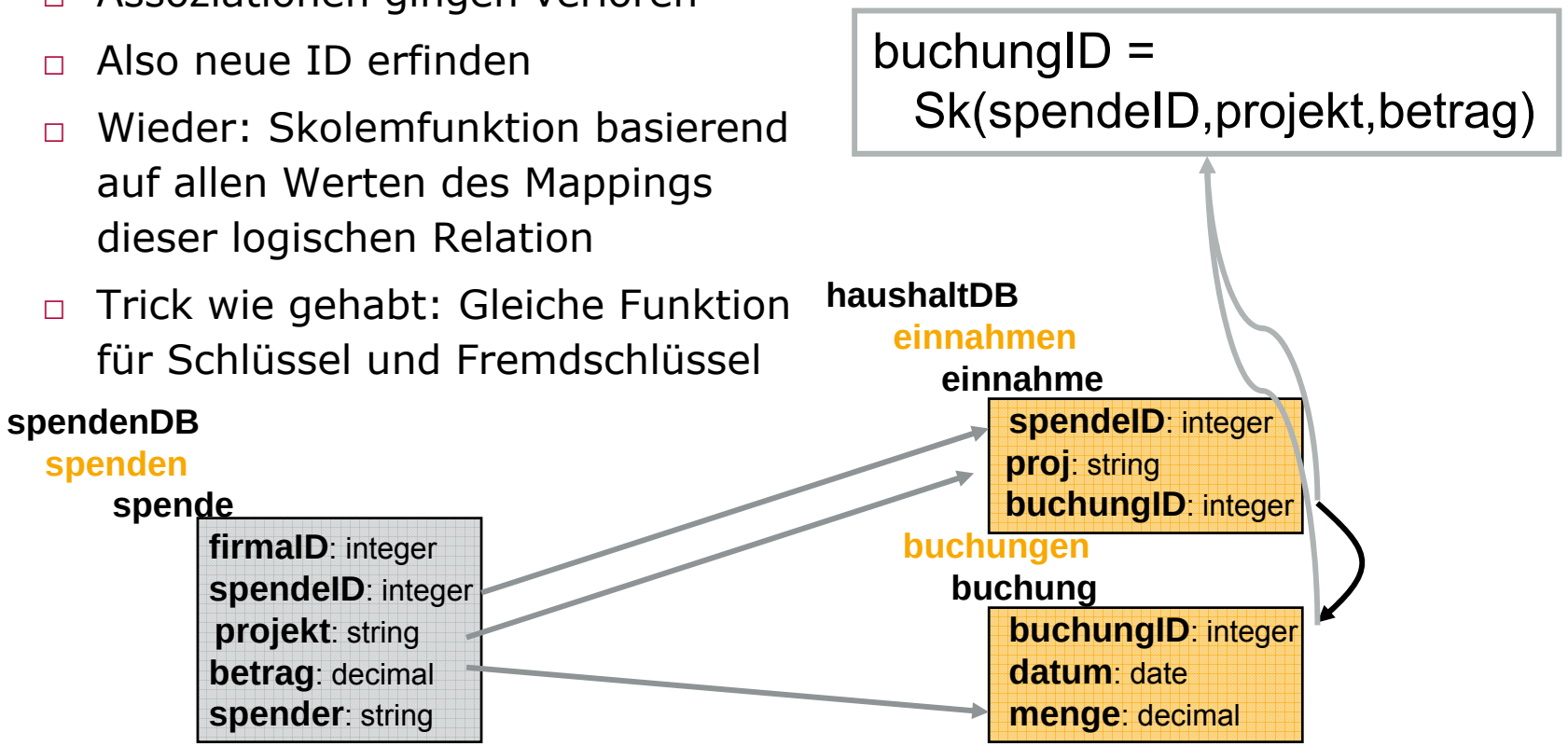
Erfinden neuer Werte (Wdh.)

- Logische Relation: buchungen
- Wert für buchungID wird nicht gefüllt
  - Entweder: Egal
  - Oder: Not-null: Dann Default-Wert, z.B. „null“
  - Oder ID: Dann eindeutigen Werte erzeugen
  - Skolemfunktion, basierend auf allen Werten des Mappings dieser logischen Relation

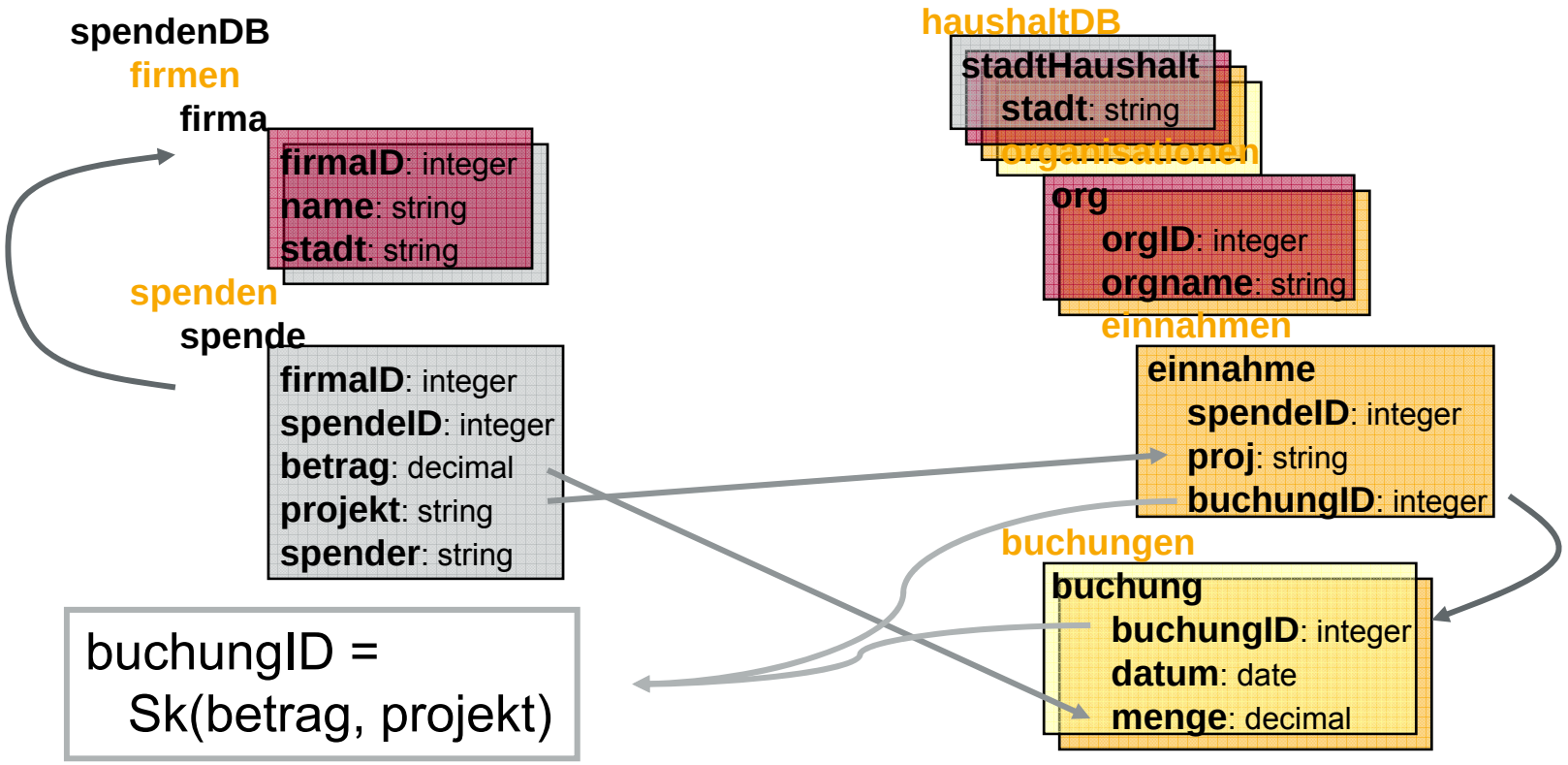


Erfinden neuer Werte

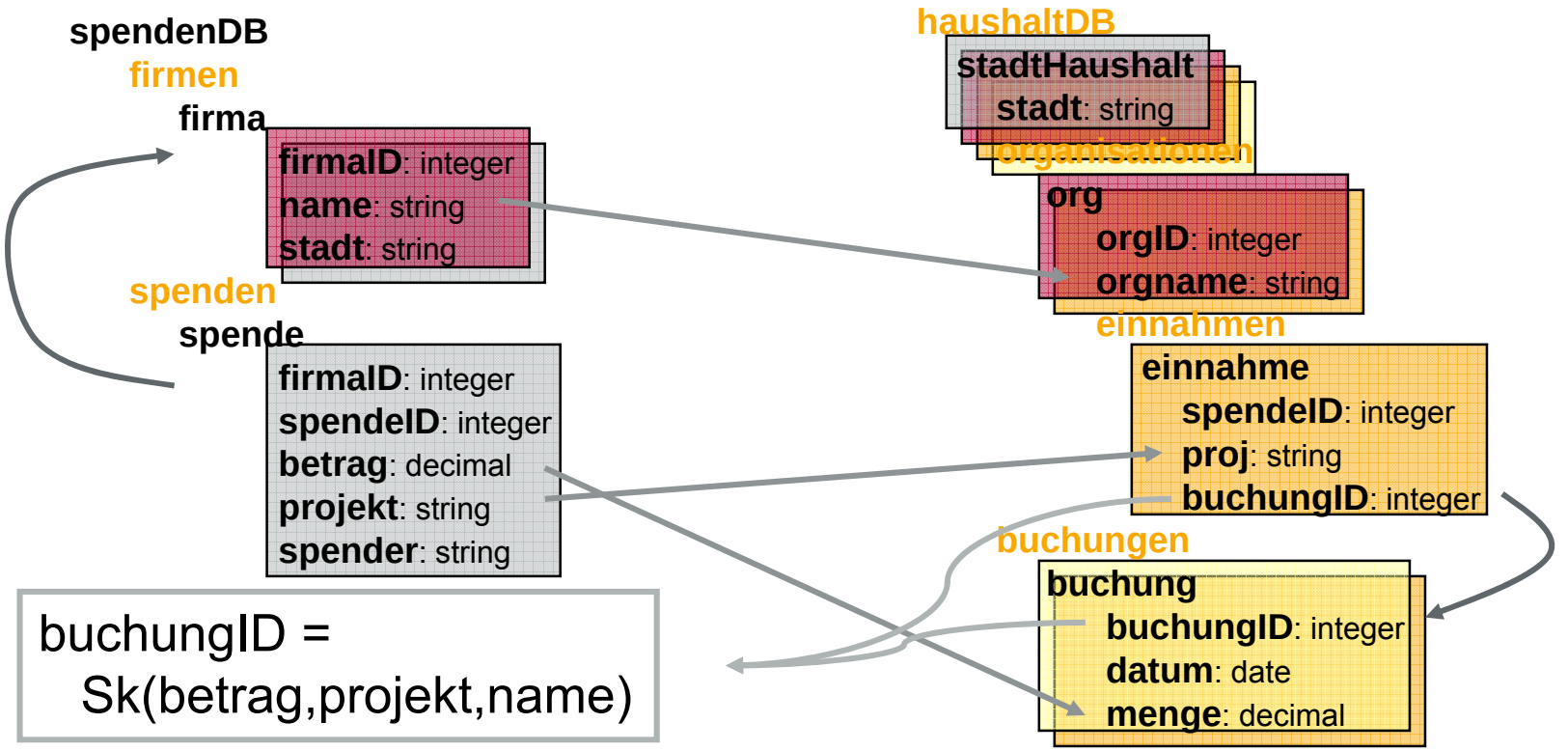
- Logische Relation: einnahmen, buchungen
- buchungID-Attribute haben keine Korrespondenz
  - Assoziationen gingen verloren
  - Also neue ID erfinden
  - Wieder: Skolemfunktion basierend auf allen Werten des Mappings dieser logischen Relation
  - Trick wie gehabt: Gleiche Funktion für Schlüssel und Fremdschlüssel



Erfinden neuer Werte



Erfinden neuer Werte



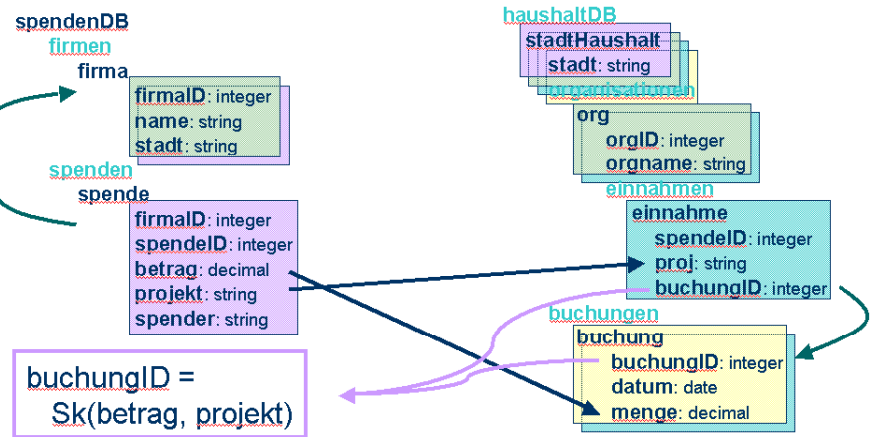
Jetzt erst recht: Warum nicht Sk(betrag,projekt,name,firmaID)?

Erfinden neuer Werte

Gegenfrage: Warum nicht gleiche ALLE ungemappten Werte?

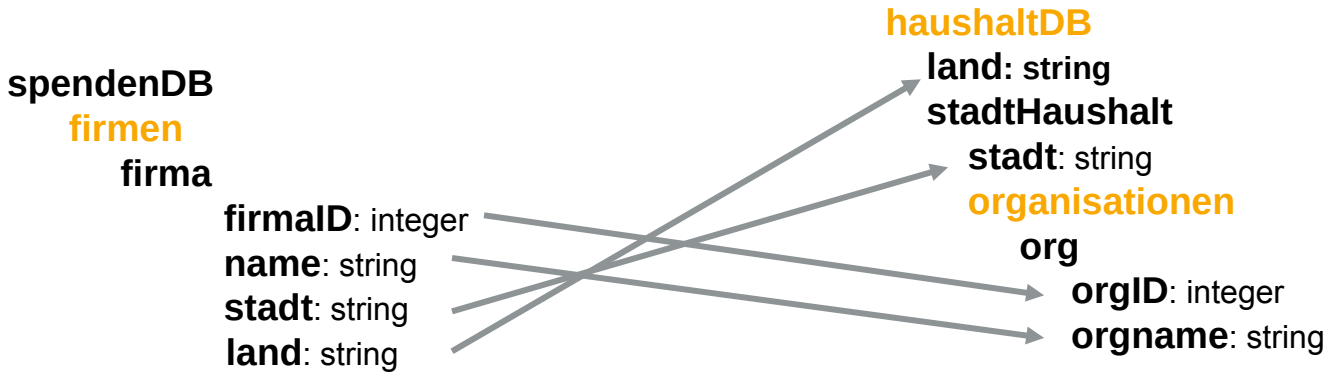
Vier Antworten (von Lucian Popa)

- Prinzipiell ginge das, aber
- „Duplikate" könnten entstehen
  - 2 Spenden, mit gleichem Betrag und gleichem Projekt aber von unterschiedlichen Firmen
- firmenID könnte für Nutzer völlig uninteressant sein.
  - Mapping hätte dann die Rolle einer Projektion, in der Duplikate fehl am Platz wären
  - Sie wären sogar völlig unerklärlich für den Nutzer.
- Minimalität
  - Beide Varianten sind in Ordnung, aber eine ist kleiner.



Gruppierung

- Alle Attribute erhalten Werte
- Aber:
  - Assoziationen könnten verloren gehen.
  - Neue (falsche) Assoziationen könnten erzeugt werden.
- Deshalb: Gruppierung notwendig
- Trick: Virtuelle ID Generierung mittels Skolemfunktion basierend auf allen (gemapten) Werten hierarchisch über der aktuellen Relation.
  - Im Beispiel: Jedes Tupel haushaltDB erhält ID Sk(land, stadt)
  - Jedes weitere Tupel aus firmen mit gleichen Werten für stadt und land errechnet gleiche ID und wird unter gleichem Element geschachtelt.



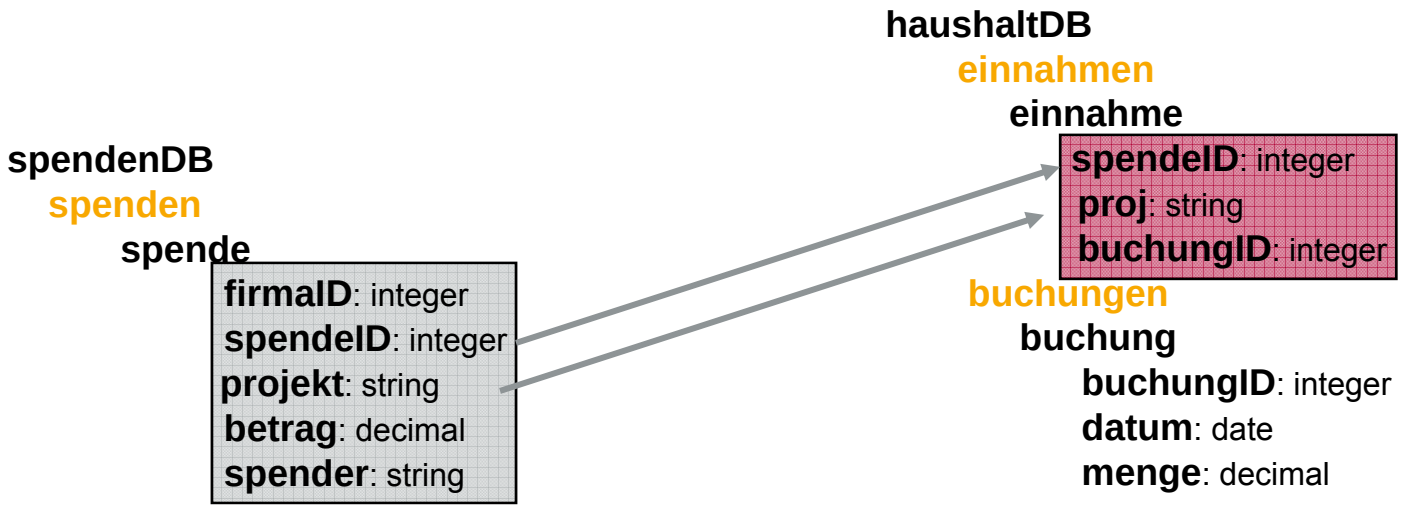
Erzeugung der Anfragen

- Implementationsspezifisch
- Abhängig von Datenmodell
  - Relational → Relational: SQL
  - Relational → XML: SQL/XML
    - Schachtelung und tagging des Ergebnisses
  - XML → Relational: XQuery oder XSLT
    - Tags weglassen
  - XML → XML: XQuery oder XSLT
- In Clio
  - Erzeugung proprietärer Regeln
  - Dann: Übersetzung der Regeln in jeweilige Anfragesprache

Erzeugung der Anfragen

Erzeugung proprietärer Regeln in Clio

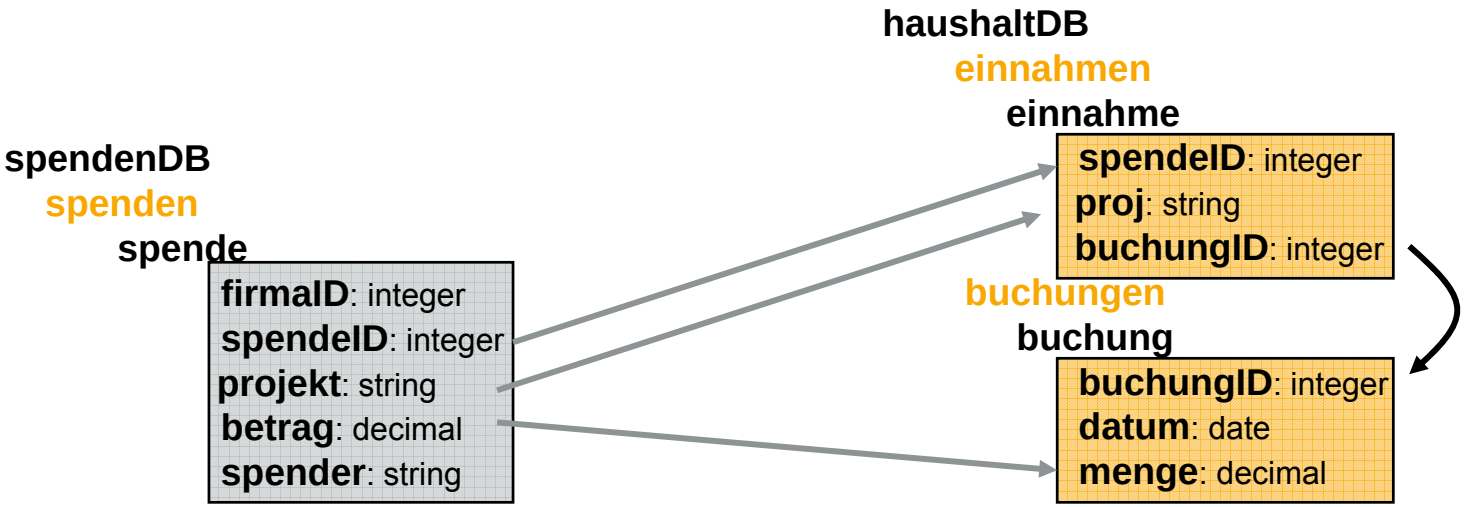
- for x in spenden
  let a = x.spende.spendeID, b = x.spende.projekt
  return <einnahme = <spendeID = a, proj = b, buchungID = null>>
  in einnahmen



Erzeugung der Anfragen

Erzeugung proprietärer Regeln in Clio

- for x in spenden
  let a = x.spende.spendeID, b = x.spende.projekt, c = x.spende.betrag
  return <einnahme = <spendeID = a, proj = b, buchungID = Sk(a,b,c)>>
  in einnahmen,
  <buchung = <buchingID = Sk(a,b,c), datum = null, menge = c>>
  in buchungen



Erzeugung der Anfragen – XQuery

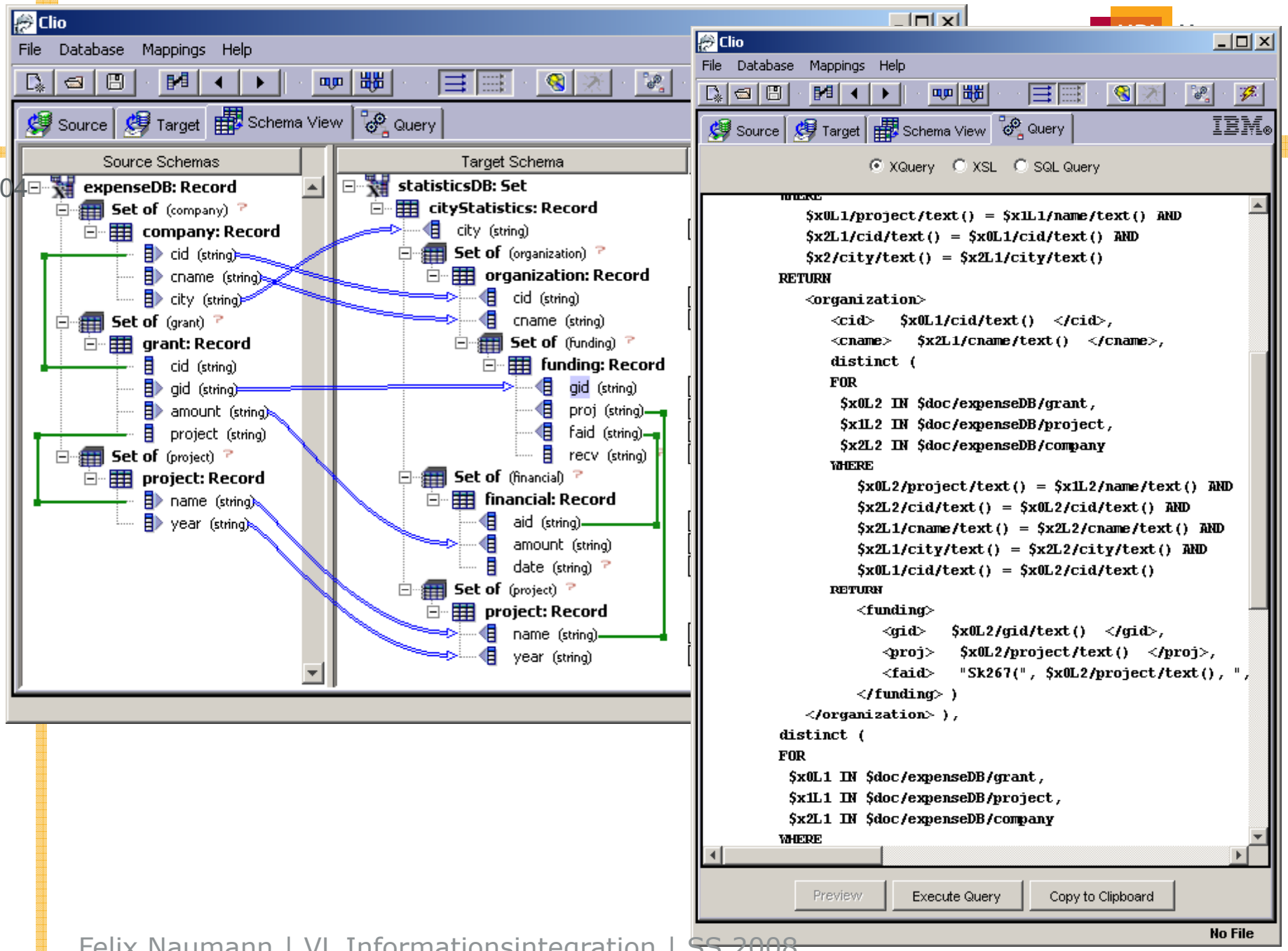

```
for     x in spenden
let     a = x.spende.spendeID, b = x.spende.projekt, c = x.spende.betrag
return <einnahme = <spendeID = a, proj = b, buchungID = Sk(a,b,c)>> in einnahmen,
       <buchung = <buchingID = Sk(a,b,c), datum = null, menge = c>> in buchungen
```

```
LET $doc0 := document("input XML file goes here")
RETURN <haushaltDB>   {
  distinct-values (
  FOR   $x0 IN $doc0/spendenDB/spende
  RETURN      <einnahme>
       <spendeID>  {  $x0/spendeID/text() }  </spendeID>
       <proj>  {  $x0/projekt/text() }  </proj>
       <buchungID>  {  "Sk32(", $x0/betrag/text(), ", ", $x0/spendeID/text(), ", ", $x0/projekt/text(), ")" }  </buchungID>
     </einnahme> )   }
  {   distinct-values (
  FOR     $x0 IN $doc0/spendenDB/spende
  RETURN      <buchung>
       <buchungID>  {  "Sk32(", $x0/betrag/text(), ", ", $x0/spendeID/text(), ", ", $x0/projekt/text(), ")" }  </buchungID>
       <menge>  {  $x0/betrag/text() }  </menge>
     </buchung> )   }
</haushaltDB>
```

Erzeugung der Anfragen – SQL


```
for     x in spenden
let     a = x.spende.spendeID, b = x.spende.projekt, c = x.spende.betrag
return <einnahme = <spendeID = a, proj = b, buchungID = Sk(a,b,c)>> in einnahmen,
       <buchung = <buchungID = Sk(a,b,c), datum = null, menge = c>> in buchungen
```

```
-- CREATE VIEW einnahme AS
SELECT
        x0.spendeID AS spendeID,
        x0.projekt AS proj,
        RTRIM('Sk32(` CHAR(x0.betrag) || ','  || (CHAR(x0.spendeID) || ','  || CHAR(x0.projekt) || ')') AS buchungID
FROM
        spendenDB.spende x0
-------------------------------------------------
-- CREATE VIEW buchung AS
SELECT
        RTRIM('Sk32(' || CHAR(x0.betrag) || ','  || CHAR(x0.spendeID) || ','  || CHAR(x0.projekt) || ')') AS buchungID,
        x0.betrag AS menge
FROM
        spendenDB.spende x0
```


Clio
File Database Mappings Help
Source Target Schema View Query
Source Schemas
expenseDB: Record
Set of (company)
company: Record
cid (string)
cname (string)
city (string)
Set of (grant)
grant: Record
cid (string)
gid (string)
amount (string)
project (string)
Set of (project)
project: Record
name (string)
year (string)
Target Schema
statisticsDB: Set
cityStatistics: Record
city (string)
Set of (organization)
organization: Record
cid (string)
cname (string)
Set of (funding)
funding: Record
gid (string)
proj (string)
faid (string)
recv (string)
Set of (financial)
financial: Record
aid (string)
amount (string)
date (string)
Set of (project)
project: Record
name (string)
year (string)
Clio
File Database Mappings Help
Source Target Schema View Query
IBM
XQuery XSL SQL Query
$x0L1/project/text() = $x1L1/name/text() AND
$x2L1/cid/text() = $x0L1/cid/text() AND
$x2/city/text() = $x2L1/city/text()
RETURN
<organization>
<cid> $x0L1/cid/text() </cid>,
<cname> $x2L1/cname/text() </cname>,
distinct (
FOR
$x0L2 IN $doc/expenseDB/grant,
$x1L2 IN $doc/expenseDB/project,
$x2L2 IN $doc/expenseDB/company
WHERE
$x0L2/project/text() = $x1L2/name/text() AND
$x2L2/cid/text() = $x0L2/cid/text() AND
$x2L1/cname/text() = $x2L2/cname/text() AND
$x2L1/city/text() = $x2L2/city/text() AND
$x0L1/cid/text() = $x0L2/cid/text()
RETURN
<funding>
<gid> $x0L2/gid/text() </gid>,
<proj> $x0L2/project/text() </proj>,
<faid> "Sk267(", $x0L2/project/text(), ",
</funding>)
</organization>),
distinct (
FOR
$x0L1 IN $doc/expenseDB/grant,
$x1L1 IN $doc/expenseDB/project,
$x2L1 IN $doc/expenseDB/company
WHERE
Preview
Execute Query
Copy to Clipboard
No File

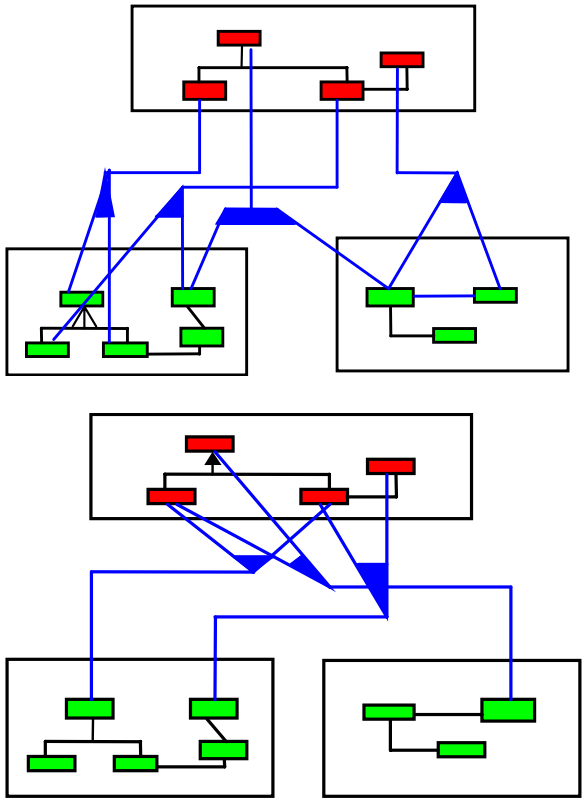
Vorschau:
Global as View / Local as View

Global as View (GaV)

- Relationen des globalen Schemas werden als Sichten auf die lokalen Schemas der Quellen ausgedrückt.

Local as View (LaV)

- Relationen der Schemas der Quellen werden als Sichten auf das globale Schema ausgedrückt.



Einbettung in GaV/LaV

Ergebnis des Schema Mapping Prozess sind Anfragen

- Eine oder mehrere Anfragen pro Assoziation

Relationales Modell:

- Jede Ziel-Relation entspricht einem primären Pfad
- Verwerfe Fremdschlüssel
- Jede Ziel-Relation entspricht einer Assoziation
- Jede Anfrage (oder Vereinigung von Anfrage) liefert Daten für eine Relation, also wie GaV.

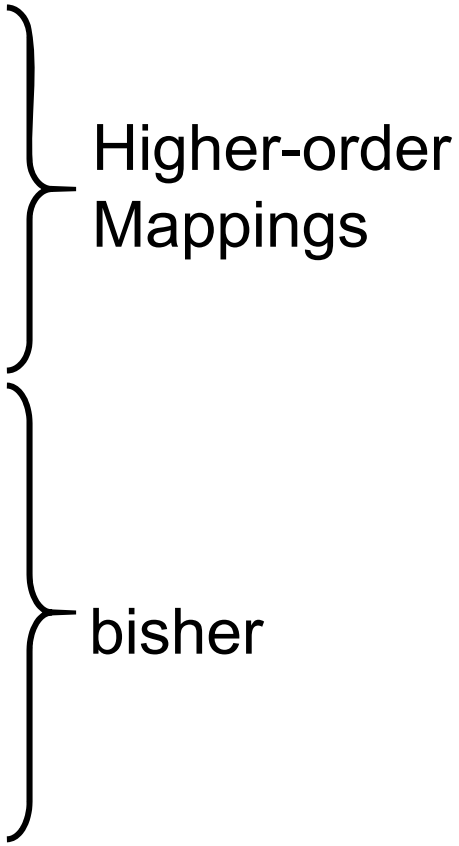
Schema Mapping leistet aber mehr!

- Erzeugung von Daten für mehrere Zielrelationen zugleich
- Unter Berücksichtigung von Integritätsbedingungen in Quell- und Zielschema
- „Kombination von LaV und GaV": GLaV

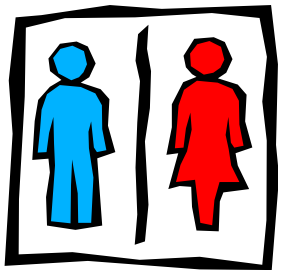
Schematische Heterogenität

Struktur

- Modellierung
  - Relation vs. Attribut
  - Attribut vs. Wert
  - Relation vs. Wert
- Benennung
  - Relationen
  - Attribute
- Normalisiert vs. Denormalisiert
- Geschachtelt vs. Fremdschlüssel
- Geschachtelt vs. Flach



High-order Mappings



Männer(Id, Vorname, Nachname)
Frauen(Id, Vorname, Nachname)

Relation vs. Attribut

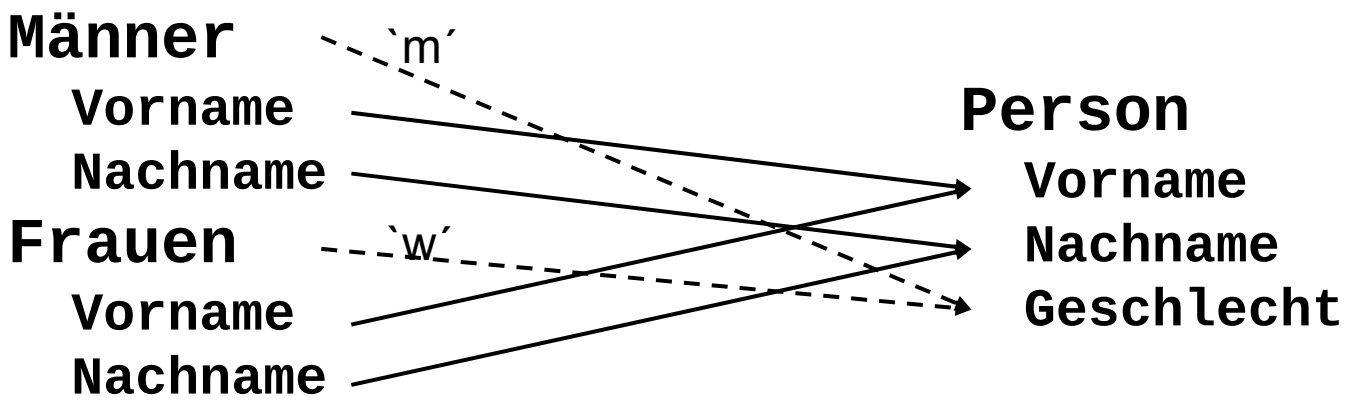
Person(Id, Vorname, Nachname,männlich, weiblich)

Relation vs. Wert

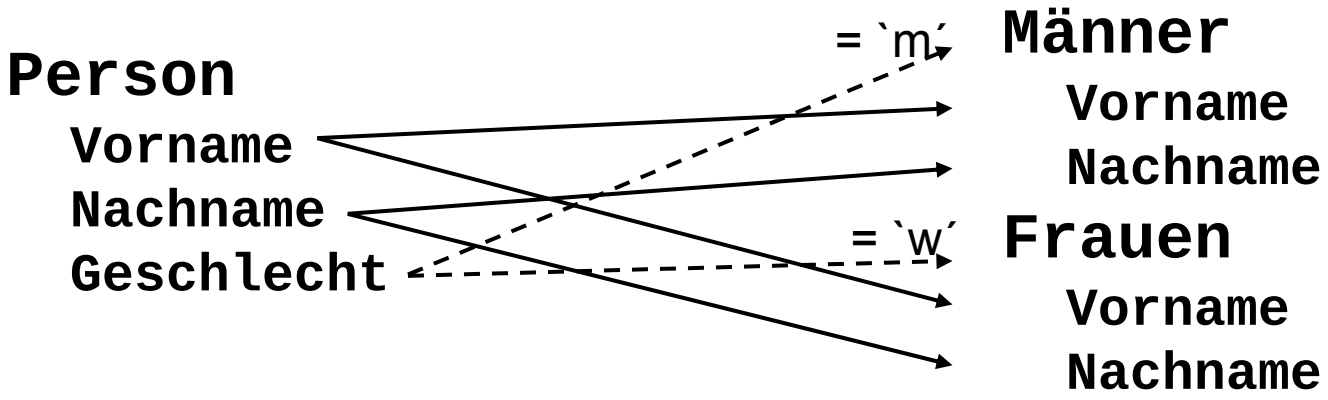
Attribut vs. Wert

Person(Id, Vorname, Nachname, Geschlecht)

High-order Mappings



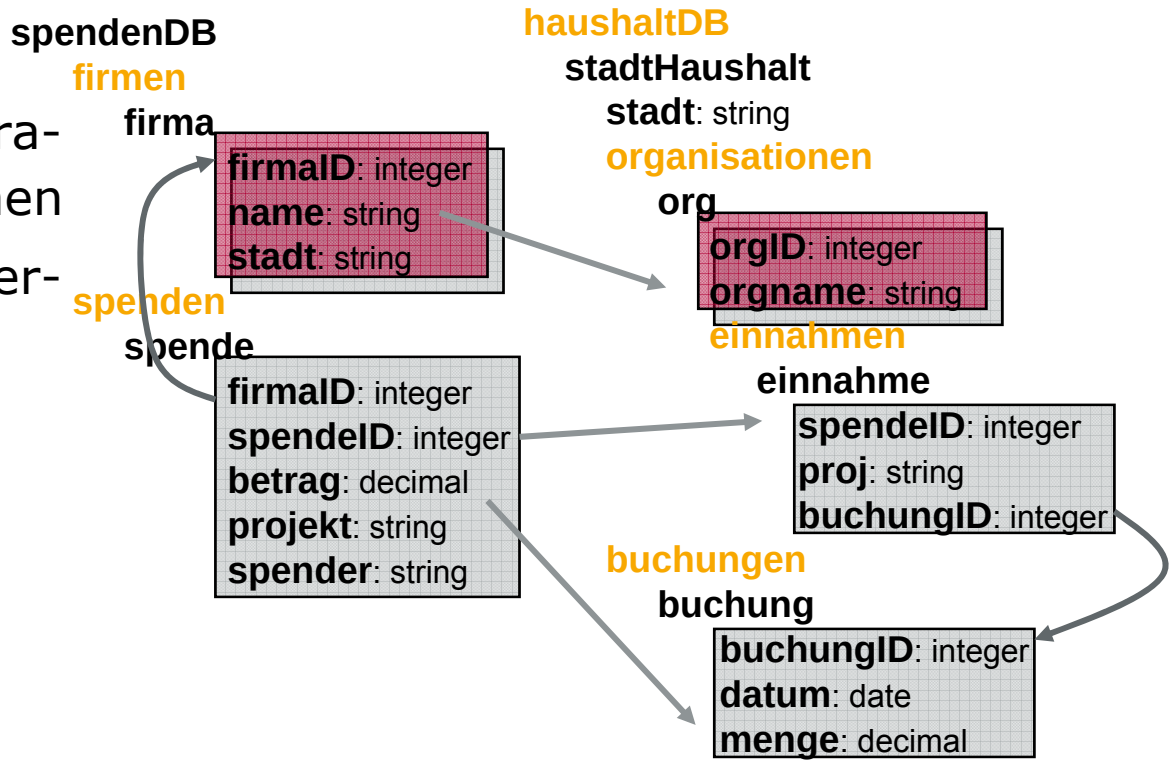
High-order Mappings



Zusammenfassung

Drei Schritte

1. Entdeckung von intra-Schema Assoziationen
2. Entdeckung von inter-Schema logischen Mappings
3. Anfrage-erzeugung



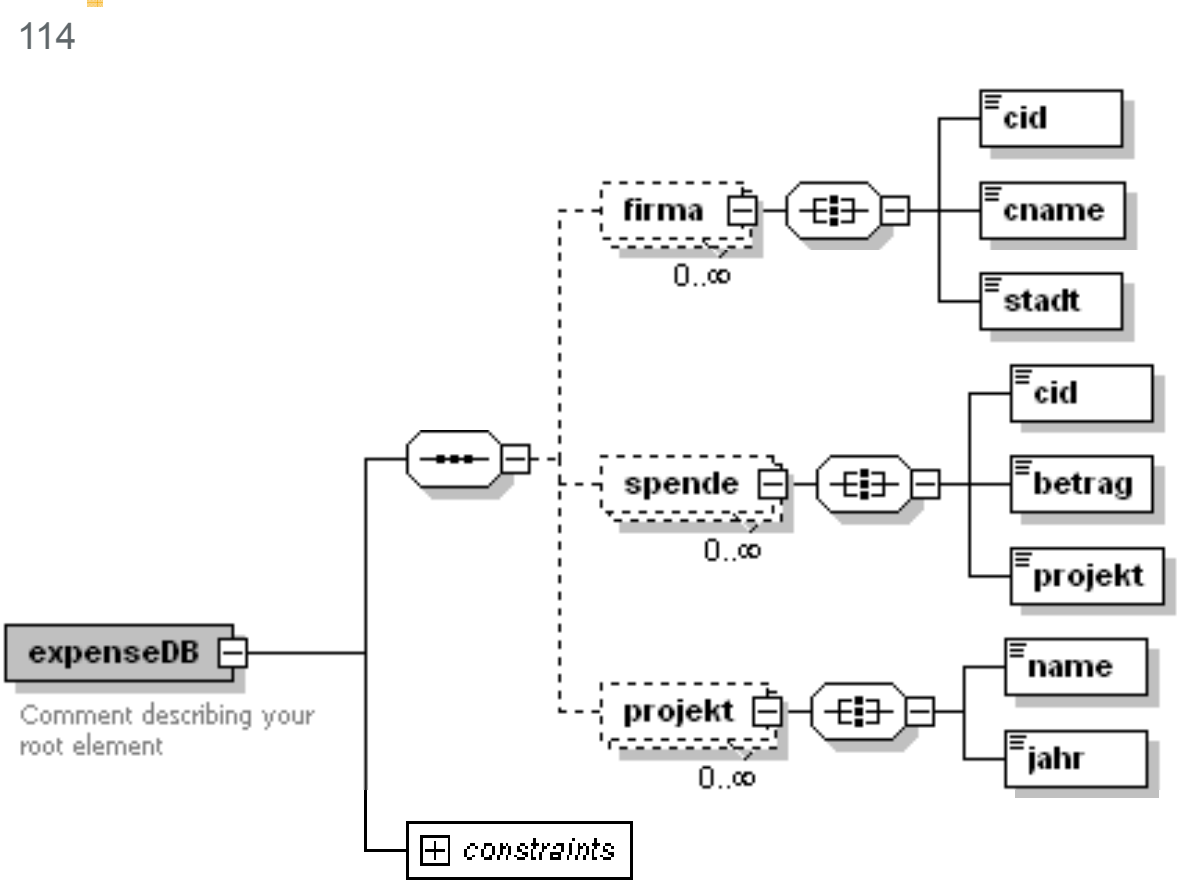
Überblick

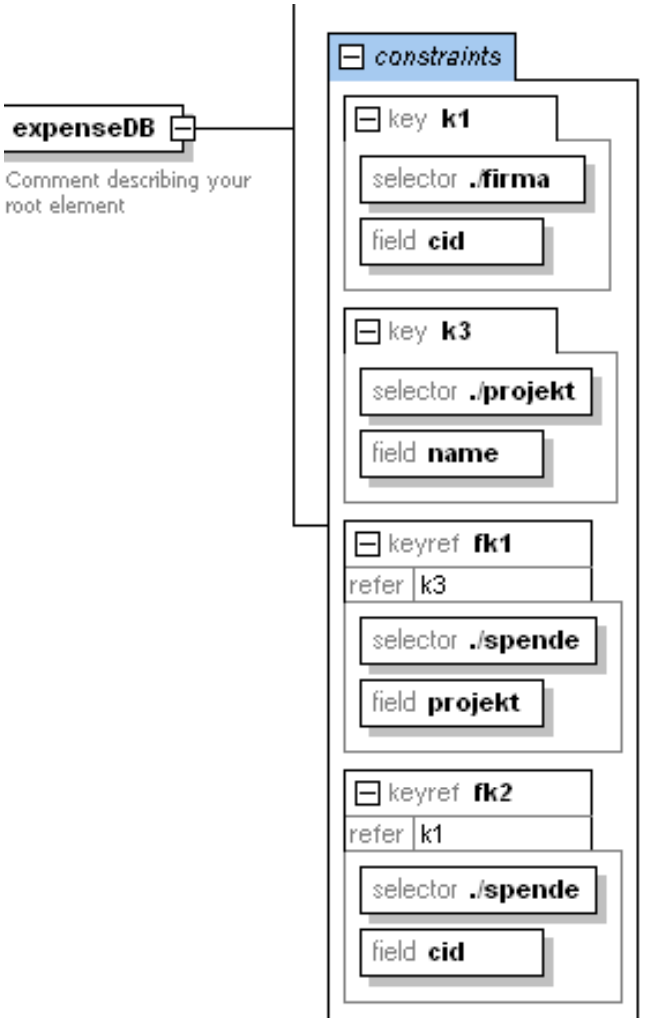
- Motivation
- Schema Mapping
- Schema Matching
- Mapping Interpretation
  - Aus [FHP+02] und VLDB 2002 Vortragsfolien
- Mapping Werkzeuge

Vergleich dreier Tools

- Altovas MapForce
- IBMs Clio
- [IBMs WSAD]
- Szenario
  - expenseDB -> statisticsDB
  - Definition des Mappings
  - Generierung der Transformation
  - Vergleich der XML Ergebnisse
- XML Screenshots aus XMLSpy
  - www.xmlspy.com

Szenario - ExpenseDB





Szenario - ExpenseDB

expenseDB

= xmlns:xsi	http://www.w3.org/2001/XMLSchema-instance
= xsi:noNamespac...	workspace\Clio\schemas\expenseDB.xsd

firma (5)

	cid	cname	stadt
1	GG	Garlic	Gilroy
2	CMPQ	Compaq	San Jose
3	NA	Network Associates	San Jose
4	IBM	International Business Machines	San Jose
5	MSFT	Microsoft	Redmond

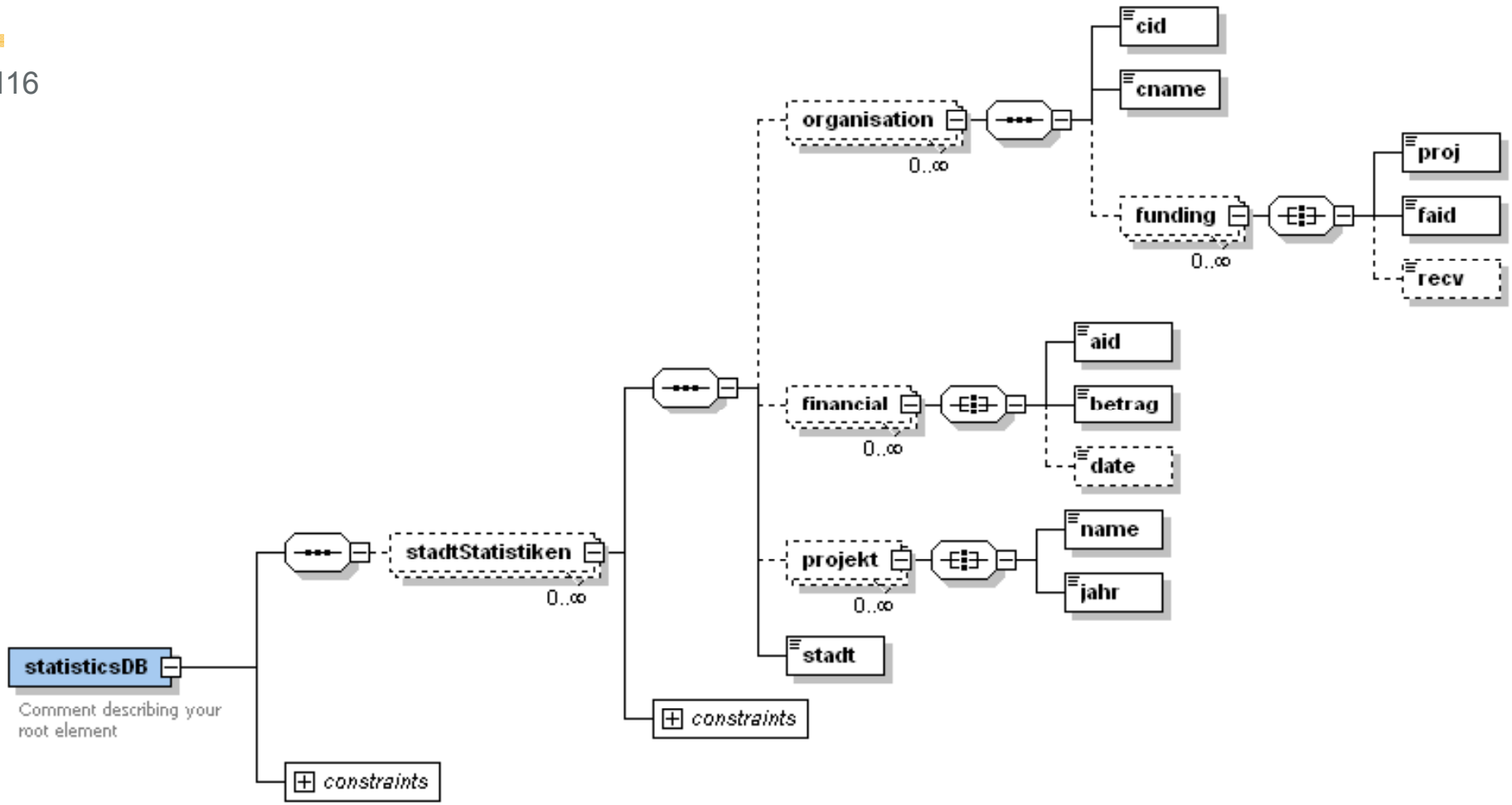
spende (6)

	cid	betrag	projekt
1	MSFT	12350	SQLServer
2	IBM	3500	Clio
3	IBM	25000	DB2
4	NA	5000	FastNetwork
5	MSFT	500	Leo
6	IBM	10000	Clio

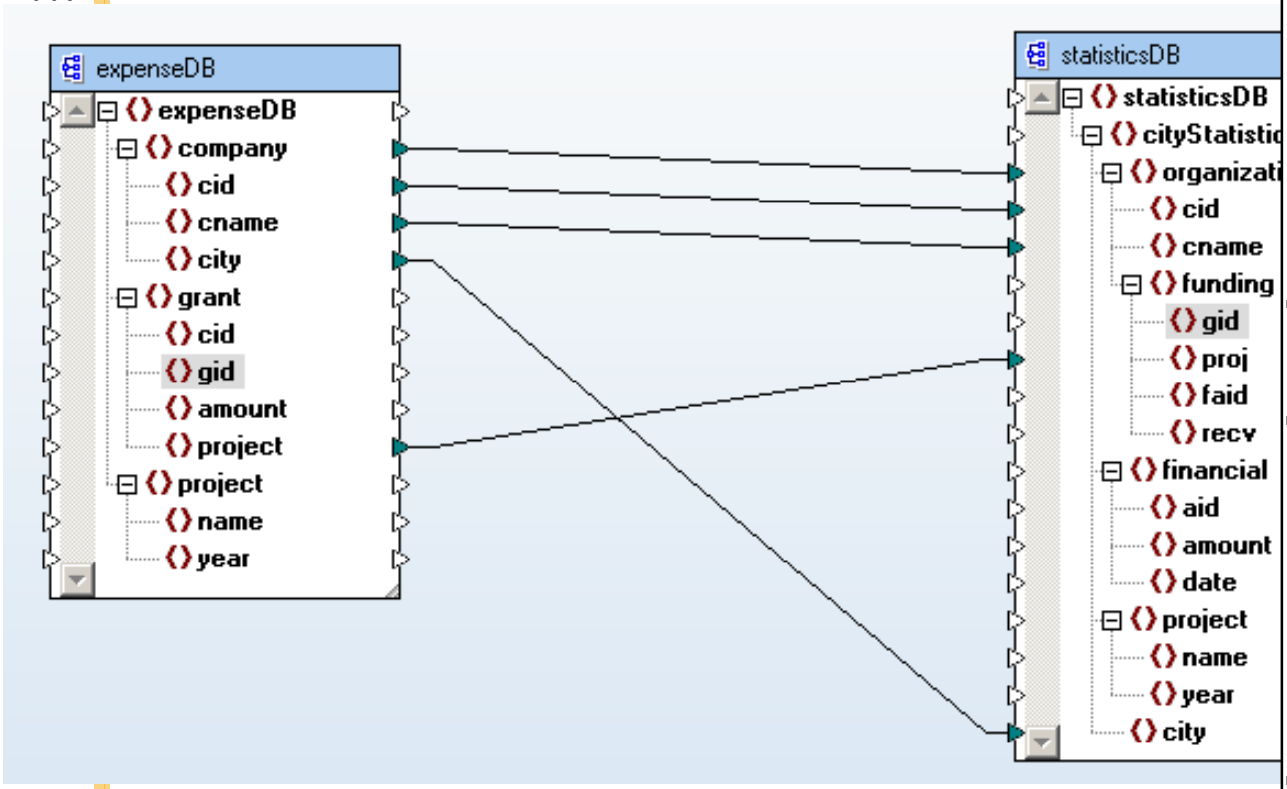
projekt (5)

	name	jahr
1	SQLServer	1997
2	FastNetwork	1996
3	DB2	1987
4	Clio	1999
5	Leo	2000

Szenario StatisticsDB

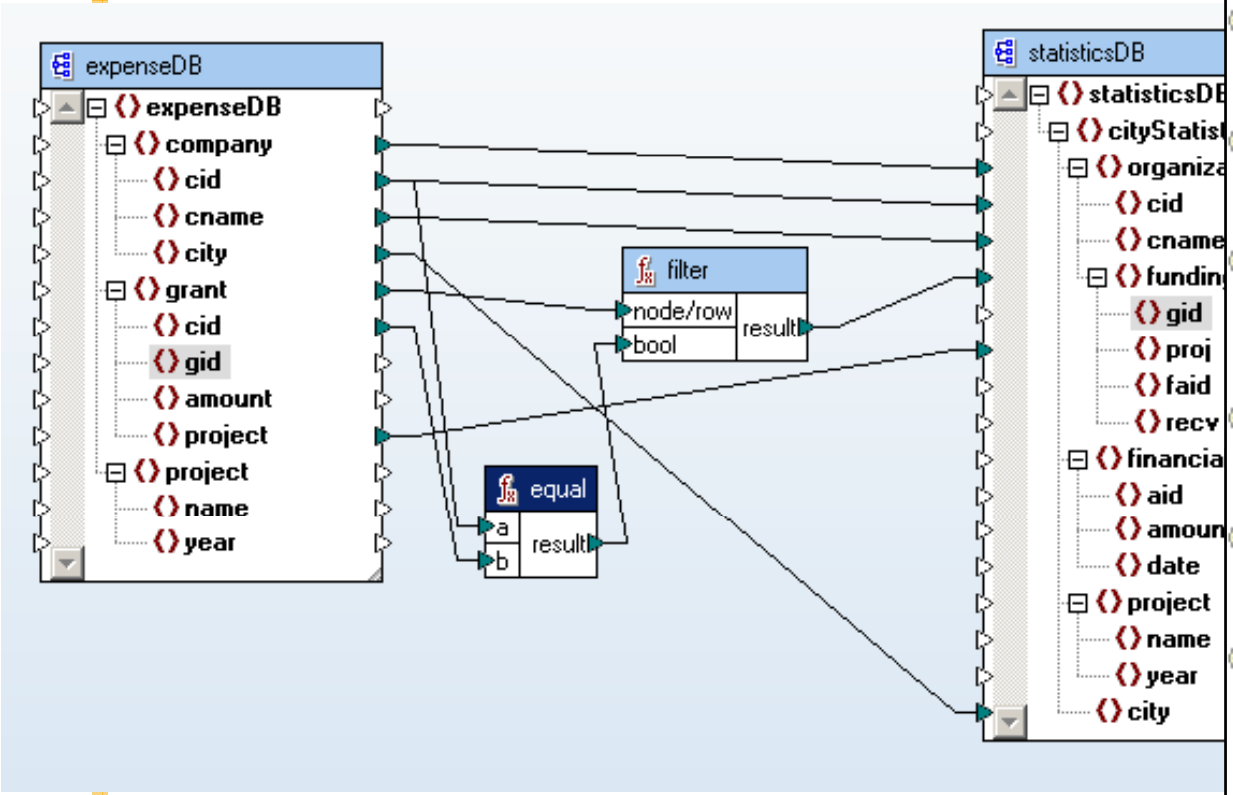


Altovas MapForce




```
<?xml version="1.0" encoding="UTF-8"?>
<statisticsDB xmlns:xsi="http://www.w3
  <cityStatistics>
    <organization>
      <cid>GG</cid>
      <cname>Garlic</cname>
      <funding>
        <proj>SQLServer</proj>
        <proj>Clio</proj>
        <proj>DB2</proj>
        <proj>FastNetwork</proj>
        <proj>Leo</proj>
        <proj>Clio</proj>
      </funding>
    </organization>
    <organization>
      <cid>CMPQ</cid>
      <cname>Compaq</cname>
      <funding>
        <proj>SQLServer</proj>
        <proj>Clio</proj>
        <proj>DB2</proj>
        <proj>FastNetwork</proj>
        <proj>Leo</proj>
        <proj>Clio</proj>
      </funding>
    </organization>
    <organization>
    <organization>
    <organization>
    <city>Gilroy</city>
    <city>San Jose</city>
    <city>San Jose</city>
    <city>San Jose</city>
    <city>Redmond</city>
```

Altovas MapForce



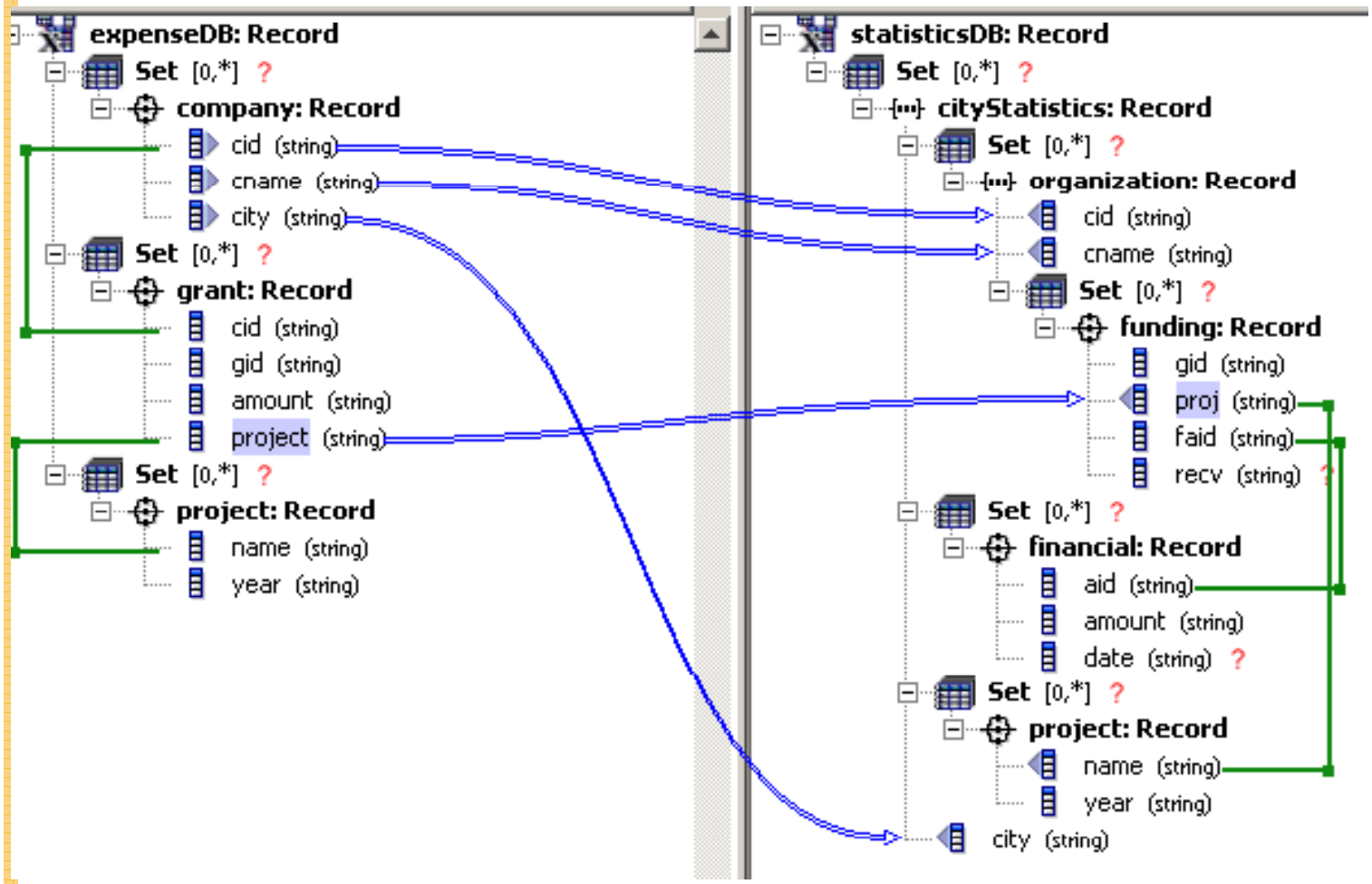

```
<organization>
  <cid>GG</cid>
  <cname>Garlic</cname>
</organization>
<organization>
<organization>
<organization>
  <cid>IBM</cid>
  <cname>International Business Machines</cname>
  <funding>
    <proj>Clio</proj>
  </funding>
  <funding>
    <proj>DB2</proj>
  </funding>
  <funding>
    <proj>Clio</proj>
  </funding>
</organization>
<organization>
  <cid>MSFT</cid>
  <cname>Microsoft</cname>
  <funding>
    <proj>SQLServer</proj>
  </funding>
  <funding>
    <proj>Leo</proj>
  </funding>
</organization>
<city>Gilroy</city>
<city>San Jose</city>
<city>San Jose</city>
<city>San Jose</city>
<city>Redmond</city>
```

Altovas MapForce

Zusammenfassung

- Informationen gehen nicht verloren
- Joins
  - werden nicht erkannt
  - können manuell in GUI erstellt werden
- Gruppierung wird nicht erkannt

IBMs Clio


expenseDB: Record
Set [0,*] ?
company: Record
cid (string)
cname (string)
city (string)
Set [0,*] ?
grant: Record
cid (string)
gid (string)
amount (string)
project (string)
Set [0,*] ?
project: Record
name (string)
year (string)
statisticsDB: Record
Set [0,*] ?
cityStatistics: Record
Set [0,*] ?
organization: Record
cid (string)
cname (string)
Set [0,*] ?
funding: Record
gid (string)
proj (string)
faid (string)
recv (string) ?
Set [0,*] ?
financial: Record
aid (string)
amount (string)
date (string) ?
Set [0,*] ?
project: Record
name (string)
year (string)
city (string)

IBMs Clio

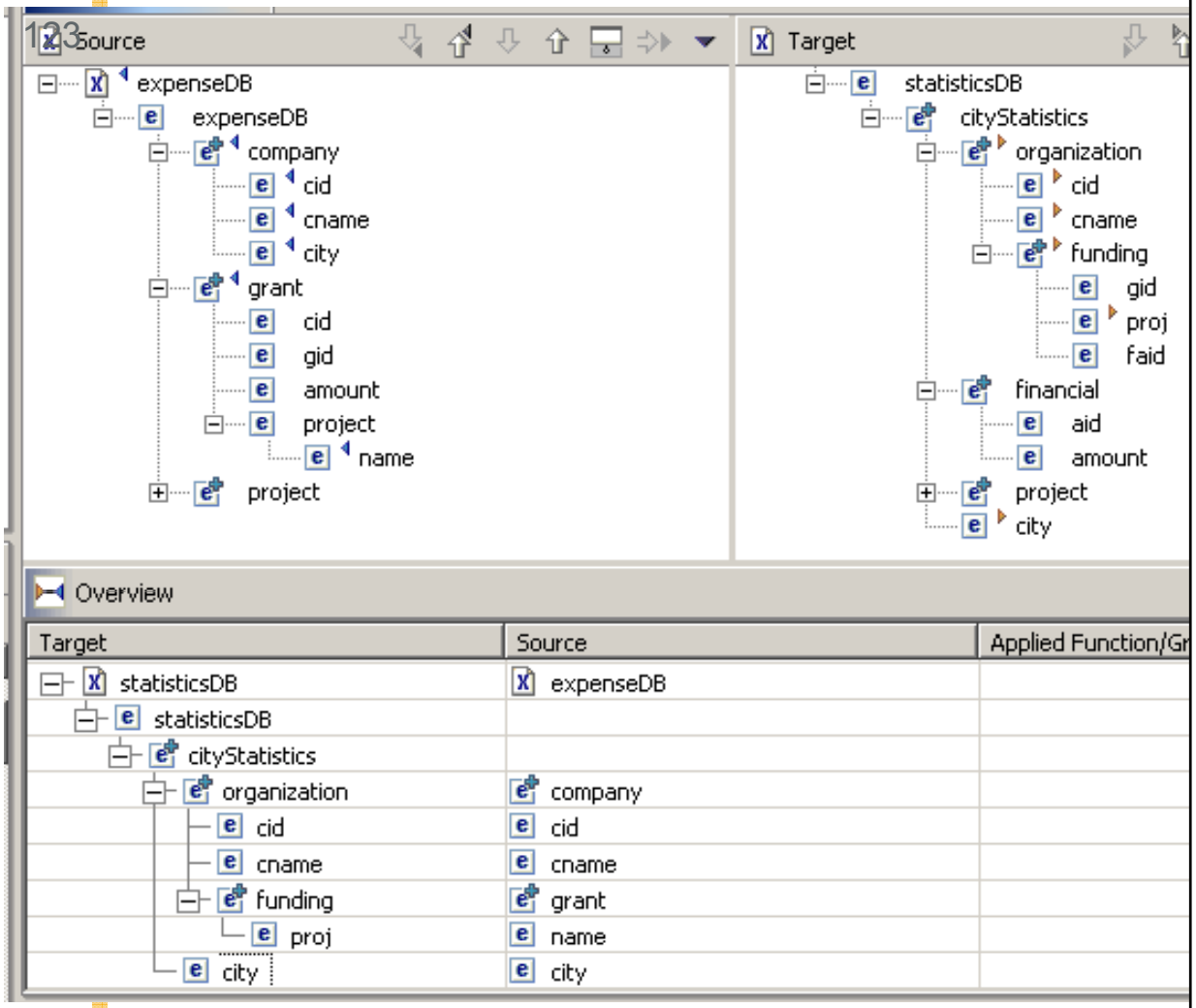

```
<?xml version="1.0" encoding="UTF-8" ?>
- <quip:result xmlns:quip="http://namespaces.softwareag.com/tamino/quip/">
  - <statisticsDB>
    + <cityStatistics>
    - <cityStatistics>
      - <organization>
          <cid>IBM</cid>
          <cname>International Business Machines</cname>
        - <funding>
            <gid>Sk72(Clio, International Business Machines, San Jose, IBM)</gid>
            <proj>Clio</proj>
            <faid>Sk67(Clio, International Business Machines, San Jose, IBM)</faid>
          </funding>
        - <funding>
            <gid>Sk72(DB2, International Business Machines, San Jose, IBM)</gid>
            <proj>DB2</proj>
            <faid>Sk67(DB2, International Business Machines, San Jose, IBM)</faid>
          </funding>
        </organization>
      - <organization>
          <cid>NA</cid>
          <cname>Network Associates</cname>
        - <funding>
            <gid>Sk72(FastNetwork, Network Associates, San Jose, NA)</gid>
            <proj>FastNetwork</proj>
            <faid>Sk67(FastNetwork, Network Associates, San Jose, NA)</faid>
          </funding>
        </organization>
      + <financial>
      + <financial>
      + <financial>
      + <project>
      + <project>
      + <project>
        <city>San Jose</city>
      </cityStatistics>
    </statisticsDB>
</quip:result>
```

IBMs Clio

Zusammenfassung

- Joins werden erkannt
  - Aber nicht erzwungen: Interpretation
- Gruppierung wird erkannt

IBMs WSAD




```
<?xml version="1.0" encoding="UTF-8" ?>
- <statisticsDB>
  - <cityStatistics>
    - <organization>
        <cid>GG</cid>
        <cname>Garlic</cname>
      + <funding>
      + <funding>
      + <funding>
      + <funding>
      + <funding>
      + <funding>
      + <funding>
      - <funding>
          <gid />
          <proj>SQLServer</proj>
          <faid />
        </funding>
      - <funding>
          <gid />
          <proj>FastNetwork</proj>
          <faid />
        </funding>
      + <funding>
      + <funding>
      + <funding>
      </organization>
    + <organization>
    + <organization>
    + <organization>
    + <organization>
    + <financial>
    - <project>
        <name />
        <year />
      </project>
      <city>Gilroy</city>
    </cityStatistics>
  </statisticsDB>
```

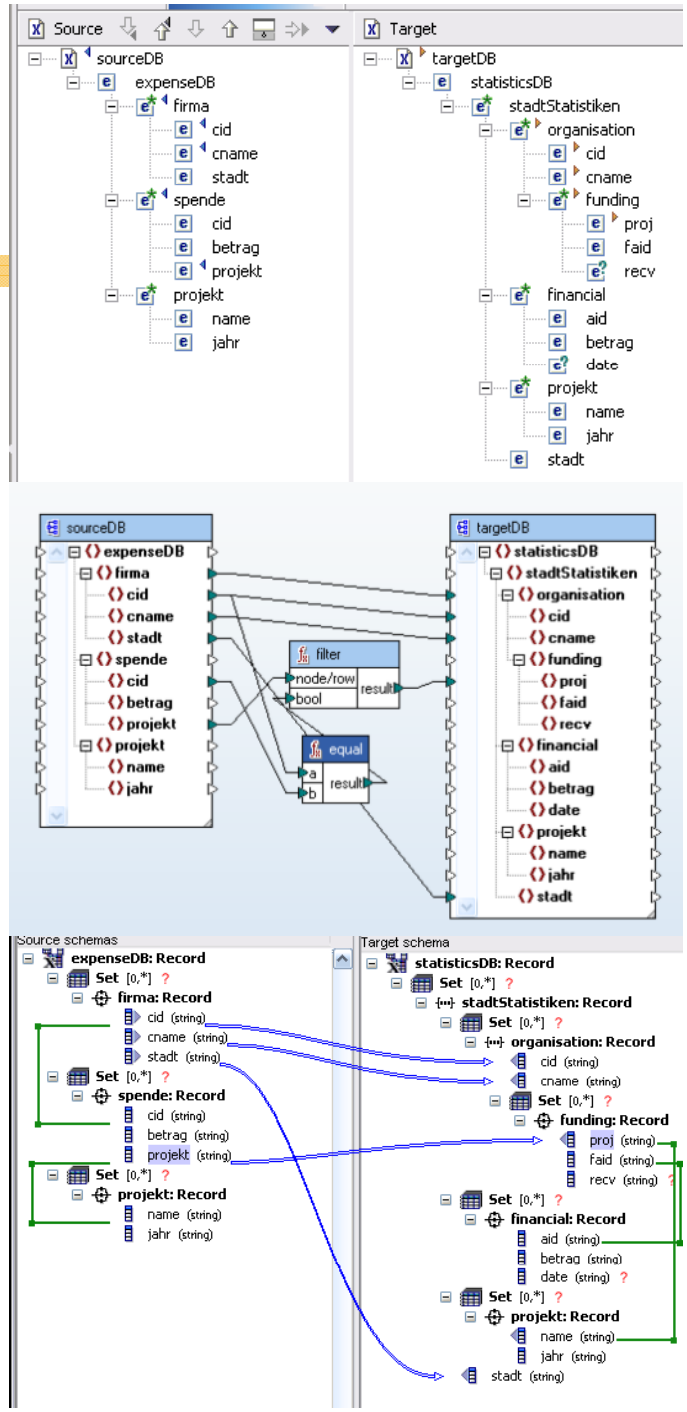
IBMs WSAD

Zusammenfassung

- Mögliche Joins werden nicht erkannt
- Gruppierung wird nicht erkannt
- Informationen werden verworfen
  - Nur erste Stadt
- Information wird „erfunden“
  - Jede Organization erhält jedes Projekt

Zusammenfassung der Demo

- IBMs WSAD
  - Mögliche Joins werden nicht erkannt.
  - Gruppierung wird nicht erkannt.
  - Daten werden verworfen.
  - Assoziationen zw. Daten werden „erfunden“.
- Altovas MapForce
  - Daten gehen nicht verloren.
  - Joins werden nicht erkannt, aber können manuell in GUI erstellt werden.
  - Gruppierung wird nicht erkannt.
- IBMs Clio
  - Joins werden erkannt, aber nicht erzwungen: Interpretationen möglich
  - Gruppierung wird erkannt.



Literatur – Schema Mapping

- [FHP+02] Ron Fagin, Mauricio Hernandez, Lucian Popa, Renee Miller, and Yannis Velegrakis, Translating Web Data, VLDB 2002, Hong Kong, China.
- [RB01] Erhard Rahm and Philip Bernstein, A survey of approaches to automatic schema matching, VLDB Journal 10(4), 2001.
- Zu der Problematik, ob Duplikate Teil des Outputs sein sollten:
  - Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, Lucian Popa: Data Exchange: Semantics and Query Answering. ICDT 2003: 207-224
- Clio:
  - http://www.almaden.ibm.com/software/projects/criollo/
  - oder http://www.cs.toronto.edu/db/clio/

Literatur – Schema Matching

Artikel mit der Klassifikation:

- [RB01] Erhard Rahm and Philip Bernstein, A survey of approaches to automatic schema matching, VLDB Journal 10(4), 2001.

Spezielle Algorithmen

- [MGMR02] Sergey Melnik, Hector Garcia-Molina, Erhard Rahm: Similarity Flooding: A Versatile Graph Matching Algorithm and Its Application to Schema Matching. ICDE 2002: 117-128
- [MH03] Jayant Madhavan, Alon Y. Halevy: Composing Mappings Among Data Sources. VLDB 2003: 572-583.
- uvam.