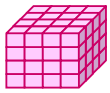
4. ETL: Schemaintegration + Data Cleaning

- ETL-Überblick
- Schemaintegration
- (Semi-)automatisches Schema Matching
  - COMA++
- Data Cleaning
  - Probleme
  - Teilaufgaben
- Tool-Unterstützung
  - MS SQL-Server 2005
  - MOMA

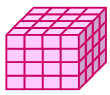


ETL-Prozess: Definition

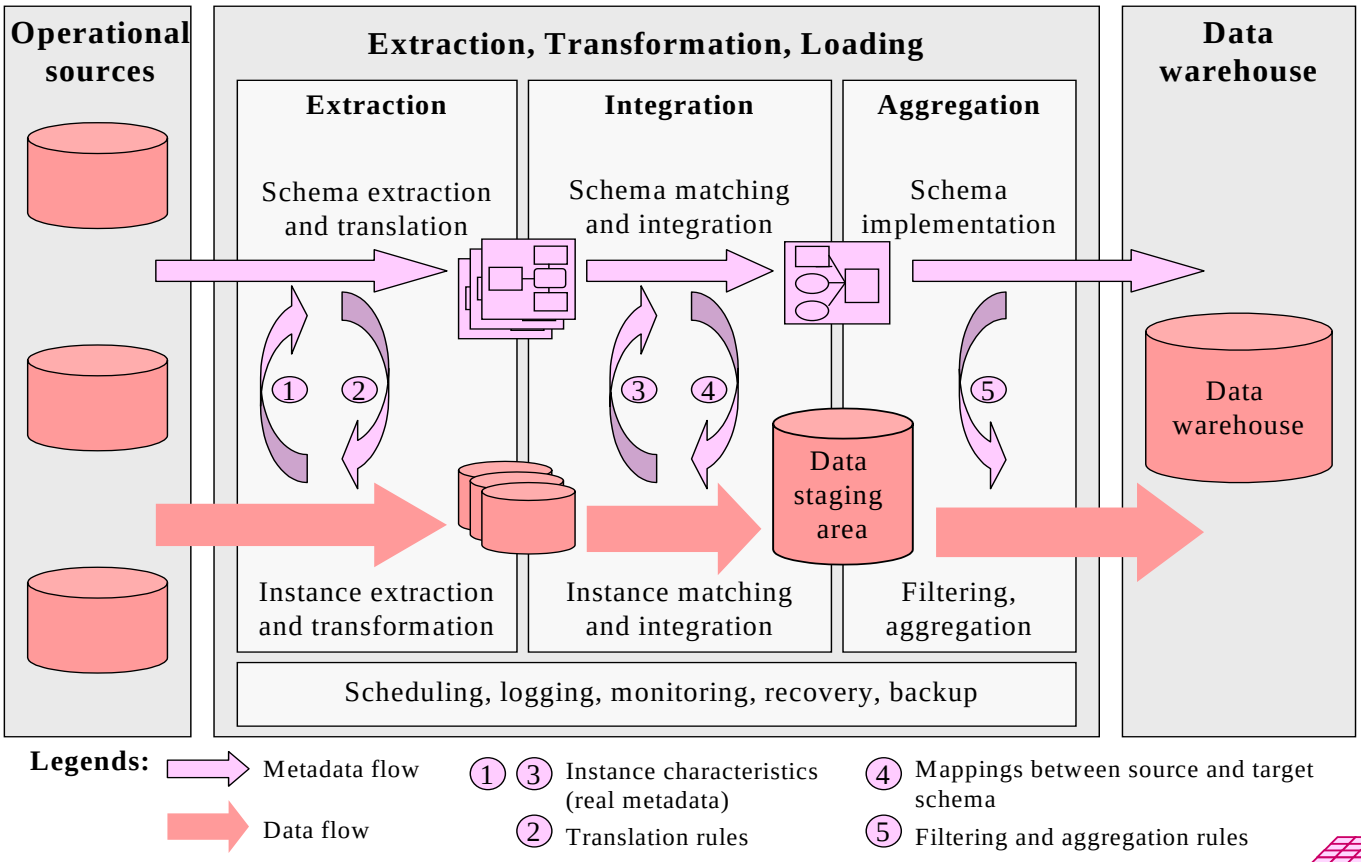
- *Extraktion*: Selektion eines Ausschnitts der Daten aus relevanten Quellen
  - ausgeführt an den entsprechenden Quellen
- *Transformation*: Aufbereitung und Anpassung der Daten an vorgegebene Schema- und Qualitätsanforderungen
  - ausgeführt im temporären Arbeitsbereich (Data Staging Area)
- *Laden*: physisches Einbringen der Daten aus dem Arbeitsbereich in das Data Warehouse, einschließlich evtl. notwendiger Aggregationen
  - ausgeführt in der Data Warehouse-Datenbank

- Aufwändigster Teil des Data Warehousing
  - Vielzahl von operativen Quellen
  - Überwindung von Heterogenität zwischen Datenquellen (DBMS, Schemata, Daten)
  - Gewährleistung hoher Qualität der Warehouse-Daten
- Entscheidende Rolle im Data Warehousing, da großer Einfluß auf
  - Genauigkeit und Richtigkeit der später durchgeführten Analysen
  - die darauf basierenden Entscheidungen

  „Garbage In, Garbage Out“



ETL-Prozess: Ablauf



ETL als Integrationsprozess

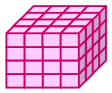
- ETL: Integration auf 2 Ebenen, Schemaintegration und Datenintegration
- Schemaintegration
  - Konstruktion eines Data Warehouse-Schemas aus existierenden Quellschemata
  - Ableitung von Korrespondenzen zwischen dem Data Warehouse-Schema und existierenden Quellschemata: *Schema Matching*
- Datenintegration
  - Transformation heterogener Daten in die einheitliche, durch das Data Warehouse-Schema vorgeschriebene Repräsentation
  - Entdeckung und Behebung von Datenqualitätsproblemen bei der Quellzusammenführung
  - Entdeckung äquivalenter Objekte/Sätze aus verschiedenen Quellen (Korrespondenzen auf Instanzenebene): *Objekt Matching* / Duplikaterkennung
- Data Warehousing und ETL: materialisierter Ansatz zur Datenintegration
  - Erzeugung einer aggregierten, materialisierten Datenquelle für Online-Analysen
  - komplexer, aufwendiger Integrationsprozeß
  - Offline-Durchführung erlaubt höhere Datenqualität gegenüber virtueller Datenintegration (Datentransformation während Query-Verarbeitung)

Schemaintegration - Anforderungen

- Minimalität
  - keine Redundanz im integrierten Schema
  - Abbildung mehrerer gleicher/ähnlicher Konzepte in lokalen Schemata auf ein Konzept im integrierten Schema)
- Korrektheit
  - Äquivalenz der im integrierten Schema enthaltenen Informationen mit denen in den lokalen Schemata
  - Konsistenz der während der Integration ergänzten/hinzugefügten Informationen, z.B. Beziehungen zwischen Konzepten im integrierten Schema
- Verständlichkeit

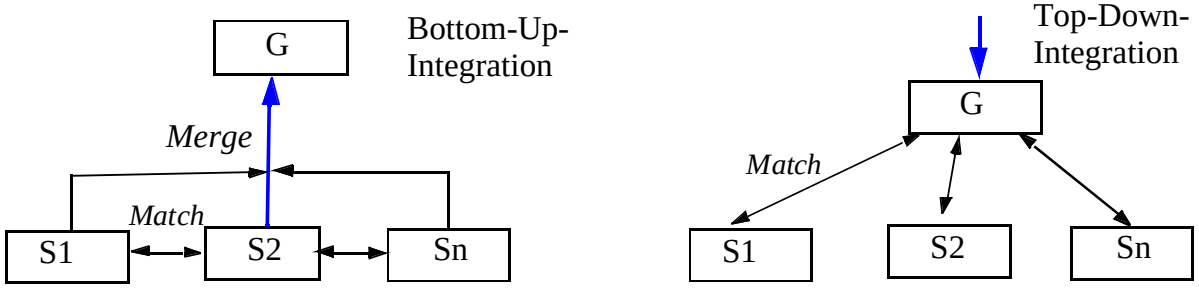
Vollständigkeit (Beibehaltung aller Informationen aus Quellschemas) i.a. nicht erforderlich !

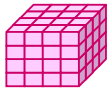
- Probleme der Schemaintegration
  - Heterogenität der Schemarepräsentationen, z.B. relational (SQL), XML, Entity-Relationship (ER), objekt-orientiert (UML), ...
  - Semantische Heterogenität der Schemaelemente (Namenskonflikte, strukturelle Konflikte)



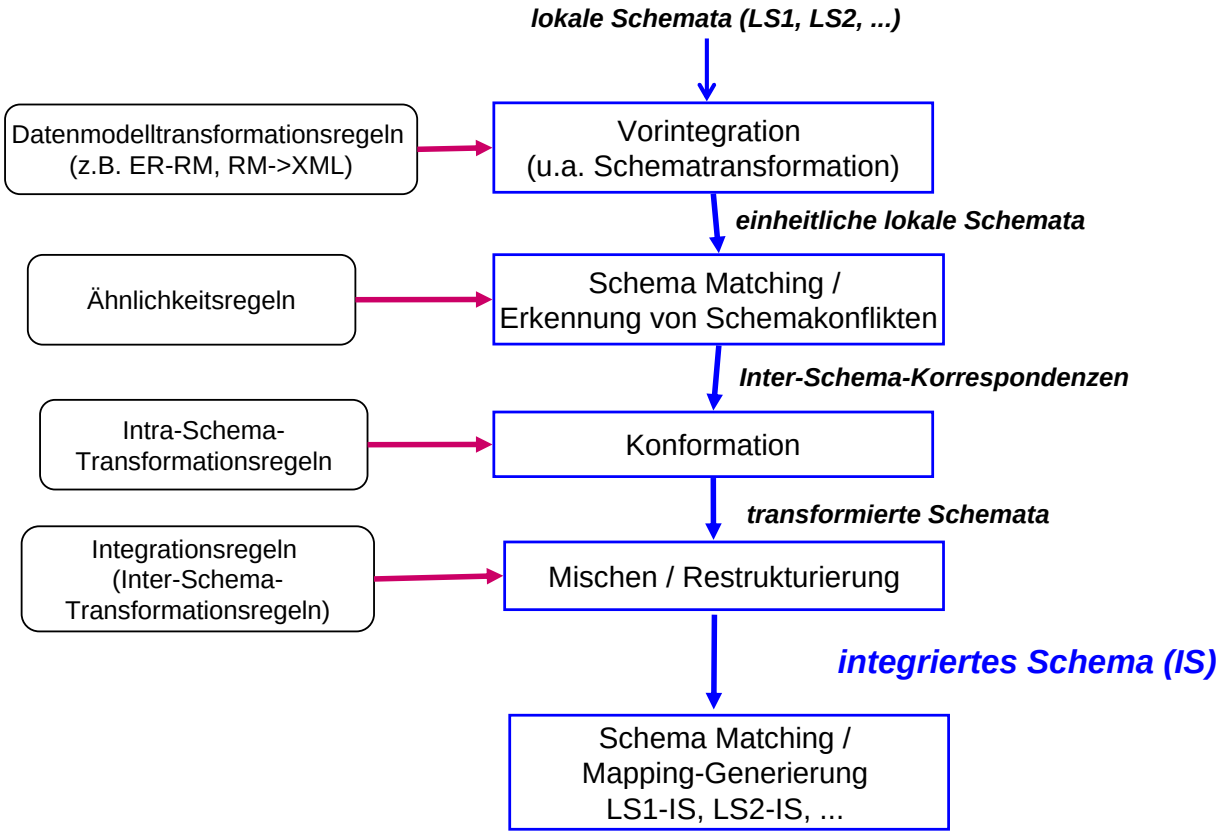
Schemaintegration: Alternativen

- Bottom-Up-Integration (*Global as View*)
  - vollständiges Mischen aller Source-Schemata in globales Schema
  - setzt Abgleich zwischen Source-Schemas voraus, insbesondere Bestimmung von Korrespondenzen / Konflikten
  - globales Schema entspricht Sicht auf die zugrundeliegenden Quellen
  - neue Quelle ändert globales Schema
- Top-Down-Integration (*Local as View*)
  - globales Schema G ist vorgegeben
  - jede Source S wird unabhängig von anderen Sources mit globalem Schema abgeglichen, d.h. ein Mapping G - S erstellt (Mapping beschreibt Inhalt der Quelle)
  - aufwändige Query-Verabeitung bei virtueller Integration
  - G berücksichtigt i.a. nur Teile der lokalen Schemata

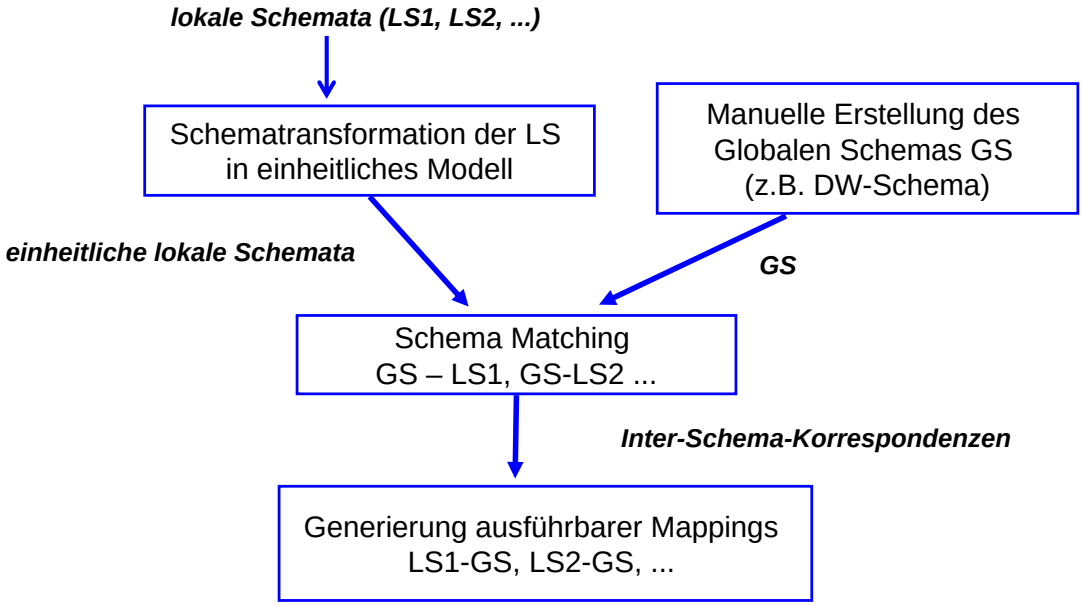




Bottom-Up-Schemaintegration



Top-Down-Schemaintegration

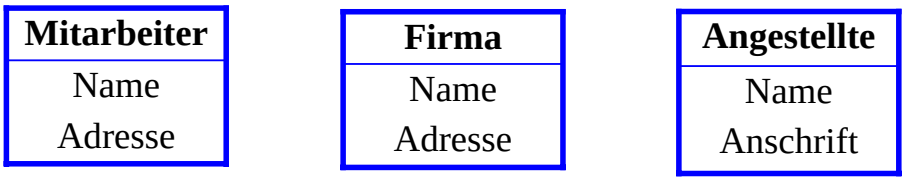


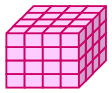
Namenskonflikte

- **Synonyme**: Repräsentation ein und desselben Konzepts durch unterschiedliche Namen:

- **Homonyme**: Nutzung gleicher Namen für verschiedene Konzepte

- **Hypernyme:** Oberbegriffe





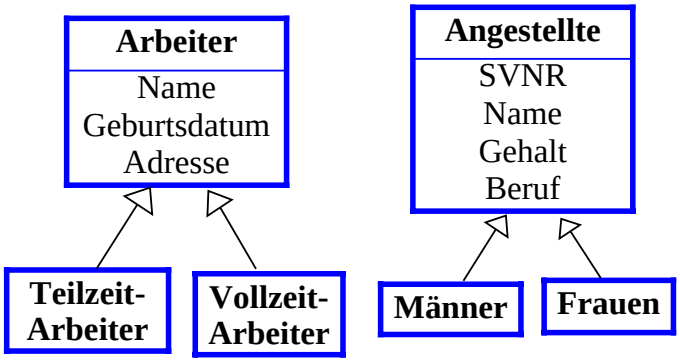
Strukturelle Konflikte

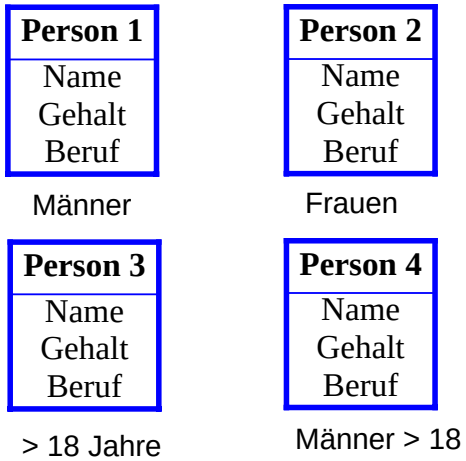
- Relation vs. Relation / Entity vs. Entity
  - **unterschiedliche Schlüssel**
  - **unterschiedliche Attributmengen, fehlende Attribute**
  - **unterschiedliche Abstraktionsebenen** (Generalisierung, Aggregation)
  - **unterschiedliche Realitätsauschnitte** (RWS, real world states)

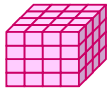
disjunkt (disjoint):

überlappend (overlaps):

enthalten (contains):

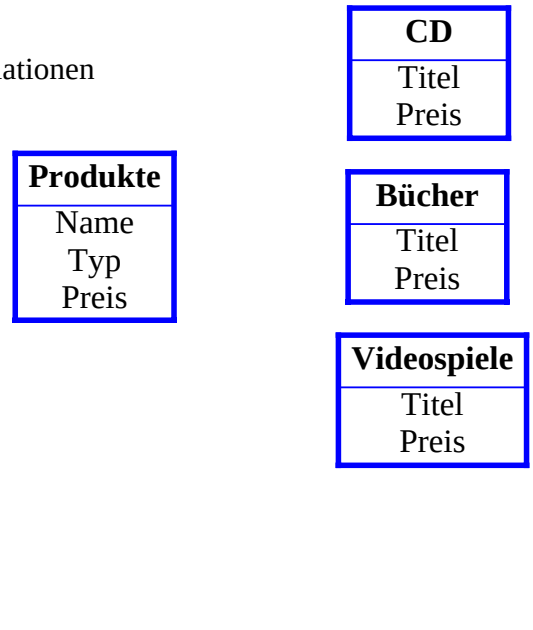




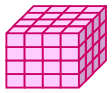


Strukturelle Konflikte (2)

- Attribut vs. Entity-Konflikte
  - Repräsentation von Attributen als eigenständige Entities/Relationen

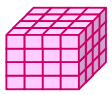


- Attribut vs. Attribut-Konflikte
  - unterschiedliche Datentypen
    Preis (Float) vs. Preis (String)
  - unterschiedliche Detailgrade
    Name vs. Vorname und Nachname
  - unterschiedliche Einheiten: $ vs. Euro
  - unterschiedliche Genauigkeiten: Tausend Euro vs. Euro
  - unterschiedliche Integritätsbedingungen, Wertebereiche, Default-Werte ...
  - Alter >18 vs. Alter > 21
  - unterschiedliche Behandlung von Nullwerten
  - unterschiedliche Verwaltung der referentieller Integrität

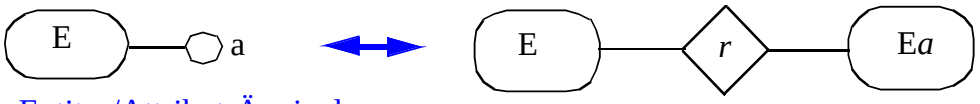


Behandlung von Konflikten

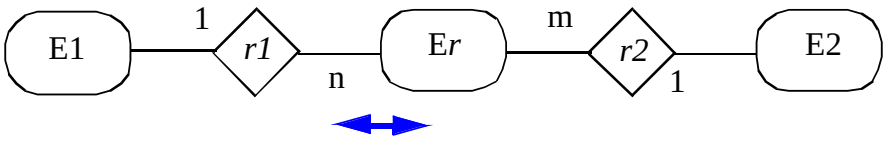
- Konflikterkennung: Vergleich der Schemata
  - Identifikation der ähnlichen/gleichen in den Schemata enthaltenen Information
  - Identifikation der verschiedenen Strukturen, die ähnliche Informationen in den Schemata repräsentieren
- Repräsentation der Interschema-Korrespondenzen
  - Synonyme, Matches: Kunde = Klient
  - Is-A-Korrespondenzen: Angestellte is-a Person
  - RWS-Korrespondenzen: RWS(Produkte) contains RWS(Bücher)
- Behebung von Schemakonflikten v.a. bei Bottom-Up-Integration (Merge) erforderlich
  - Umbenennungen zur Behebung von Namenskonflikten
  - Schemakonformation und –restrukturierungen
  - Top-Down-Integration: Spezifikation notwendiger Transformationen bei Mapping-Erzeugung
- bisherige Schemaintegrationsansätze weitgehend manuell
  - nutzerdefinierte Korrespondenzen und Konfliktbehandlung
  - Nutzung spezifischen Schema-/Domain-Wissens
  - aufwendig / fehleranfällig vor allem für größere Schemata
- (Teil-) Automatisches Schema-Matching
  - für große Schemata sinnvoll
  - Nutzer-Feedback notwendig, jedoch im begrenzten Umfang

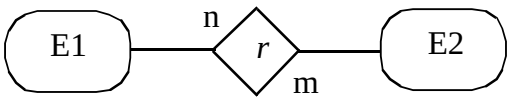


Schema-Konformationstransformationen

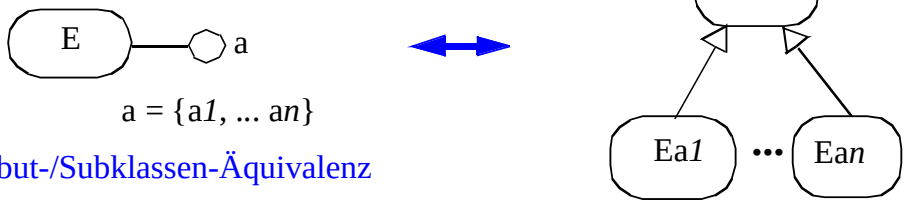


Entity-/Attribut-Äquivalenz

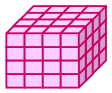




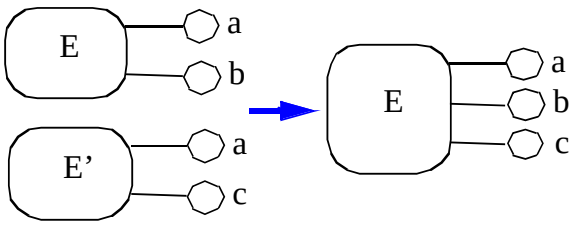
Entity-/Relationship-Äquivalenz



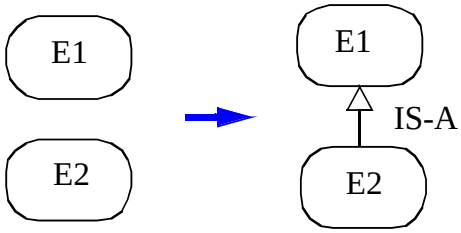
Attribut-/Subklassen-Äquivalenz



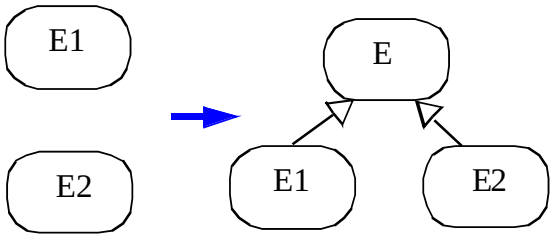
Schema-Merging-Transformationen



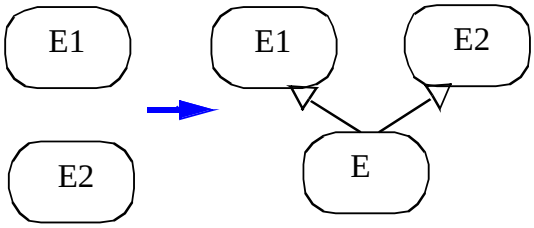
Vereinigung der Attribute



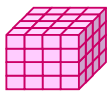
Einführung einer IS-A-Beziehung
RWS(E1) contains RWS(E2)



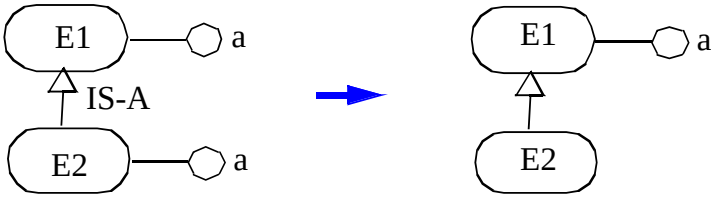
Einführung einer Superklasse
RWS(E1) disjoint RWS(E2)



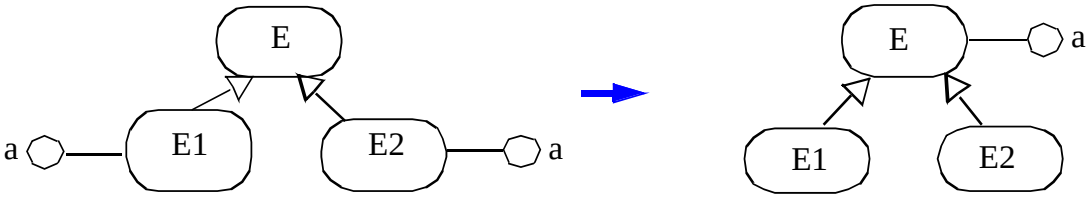
Einführung einer Subklasse
RWS(E1) overlaps RWS(E2)



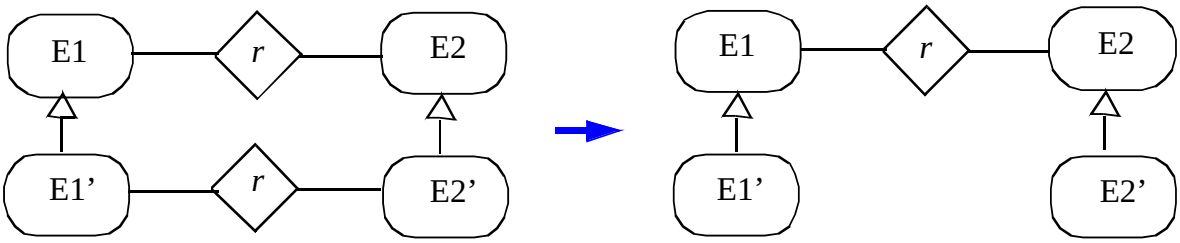
Schema-Restrukturierungstransformationen



Entfernung redundanter Attribute



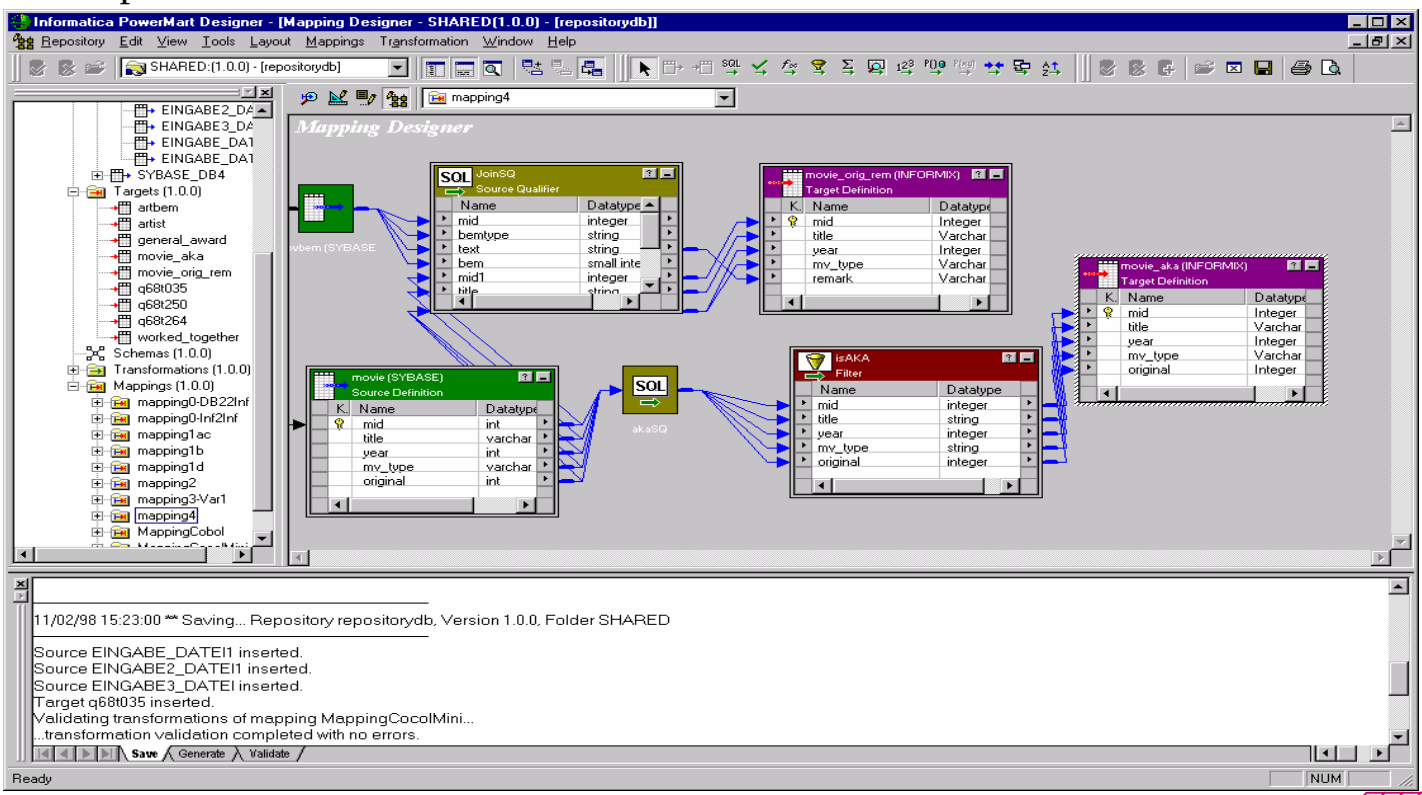
Generalisierung von Attributen



Generalisierung von Beziehungen

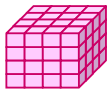
Manuelle Mapping-Definitionen

- Beispiel: PowerMart

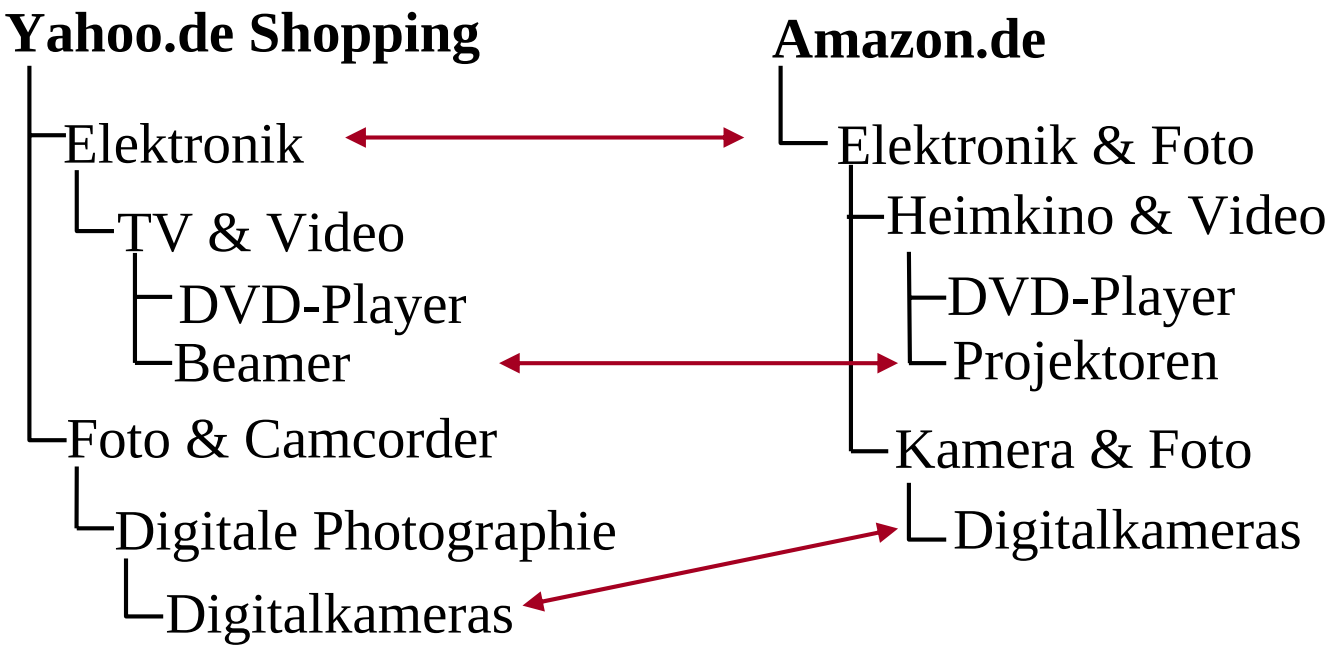


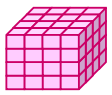
Schema Matching

- Finden semantischer Korrespondenzen zwischen 2 Schemas
  - DB-Schemas, XML-Schemas, Ontologien, …
- Kritischer Schritt in zahlreichen Applikationen
  - Datenintegration: Data Warehouses, Mediatoren, P2P
  - E-Business: XML Message Mapping; Katalogintegration
  - Semantic Web: Ontology Matching
- Input:
  - 2 Schemas S_1 and S_2
  - Evtl. Dateninstanzen zu S_1 und S_2
  - *Hintergrundwissen*
- Output: Mapping zwischen S_1 und S_2
- Skalierbarkeit erfordert semi-automatische Lösungen / Tools!
  - Vollautomatische Lösungen aufgrund semantischer Heterogenität nicht möglich
  - Namensproblematik (Synonyme, Homonyme)
  - Begrenzte Mächtigkeit von Metadaten / Schemasprachen



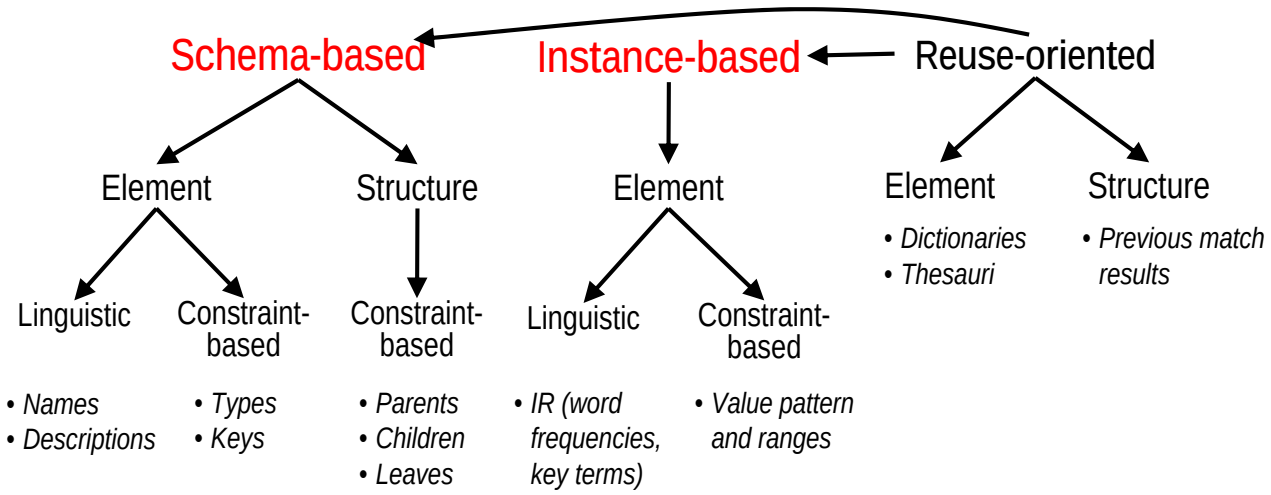
Match-Beispiel: Produktkataloge





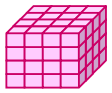
Automatische Match-Ansätze*

- Einzelne Ansätze



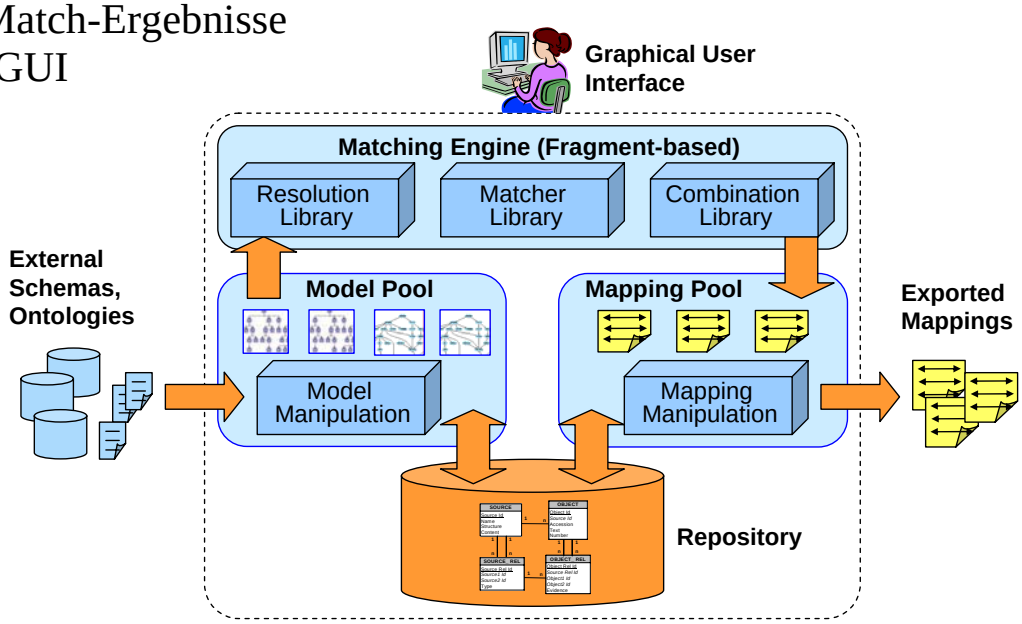
- Kombinierende Ansätze: Hybrid vs. Composite

* Rahm, E., P.A. Bernstein: *A Survey of Approaches to Automatic Schema Matching*. VLDB Journal 10 (4), 2001



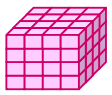
COMA++: Generische Match-Platform*

- Unterstützt XML, relationale Schemas und OWL-Ontologien
- Composite-Ansatz mit flexibler Konfiguration von Matchern
- Match-Strategien für große Schemas: Fragment-basiert / Wiederverwendung vorhandener Match-Ergebnisse
- Umfassendes GUI



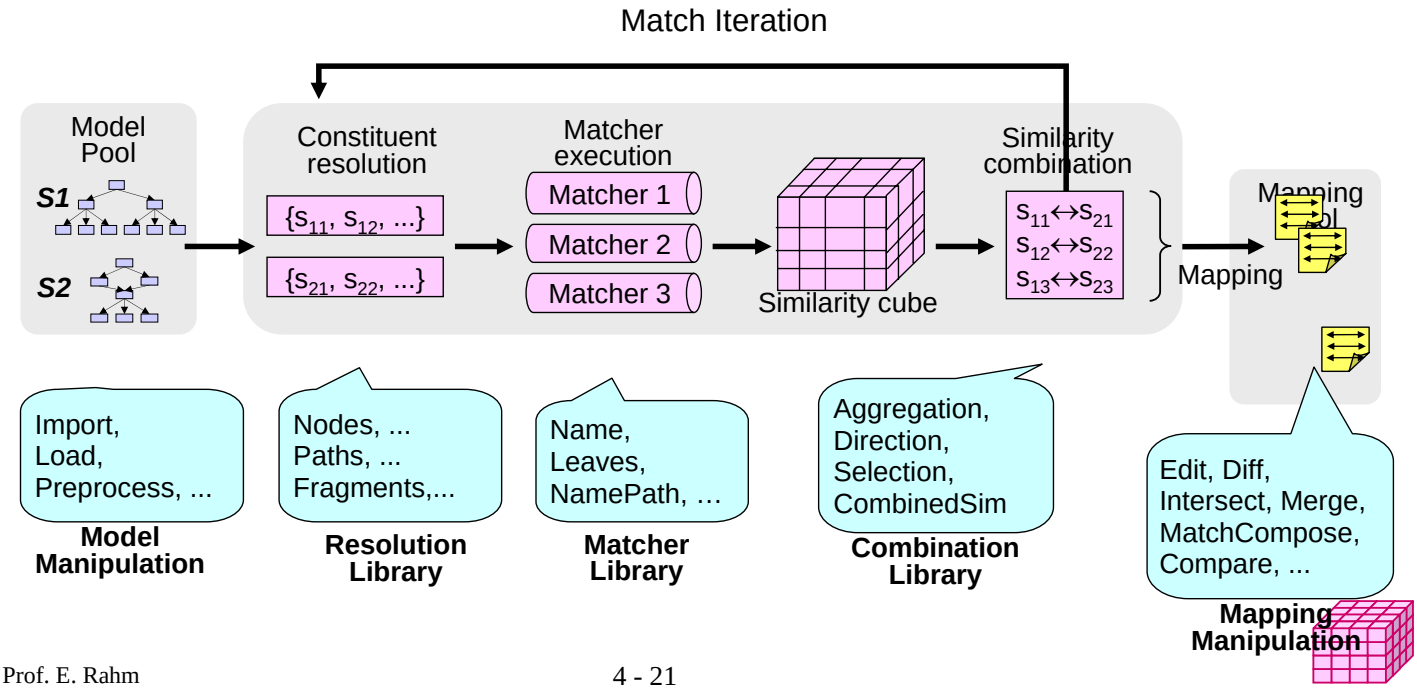
*Do, H.H., E. Rahm: *COMA - A System for Flexible Combination of Schema Matching Approaches*. VLDB 2002

Aumüller D., H.-H. Do, S. Massmann, E. Rahm: *Schema and Ontology Matching with COMA++*. Sigmod 2005



Match-Verarbeitung

- Ausführung verschiedener Match-Algorithmen
- Manipulation/Kombination der erzeugten Mappings



Matcher-Bibliothek

- Basis-Matcher:
  - String-Matcher: Synonym, Type, Trigram, Affix, EditDistance
  - Type-Matcher
  - Taxonomie-Matcher
  - Reuse-Matcher: Wiederverwendung von Mappings
- Hybrid-Matcher: feste Kombination anderer Matcher

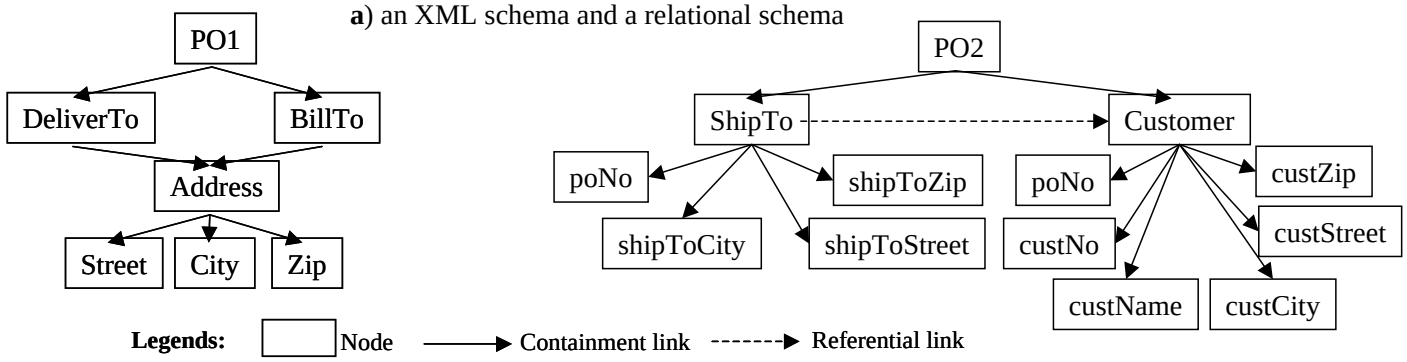
Name	*Constituents*	*Matchers/Sim measures*	*Combination*
Name	Name tokens	Synonym/Taxonomy, Trigram	Avg, Both, Max1, Avg
NameType	Node	Name, Type	Wgt(0.7,03), Both, Max1, Avg
NameStat	Node	Name, Statistics	
Children	Children	NameType	Avg, Both, Max1, Avg
Leaves	Leaves	NameType	
Parents	Parents	Leaves	
Siblings	Siblings	Leaves	
NamePath	Ascendants	Name	

Schema-Beispiel

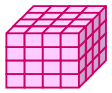
```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:complexType name="PO1" >
  <xsd:sequence>
        <xsd:element name="DeliverTo" type="Address"/>
        <xsd:element name="BillTo" type="Address"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Address" >
   <xsd:sequence>
       <xsd:element name="Street" type="xsd:string"/>
       <xsd:element name="City" type="xsd:string"/>
       <xsd:element name="Zip" type="xsd:decimal"/>
  </xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

```
CREATE TABLE PO2.ShipTo (
  poNo          INT,
  custNo        INT REFERENCES PO2.Customer,
  shipToStreet VARCHAR(200),
  shipToCity    VARCHAR(200),
  shipToZip     VARCHAR(20),
  PRIMARY KEY (poNo)
  ) ;
CREATE TABLE PO2.Customer (
  custNo        INT,
  custName      VARCHAR(200),
  custStreet    VARCHAR(200),
  custCity      VARCHAR(200),
  custZip       VARCHAR(20),
  PRIMARY KEY (custNo)
  ) ;
```

a) an XML schema and a relational schema

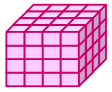


b) Their corresponding graph representation



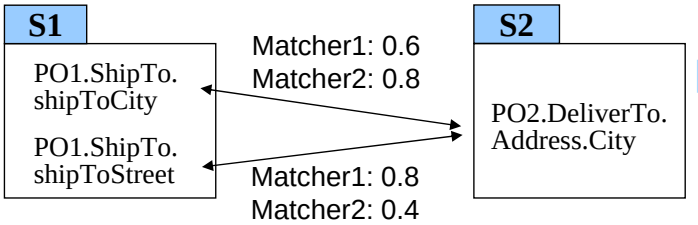
String-Matching: Beispiel

- „Street“ vs. „ShipToStreet“
- Edit Distance:
  - Distanz entspricht Anzahl notwendiger Einfüge-, Lösch- bzw. Änderungsschritte auf Zeichen bezogen auf maximale Stringlänge
  - Ähnlichkeit: 1 – Distanz
- n-Gram (z.B. n=3 / Trigram):
  - Bestimme alle Zeichensequenzen der Länge n (n-grams)
  - Ähnlichkeit entspricht dem Anteil übereinstimmender n-grams

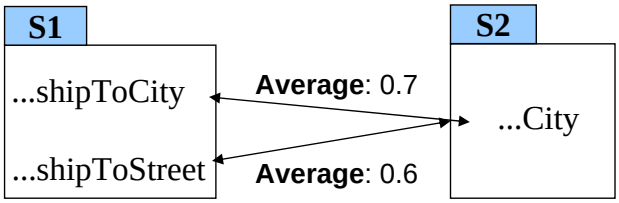


Kombination von Match-Ergebnissen: Beispiel

1. Matcher execution



2. Aggregation



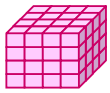
3. Selection

Max1

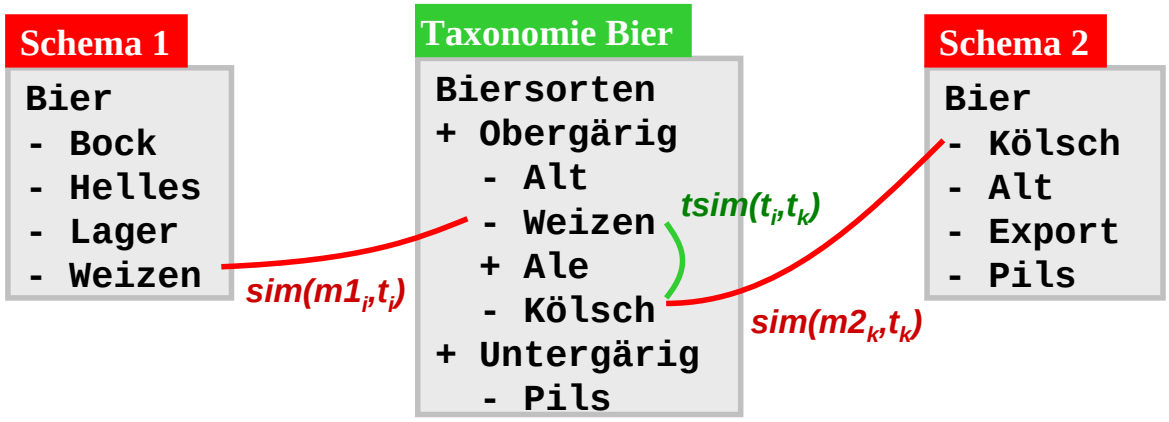
S2 elements	*S1 elements*	*Sim*
...City	...shipToCity	0.7

Threshold(0.5)

S2 elements	*S1 elements*	*Sim*
...City	...shipToCity	0.7
...City	...shipToStreet	0.6



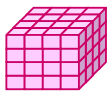
Taxonomie-Matcher



- Taxonomie als Referenz zwischen Schemas/Modellen

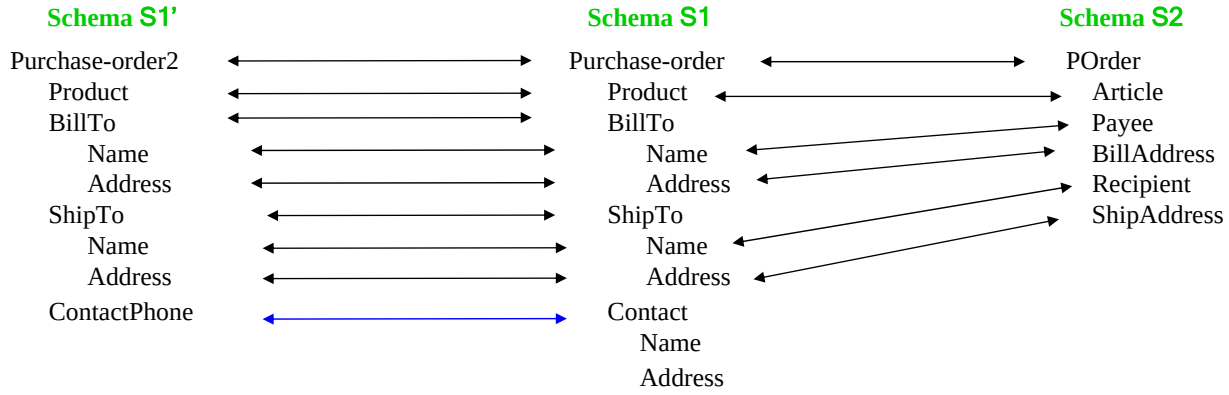
```
sim(Weizen,Kölsch) = 0.8
```

- Ähnlichkeit zweier Schema-Elemente → Kombination aus *sim(m,t)* und *tsim(t,t)*:
  - lexikalischen Ähnlichkeiten der Schema-Elemente *mit* der Taxonomie
  - semantische Ähnlichkeit durch Distanz der Begriffe *innerhalb* der Taxonomie (verschiedene Heuristiken denkbar)



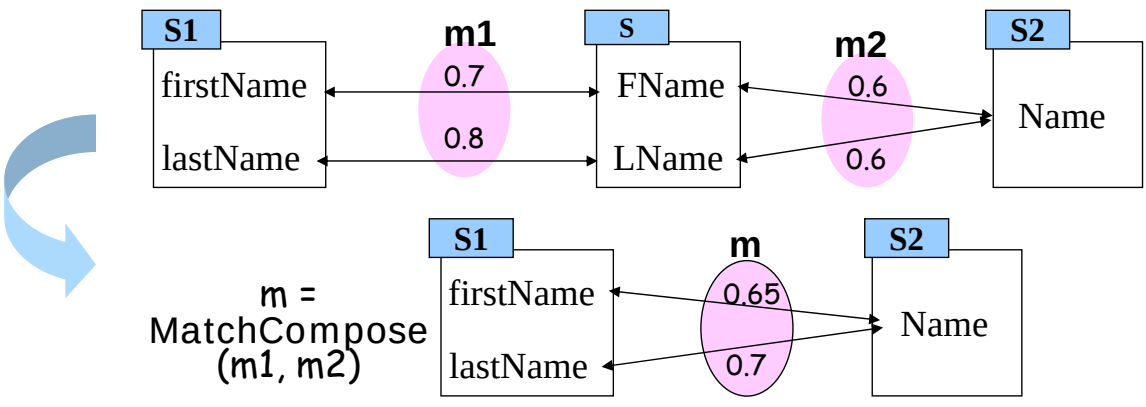
Wiederverwendung (Reuse)

- Nutzung von Hilfsquellen
  - Nutzerspezifizierte Synonymtabellen
  - Allgemeine/Domänenspezische Vokabulare, Wörterbücher
  - Gemeinsame Ontologien
- Nutzung bereits bestätigter Match-Ergebnisse für ähnliche Match-Probleme
  - Speichern von Schemas und Mappings in Repository
  - Besonders vorteilhaft für Abgleich neuer Schema-Versionen (Schema-Evolution)
- Beispiel: Wiederverwendung des vorhandenen (bestätigten) Mappings S1—S2 zur Lösung des neuen Match-Problems S1'—S2



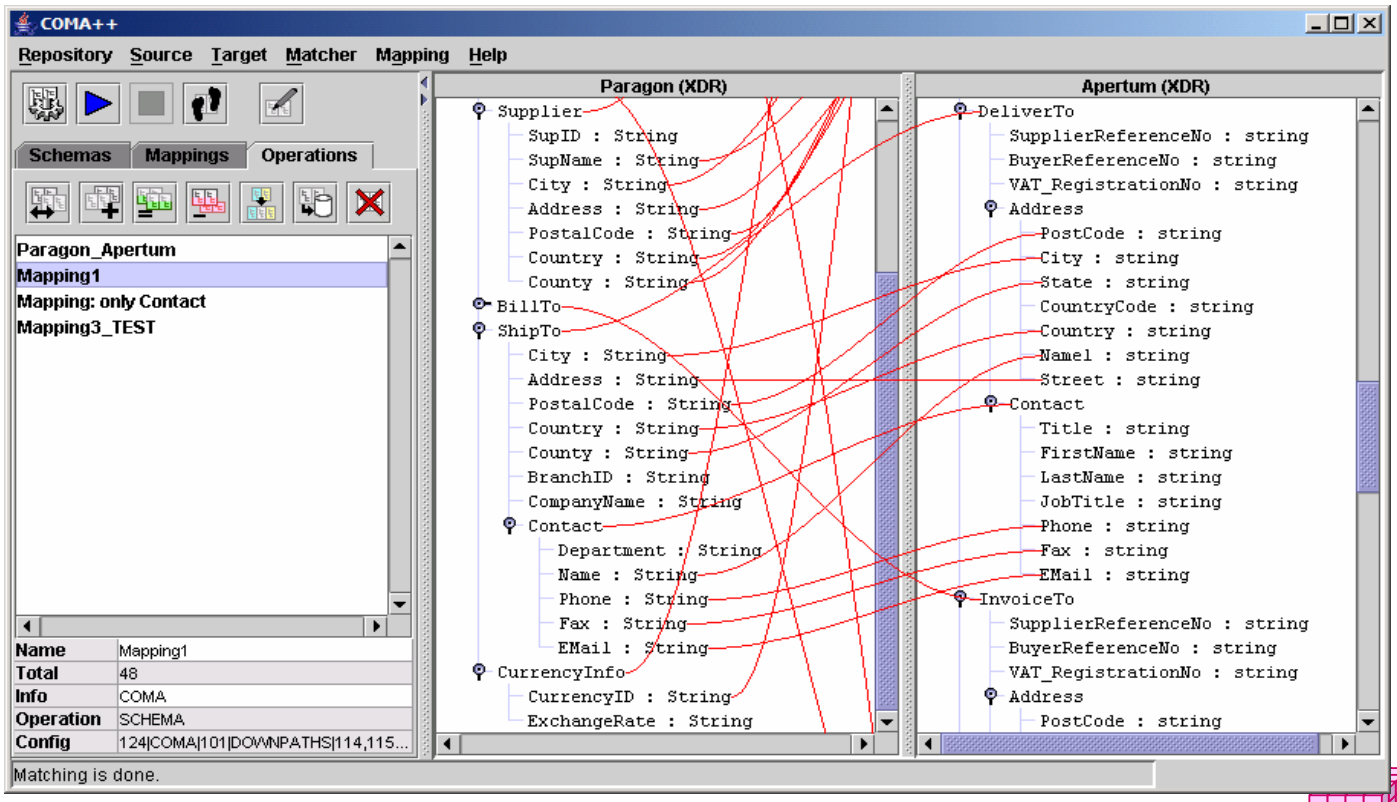
Wiederverwendung von Mappings

- MatchCompose-Operation: Ähnlichkeit/Match als transitive Beziehung



- Wiederverwendungsmöglichkeiten für neues Match-Problem S1-S2
  - Direkte Mappings S1-S2
  - Mapping-Pfade (S1-S3-S2, S2-S4-S5-S1, …)
  - Nutzung ähnlicher Mappings, z.B. mit unterschiedlichen Schemaversionen

Coma++ Nutzerschnittstelle



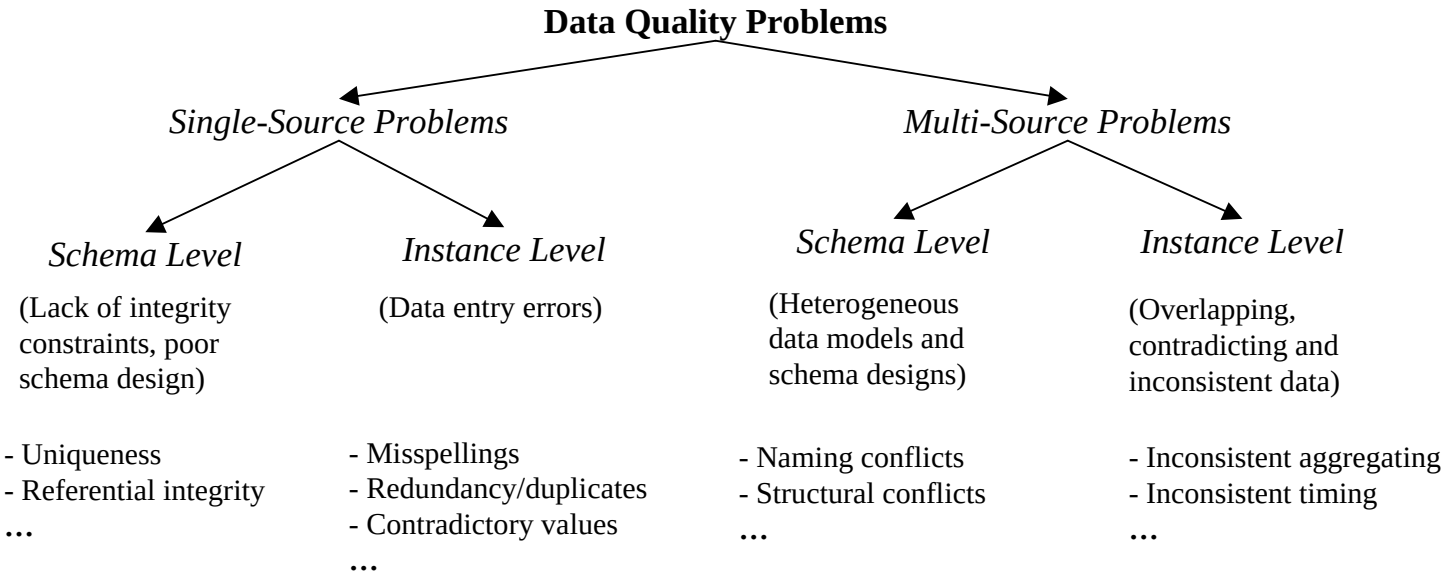
Data Cleaning*

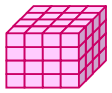
- Datenanalyse
  - Entdeckung von Datenfehlern und -inkonsistenzen
  - manuell bzw. Einsatz von Analyse-Tools
- Definition von Mapping-Regeln und Transformations-Workflows
  - Datentransformationen auf Schemaebene
  - Cleaning-Schritte zur Behandlung von Instanzdaten
  - deklarative Spezifikation erlaubt automatische Generierung von ausführbaren Skripts
- Verifizierung
  - Test der Transformations-Workflows auf Korrektheit und Effektivität aus Kopien/Ausschnitt der Daten
- Transformation
  - regelmäßige Ausführung der spezifizierten und geprüften Transformationsschritte
- ggf. Rückfluss korrigierter Daten in operative Quellsysteme

* E. Rahm, H. H. Do: *Data Cleaning: Problems and Current Approaches.*
IEEE Techn. Bulletin on Data Engineering, Dec. 2000, http://dol.uni-leipzig.de/pub/2000-45

Probleme bezüglich Datenqualität

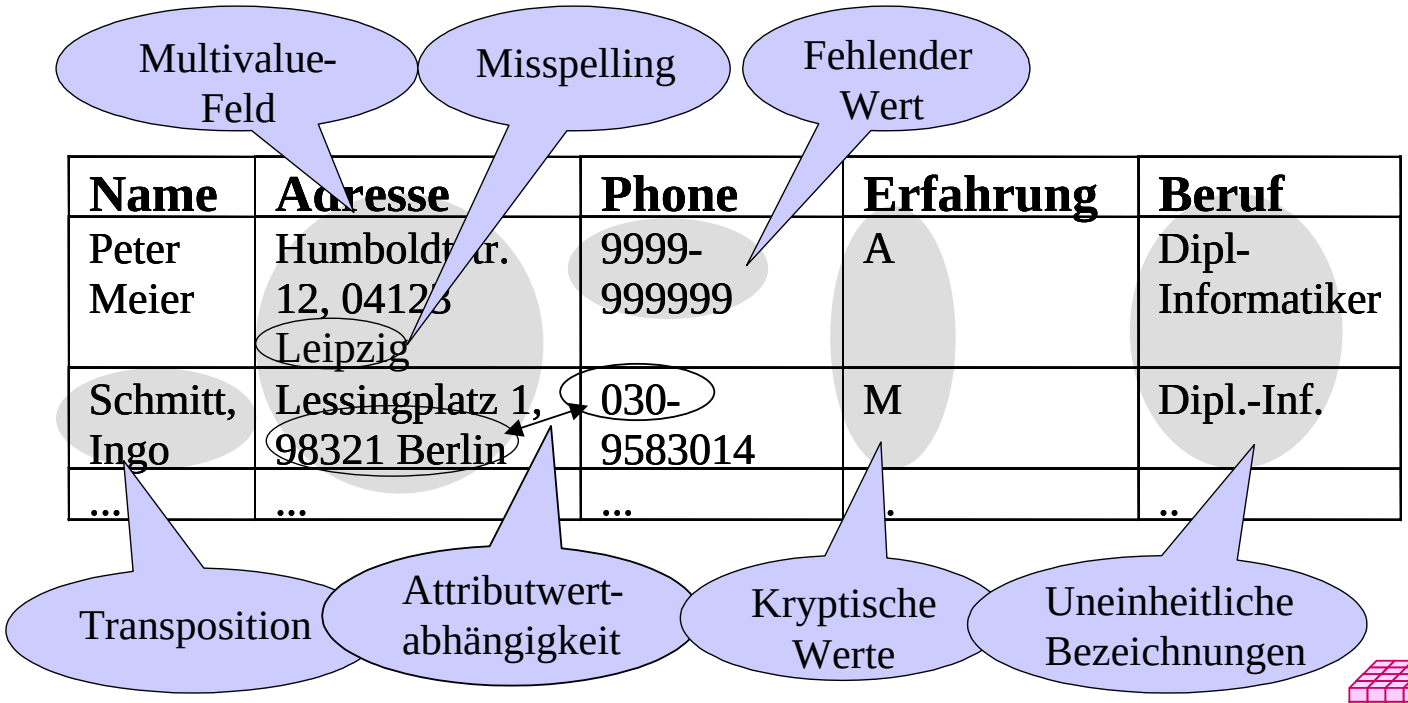
- Probleme auf Schema- und auf Instanzebene
- Probleme bezüglich einer oder mehrerer Datenquellen (Single-Source vs. Multi-Source)

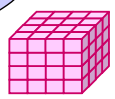




Single-Source Probleme

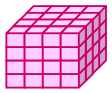
- Ursachen:
  - Fehlen von Schemata (z.B. bei dateibasierter Datenverwaltung), Fehlen von modell- bzw. applikationsspezifischen Integritäts-Constraints
  - Eingabefehler oder unterschiedliche Änderungsstände





Multi-Source-Probleme

- Ursache: überlappende, widersprüchliche bzw. inkonsistente Daten, die unabhängig voneinander in den entsprechenden Quellen erzeugt und gespeichert wurden
- Hauptproblem: überlappende Daten
  - Gängige Bezeichnungen: *Duplikate, Merge/Purge-Problem, Object Identity Problem, Record Linkage*
  - Beschreibung einer Instanz der realen Welt durch mehrere Datensätze unterschiedlicher Quellen
  - Oft nur teilweise Redundanz (einzelne Attribute, nur in Teilmenge der Datenquellen)
- Unterschiedliche Repräsentationen der Instanzdaten
  - verschiedene Wertebereiche (z.B. *Geschlecht = {1,2} vs. Gender = {m,w}*)
  - verschiedene Einheiten (z.B. *Verkauf in EUR vs. Verkauf in Tsd.EUR*)
  - verschiedene Genauigkeiten
- inkonsistentes Timing: unterschiedliche Änderungsstände der Quelldaten
- unterschiedliche Aggregationsstufen der Quelldaten



Beispiel für Multi-Source-Dateninkonsistenzen

Customer (source 1)

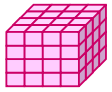
CID	*Name*	*Street*	*City*	*Sex*
11	Kristen Smith	2 Hurley Pl	South Fork, MN 48503	0
24	Christian Smith	Hurley St 2	S Fork MN	1

Client (source 2)

Cno	*LastName*	*FirstName*	*Gender*	*Address*	*Phone/Fax*
24	Smith	Christoph	M	23 Harley St, Chicago IL, 60633-2394	333-222-6542 / 333-222-6599
493	Smith	Kris L.	F	2 Hurley Place, South Fork MN, 48503-5998	444-555-6666

Customers (integrated target with cleaned data)

No	*LName*	*FName*	*Gender*	*Street*	*City*	*State*	*ZIP*	*Phone*	*Fax*	*CID*	*Cno*
1	Smith	Kristen L.	F	2 Hurley Place	South Fork	MN	48503-5998	444-555-6666		11	493
2	Smith	Christian	M	2 Hurley Place	South Fork	MN	48503-5998			24	
3	Smith	Christoph	M	23 Harley Street	Chicago	IL	60633-2394	333-222-6542	333-222-6599		24

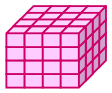


Datenanalyse

- Entdeckung von Fehlern / Verifikation korrekter Werte
- Ableitung von (wirklichen) Metadaten
- Berechnung der Statistiken zu Attributen auf Basis ihrer Instanzen
  - Datentyp, Länge, Maximum und Minimum, Null-, Default-Werte, Kardinalität, ...
  - Ermitteln von Wertebereichen, Häufigkeiten und Mustern von Attributwerte
- Erkennung von Ausreißern, funktionalen Abhängigkeiten

Attribute Values	#occurences
IBM	3000
I.B.M.	360
Intel Bus Mach	213
International Business Machine	36

Actual Data	Pattern	Identified Category
(978) 555-1212	(nnn) nnn-nnnn	US Phone Number
036-55-1234	nnn-nn-nnnn	Social Security Number
abc@web.de	aaa@aaa.aa	Email Address
12.03.1999	nn.nn.nnnn	Date



Behandlung von Single-Source-Problemen

- Extraktion von individuellen Werten aus Freiform-Attributen
  - Parsing und Attribut-Splitting, *z.B. Name -> Vorname / Nachname*
  - Reorganisierung der Wortreihenfolge

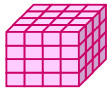
Parsing the elements

Input Data
BETH CHRISTINE PARKER, SLS MGR
REGIONAL PORT AUTHORITY
FEDERAL BUILDING
12800 LAKE CALUMET
HEGEWISCH IL

Parsed data

First Name:	BETH
Middle Name:	CHRISTINE
Last Name:	PARKER
Title:	SLS MGR
Firm:	REGIONAL PORT AUTHORITY
Firm location:	FEDERAL BUILDING
Range:	12800
Street:	LAKE CALUMET
City:	HEGEWISCH
State:	IL

Parsing is a vital step for the cleansing and matching stages. This example shows how parsing identifies and isolates the individual elements of an input record.

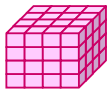


Behandlung von Single-Source-Problemen (2)

- Definition und Einführung von Standardrepräsentationen
  - einheitliches Format für Datums-/Zeit-Angaben
  - einheitliche Groß/Kleinschreibungsform für Namen / String-Attribute
  - einheitliche Abkürzungen, Kodierungsschemas
- Bereitstellung von (Konversions-)Tabellen zur expliziten Werteabbildung
  - z.B. bei unterschiedlichen Aufzählungstypen

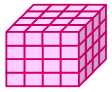
Legacy Value	New Value
IBM	IBM
I.B.M	IBM
Intel Bus Mach	IBM
...	...

- Behandlung von Textfeldern
  - Textreduktion, z.B. durch Wortstammbildung (Stemming), Entfernung von Präfixen/Suffixen, Stop-Wörtern
  - Textersetzung, z.B. durch einheitliche Synonyme
- Validierung / Korrektur durch Hinzuziehen von Hintergrundwissen
  - Überprüfung/Spell checking mit speziellen Datenbanken: Wörterbücher, Datenbanken mit Adressen, Produktbezeichnungen, Akronymen/Abkürzungen, etc.)
  - Nutzung bekannter Attributabhängigkeiten zur Korrektur von fehlenden / falschen Attributwerten



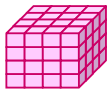
Behandlung von Multi-Source-Problemen

- Hauptproblem: Entdecken von Duplikaten bzw. korrespondierender Objekte (Objekt Matching)
- Durchführung auf aufbereiteten und gesäuberten Quellen
- Hauptschritte: Identifikation von “ähnlichen” Records (Matching) und Mischen (Merge) zu einem Record mit allen relevanten Attribute ohne Redundanz
- *Exact Matching*: Existenz eines Attributs oder eine Attributkombination zur eindeutigen Identifikation der einzelnen Records
  - Nutzung der Standard-Equijoin-Operationen zur Zusammenführung der zugehörigen Records
  - Sortierung der Records über die Schlüsselattribute und Vergleich der benachbarten Records zur Duplikatidenfikation
- *Fuzzy Record Matching*: keine gemeinsamen Schlüsselattribute
  - Suche und Ranking ähnlicher Records auf Basis von (nutzerdefinierten) Matching-Regeln
  - Berechnung von Ähnlichkeitsmaßen zwischen 0 und 1
  - Robustheit gegenüber Tippfehlern und leicht unterschiedlichen Schreibweisen)
  - Verwendung von Distanzfunktionen für String-Vergleiche: Zeichenhäufigkeit, Edit-, Keyboard-Distanz, phonetische Ähnlichkeit (Soundex), ...
  - Kombination der Attribut-Ähnlichkeiten zur Abschätzung der Satz-Ähnlichkeit
  - ggf. Gewichtung der Felder (z.B. hohe Gewichte für Namens-, Adressenfelder beim Matching von Person-Records)



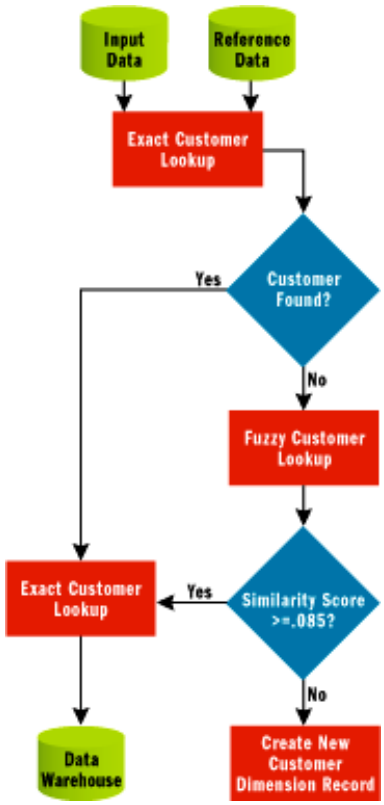
Tool-Unterstützung

- ETL-Tools
  - z.B. CopyManager (Information Builders), Extract(ETI), PowerMart (Informatica), DecisionBase (CA/Platinum), DataTransformationService (Microsoft), etc.
  - Spezialisierung auf Datentransformationen zur Population von Data Warehouses
  - proprietäre Transformationssprachen mit vordefinierten Funktionen für häufig benötigte Konvertierungen
  - Quellzugriff über Standard-Gateways (ODBC, OLE DB) oder native Schnittstellen für Daten- und Metadatenextraktion
  - wenig eingebaute Cleaning-Funktionalität (proprietäre APIs zum Aufruf eigener Routinen)
- Datenanalyse-Tools, z.B. Migration Architect, Information Discovery
- Cleaning-Tools
  - Data Reengineering-Tools (z.B. Vality Integrity): Datenstandardisierung mit Transformationen für Attribute/Sätze, z.B. split, delete, merge
  - spezielle Tools für Namen- und Adressen-Cleaning, z.B. Trillium mit über 200.000 Regeln
  - Duplikat-Eliminierung (auf bereits gesäuberten Datenquellen): DataCleanser (EDD), Merge/Purge Library (Sagent/QM Software), MasterMerge (Pitnew Bowes)
    - Bereitstellung mehrerer Ansätze zum Attributvergleich und zur Kombination der Ähnlichkeitsmaße



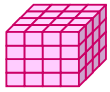
MS SQL-Server 2005: Data Cleaning Operatoren

- Bestandteil von SQL-Server Integration Services (SSIS; vormals DTS)*
  - Definition komplexer ETL-Workflows
  - zahlreiche Operatoren
- Fuzzy Lookup
  - "Fuzzy Join" zwischen Eingaberelation und sauberen Sätzen einer Referenztabelle
  - Parameter: Schwellwerte bzgl. String-Ähnlichkeit (Edit Distance) sowie Gewichte zur Kombination von Ähnlichkeiten



- Fuzzy Grouping
  - Gruppierung ähnlicher Sätze (potentielle Duplikate) innerhalb einer Tabelle über String-Matching (Edit Distance)

* http://msdn.microsoft.com/sql/bi/integration/default.aspx

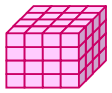


MOMA-Prototyp (U Leipzig)

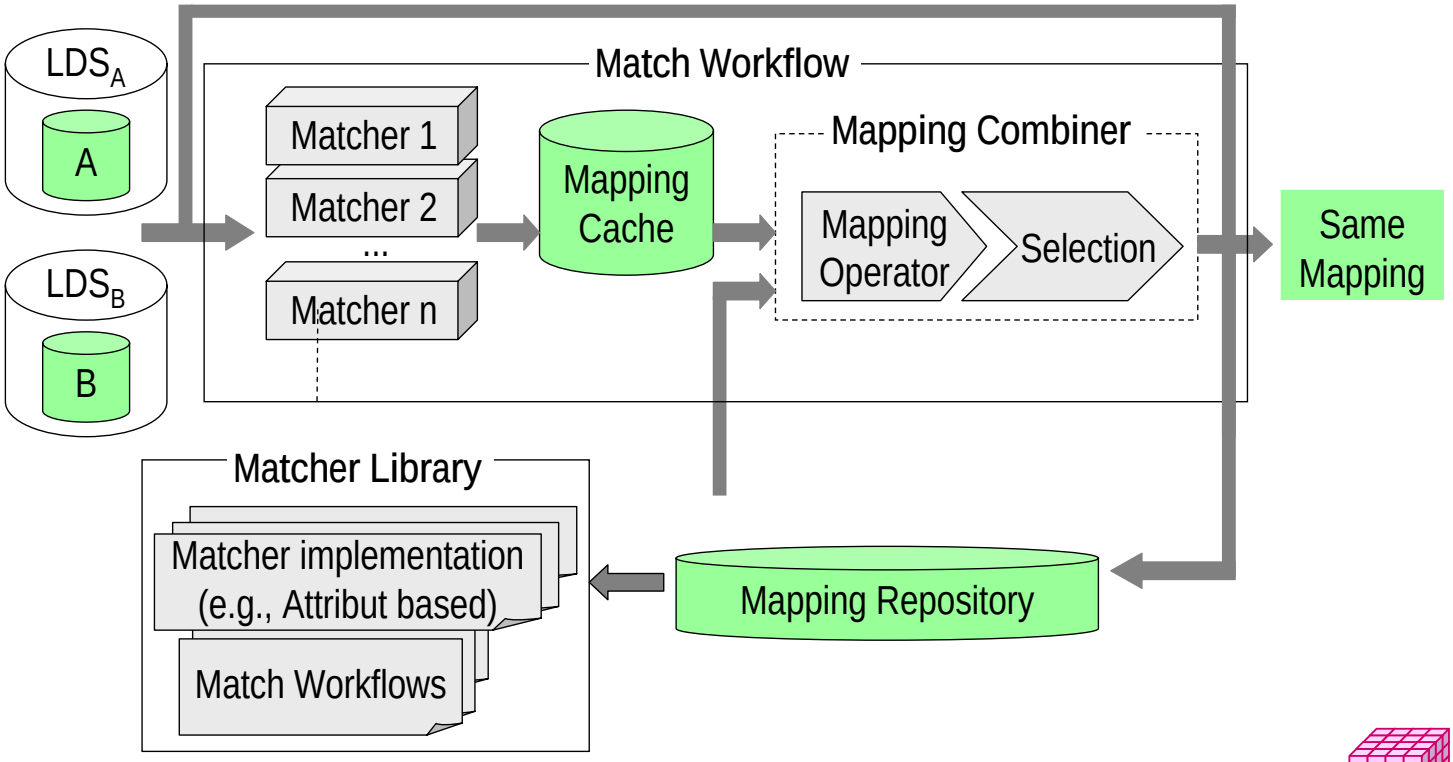
- MOMA = **M**apping based **O**bject **Ma**tching
- Framework für Objekt-Matching (Fuzzy Match)
  - Unterstützung komplexer Match-Workflows
  - Kombination mehrerer Matcher / Ergebnisse
  - Wiederverwendung bereits berechneter Mappings
  - Unterstützuung heterogener Datenquellen, v.a. von Web-Daten
- Mapping-basierter Ansatz
  - Match-Ergebnis ist Mapping bestehend aus Instanz-Korrespondenzen („Same mapping“)

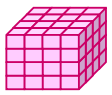
$Quelle_A$	$Quelle_B$	Sim
a_1	b_1	0.9
a_2	b_2	0.7
a_3	b_3	1

  - bereits existierende Mappings (z.B. Web-Links) werden ausgenutzt
  - semantische Beziehungen zwischen Objekten („association mappings“) werden durch spezielle Matcher bzw. Workflows ausgenutzt
- Implementierung im Rahmen der iFuice-P2P-Architektur zur Datenintegration



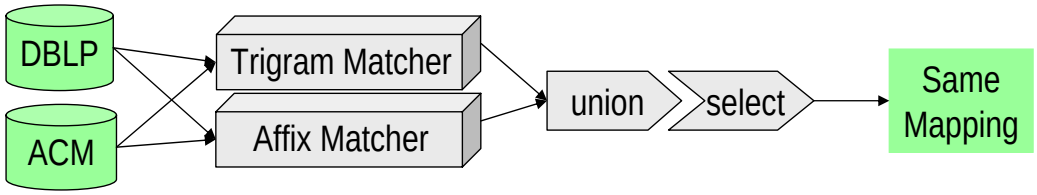
MOMA-Architektur



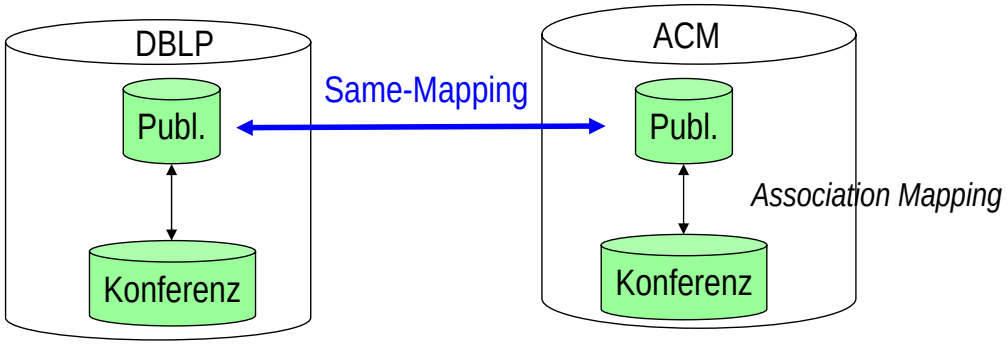


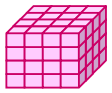
Beispiel-Workflows

- Kombination unabhängiger Attribut-Matcher-Ausführungen



- Komposition von Mappings, z.B. DBLP – IEEE und IEEE – ACM
- Nutzung existierender Mappings sowie semantischer Mapping-Typen
  - Beispielaufgabe: Bestimme Objekt-Matching für Konferenzen





Zusammenfassung

- ETL als komplexer, aufwendiger Integrationsprozeß
- Schema- und Datenintegration / Data Cleaning
  - zahlreiche Schema- und Datenkonflikte
  - begrenzte Automatisierbarkeit bezüglich Konflikterkennung und -behandlung
  - möglichst deskriptive Spezifikation aller Datentransformationen zur Behandlung von Schema- und Datenkonflikten
- Fokussierung auf Data Warehouse-spezifisches Zielschema erleichtert Schemaintegration
  - Top-Down-Schemaintegration
  - keine vollständige Integration aller Quell-Schemata erforderlich
- wichtiges Teilproblem: Schema-Matching
  - Nutzung und Kombination mehrerer Lösungsalgorithmen (Matcher)
  - Reuse früherer Match-Ergebnisse
- Unterscheidung quell-lokaler und -übergreifender Datenkonflikte
  - Data Cleaning zunächst auf einzelnen Quellen
  - Duplikat-Identifikation und –Behandlung (Objekt Matching)