Duplikaterkennung und Datenfusion

Kolloquium
Stuttgart, 6.7.2004
Felix Naumann
Humboldt-Universität zu Berlin

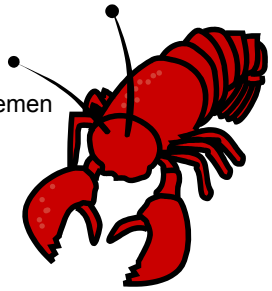
Humboldt-Universität zu Berlin

Humboldt-Universität zu Berlin

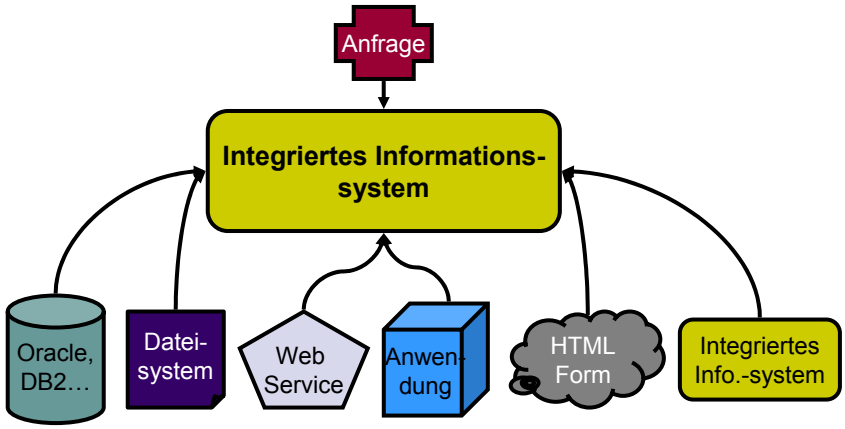
- Wilhelm und Alexander von Humboldt
- Einheit von Lehre und Forschung
- Freiheit und Unabhängigkeit der Wissenschaft
- 29 Nobelpreisträger
  - Mommsen, Hertz, Koch, Hahn, Planck, Einstein,...
- 38,000 Studenten, (1100 Informatik)
- 560 Professoren (-90)

Forschungsgruppe Informationsintegration

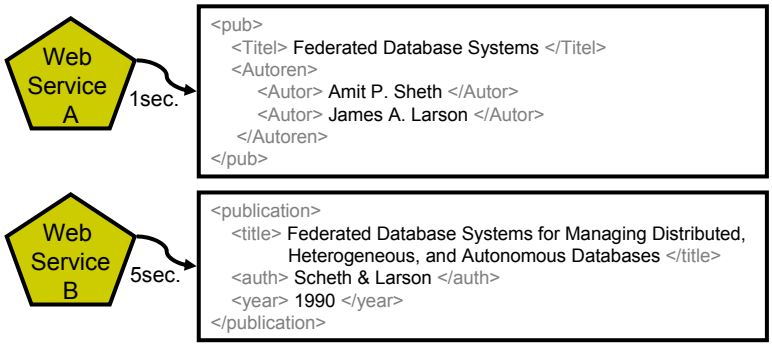
- Leitung: Felix Naumann (naumann@informatik.hu-berlin.de)
- Mitarbeiter
  - Jens Bleiholder (bleiho@informatik.hu-berlin.de)
    - Informationsfusion in relationalen Daten
  - Melanie Weis (mweis@informatik.hu-berlin.de)
    - Objektidentifikation in XML Daten
- Affiliated
  - Armin Roth (aroth@informatik.hu-berlin.de)
    - Datenqualität in Peer-Data-Management-Systemen
  - Alexander Bilke (bilke@cs.tu-berlin.de)
    - Schema Matching
- Forschungsthemen
  - Objektidentifikation
  - Informationsfusion
  - Optimierung
  - Visualisierung

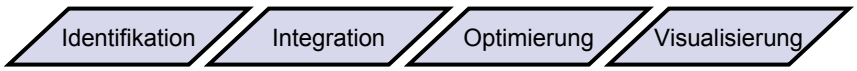


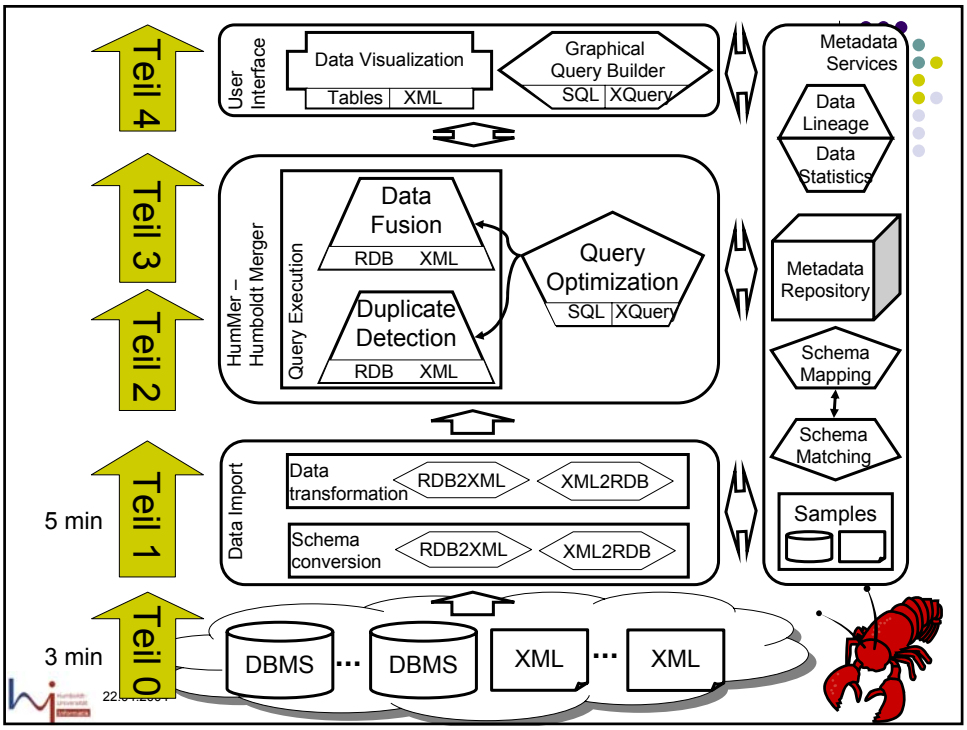
Integrierte Informationssysteme



Beispiel der Informationsintegration







HumMer
Der Humboldt Merger

Schematische Heterogenität

```
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)
```


Schema Mapping Beispiel

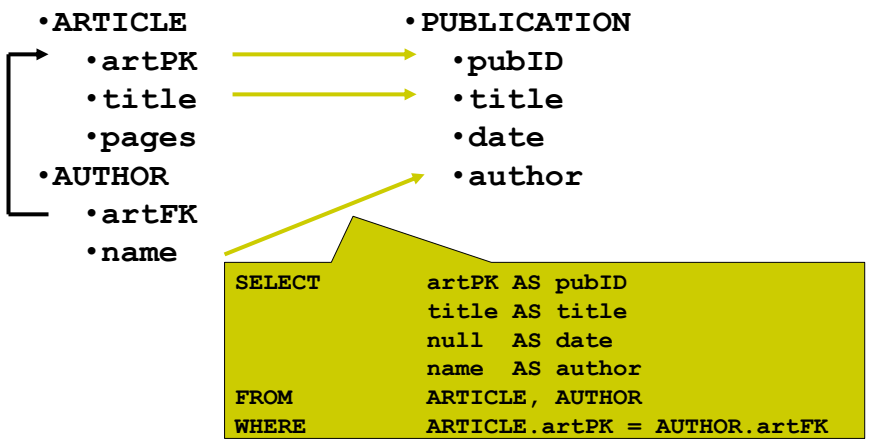
- ARTICLE
  - artPK
  - title
  - pages
- AUTHOR
  - artFK
  - name

- PUBLICATION
  - pubID
  - title
  - date
  - author

```
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
```

Mit Clio Team: Howard Ho, Lucian Popa, Mauricio Hernandez,...

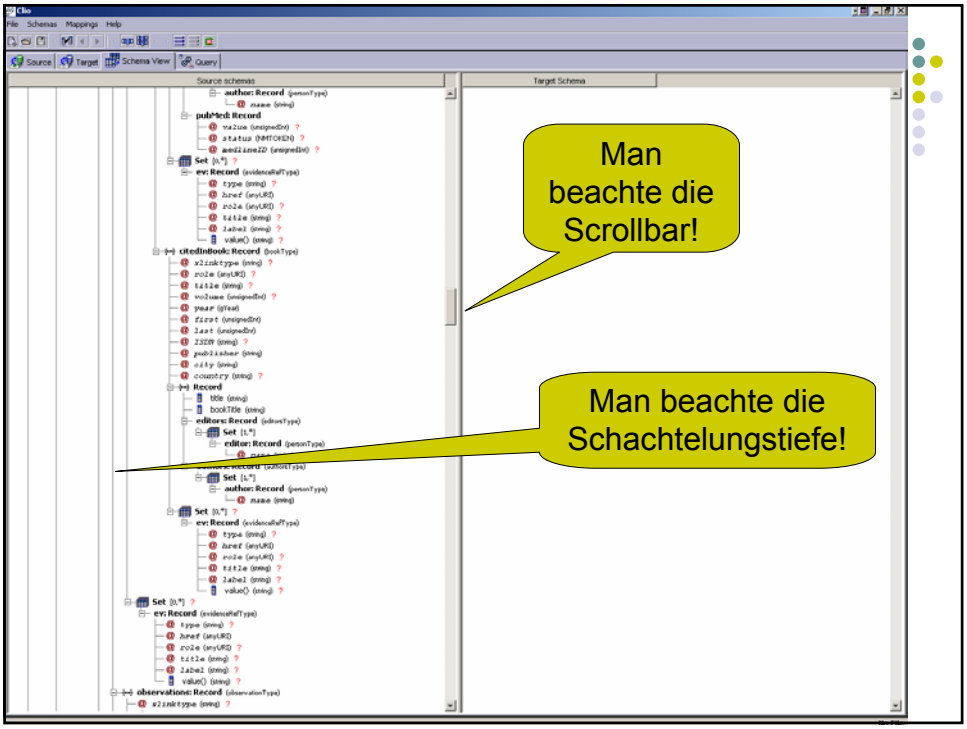
Schema Mapping Beispiel



Schema Matching – Motivation

- Große Schemata
  - > 100 Tabellen, viele Attribute
  - Bildschirm nicht lang genug
- Unübersichtliche Schemata
  - Tiefe Schachtelungen
  - Fremdschlüssel
  - Bildschirm nicht breit genug
- Fremde Schemata
- Irreführende Schemata
- Fremdsprachliche Schemata
- Kryptische Schemata
- XML Schemata

Mit Clio Team: Howard Ho

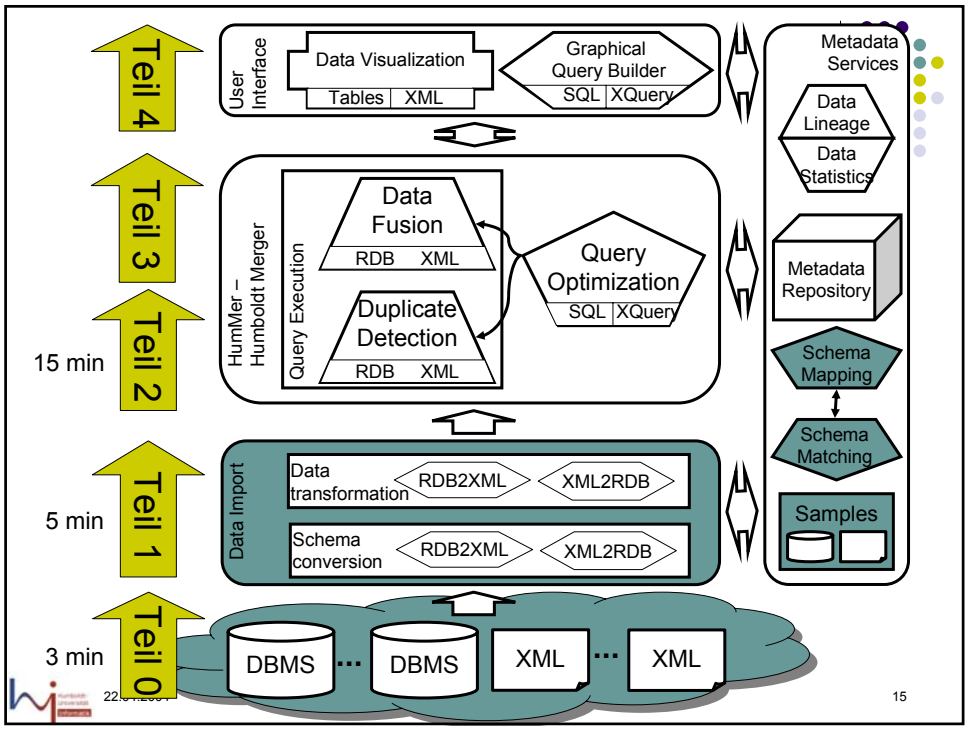


Schema Matching Klassifikation

- Schema Matching basierend auf
  - Namen der Schemaelemente (label-based)
  - Darunterliegende Daten (instance-based)
  - Struktur des Schemas (structure-based)
  - Mischformen, Meta-Matcher

später

- Kernalgorithmus des Schema Matching:
  - Gegeben zwei Schemata mit Attributmengen A und B.
  - Bilde Kreuzprodukt aller Attribute aus A und B.
  - Für jedes Paar vergleiche Ähnlichkeit.
  - Ähnlichste Paare sind Matches.



Duplikaterkennung

- Erkennung mehrere (ähnlicher) Darstellungen des gleichen Objekts
- Auch
  - Duplicate Detection
  - Data Cleansing
  - Record Linkage
- Domänenspezifische Algorithmen
  - Adressdaten
  - Mikrobiologische Daten
- Manchmal gibt es IDs
  - Bücher: ISBN
  - Personen: SSN / Personalausweisnummer
  - Webseiten: URL

Duplikaterkennung in XML Daten

```
- <author>
    <name>Bernd Amann</name>
  - <publication>
      <title>XML Repository and Active Views Demonstration.</title>
      <year>1999</year>
    </publication>
  - <publication>
      <title>Active Views for Electronic Commerce.</title>
      <year>1999</year>
    </publication>
  </author>
- <author>
    <name>Sophie Cluet</name>
  - <publication>
      <title>XML Repository and Active Views Demonstration.</title>
      <year>1999</year>
    </publication>
  - <publication>
      <title>Active Views for Electronic Commerce.</title>
      <year>1999</year>
    </publication>
  - <publication>
      <title>Views in a Large Scale XML Repository.</title>
      <year>2001</year>
    </publication>
  </author>
```

- Vergleiche <author>
  - Mit Subelementen (<publication>)?
  - Wie tief?
- Vergleiche <publication>
  - Mit parallelen Elementen (<year>)?
  - Schema, oder Daten?
- Kurz: Was ist ein Duplikat?

Mit Melanie Weis

DELPHI - Data Warehouse Duplikate [AC02]

ID	country
1	USA
2	United States
3	Unitd States

ID	City	County_ID
1	New York	1
2	Los Angeles	1
3	Now York	2
4	Los Angeles	2
5	New York	3
6	Los Angels	3

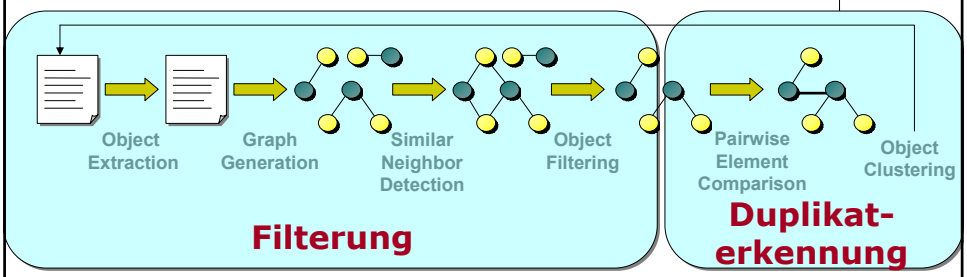
Token Ähnlichkeit
→ 2 ≈ 3

Gemeinsame Kinder
→ 1 ≈ 2 ≈ 3

XML Duplikate

- Input: XML Dokument, kein Schema
- Output: Alle Duplikate im Dokument
- Zwei Elemente sind Duplikate falls:
  - Gleiche oder ähnliche Eltern
  - Gleicher Name (Tag)
  - Gleiche Tiefe im Dokument
  - Ähnliche Daten
  - Ähnliche Kinder
- Ähnlichkeit: Basiert auf edit distance mit Schwellwert

Der Algorithmus

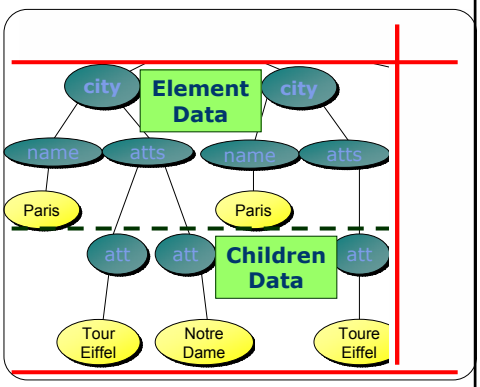


- Experimente auf “sauberen” Länderdaten, die mit vier Fehlerarten verschmutzt wurden.
- 260 Länder
- Außerdem: Filmdaten, CD Daten, usw.

Algorithmus - Objektextraktion

Ziel: Relevante Elemente extrahieren

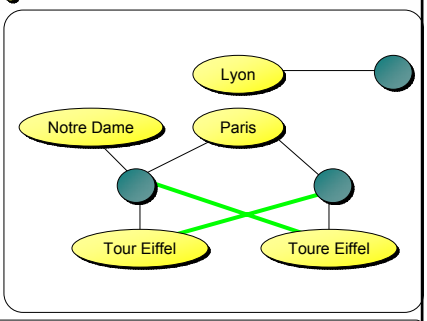
- XQuery
- Resultat hat festes Schema
- Unterscheidung:
  - Elementdaten
  - Kinddaten



Algorithmus – Entdeckung ähnlicher Nachbarn

Ziel: Finde ähnliche Daten zwischen Elementen

- Edit distance.
- Filter um Anzahl der Vergleiche zu minimieren:
  - Längen-Filter
  - Dreiecksungleichung
  - Bag distance



Selektivität der Filter (in Reihenfolge):

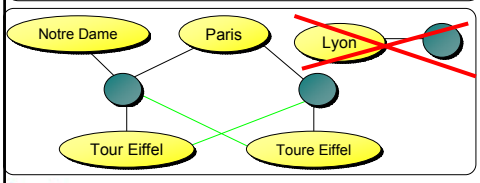
Längenfilter: 83 %
Dreiecksungleichung: 23%
Bag distance Filter: 97%

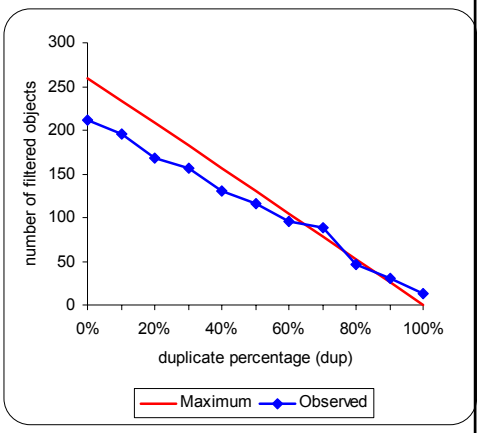
>99%

Algorithmus - Objektfilter

Ziel: Anzahl der Vergleiche zwischen Objekten minimieren

- Elemente filtern, die nicht Duplikate sein können
- Obere Schranke für Ähnlichkeitsmaß

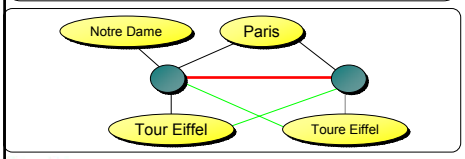


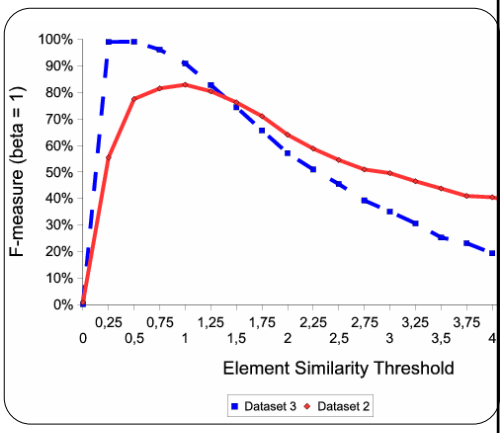


Algorithmus – Elementvergleich

Ziel: Identifikation von Duplikaten

- Ähnlichkeitsmaß basiert auf
  - Relevanz der gemeinsamen Daten
  - Relevanz der nicht-gemeinsamen Daten
- Schwellwert





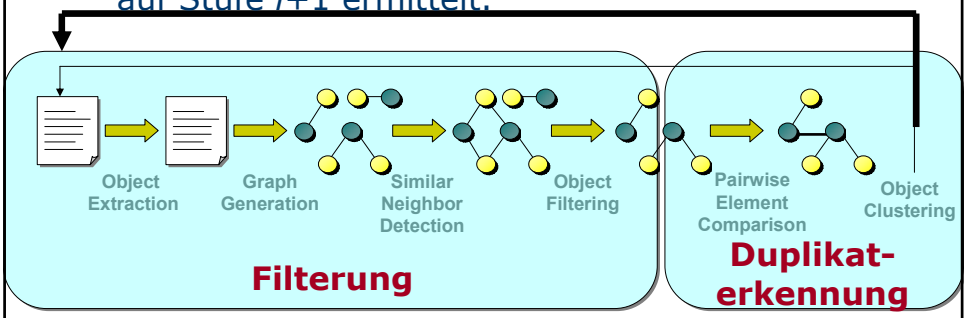
Algorithmus – Clustering

Ziel: Bestimmung von Duplikatclustern

- Transitivität von “ist Duplikat mit”
- Transitive Hülle bilden

Algorithmus – Top-down

Mittels Duplkate in Tiefe *I* werden Duplikate auf Stufe *i*+1 ermittelt.



Ausblick

- Weitere Experiment
  - Filmdaten
- Skalierbarkeit
  - Speicherbedarf
  - Geschwindigkeit
- Schema nutzen
  - Vergleiche sparen (content model, maxOccurs)
  - Datentypen → Andere Ähnlichkeitsmaße

Duplikaterkennung in XML Daten

```
- <author>
    <name>Bernd Amann</name>
  - <publication>
      <title>XML Repository and Active Views Demonstration.</title>
      <year>1999</year>
    </publication>
  - <publication>
      <title>Active Views for Electronic Commerce.</title>
      <year>1999</year>
    </publication>
  </author>
- <author>
    <name>Sophie Cluet</name>
  - <publication>
      <title>XML Repository and Active Views Demonstration.</title>
      <year>1999</year>
    </publication>
  - <publication>
      <title>Active Views for Electronic Commerce.</title>
      <year>1999</year>
    </publication>
  - <publication>
      <title>Views in a Large Scale XML Repository.</title>
      <year>2001</year>
    </publication>
  </author>
```

```
- <inproceedings key="conf/vldb/AbiteboulAAACHHMMMMSTV9
    <author>Serge Abiteboul</author>
    <author>Vincent Aguilera</author>
    <author>Sébastien Ailleret</author>
    <author>Bernd Amann</author>
    <author>Sophie Cluet</author>
    <author>Brendan Hills</author>
    <author>Frédéric Hubert</author>
    <title>XML Repository and Active Views Demonstration.</ti
    <pages>742-745</pages>
    <year>1999</year>
    <booktitle>VLDB</booktitle>
    <url>db/conf/vldb/vldb99.html#AbiteboulAAACHHMMMMS
    <crossref>conf/vldb/99</crossref>
    <ee>db/conf/vldb/AbiteboulAAACHHMMMMSTV99.html</e
    <cdrom>VLDB99/P73.pdf</cdrom>
    <cite>conf/edbt/SantosAD94</cite>
    <cite>www/org/w3/dom</cite>
  </inproceedings>
```

Duplikat-gesteuertes Schema Matching

- Umgekehrte Idee
- Schema Matching
- Herkömmliche Lösungen (siehe zuvor)
  - Namen der Schemaelemente (*label-based*)
  - Darunterliegende Daten (*instance-based*)
    - Insbesondere Eigenschaften von Spalten
  - Struktur des Schemas (*structure-based*)
- Nun
  - Finde Duplikate (trotz mangelnder Schemata)
  - Korrespondenzen zwischen gleichen Attributwerten

Mit Alexander Bilke

Duplikat-gesteuertes Schema Matching

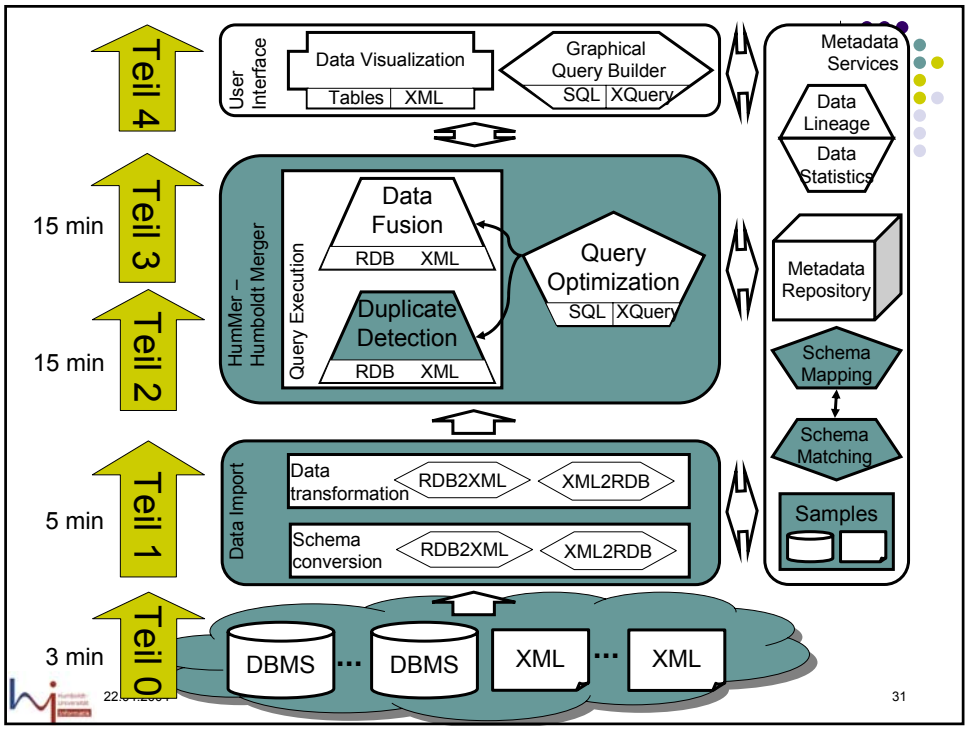
R	A	B	C	D	E
r_1	John	Doe	m	(408) 757 33 39	(408) 757 33 38
r_2	Joe	Smith	m	(249) 361 56 16	(249) 234 23 66
r_3	Suzy	Klein	f	(358) 243 63 21	(358) 243 63 21
r_4	Sam	Adams	m	(541) 812 71 00	(541) 812 11 64
r_5	Mark	Spitz	m	(901) 831 93 11	(901) 861 23 82
r_6	Jim	Beam	-	(782) 123 89 57	(781) 188 37 44
r_7	Kate	Moss	f	(124) 965 45 65	-
r_8	Sam	Wong	f	(124) 495 56 70	(999) 999 99 99
r_9	John	Dean	m	(369) 366 36 24	(367) 366 36 25

S	B'	F	E'	G
s_1	Doe	jdoe	408-9182043	XP
s_2	Deen	jdean	369-3663625	XP
s_3	Klein	suzy	358-2436321	UNIX
s_4	Adams	adams	541-8121164	W2000
s_5	Wong	howard	923-6363443	Linux
s_6	Kurz	itsme	-	UNIX

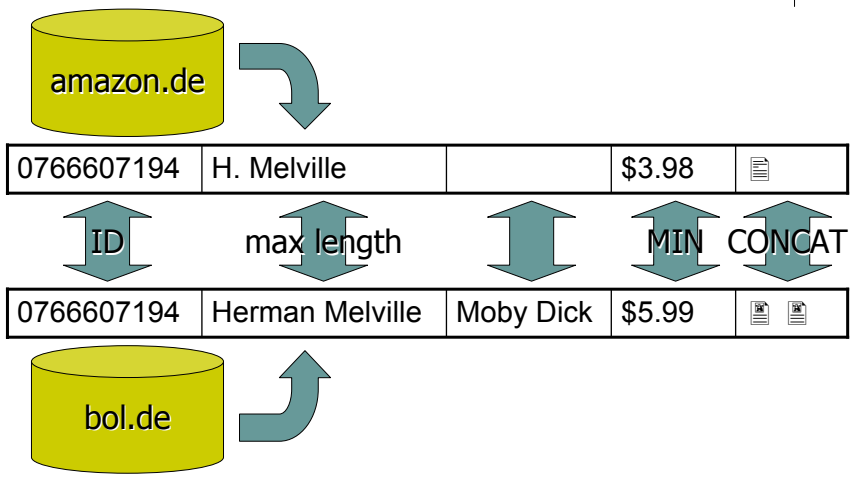
Duplikate

	A	B	C	D	E
r_3	Suzy	Klein	f	(358) 243 63 21	(358) 243 63 21

	B'	F	E'	G
s_3	Klein	suzy	358-2436321	UNIX

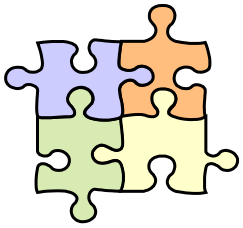


Daten Fusion



Relationale Datenfusion

- Union ∪
  - Duplikat-Eliminierung
- Outer union ⊎ (Codd 1979)
  - Union bei heterogenen Schemata
- Minimum union ⊕ (Ullmann 1989, Galindo-Legaria 1994)
  - Eliminierung subsummierter Tupel
- Data Fusion
  - Duplikatintegration
  - Konfliktlösung



Wie Union integriert

Quelle 1(A,B,C) →
Quelle 2(A,B,C) →

a, b, c	→	a, b, c
a, b, c		

⇒ Elimination identischer Tupel

a, b, c	→	a, b, c
a, b, -		a, b, -

⇒ Keine Elimination subsummierter Tupel

a, b, c	→	a, b, c
a, b, d		a, c, d

⇒ Keine Konfliktlösung

a, -, c	→	a, -, c
a, b, -		a, b, -

⇒ Keine Integration komplementärer Tupel

Wie Outer Union integriert

Quelle 1(A,B,C) →
Quelle 2(A,B,D) →

a, b, c	a, b, c, -	a, b, c, -	Keine Integr. komplement. Tupel
a, b, d	a, b, -, d	a, b, -, d	

a, b, -	a, b, -, -	a, b, -, -	Elimination identischer Tupel
a, b, -	a, b, -, -		

a, b, c	a, b, c, -	a, b, c, -	Keine Elimination subsummierter Tupel
a, b, -	a, b, -, -	a, b, -, -	

a, b, c	a, b, c, -	a, b, c, -	Keine Konfliktlösung
a, e, d	a, e, -, d	a, e, -, d	

Wie Minimum Union integriert

Quelle 1(A,B,C) →
Quelle 2(A,B,D) →

a, b, c	a, b, c, -	a, b, c, -	Keine Integr. komplement. Tupel
a, b, d	a, b, -, d	a, b, -, d	

a, b, -	a, b, -, -	a, b, -, -	Elimination identischer Tupel
a, b, -	a, b, -, -		

a, b, c	a, b, c, -	a, b, c, -	Elimination subsummierter Tupel
a, b, -	a, b, -, -		

a, b, c	a, b, c, -	a, b, c, -	Keine Konfliktlösung
a, e, d	a, e, -, d	a, e, -, d	

Wie Data Fusion integrieren sollte

Quelle 1(A,B,C) →
Quelle 2(A,B,D) →

a, b, c	→	a, b, c, -	→	a, b, c, d
a, b, d		a, b, -, d		

⇒ Integration komplement. Tupel

a, b, -	→	a, b, -, -	→	a, b, -, -
a, b, -		a, b, -, -		

⇒ Elimination identischer Tupel

a, b, c	→	a, b, c, -	→	a, b, c, -
a, b, -		a, b, -, -		

⇒ Elimination subsummierter Tupel

a, b, c	→	a, b, c, -	→	a, *f(* b,e*)*, c, d
a, e, d		a, e, -, d		

⇒ Konfliktlösung

Konfliktlösungsfunktionen

- Numerisch:
  - SUM, AVG, MAX, MIN, …
- Nicht-numerisch:
  - MAXLENGTH, CONCAT, AnnCONCAT,…
- Spezialisiert:
  - RANDOM, COUNT, CHOOSE, FAVOR, MaxIQ,…
- Domänen-spezifisch
  - …

Fuse By – Syntaxdiagramm

```
                .----,----------------------------.
                V                                  |
>>--SELECT--+---+-------column-----------------+-+-+-->
            |   +--RESOLVE (column)------------+   |
            |   `--RESOLVE (column, function)--'   |
            `--------------*-----------------------'

        .--,-----.
        V        |
>--FROM---table--+-->

                .--,------.              .--,------.
                V         |              V         |
>--FUSE BY--(-+---column--+-+-)--ON ORDER---column--+--><
              `-------------'
```

Mit Jens Bleiholder

Fuse By – Semantik

- Idee: Gruppierung und Outer Union
- Konfliktlösungsfunktionen in den Gruppen
- Intuitives Default-Verhalten
  - Verwendung von `Coalesce`
  - Beseitigung von Unsicherheiten
  - Entfernung doppelter Tupel
  - Entfernung subsummierter Tupel

Fuse By – Anfragen

```
SELECT *
FROM Q1
FUSE BY (Name)
```

Gruppierung mit Coalesce Aggregation

```
SELECT *
FROM Q1
FUSE BY ()
```

Subsumption

```
SELECT *
FROM Q1, Q2
FUSE BY ()
```

Minimum Union

```
SELECT Name, RESOLVE(Alter, max), RESOLVE(Student,
       vote), RESOLVE(Ort), RESOLVE(Telefon)
FROM Q1, Q2
FUSE BY (Name) ON ORDER Q2.Alter DESC
```

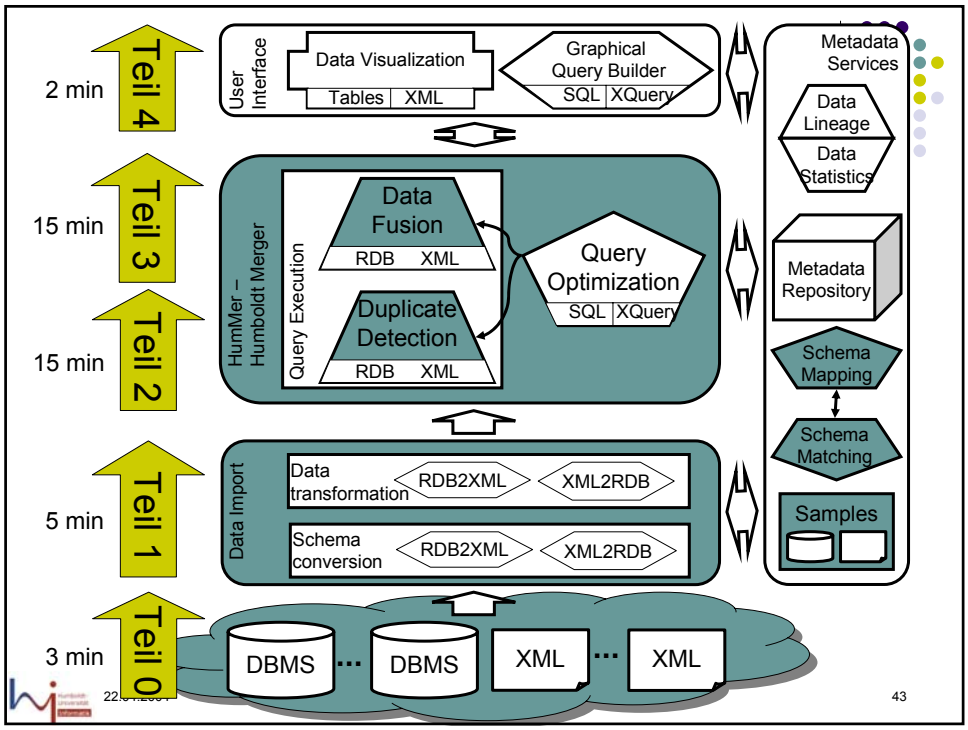

FUSE BY – Beispiel

Name	Alter	Student	Ort
Felix	⊥	Nein	Hamburg
Melanie	22	Ja	⊥
Jens	⊥	Ja	Karlsruhe
Christoph	25	Ja	Berlin
Sven	26	Ja	Berlin
Sven	⊥	Ja	Berlin

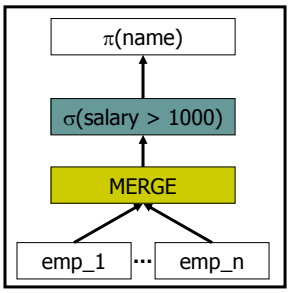
Name	Alter	Student	Telefon
Melanie	⊥	Ja	030/12345
Jens	27	⊥	030/54321
Christoph	24	Ja	⊥
Melanie	21	Nein	030/98765
Karsten	24	Ja	⊥
Karsten	24	Ja	⊥

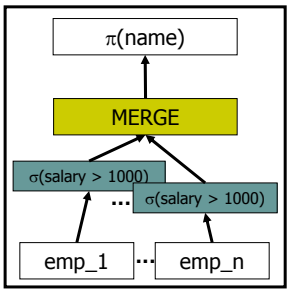
```
SELECT Name,
 RESOLVE(Alter,max),
 RESOLVE(Student,vote),
 RESOLVE(Ort),
 RESOLVE(Telefon)
FROM Q1, Q2
FUSE BY (Name)
ON ORDER Q2.Alter DESC
```

Name	Alter	Student	Ort	Telefon
Felix	⊥	Nein	Hamburg	⊥
Melanie	22	Ja	⊥	030/98765
Jens	27	Ja	Karlsruhe	030/54321
Christoph	25	Ja	Berlin	⊥
Sven	26	Ja	Berlin	⊥
Karsten	24	Ja	⊥	⊥



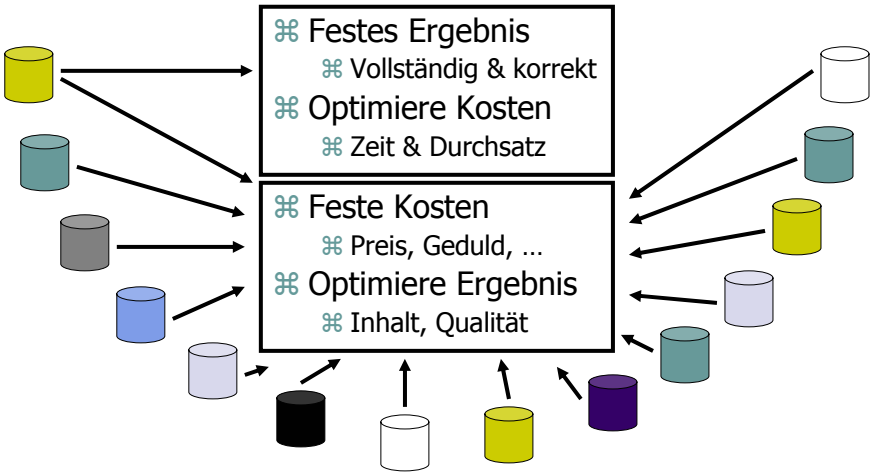
Anfrageoptimierung mit Merging





- Korrektheit?
- Vollständigkeit?
- Effizienz?

Optimierung – Ein Paradigmenwechsel



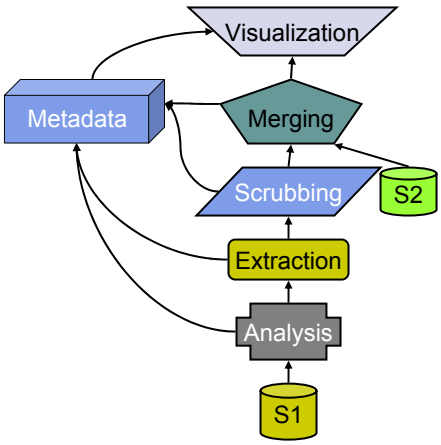
Visualisierung – Ziele

- Darstellung integrierter Ergebnisse
  - Viele Quellen
  - Viele Operationen / Transformationen
- Nutzerfreundliche Darstellung
- Drill-Down
  - Metadaten
- Kein kleinster gemeinsamer Nenner!

Mit Claudia Lozek

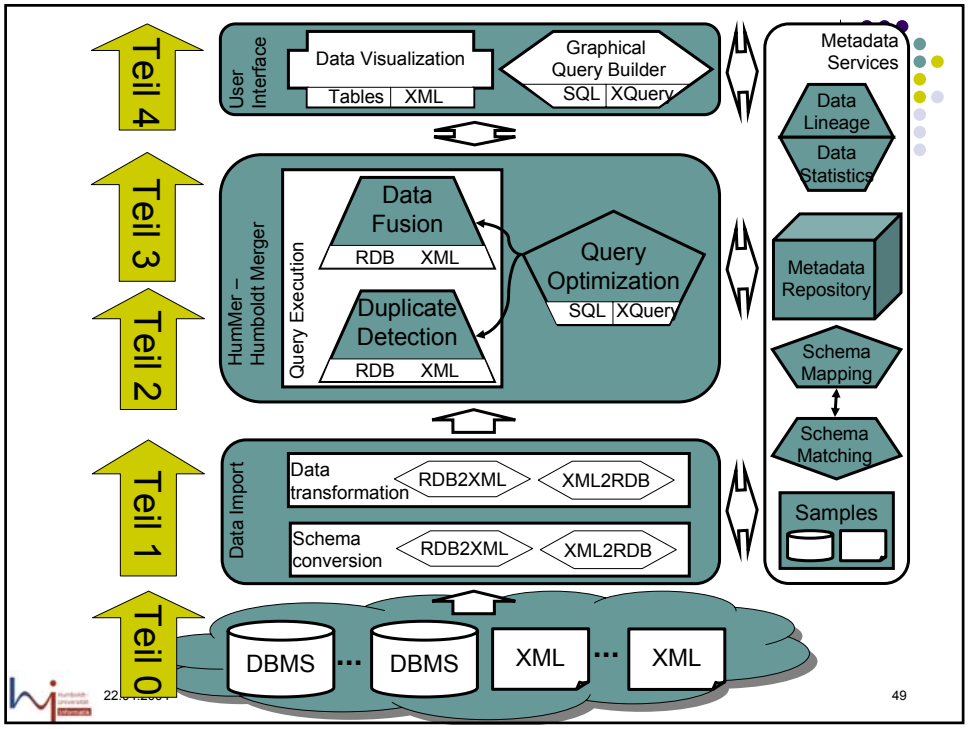
Metadaten der Integration

- Herkunft
  - Data lineage
  - Durch Operatoren hindurch
- Transformationen
  - Aggregation / Integration
- Konflikte
- Datenqualität
- Ranking



Fragen?

naumann@informatik.hu-berlin.de


Teil 4
Teil 3
Teil 2
Teil 1
Teil 0
User Interface
Data Visualization
Tables
XML
Graphical Query Builder
SQL
XQuery
HumMer – Humboldt Merger
Query Execution
Data Fusion
RDB
XML
Duplicate Detection
RDB
XML
Query Optimization
SQL
XQuery
Data Import
Data transformation
RDB2XML
XML2RDB
Schema conversion
RDB2XML
XML2RDB
DBMS
DBMS
XML
XML
Metadata Services
Data Lineage
Data Statistics
Metadata Repository
Schema Mapping
Schema Matching
Samples