Einführung Informationsintegration

Komplexe Informationssysteme

Fabian Panse

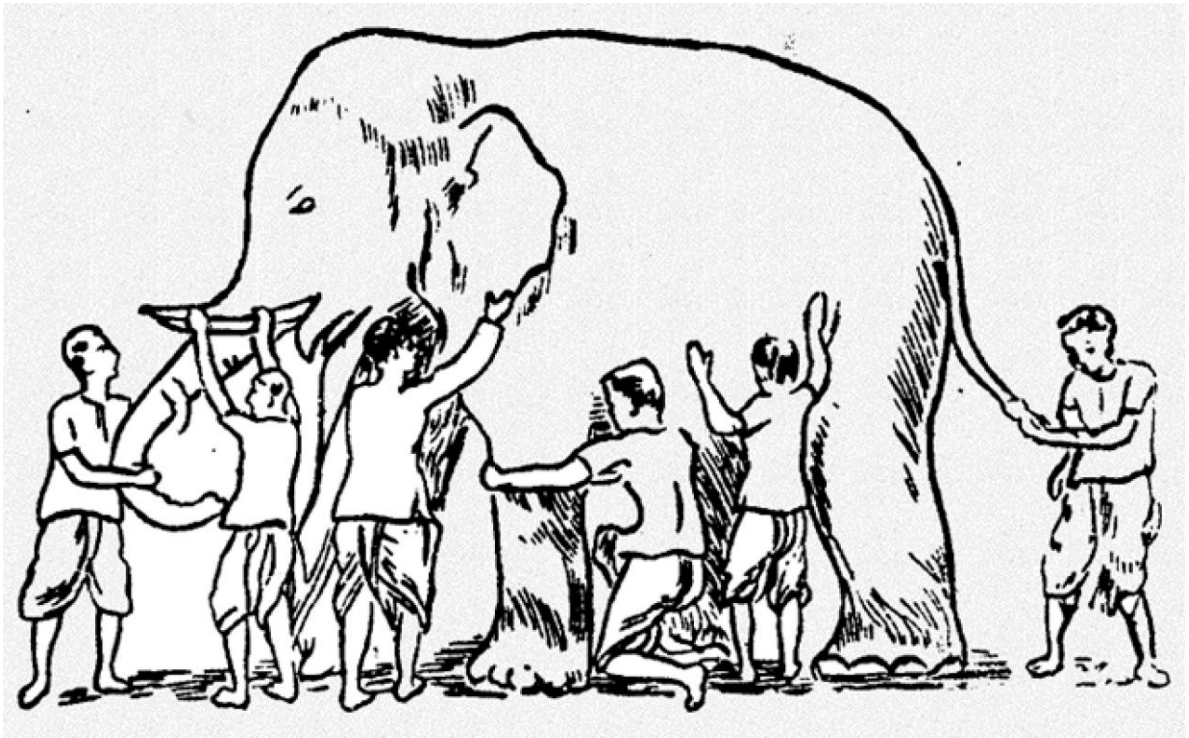
panse@informatik.uni-hamburg.de

Universität Hamburg

Universität Hamburg

DER FORSCHUNG | DER LEHRE | DER BILDUNG

Der Elefant und die sechs blinden Männer



Quelle: Laura M. Haas. Beauty and the beast: The theory and practice of information integration. ICDT, 2007.

Informationsbedarf und potentielle Einsparungen

1 von 3

Manager treffen häufiger Entscheidungen auf Basis von Informationen, denen sie nicht vertrauen oder die sie gar nicht haben.

1 von 3

Manager haben nicht Zugriff zu den erforderlichen Informationen.

Quelle: IBM: Break Away with Business Analytics and Optimization Study, IDC

Informationsbedarf und potentielle Einsparungen

1 von 3

Manager treffen häufiger Entscheidungen auf Basis von Informationen, denen sie nicht vertrauen oder die sie gar nicht haben.

1 von 3

Manager haben nicht Zugriff zu den erforderlichen Informationen.

5700 USD/a

Zeitaufwand je Wissensarbeiter für Umformatierung von Informationen zwischen Anwendungen.

5300 USD/a

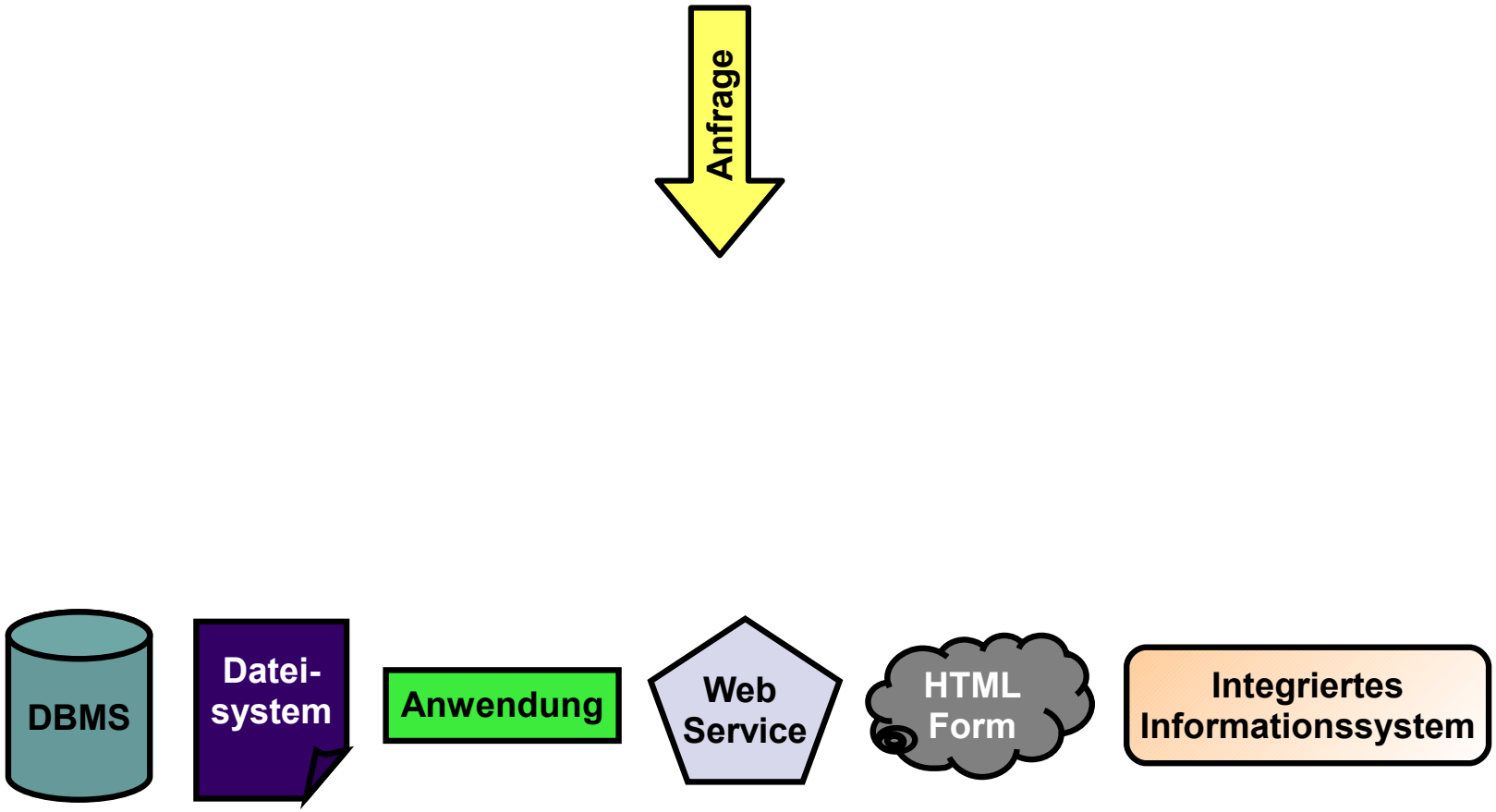
Zeitaufwand je Wissensarbeiter für Informationssuche.

Quelle: IBM: Break Away with Business Analytics and Optimization Study, IDC

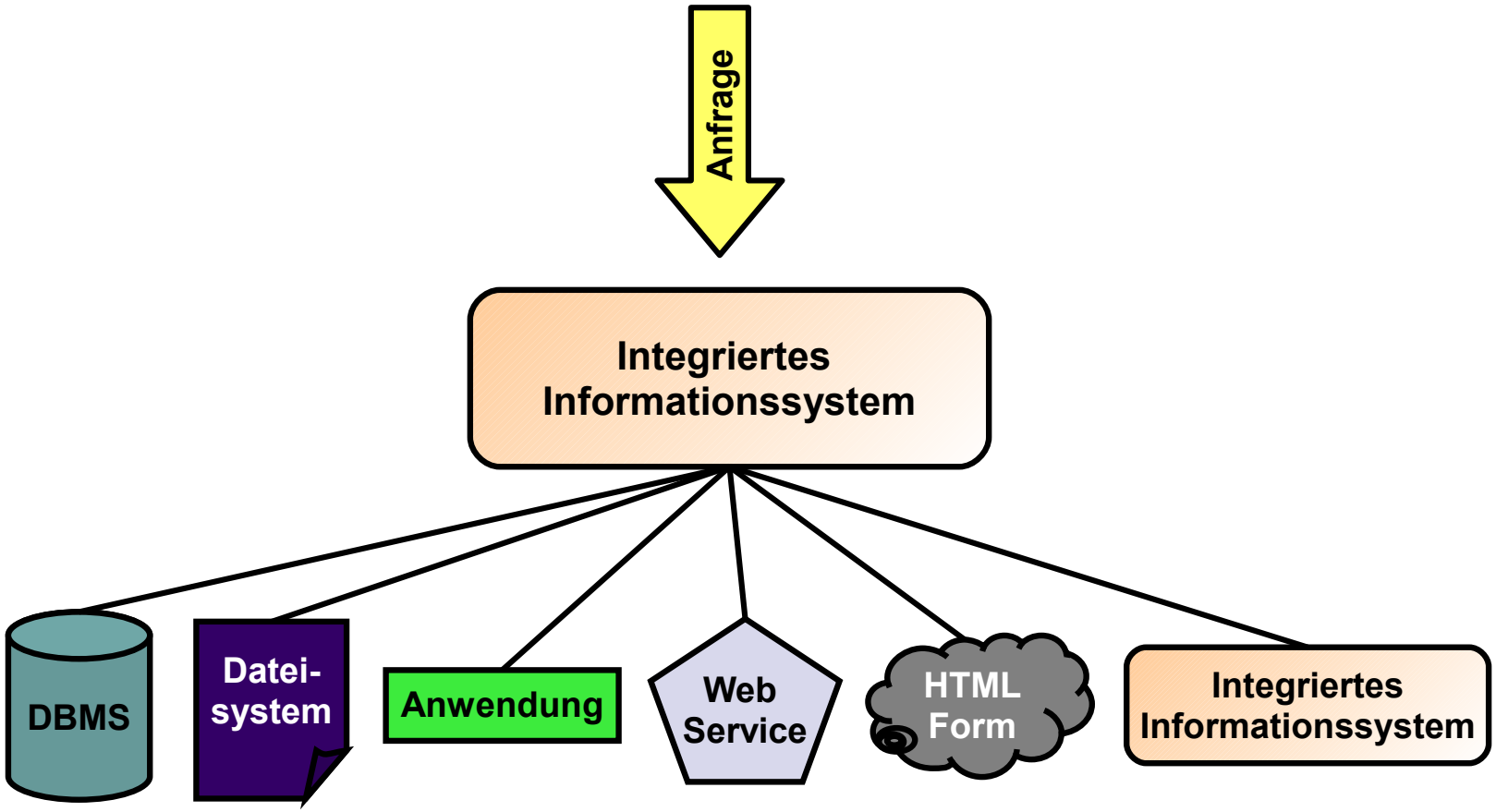
Integrierte Informationssysteme

Integrierte Informationssysteme

Integrierte Informationssysteme



Integrierte Informationssysteme



Agenda

Organisatorisches

- Voraussetzungen:
  - Grundlagen Datenbanken
    (relationale DBMS, SQL, XML)
  - Interesse an aktuellem Thema
- Prüfungsinhalt: ausschließlich Vorlesungsstoff
- Übungen: Saalübungen

- Acknowledgements: Angelehnt an Folien von
  - Dr. Armin Roth (IBM)
  - Prof. Dr. Melanie Herschel (Univ. Stuttgart)
  - Folien zum Buch *Principles of Data Integration*

Organisatorisches

1. Woche (04.09.2017 - 09.09.2017)

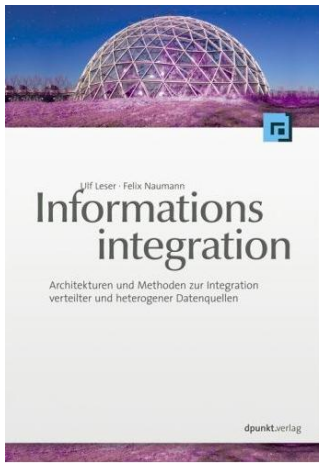
Montag	Dienstag	Mittwoch	Donnerstag	Freitag
Vorlesung	Vorlesung	Vorlesung	Vorlesung	-
9:30 13:30	9:30 13:30	9:30 13:30	9:30 13:30	- -

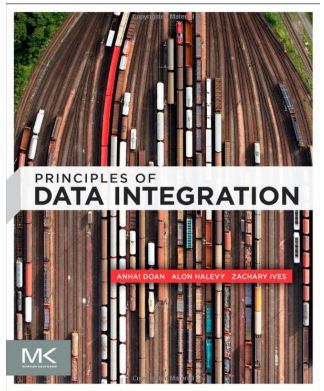
2. Woche (11.09.2017 - 15.09.2017)

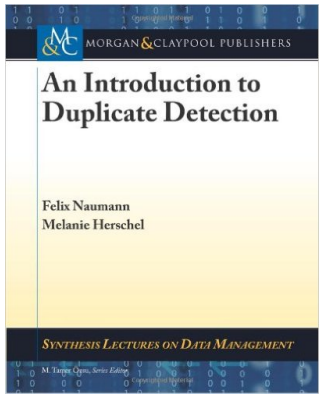
Montag	Dienstag	Mittwoch	Donnerstag	Freitag
Vorlesung	Vorlesung	Vorlesung	Seminar	Seminar
9:30 13:30	9:30 13:30	9:30 13:30	9:30 16:30	9:30 16:30

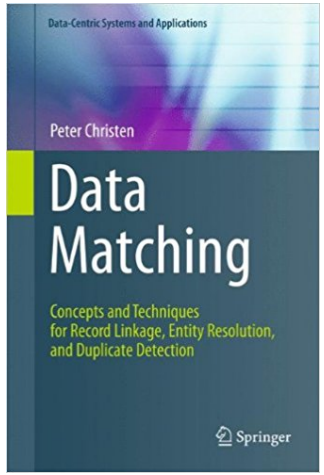
Literatur

- Ulf Leser und Felix Naumann. Informationsintegration. dpunkt.verlag, 2006 [LN06]
- Anhai Doan, Alon Halevy, Zachary Ives. Principles of Data Integration. Morgan Kaufmann, 2012 [DHI12]
- Felix Naumann und Melanie Herschel. Introduction to Duplicate Detection. Morgan & Claypool, 2010 [NH10]
- Peter Christen. Data Matching. Springer, 2012 [Chr12]









Agenda

Was ist Informationsintegration? [LN06]

Informationsintegration ist die korrekte, vollständige und effiziente Zusammenführung von Informationen verschiedener, heterogener Quellen zu einer einheitlichen und strukturierten Informationsmenge zur effektiven Interpretation durch Nutzer und Anwendungen.

Wofür brauchen wir Informationsintegration? [DHI12]

Wofür brauchen wir Informationsintegration? [DHI12]

- Informationssysteme in vielen Lebensbereichen.

Wofür brauchen wir Informationsintegration? [DHI12]

- Informationssysteme in vielen Lebensbereichen.
- In der Realität sind Informationssysteme häufig unabhängig voneinander konzipiert nur um später festzustellen, dass die von ihnen bereitgestellten Informationen kombiniert betrachtet werden müssen.

Wofür brauchen wir Informationsintegration? [DHI12]

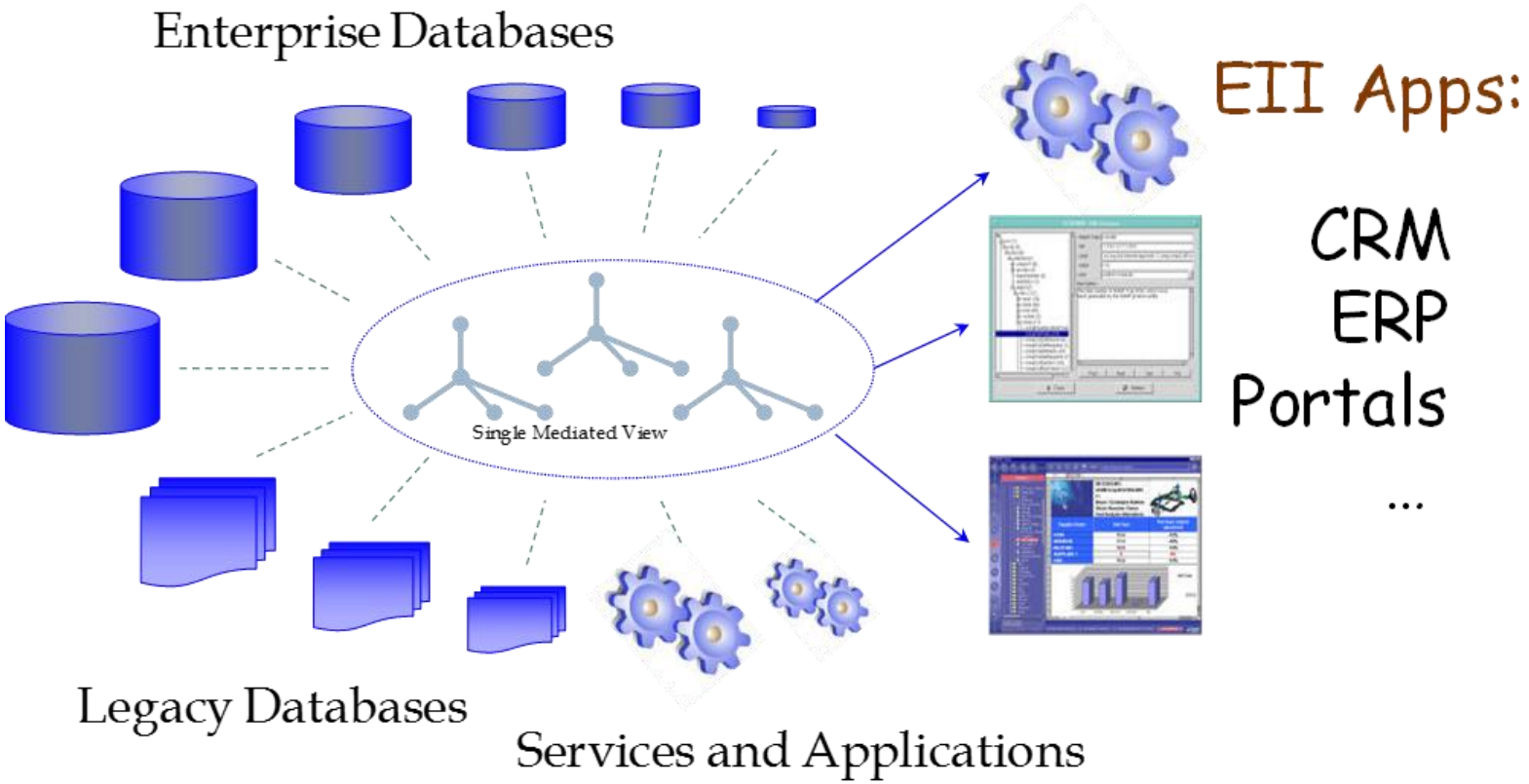
- Informationssysteme in vielen Lebensbereichen.
- In der Realität sind Informationssysteme häufig unabhängig voneinander konzipiert nur um später festzustellen, dass die von ihnen bereitgestellten Informationen kombiniert betrachtet werden müssen.
- Zu diesem Zeitpunkt benutzten die Systeme unterschiedliche Datenmodelle, unterschiedliche Schemata und bieten oft nur einen eingeschränkten Zugriff auf ihre Daten.

Wofür brauchen wir Informationsintegration? [DHI12]

- Informationssysteme in vielen Lebensbereichen.
- In der Realität sind Informationssysteme häufig unabhängig voneinander konzipiert nur um später festzustellen, dass die von ihnen bereitgestellten Informationen kombiniert betrachtet werden müssen.
- Zu diesem Zeitpunkt benutzten die Systeme unterschiedliche Datenmodelle, unterschiedliche Schemata und bieten oft nur einen eingeschränkten Zugriff auf ihre Daten.
- Das Ziel der Informationsintegration ist es verschiedene Informationsquellen unter einer Sicht zu vereinen.

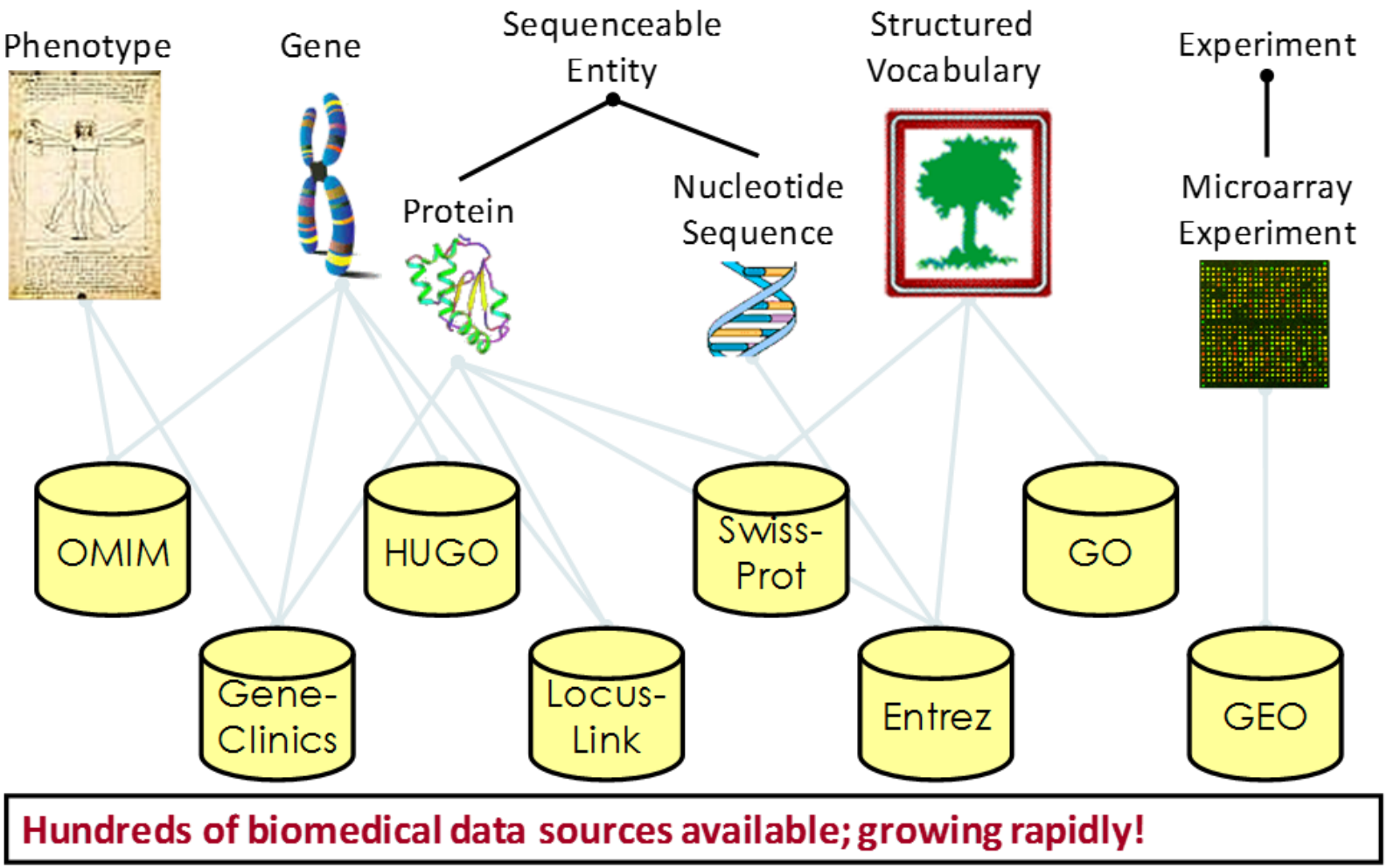
Agenda

Anwendungen: Business



Quelle: Doan, Halevy and Ives. Principles of data Integration (Slides), 2012 [DHI12]

Anwendungen: Biowissenschaften

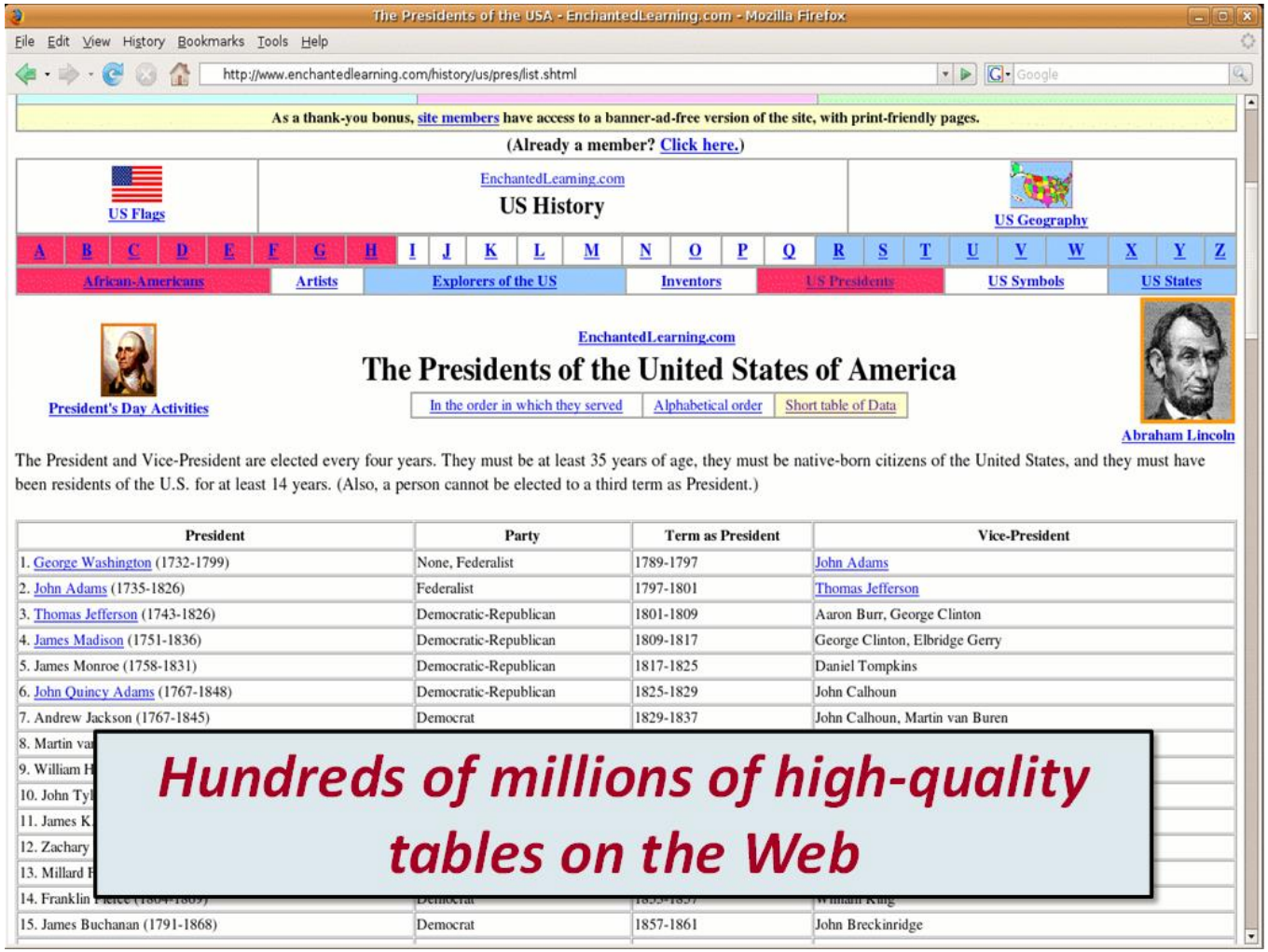


Quelle: Doan, Halevy and Ives. Principles of data Integration (Slides), 2012 [DHI12]

Anwendungen: Web Data Integration

Quelle: Doan, Halevy and Ives. Principles of data Integration (Slides), 2012 [DHI12]

Anwendungen: Historische Daten



Quelle: Doan, Halevy and Ives. Principles of data Integration (Slides), 2012 [DHI12]

Warum ist eine solche Integration so schwer? [DHI12]

- System-bedingte Gründe:
  - Verschiedene Plattformen
  - Anfragebearbeitung über mehrere (autonome) Systeme

Warum ist eine solche Integration so schwer? [DHI12]

- System-bedingte Gründe:
  - Verschiedene Plattformen
  - Anfragebearbeitung über mehrere (autonome) Systeme
- Soziale Gründe:
  - Finden relevanter Daten in Unternehmen
  - Beschaffen relevanter Daten in Unternehmen
  - Menschen zur Zusammenarbeit überreden

Warum ist eine solche Integration so schwer? [DHI12]

- System-bedingte Gründe:
  - Verschiedene Plattformen
  - Anfragebearbeitung über mehrere (autonome) Systeme
- Soziale Gründe:
  - Finden relevanter Daten in Unternehmen
  - Beschaffen relevanter Daten in Unternehmen
  - Menschen zur Zusammenarbeit überreden
- Logik-bedingte Gründe:
  - Schema- und Datenheterogenität
  - Dies ist unabhängig von der jeweiligen Integrationsarchitektur

Agenda

Beispiel: Unternehmen FullServe

Beispiel: Unternehmen FullServe

- Tätigkeitsfeld:
  - Internetprovider
  - Verkauf von Produkten wie Routern, voice-over-IP phones, etc.

Beispiel: Unternehmen FullServe

- Tätigkeitsfeld:
  - Internetprovider
  - Verkauf von Produkten wie Routern, voice-over-IP phones, etc.
- Verschiedene Abteilungen mit eigenen Datenbanken:

Beispiel: Unternehmen FullServe

- Tätigkeitsfeld:
  - Internetprovider
  - Verkauf von Produkten wie Routern, voice-over-IP phones, etc.
- Verschiedene Abteilungen mit eigenen Datenbanken:
  - *Human Resource Department*:
    - Datenbank über Angstellte (Vollzeit und Teilzeit)
    - Datenbank über Bewerbungsverfahren

Beispiel: Unternehmen FullServe

- Tätigkeitsfeld:
  - Internetprovider
  - Verkauf von Produkten wie Routern, voice-over-IP phones, etc.
- Verschiedene Abteilungen mit eigenen Datenbanken:
  - *Human Resource Department*:
    - Datenbank über Angstellte (Vollzeit und Teilzeit)
    - Datenbank über Bewerbungsverfahren
  - *Training and Development Department*:
    - Datenbank über Trainingskurse

Beispiel: Unternehmen FullServe

- Tätigkeitsfeld:
  - Internetprovider
  - Verkauf von Produkten wie Routern, voice-over-IP phones, etc.
- Verschiedene Abteilungen mit eigenen Datenbanken:
  - *Human Resource Department*:
    - Datenbank über Angstellte (Vollzeit und Teilzeit)
    - Datenbank über Bewerbungsverfahren
  - *Training and Development Department*:
    - Datenbank über Trainingskurse
  - *Sales Department*:
    - Datenbank über angebotene Dienste, Kunden und Verträge
    - Datenbank über verkaufte Produkte

Beispiel: Unternehmen FullServe

- Tätigkeitsfeld:
  - Internetprovider
  - Verkauf von Produkten wie Routern, voice-over-IP phones, etc.
- Verschiedene Abteilungen mit eigenen Datenbanken:
  - *Human Resource Department*:
    - Datenbank über Angstellte (Vollzeit und Teilzeit)
    - Datenbank über Bewerbungsverfahren
  - *Training and Development Department*:
    - Datenbank über Trainingskurse
  - *Sales Department*:
    - Datenbank über angebotene Dienste, Kunden und Verträge
    - Datenbank über verkaufte Produkte
  - *Customer Care Department*:
    - Datenbank über Anrufe an das *Help-Line Center*

Beispiel: Unternehmen FullServe

Employee Database

FullTimeEmps(ssn, empID, firstName,
middleName, lastName)
Hire(empID, hireDate, recruiter)
TempEmployees(ssn, hireStart,
hireEnd, name, hourlyRate)

Quelle: Doan, Halevy and Ives. Principles of data Integration, 2012 [DHI12]

Beispiel: Unternehmen FullServe

Employee Database

FullTimeEmps(ssn, empID, firstName, middleName, lastName)

Hire(empID, hireDate, recruiter)

TempEmployees(ssn, hireStart, hireEnd, name, hourlyRate)

Resume Database

Interviews(interviewDate, pID, recruiter, hireDecision, hireDate)

CVs(ID, resume)

Quelle: Doan, Halevy and Ives. Principles of data Integration, 2012 [DHI12]

Beispiel: Unternehmen FullServe

Employee Database

FullTimeEmps(ssn, empID, firstName, middleName, lastName)
Hire(empID, hireDate, recruiter)
TempEmployees(ssn, hireStart, hireEnd, name, hourlyRate)

Training Database

Courses(courseID, name, instructor)
Enrollments(courseID, empID, date)

Resume Database

Interviews(interviewDate, pID, recruiter, hireDecision, hireDate)
CVs(ID, resume)

Quelle: Doan, Halevy and Ives. Principles of data Integration, 2012 [DHI12]

Beispiel: Unternehmen FullServe

Employee Database

FullTimeEmps(ssn, empID, firstName, middleName, lastName)
Hire(empID, hireDate, recruiter)
TempEmployees(ssn, hireStart, hireEnd, name, hourlyRate)

Resume Database

Interviews(interviewDate, pID, recruiter, hireDecision, hireDate)
CVs(ID, resume)

Training Database

Courses(courseID, name, instructor)
Enrollments(courseID, empID, date)

Services Database

Services(packName, textDescription)
Customers(name, ID, zipCode, streedAdr, phone)
Contracts(custID, packName, startDate)

Quelle: Doan, Halevy and Ives. Principles of data Integration, 2012 [DHI12]

Beispiel: Unternehmen FullServe

Employee Database
FullTimeEmps(ssn, empID, firstName, middleName, lastName)
Hire(empID, hireDate, recruiter)
TempEmployees(ssn, hireStart, hireEnd, name, hourlyRate)

Resume Database
Interviews(interviewDate, pID, recruiter, hireDecision, hireDate)
CVs(ID, resume)

Training Database
Courses(courseID, name, instructor)
Enrollments(courseID, empID, date)

Services Database
Services(packName, textDescription)
Customers(name, ID, zipCode, streedAdr, phone)
Contracts(custID, packName, startDate)

Sales Database
Products(prodName, prodID)
Sales(prodID, custID, custName, address)

Quelle: Doan, Halevy and Ives. Principles of data Integration, 2012 [DHI12]

Beispiel: Unternehmen FullServe

Employee Database
FullTimeEmps(ssn, empID, firstName, middleName, lastName)
Hire(empID, hireDate, recruiter)
TempEmployees(ssn, hireStart, hireEnd, name, hourlyRate)

Resume Database
Interviews(interviewDate, pID, recruiter, hireDecision, hireDate)
CVs(ID, resume)

Training Database
Courses(courseID, name, instructor)
Enrollments(courseID, empID, date)

Services Database
Services(packName, textDescription)
Customers(name, ID, zipCode, streedAdr, phone)
Contracts(custID, packName, startDate)

Sales Database
Products(prodName, prodID)
Sales(prodID, custID, custName, address)

HelpLine Database
Calls(date, agent, custID, text, action)

Quelle: Doan, Halevy and Ives. Principles of data Integration, 2012 [DHI12]

Beispiel: Unternehmen FullServe

Beispiel: Unternehmen FullServe

- FullServe will nach Europa expandieren

Beispiel: Unternehmen FullServe

- FullServe will nach Europa expandieren
- Zukauf des Unternehmens EuroCard
  - Kreditkartenanbieter
  - Ermöglicht Kunden Zugang zum Internet

Beispiel: Unternehmen FullServe

- FullServe will nach Europa expandieren
- Zukauf des Unternehmens EuroCard
  - Kreditkartenanbieter
  - Ermöglicht Kunden Zugang zum Internet
- EuroCard hat eigene Datenbanken

Quelle: Doan, Halevy and Ives. Principles of data Integration, 2012 [DHI12]

Beispiel: Unternehmen FullServe

- FullServe will nach Europa expandieren
- Zukauf des Unternehmens EuroCard
  - Kreditkartenanbieter
  - Ermöglicht Kunden Zugang zum Internet
- EuroCard hat eigene Datenbanken

Employee Database

Emp(ID, firstnameMiddleInitial,
lastName, salary)
Hire(ID, hireDate, recruiter)

Quelle: Doan, Halevy and Ives. Principles of data Integration, 2012 [DHI12]

Beispiel: Unternehmen FullServe

- FullServe will nach Europa expandieren
- Zukauf des Unternehmens EuroCard
  - Kreditkartenanbieter
  - Ermöglicht Kunden Zugang zum Internet
- EuroCard hat eigene Datenbanken

Employee Database

Emp(ID, firstnameMiddleInitial,
lastName, salary)
Hire(ID, hireDate, recruiter)

Resume Database

Interviews(ID, date, location, recruiter)
CVs(candID, resume)

Quelle: Doan, Halevy and Ives. Principles of data Integration, 2012 [DHI12]

Beispiel: Unternehmen FullServe

- FullServe will nach Europa expandieren
- Zukauf des Unternehmens EuroCard
  - Kreditkartenanbieter
  - Ermöglicht Kunden Zugang zum Internet
- EuroCard hat eigene Datenbanken

Employee Database
Emp(ID, firstnameMiddleInitial, lastName, salary)
Hire(ID, hireDate, recruiter)

Resume Database
Interviews(ID, date, location, recruiter)
CVs(candID, resume)

Credit Card Database
Cards(CustID, cardNum, expiration, currentBalance)
Customers(CustID, name, address)

Quelle: Doan, Halevy and Ives. Principles of data Integration, 2012 [DHI12]

Beispiel: Unternehmen FullServe

- FullServe will nach Europa expandieren
- Zukauf des Unternehmens EuroCard
  - Kreditkartenanbieter
  - Ermöglicht Kunden Zugang zum Internet
- EuroCard hat eigene Datenbanken

Employee Database
Emp(ID, firstnameMiddleInitial, lastName, salary)
Hire(ID, hireDate, recruiter)

Resume Database
Interviews(ID, date, location, recruiter)
CVs(candID, resume)

Credit Card Database
Cards(CustID, cardNum, expiration, currentBalance)
Customers(CustID, name, address)

HelpLine Database
Calls(date, agent, custID, description, followup)

Quelle: Doan, Halevy and Ives. Principles of data Integration, 2012 [DHI12]

Beispiel: Unternehmen FullServe

Beispiel-Szenarien in denen Daten von mehreren Datenbanken benötigt werden:

Beispiel: Unternehmen FullServe

Beispiel-Szenarien in denen Daten von mehreren Datenbanken benötigt werden:

- *Human Resource Department* möchte alle Angestellte wissen

Beispiel: Unternehmen FullServe

Beispiel-Szenarien in denen Daten von mehreren Datenbanken benötigt werden:

- *Human Resource Department* möchte alle Angestellte wissen
- *Help-Line Center* braucht Kundendaten zur Problemlokalisierung und -behebung

Beispiel: Unternehmen FullServe

Beispiel-Szenarien in denen Daten von mehreren Datenbanken benötigt werden:

- *Human Resource Department* möchte alle Angestellte wissen
- *Help-Line Center* braucht Kundendaten zur Problemlokalisierung und -behebung
- Aufsetzen einer Webseite mit allen angebotenen Produkten und Diensten inkl. Kundenbereich

Beispiel: Unternehmen FullServe

Beispiel-Szenarien in denen Daten von mehreren Datenbanken benötigt werden:

- *Human Resource Department* möchte alle Angestellte wissen
- *Help-Line Center* braucht Kundendaten zur Problemlokalisierung und -behebung
- Aufsetzen einer Webseite mit allen angebotenen Produkten und Diensten inkl. Kundenbereich
- Herausfinden von Angestellten die früher bei Konkurrenzunternehmen gearbeitet haben

Beispiel: Unternehmen FullServe

Beispiel-Szenarien in denen Daten von mehreren Datenbanken benötigt werden:

- *Human Resource Department* möchte alle Angestellte wissen
- *Help-Line Center* braucht Kundendaten zur Problemlokalisierung und -behebung
- Aufsetzen einer Webseite mit allen angebotenen Produkten und Diensten inkl. Kundenbereich
- Herausfinden von Angestellten die früher bei Konkurrenzunternehmen gearbeitet haben
- Verknüpfung von Help-Line Anrufen mit anderen Daten

Beispiel: Unternehmen FullServe

Beispiel-Szenarien in denen Daten von mehreren Datenbanken benötigt werden:

- *Human Resource Department* möchte alle Angestellte wissen
- *Help-Line Center* braucht Kundendaten zur Problemlokalisierung und -behebung
- Aufsetzen einer Webseite mit allen angebotenen Produkten und Diensten inkl. Kundenbereich
- Herausfinden von Angestellten die früher bei Konkurrenzunternehmen gearbeitet haben
- Verknüpfung von Help-Line Anrufen mit anderen Daten
  - Beseitigung von Defiziten
    - Bsp. Erhöhte Fehlerhäufigkeit von Diensten/Produkten die von Angstellten installiert wurden, die einen bestimmten Kurs besucht haben

Beispiel: Unternehmen FullServe

Beispiel-Szenarien in denen Daten von mehreren Datenbanken benötigt werden:

- *Human Resource Department* möchte alle Angestellte wissen
- *Help-Line Center* braucht Kundendaten zur Problemlokalisierung und -behebung
- Aufsetzen einer Webseite mit allen angebotenen Produkten und Diensten inkl. Kundenbereich
- Herausfinden von Angestellten die früher bei Konkurrenzunternehmen gearbeitet haben
- Verknüpfung von Help-Line Anrufen mit anderen Daten
  - Beseitigung von Defiziten
    - Bsp. Erhöhte Fehlerhäufigkeit von Diensten/Produkten die von Angstellten installiert wurden, die einen bestimmten Kurs besucht haben
  - Erschliessen neuer Geschäftsideen

Agenda

Architekturparadigmen

- Materialisiert
  - A priori-Integration
  - Zentrale Datenbasis
  - Zentrale Anfragebearbeitung
  - Typisches Beispiel: Data Warehouse

Architekturparadigmen

- Materialisiert
  - A priori-Integration
  - Zentrale Datenbasis
  - Zentrale Anfragebearbeitung
  - Typisches Beispiel: Data Warehouse
- Virtuell
  - *On demand*-Integration
  - Dezentrale Daten
  - Dezentrale Anfragebearbeitung
  - Typisches Beispiel: Mediator-basiertes Informationssystem

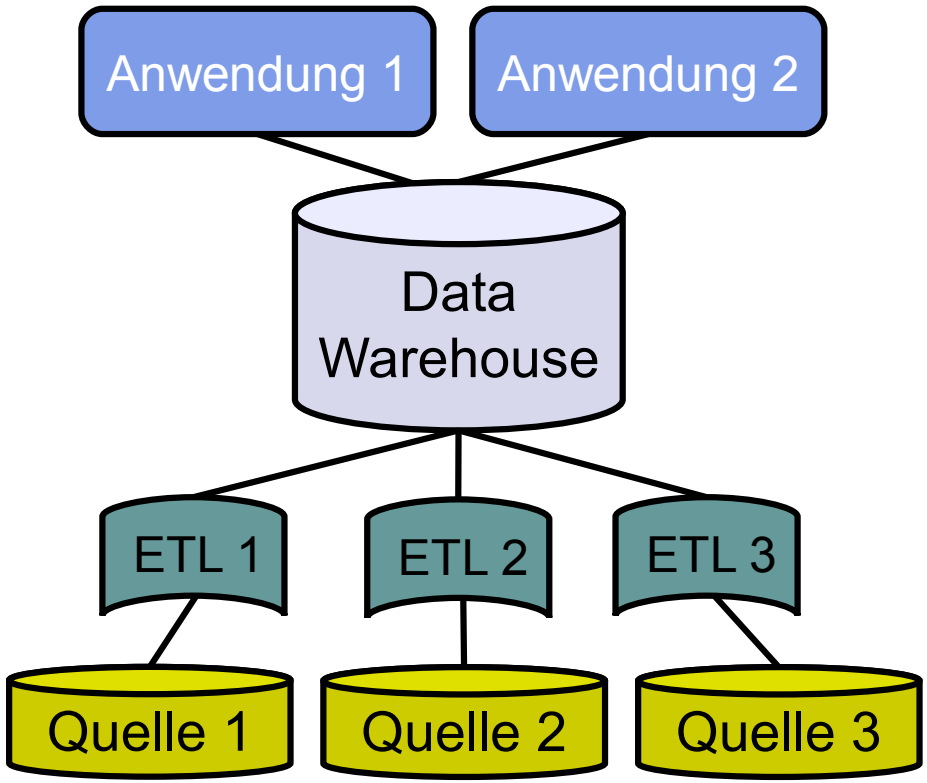
Architekturparadigmen

- Materialisiert
  - A priori-Integration
  - Zentrale Datenbasis
  - Zentrale Anfragebearbeitung
  - Typisches Beispiel: Data Warehouse
- Virtuell
  - *On demand*-Integration
  - Dezentrale Daten
  - Dezentrale Anfragebearbeitung
  - Typisches Beispiel: Mediator-basiertes Informationssystem
- Existierende Architekturen befinden sich oft zwischen diesen Extremen

$\Rightarrow$ einige Daten werden materialisiert vorgehalten (z.B. durch den Einsatz von Caching)

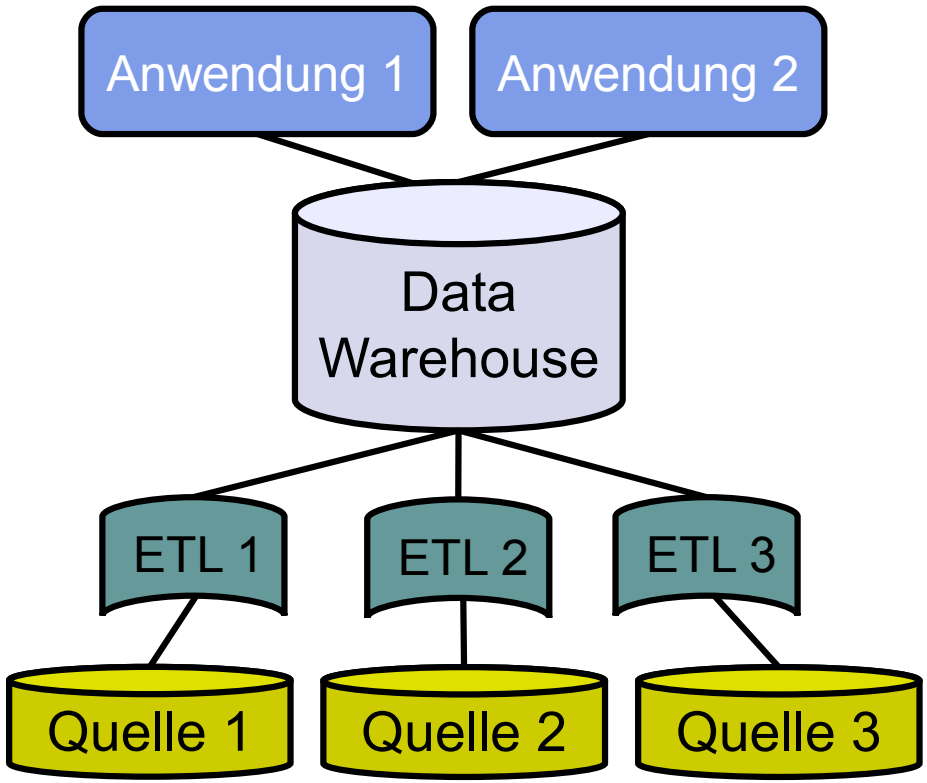
Materialisierte Integration – Datenfluss

- *Push*
- Erstmaliges Laden (*population*) des DW (inkl. *Data Cleaning*)
- Periodischer Datenimport: *Updating materialized views*
- Redundante Datenhaltung
- Aggregation und Löschung alter Daten



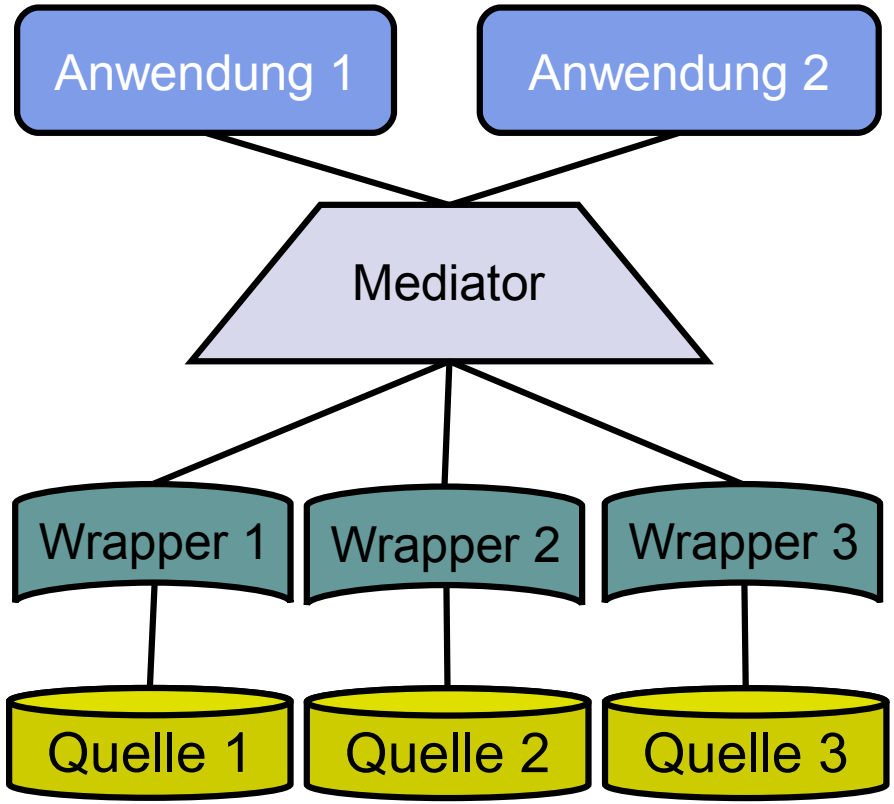
Materialisierte Integration – Anfragebearbeitung/Schema

- Anfragebearbeitung:
  - Wie normale DBMS
  - Oft Aggregationsanfragen
  - *Decision Support*
- Schema:
  - Meist *Bottom-Up*-Entwurf
  - Schemaintegration
  - *Star*-Schema
    - *Fact Table*
    - *Dimension Tables*



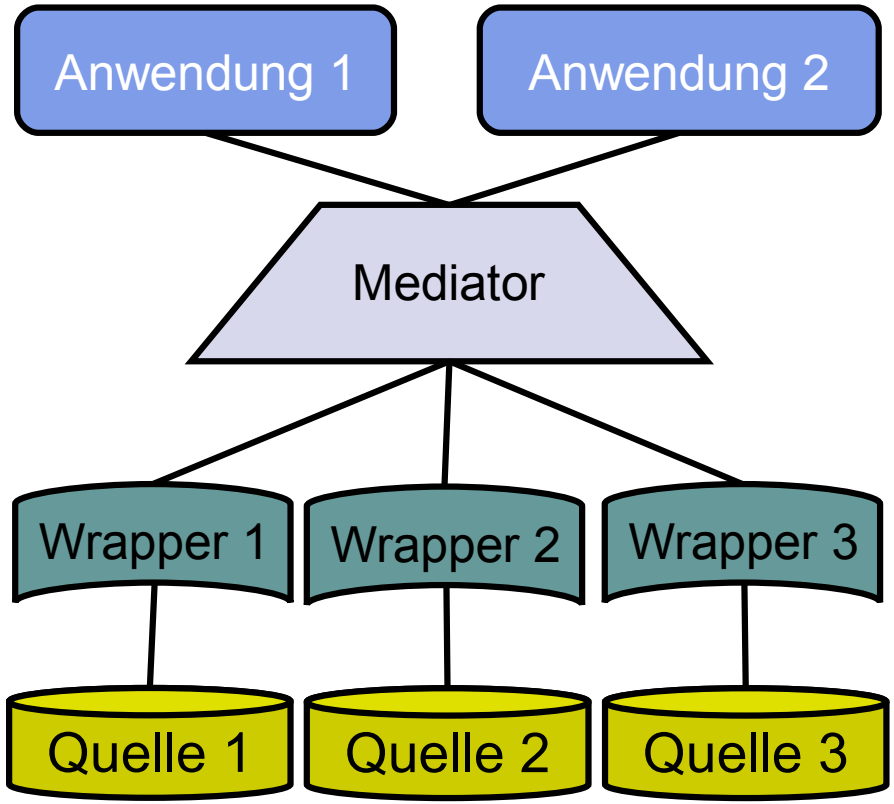
Virtuelle Integration – Datenfluss

- *Pull*
- Daten sind in Quellen gespeichert
- Nur Anfragen und Ergebnisse werden übertragen (*Query Shipping*)
- *Data Cleaning* nur online möglich



Virtuelle Integration – Anfragebearbeitung/Schema

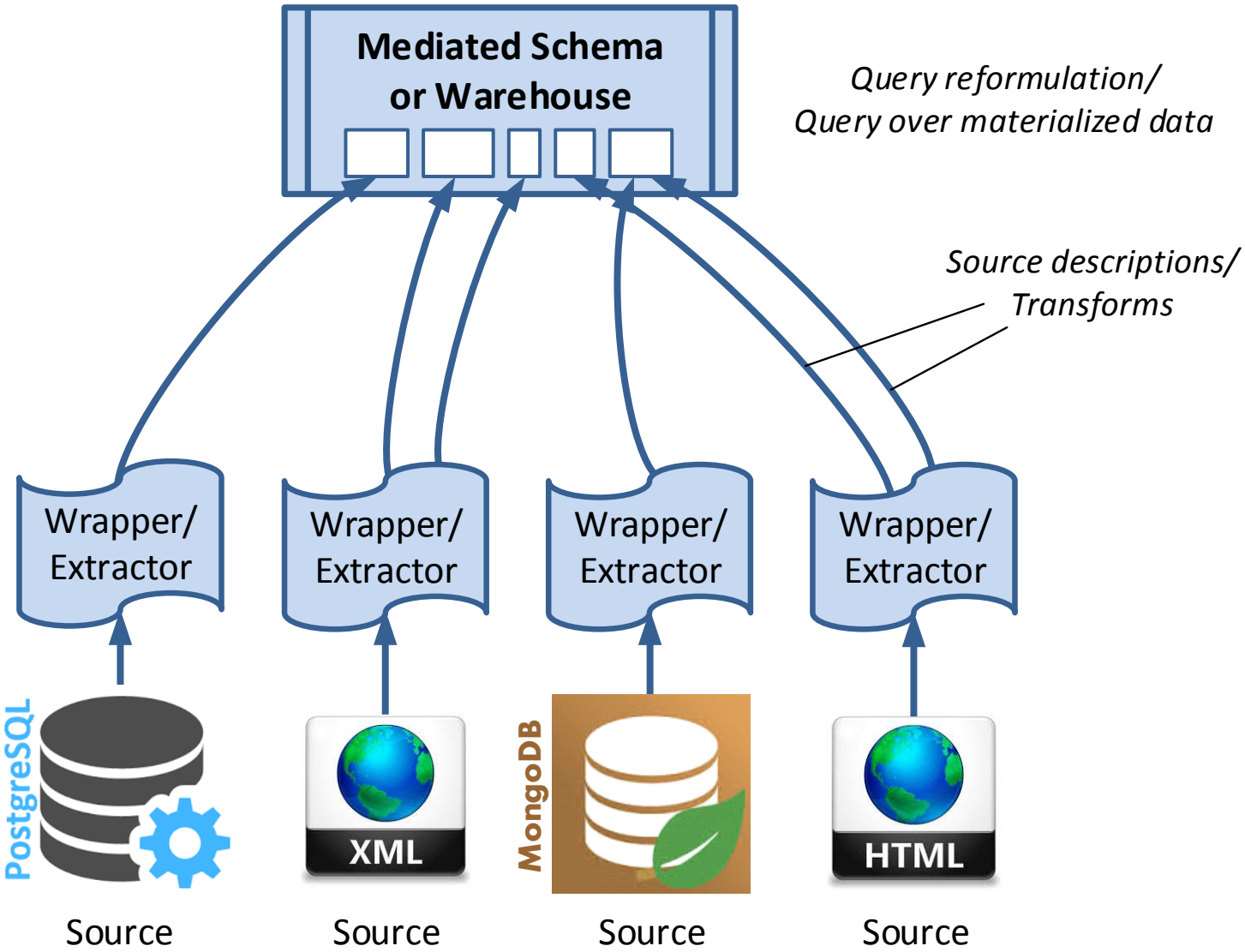
- Anfragebearbeitung:
  - Optimierung schwierig (Geschwindigkeiten und Fähigkeiten der Quellen)
  - Viele mögliche Pläne zur Berechnung des Anfrageergebnisses
- Schema:
  - Meist *Top-down* Entwurf
  - Leicht erweiterbar
    - neue Quellen
    - neue/geänderte *Mappings*
  - *Schema Mapping* statt Schemaintegration



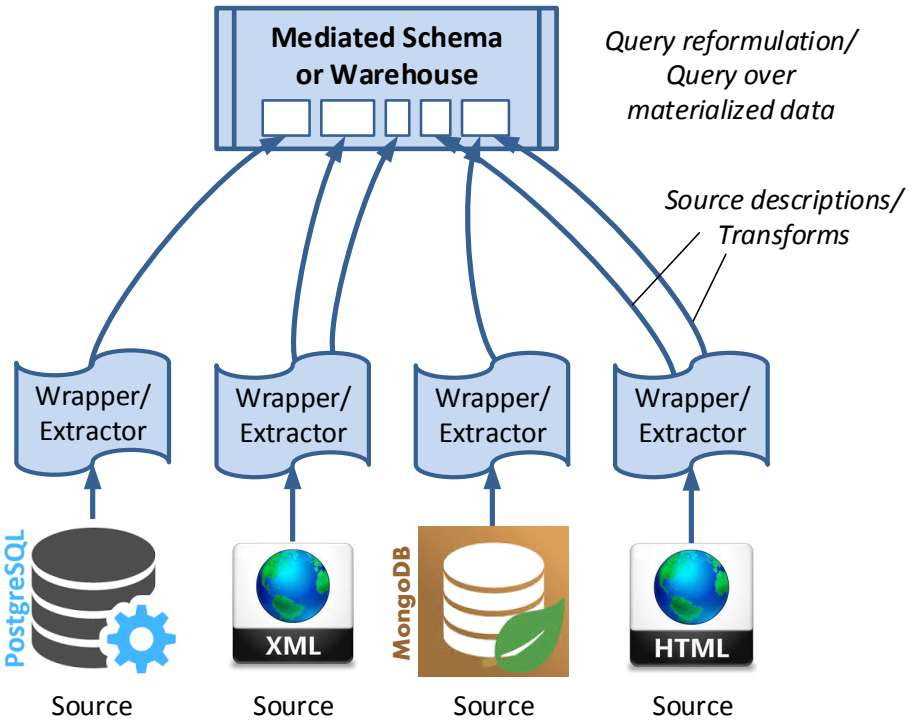
Materialisiert vs. Virtuelle Integration – Vergleich

	Materialisiert	**Virtuell**
Aktualität	– (Cache)	+
Antwortzeit	+	–
Flexibilität	– (GaV)	+ (LaV)
Komplexität	+	++
Autonomie	–	+
Anfragemächtigkeit	+	–
Read/Write	+/+	+/–
Ressourcenbedarf	? (workload)	? (workload)
Vollständigkeit	+	? (OWA, CWA)
Datenreinigung	+	–
Informationsqualität	+	–

Komponenten Virtueller Architekturen

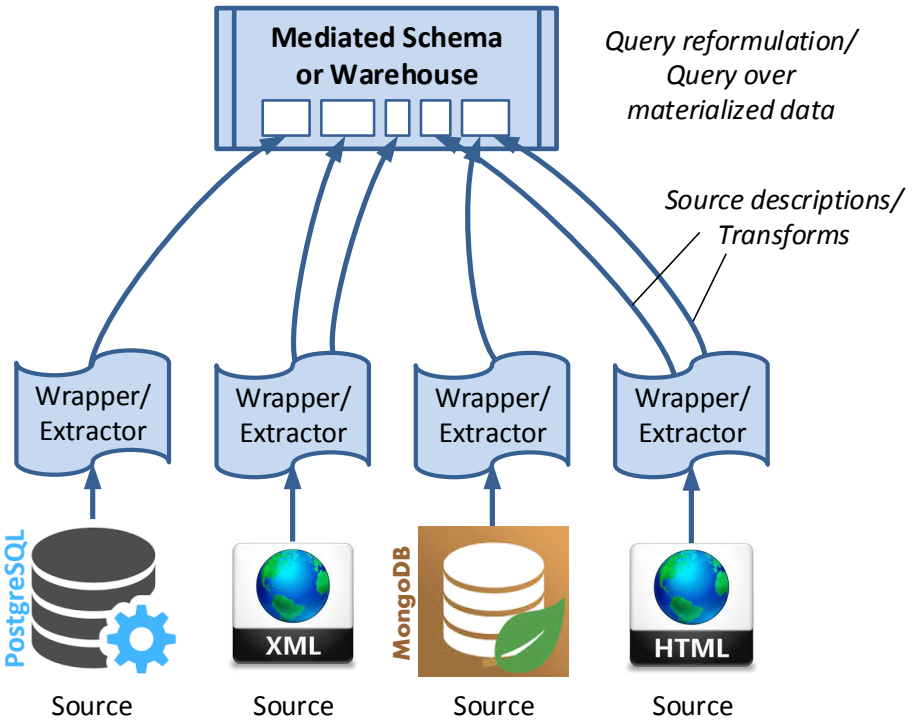


Komponenten Virtueller Architekturen



Quellen:

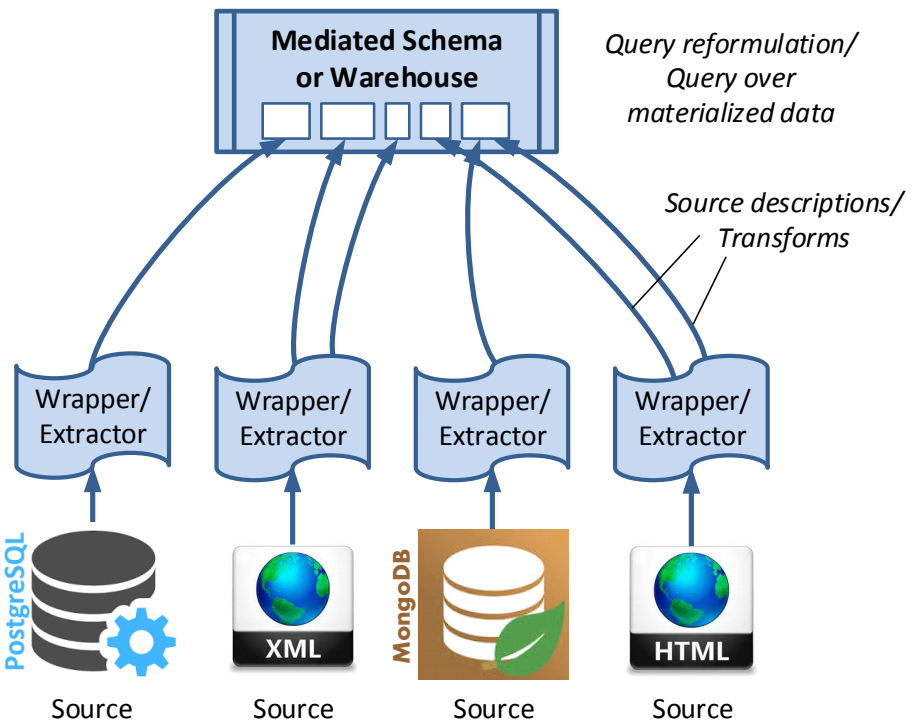
Komponenten Virtueller Architekturen



Quellen:

- verschiedene Datenmodelle

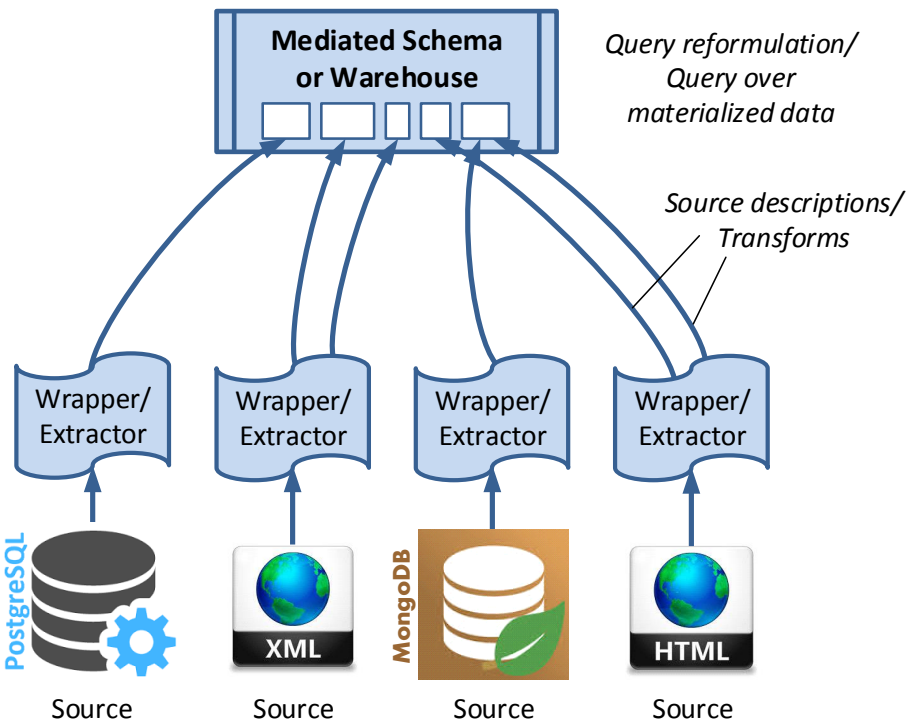
Komponenten Virtueller Architekturen



Quellen:

- verschiedene Datenmodelle
- verschiedene Anfragemächtigkeiten

Komponenten Virtueller Architekturen

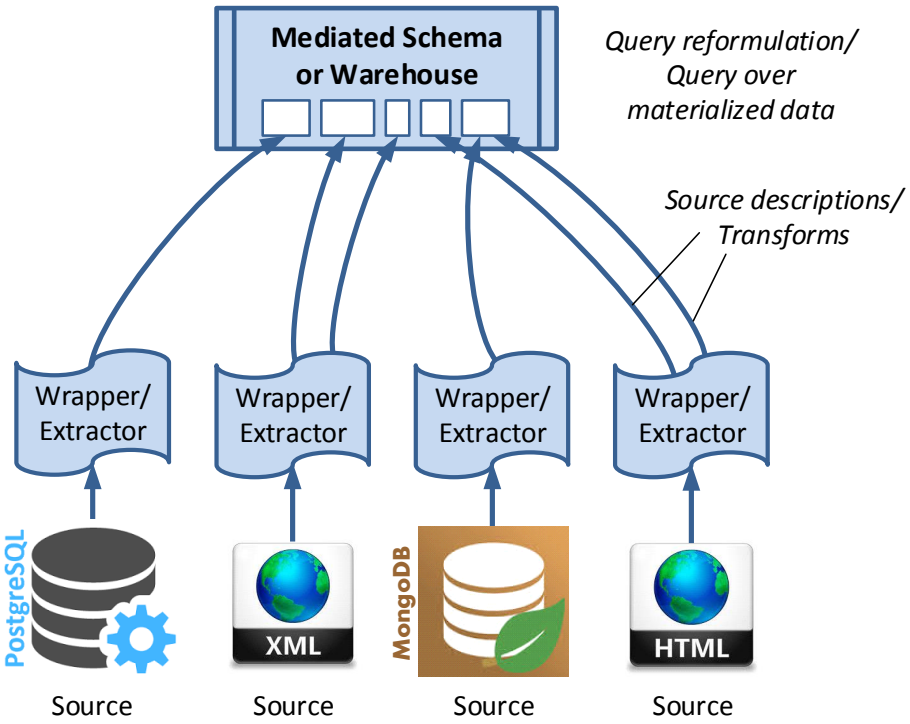


Quellen:

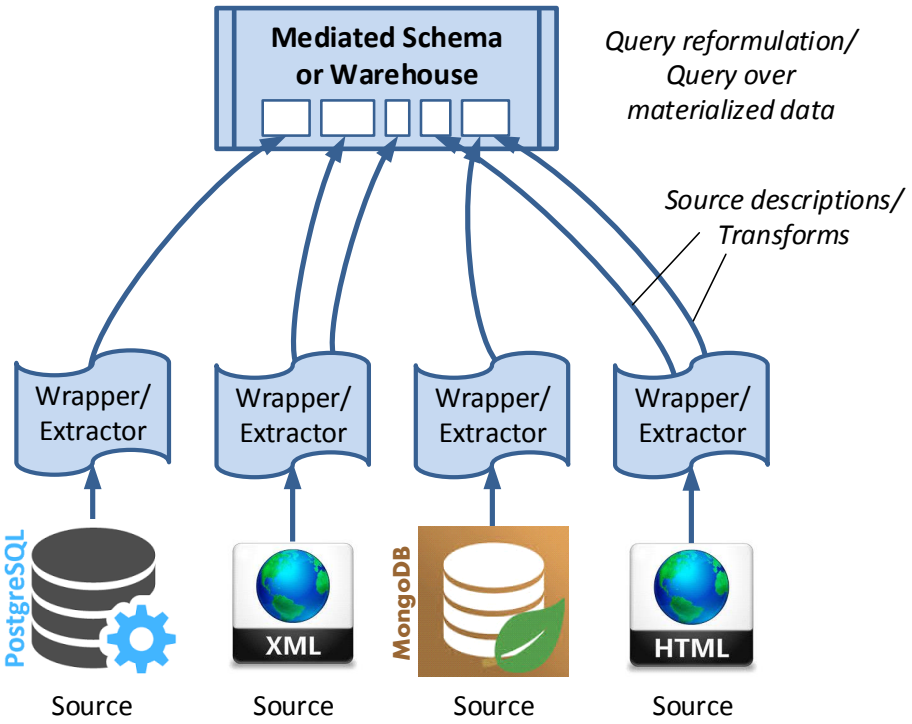
- verschiedene Datenmodelle
- verschiedene Anfragemächtigkeiten
- Quelle kann eine Anwendung sein, die wiederum komplexe Bearbeitungsschritte vollzieht

Komponenten Virtueller Architekturen

Wrapper:



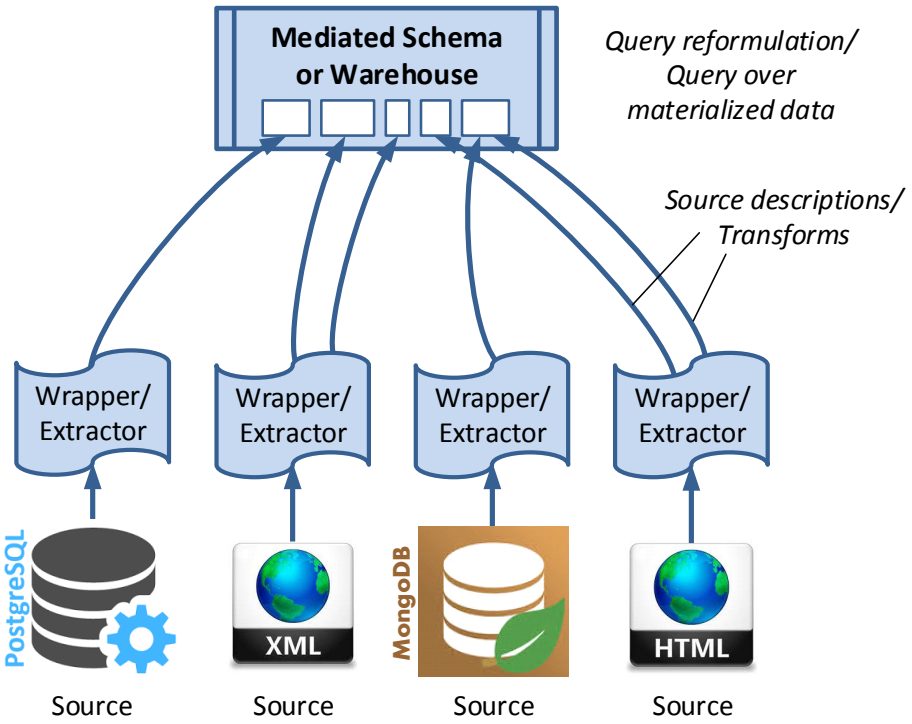
Komponenten Virtueller Architekturen



Wrapper:

- bekommt Anfrage in Sprache des Integrationssystems (z.B. relational oder XML)

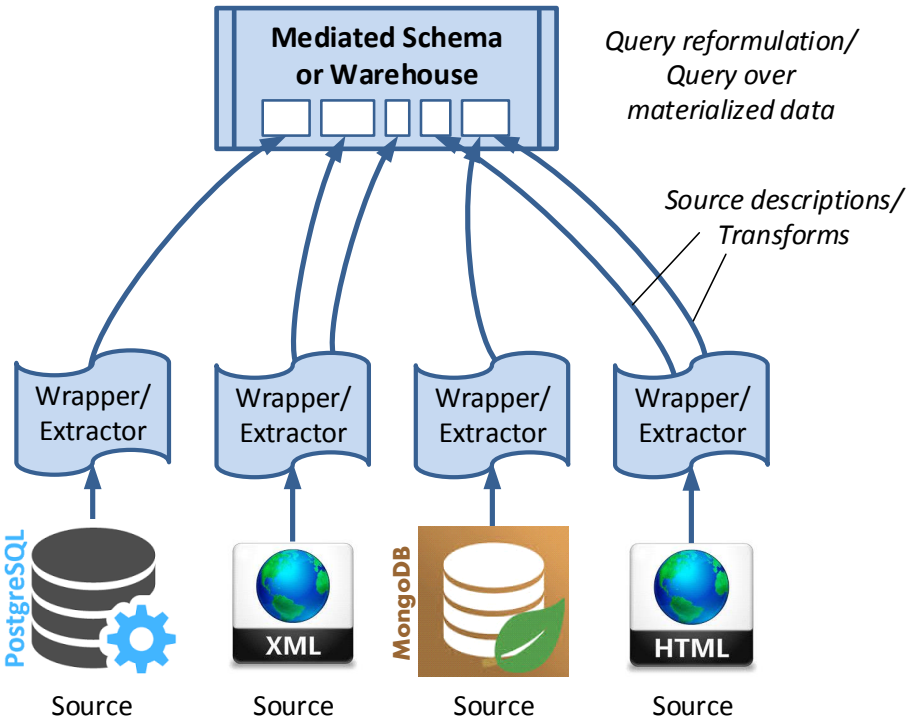
Komponenten Virtueller Architekturen



Wrapper:

- bekommt Anfrage in Sprache des Integrationssystems (z.B. relational oder XML)
- übersetzt Anfrage in Sprache der Quelle (z.B. HTTP Request)

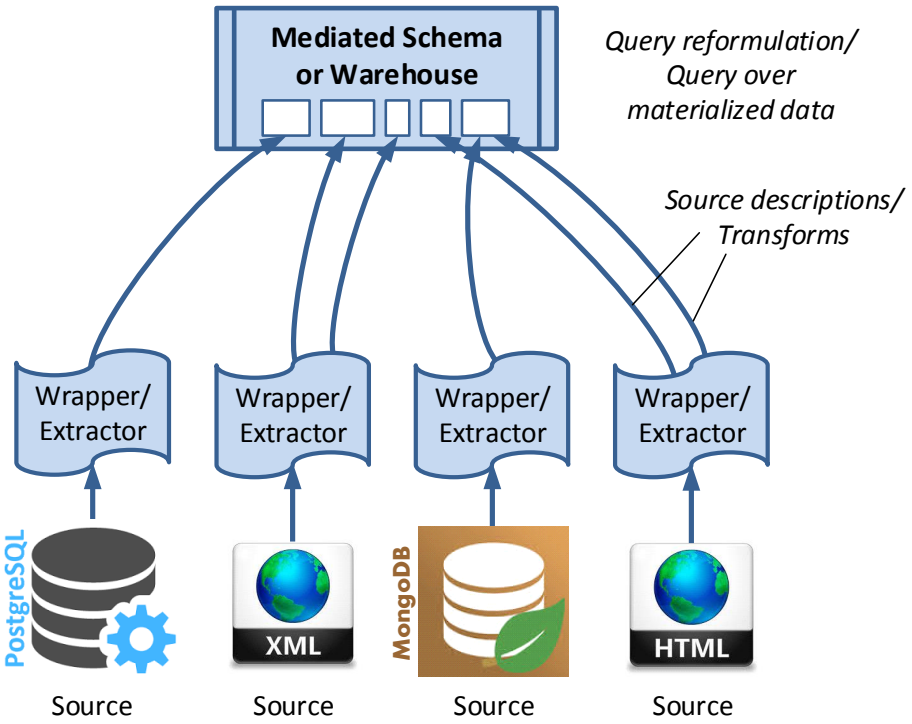
Komponenten Virtueller Architekturen



Wrapper:

- bekommt Anfrage in Sprache des Integrationssystems (z.B. relational oder XML)
- übersetzt Anfrage in Sprache der Quelle (z.B. HTTP Request)
- sendet Anfrage an Quelle

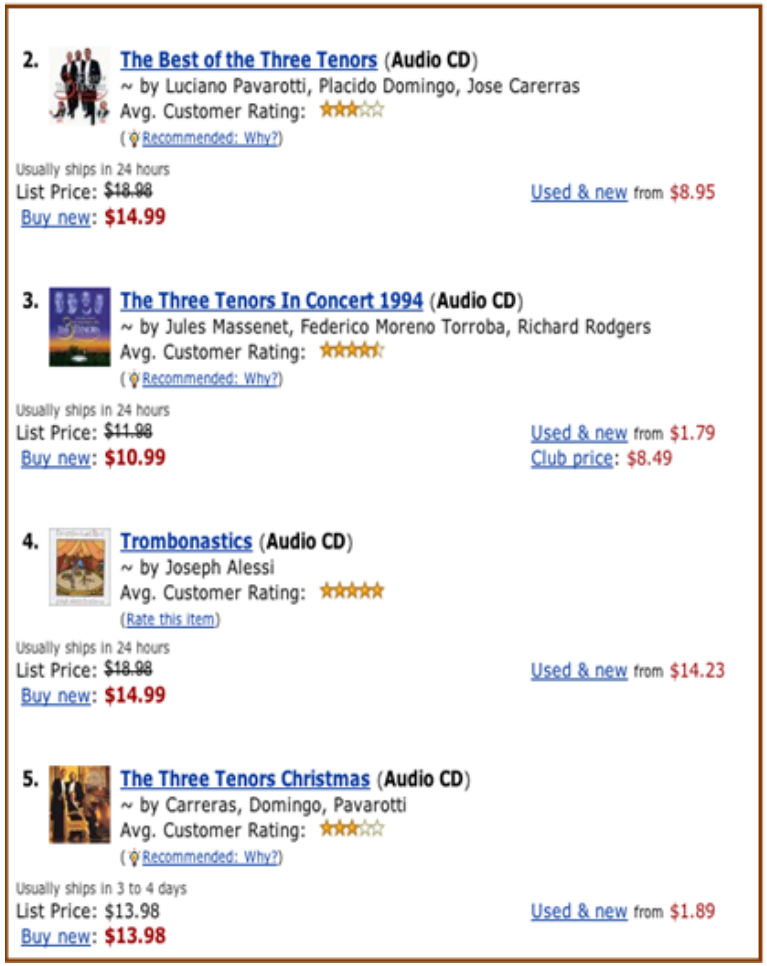
Komponenten Virtueller Architekturen



Wrapper:

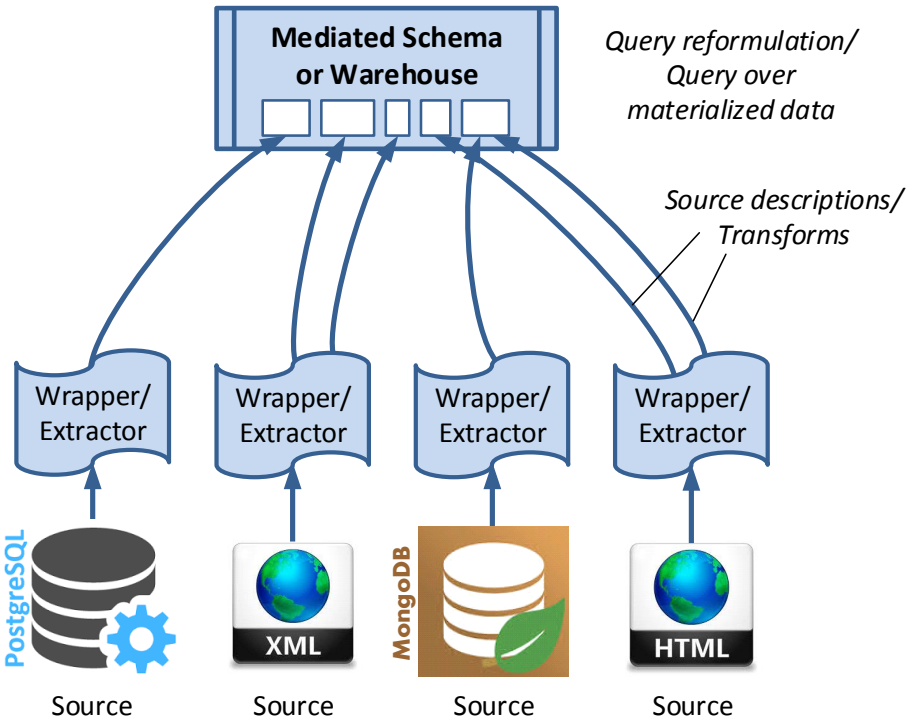
- bekommt Anfrage in Sprache des Integrationssystems (z.B. relational oder XML)
- übersetzt Anfrage in Sprache der Quelle (z.B. HTTP Request)
- sendet Anfrage an Quelle
- transformiert Ergebnis (z.B. HTML Datei) in Datenmodell des Integrationssystems (z.B. Tupelmenge oder XML Datei)

Wrappers



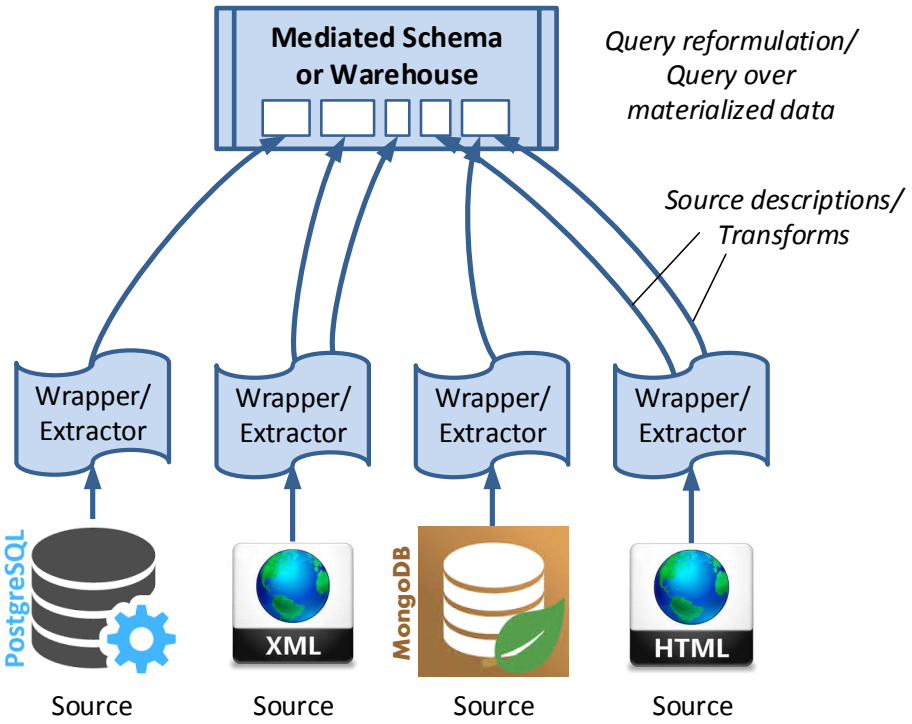

```
<cd>   <title> The best of … </title>
          <artist> Abiteboul </artist>
          <artist> Pavarotti  </artist>
          <artist> Domingo  </artist>
          <price> 19.95      </price>
    </cd>
      …
```

Komponenten Virtueller Architekturen



Mediated/Globales Schema:

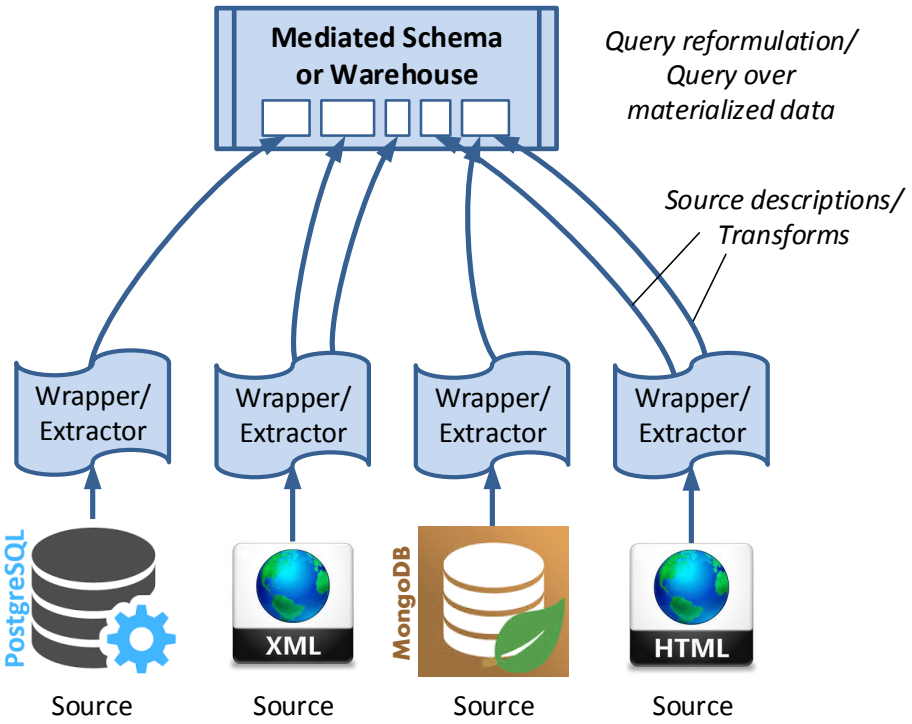
Komponenten Virtueller Architekturen



Mediated/Globales Schema:

- dient zur Interaktion mit dem Benutzer

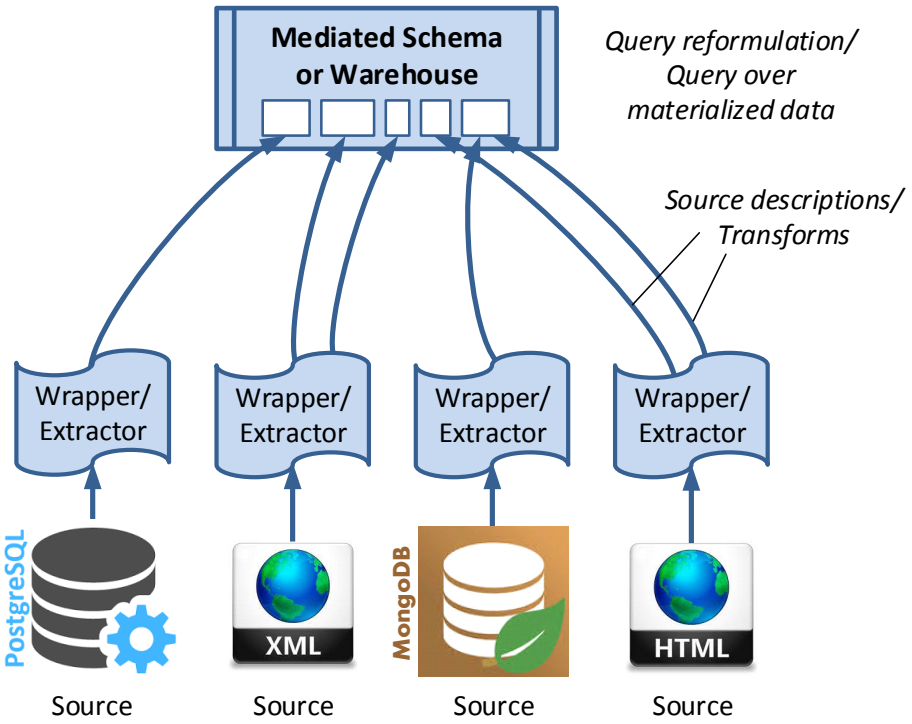
Komponenten Virtueller Architekturen



Mediated/Globales Schema:

- dient zur Interaktion mit dem Benutzer
- konzipiert für die Integrationsanwendung (beinhaltet daher nur einen Teil der Aspekte aus den Quellen)

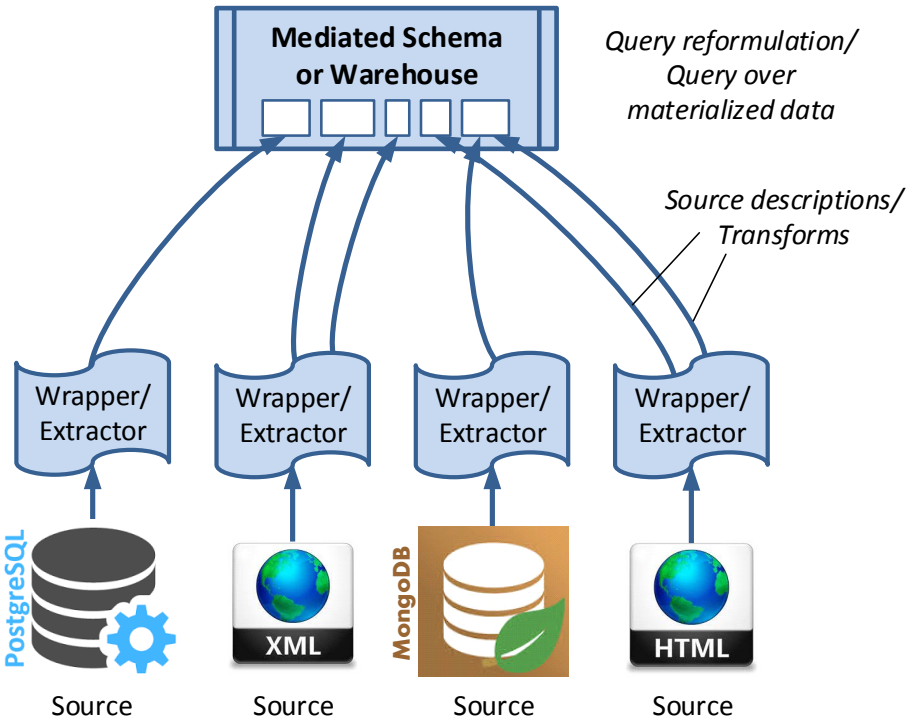
Komponenten Virtueller Architekturen



Mediated/Globales Schema:

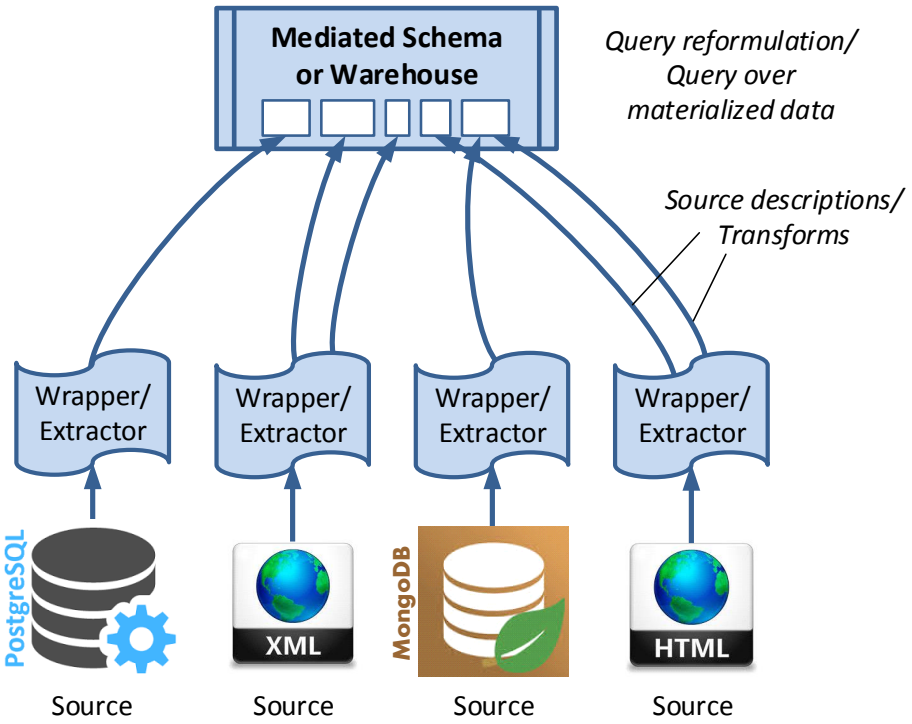
- dient zur Interaktion mit dem Benutzer
- konzipiert für die Integrationsanwendung (beinhaltet daher nur einen Teil der Aspekte aus den Quellen)
- ist lediglich logisch und dient zur Formulierung von Anfragen

Komponenten Virtueller Architekturen



Quellbeschreibungen:

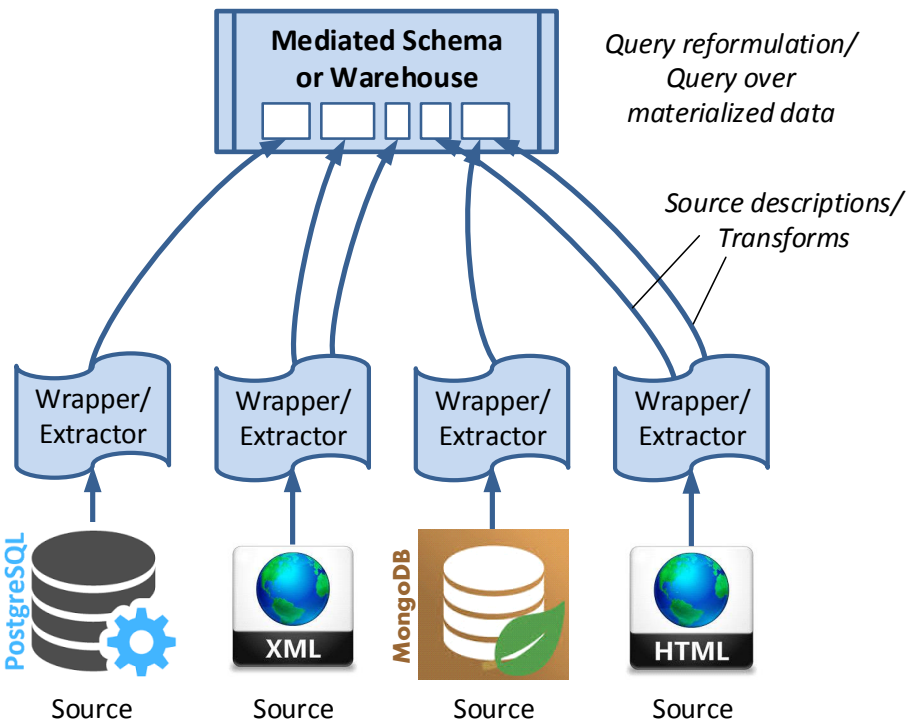
Komponenten Virtueller Architekturen



Quellbeschreibungen:

- eine Beschreibung pro Quelle

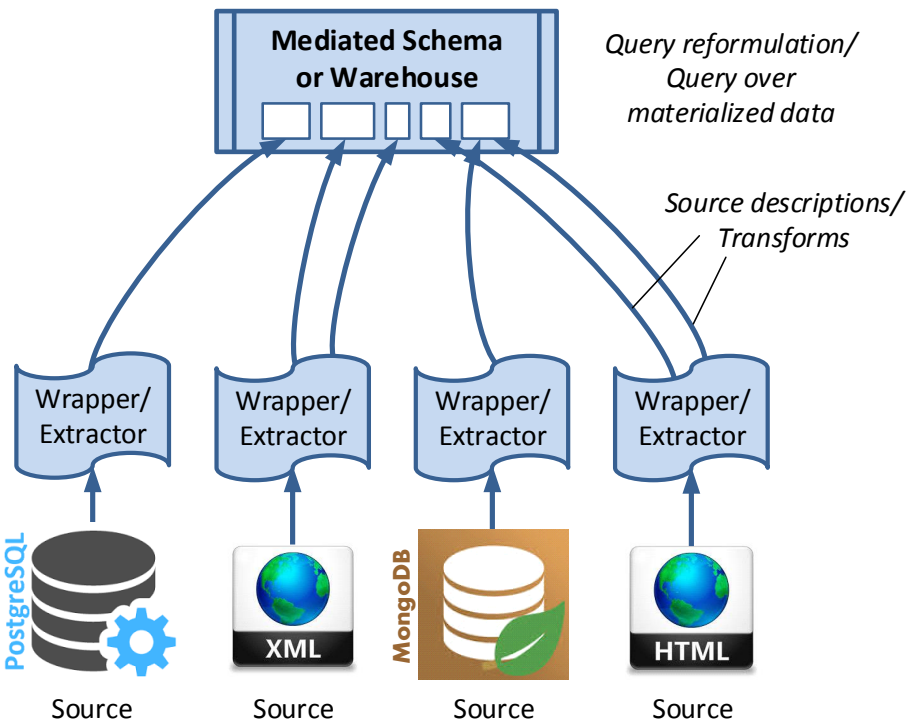
Komponenten Virtueller Architekturen



Quellbeschreibungen:

- eine Beschreibung pro Quelle
- enthält alle Informationen die das System braucht um die Quelle zu nutzen

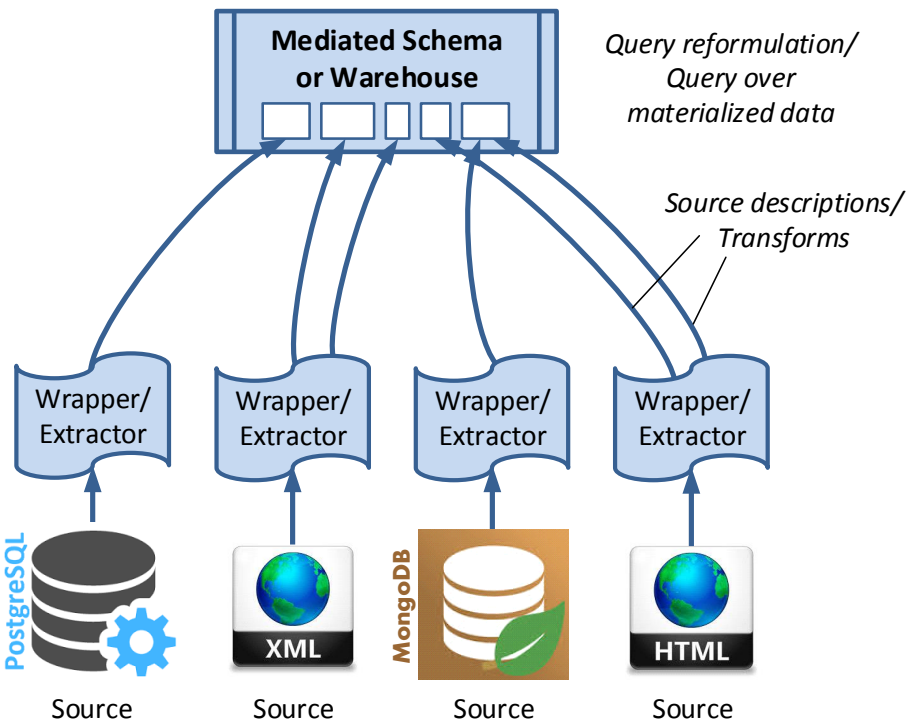
Komponenten Virtueller Architekturen



Quellbeschreibungen:

- eine Beschreibung pro Quelle
- enthält alle Informationen die das System braucht um die Quelle zu nutzen
- mappt deklarativ Konzepte zw. globalem Schema und Quellschema

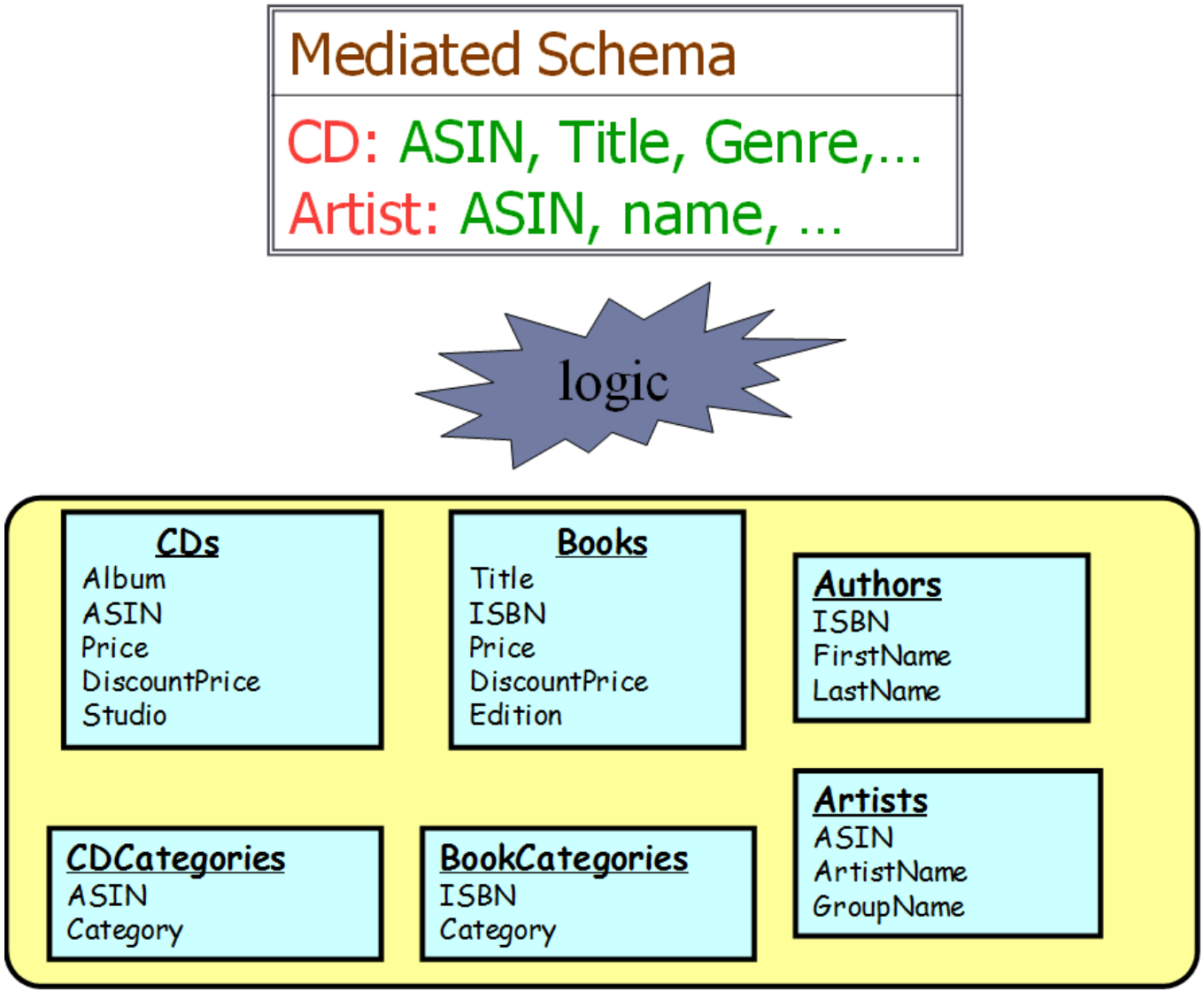
Komponenten Virtueller Architekturen



Quellbeschreibungen:

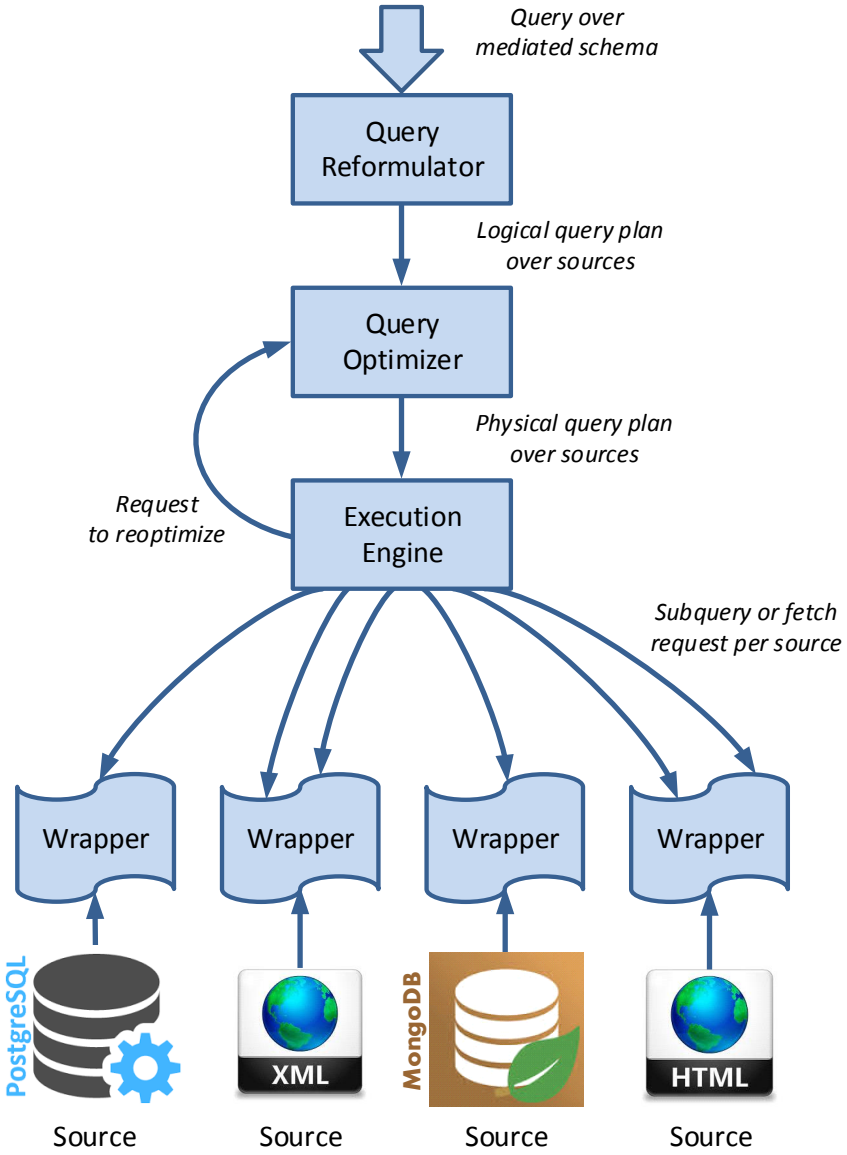
- eine Beschreibung pro Quelle
- enthält alle Informationen die das System braucht um die Quelle zu nutzen
- mappt deklarativ Konzepte zw. globalem Schema und Quellschema
- beschreibt Transformation auf Datenwertebene (z.B. für Konventionen, Einheiten)

Quellbeschreibungen

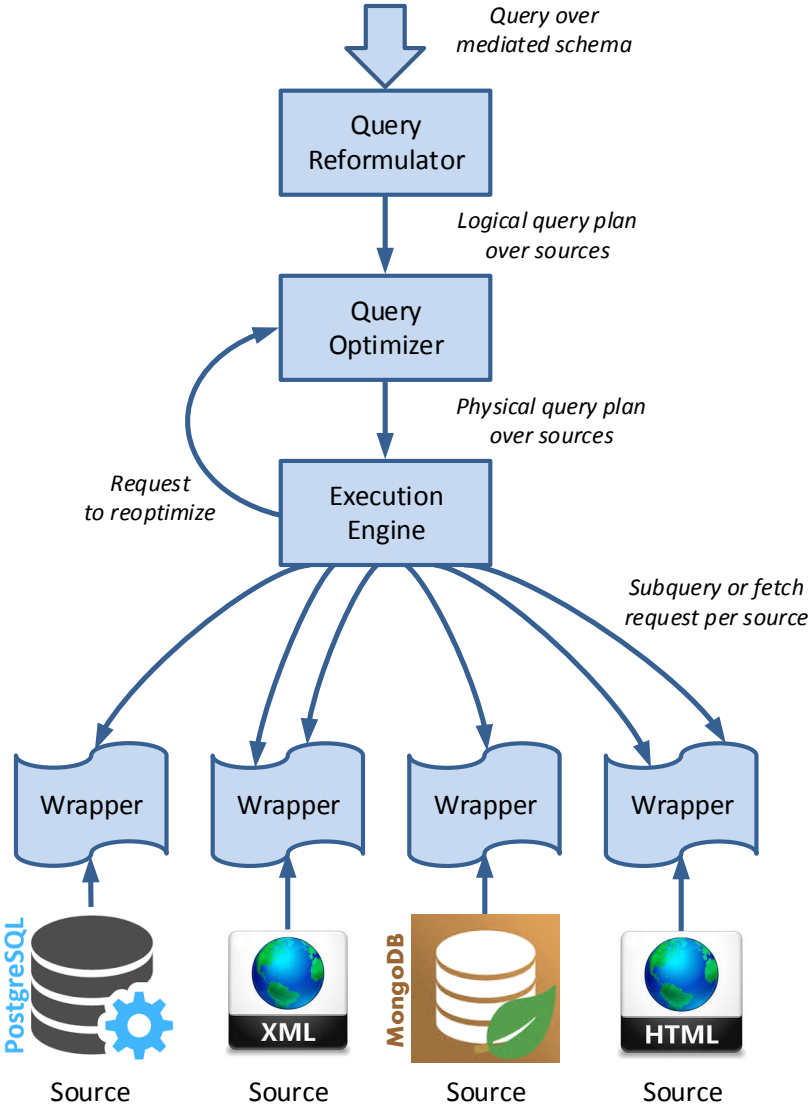


Agenda

Anfragebearbeitung (Virtuell)

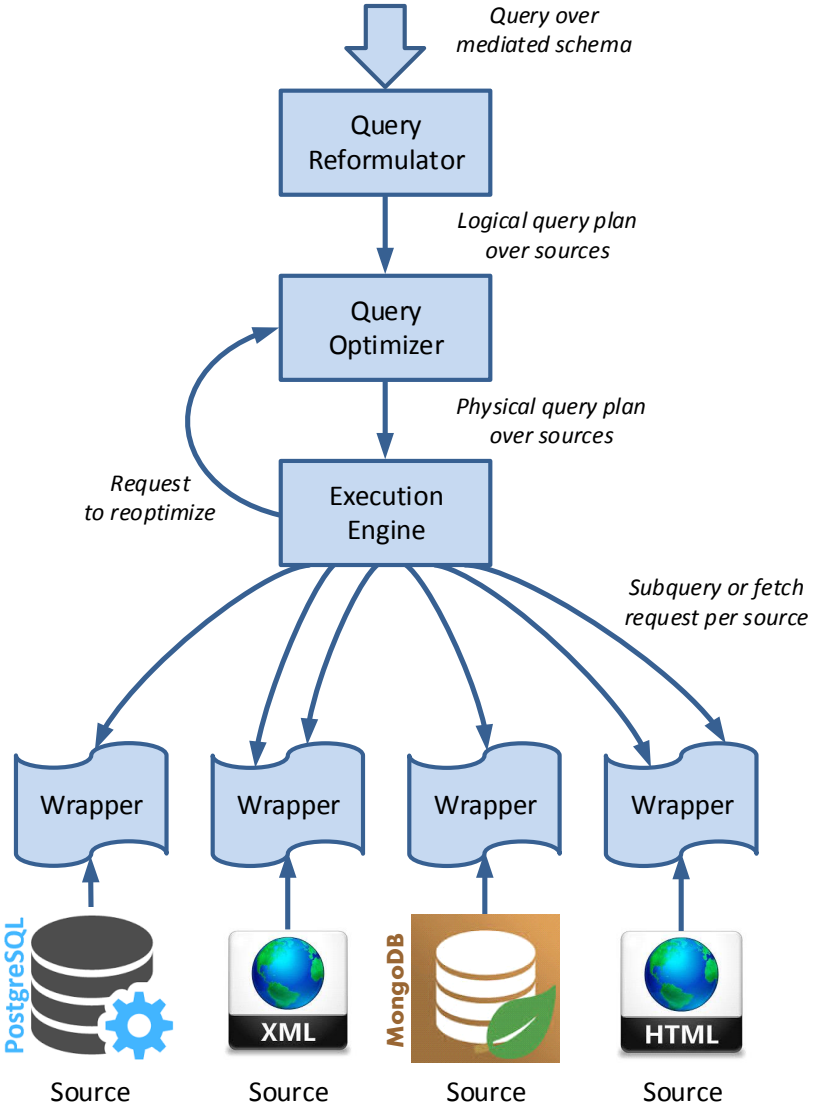


Anfragebearbeitung (Virtuell)



Anfrageumschreibung:

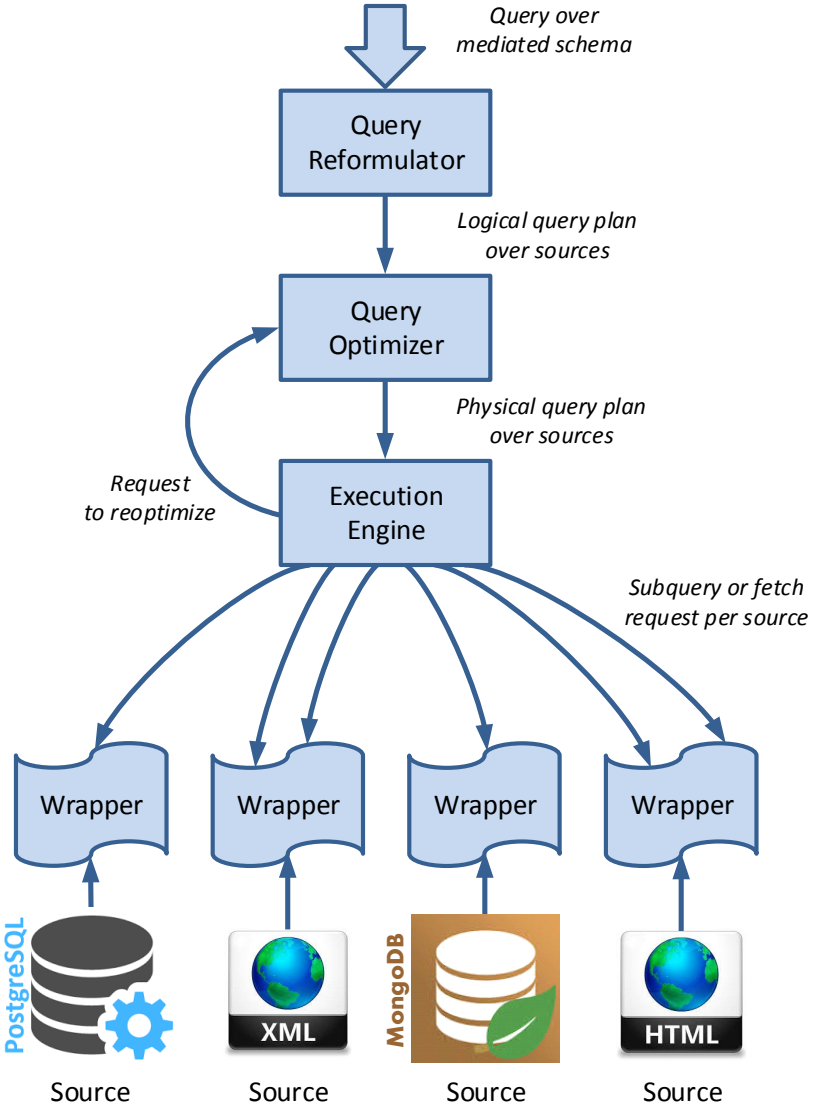
Anfragebearbeitung (Virtuell)



Anfrageumschreibung:

- Gegeben: Anfrage auf globales Schema

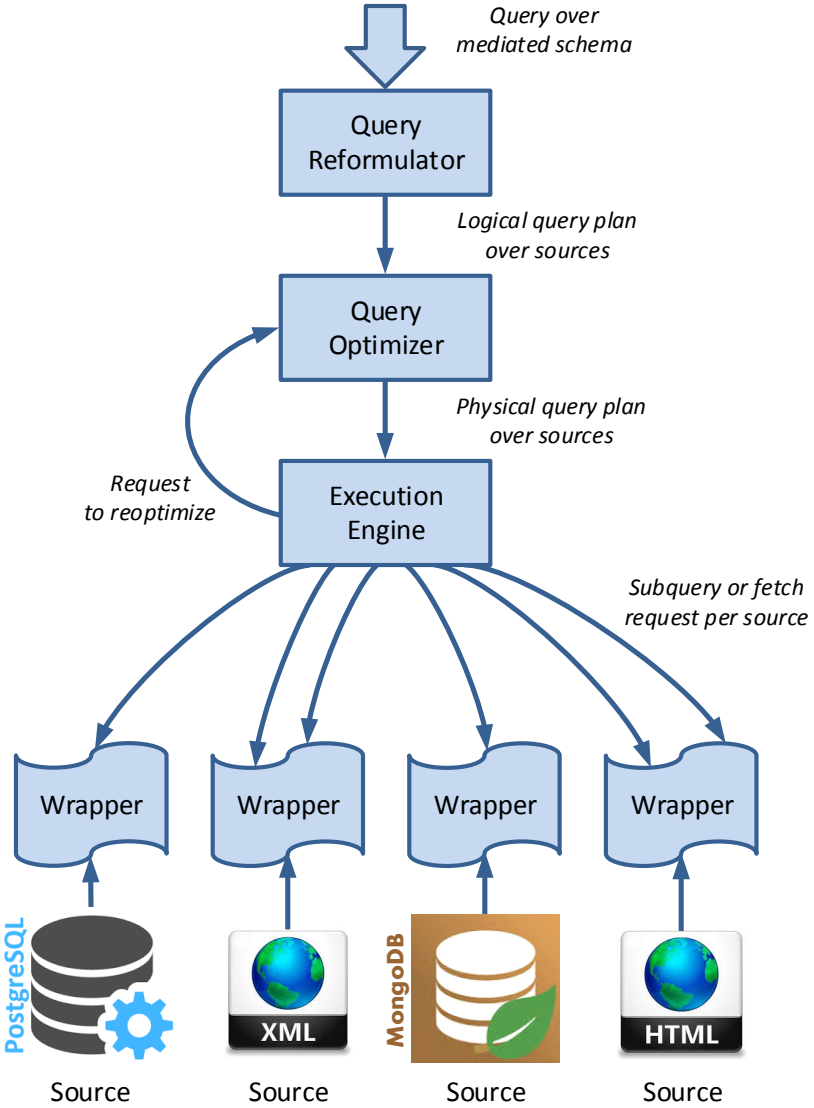
Anfragebearbeitung (Virtuell)



Anfrageumschreibung:

- Gegeben: Anfrage auf globales Schema
- Benötigt: Anfragen auf Quellschemata

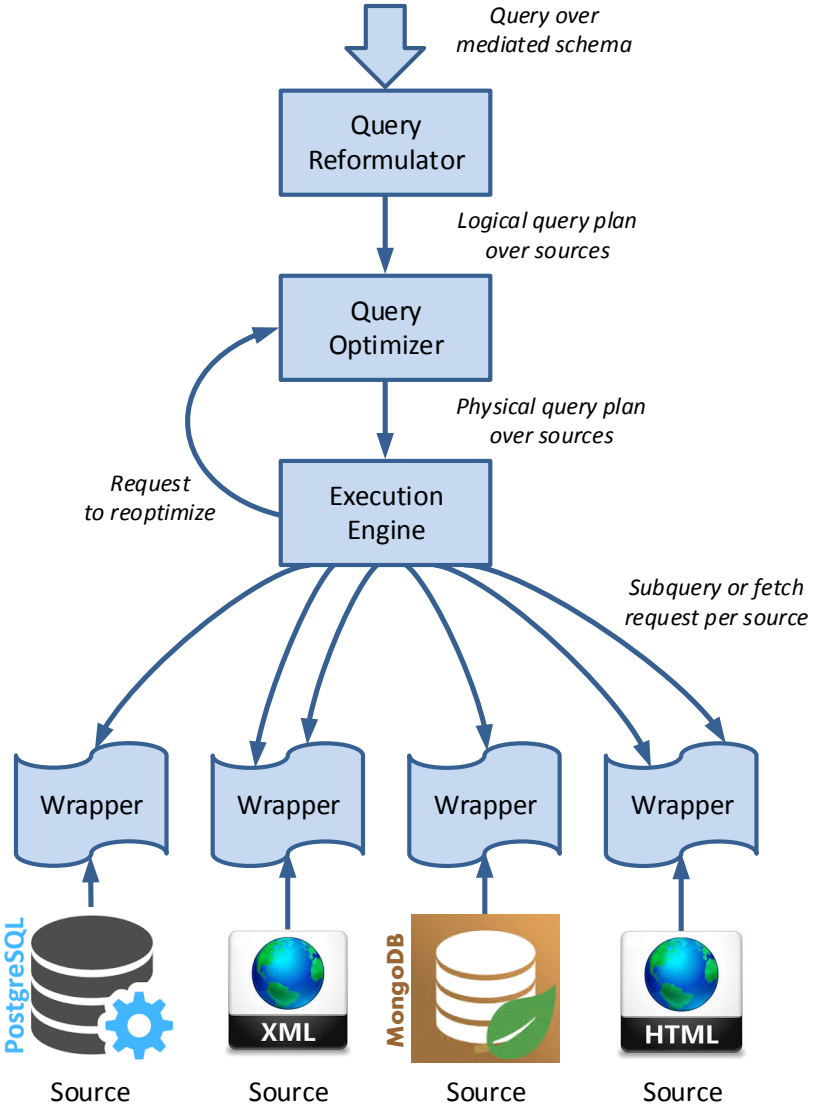
Anfragebearbeitung (Virtuell)



Anfrageumschreibung:

- Gegeben: Anfrage auf globales Schema
- Benötigt: Anfragen auf Quellschemata
- Umschreibung mit Hilfe der Quellbeschreibungen

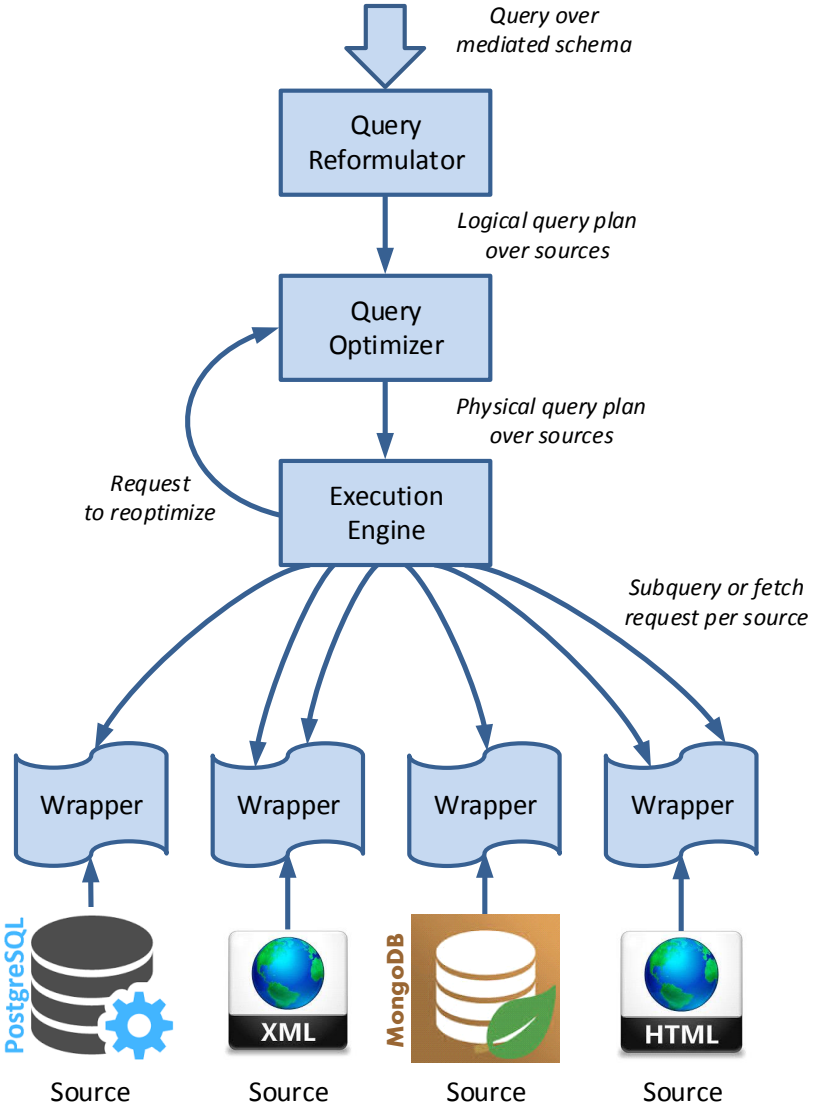
Anfragebearbeitung (Virtuell)



Anfrageumschreibung:

- Gegeben: Anfrage auf globales Schema
- Benötigt: Anfragen auf Quellschemata
- Umschreibung mit Hilfe der Quellbeschreibungen
- Ergebnis: Logischer Anfrageplan (inkl. Kombination der Quellanfragen)

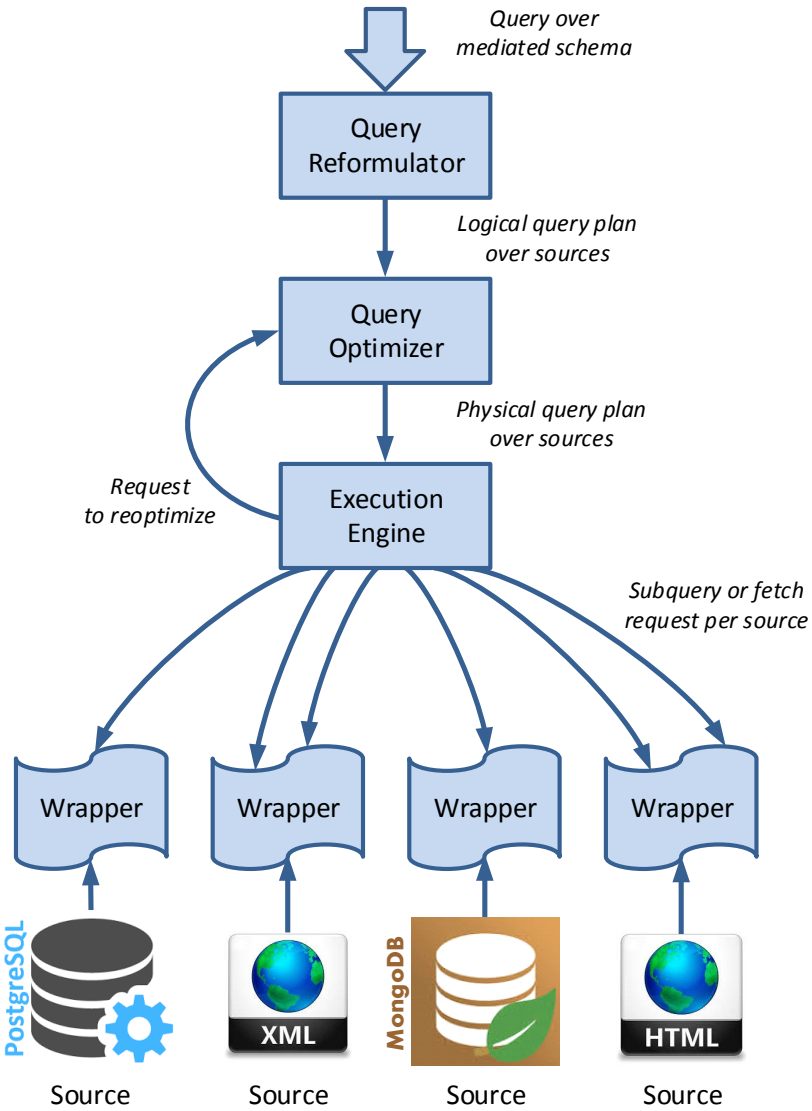
Anfragebearbeitung (Virtuell)



Anfrageumschreibung:

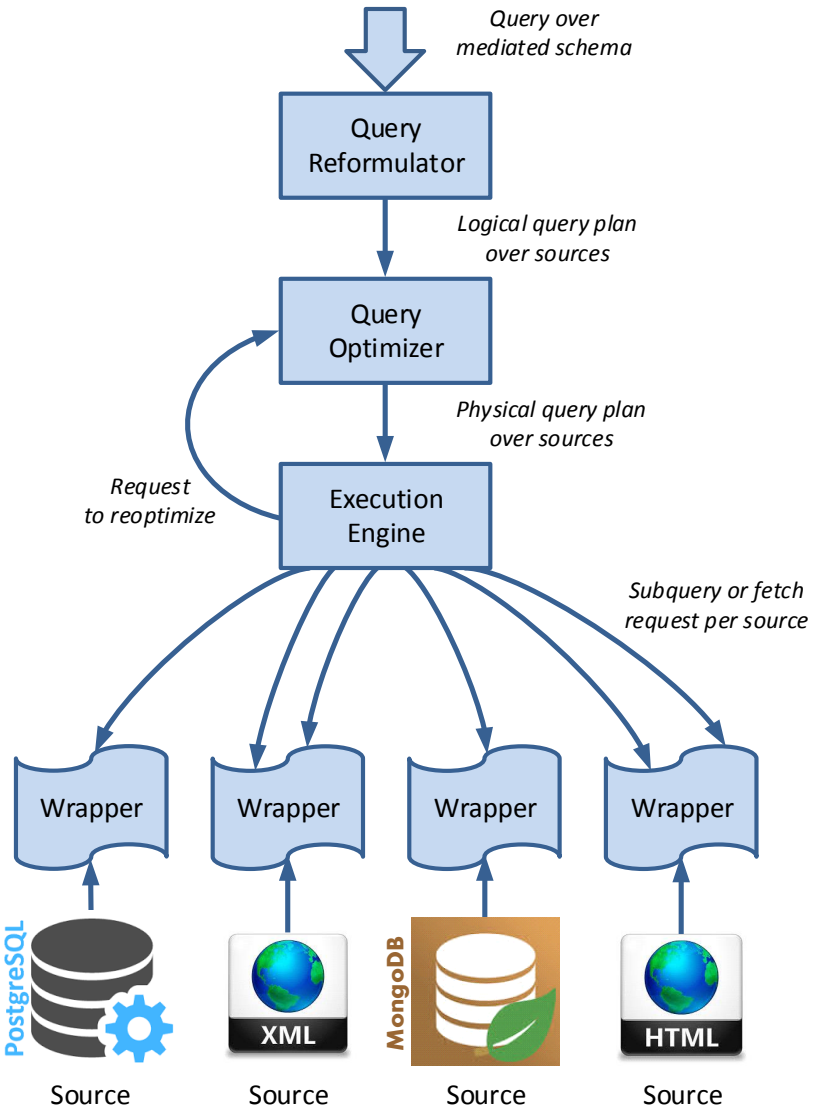
- Gegeben: Anfrage auf globales Schema
- Benötigt: Anfragen auf Quellschemata
- Umschreibung mit Hilfe der Quellbeschreibungen
- Ergebnis: Logischer Anfrageplan (inkl. Kombination der Quellanfragen)
- mehrere Logische Anfragepläne möglich

Anfragebearbeitung (Virtuell)



Anfrageoptimierung:

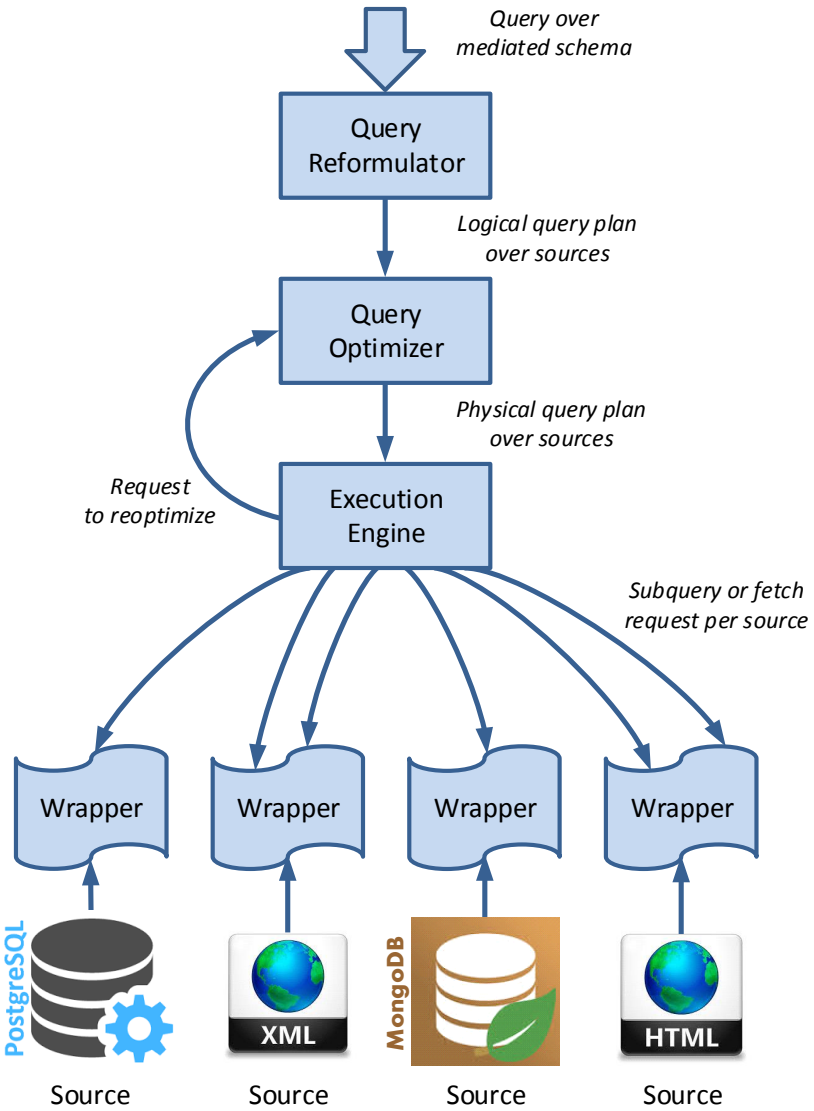
Anfragebearbeitung (Virtuell)



Anfrageoptimierung:

- Ergebnis: Physischer Anfrageplan

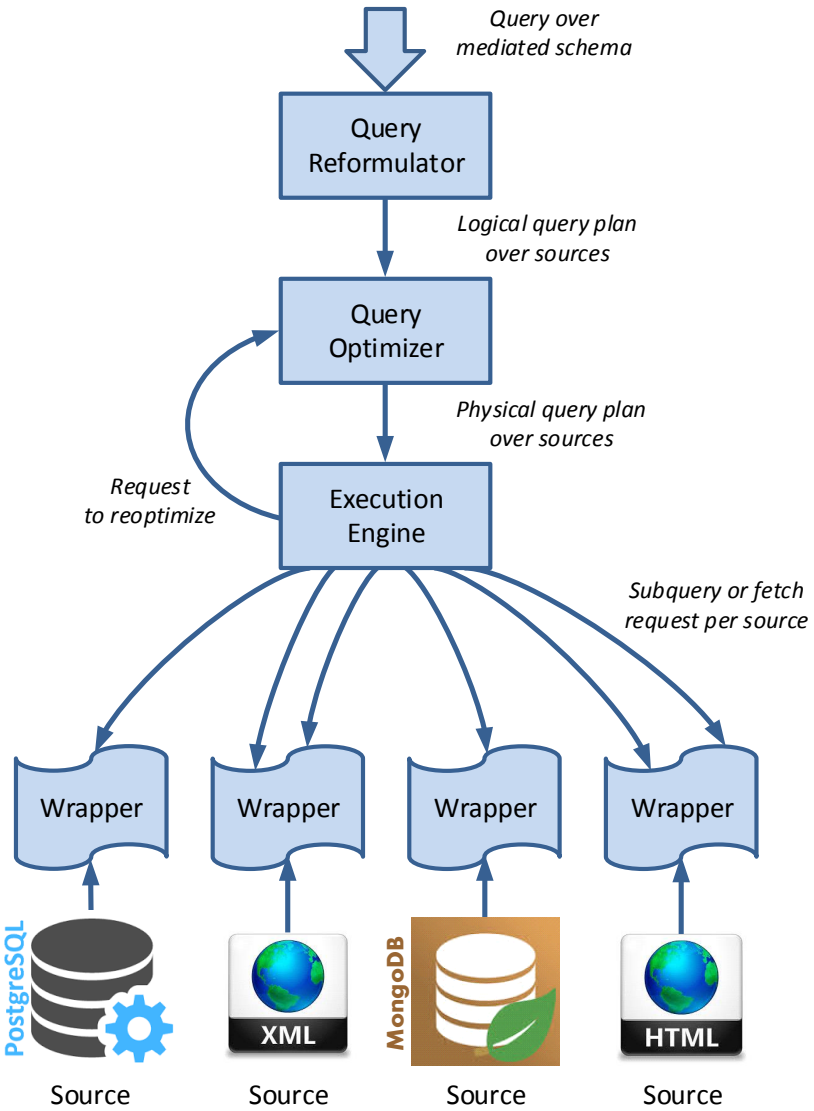
Anfragebearbeitung (Virtuell)



Anfrageoptimierung:

- Ergebnis: Physischer Anfrageplan
- bestimmt exakte Reihenfolge in der die Quellen angefragt werden

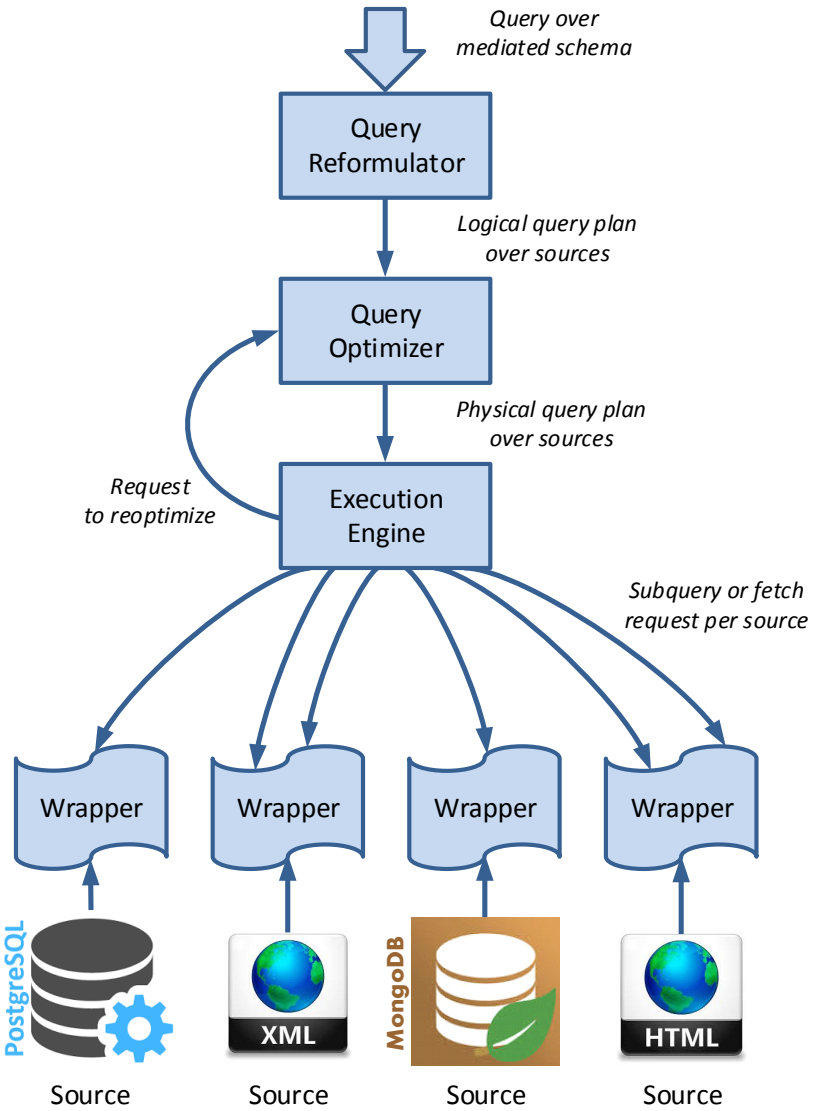
Anfragebearbeitung (Virtuell)



Anfrageoptimierung:

- Ergebnis: Physischer Anfrageplan
- bestimmt exakte Reihenfolge in der die Quellen angefragt werden
- bestimmt wann, wie (z.B. Join, Union) und wo (in Quelle oder im Zielsystem) Quellergebnisse kombiniert werden

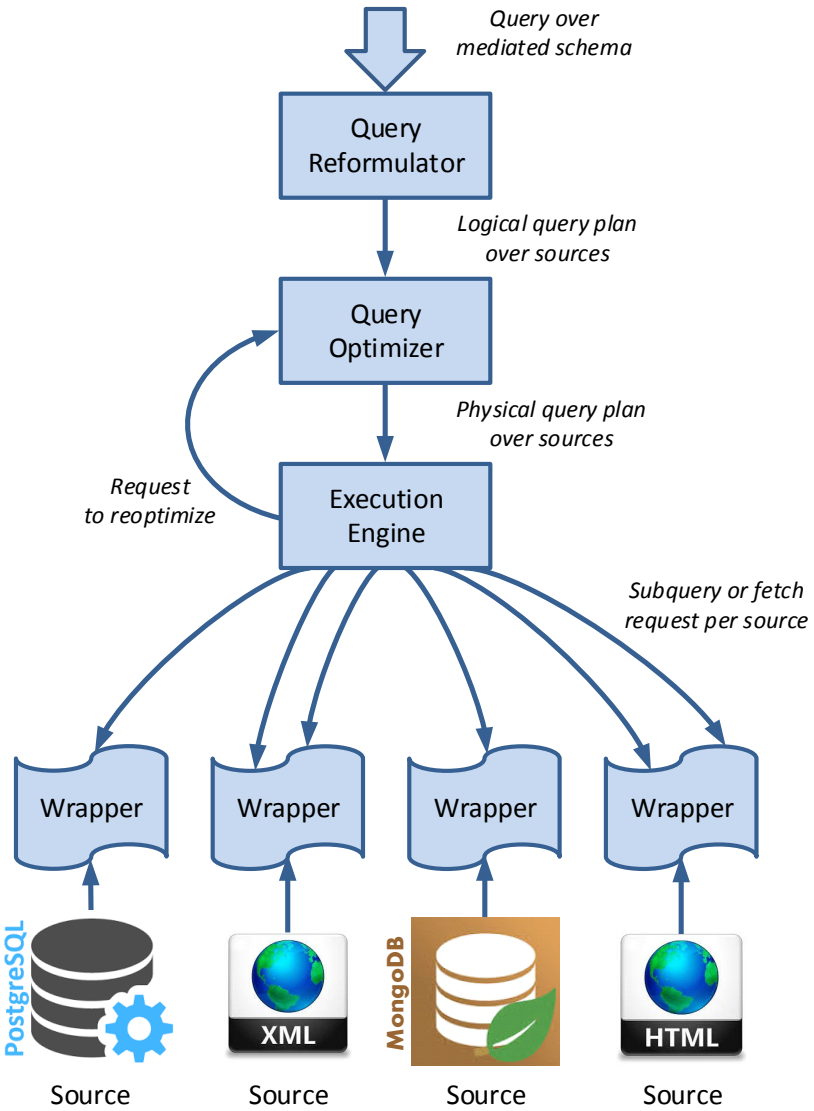
Anfragebearbeitung (Virtuell)



Anfrageoptimierung:

- Ergebnis: Physischer Anfrageplan
- bestimmt exakte Reihenfolge in der die Quellen angefragt werden
- bestimmt wann, wie (z.B. Join, Union) und wo (in Quelle oder im Zielsystem) Quellergebnisse kombiniert werden
- bestimmt wann und wo Selektionen durchgeführt werden

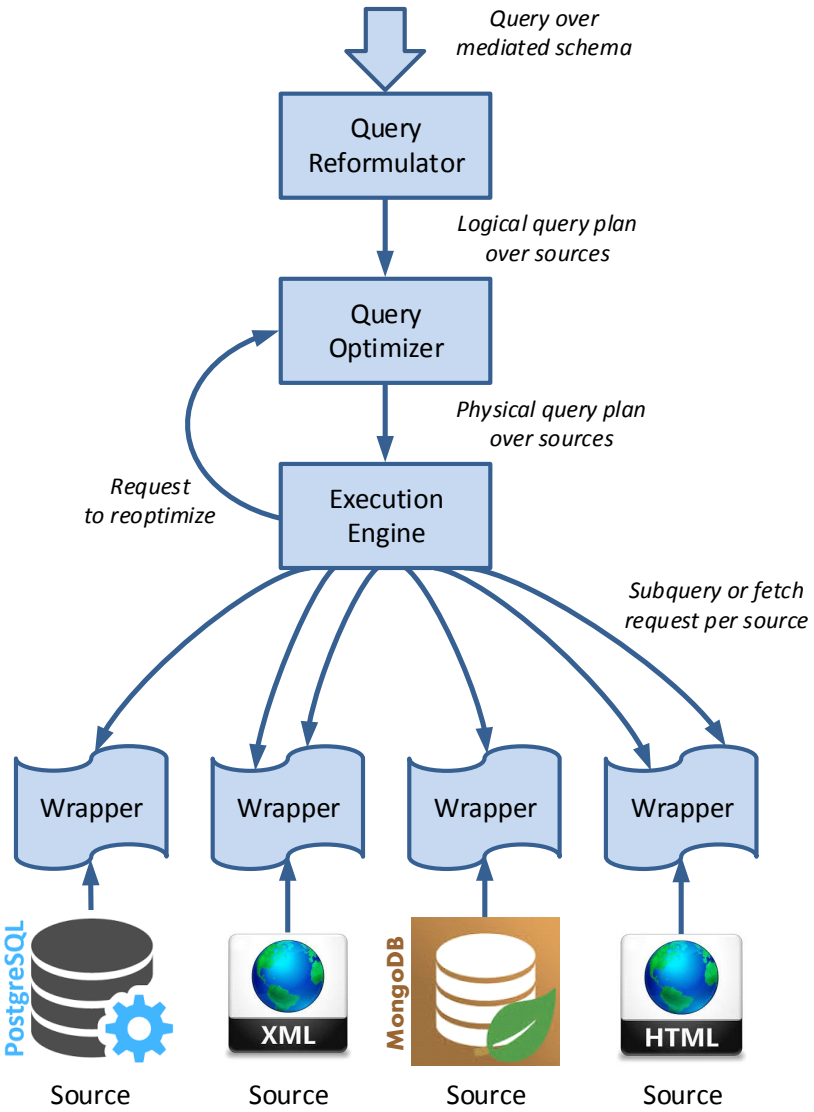
Anfragebearbeitung (Virtuell)



Anfrageoptimierung:

- Ergebnis: Physischer Anfrageplan
- bestimmt exakte Reihenfolge in der die Quellen angefragt werden
- bestimmt wann, wie (z.B. Join, Union) und wo (in Quelle oder im Zielsystem) Quellergebnisse kombiniert werden
- bestimmt wann und wo Selektionen durchgeführt werden
- Verteilung von Ressourcen des Zielsystems (Speicher,Prozessor)

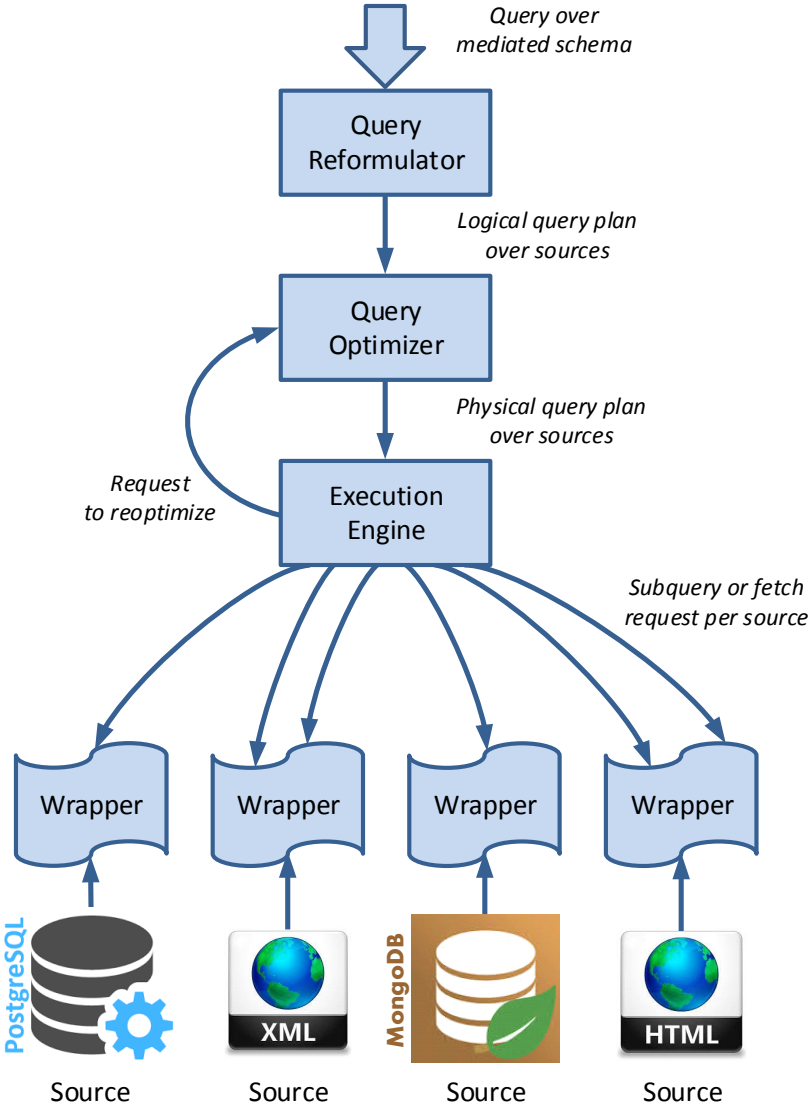
Anfragebearbeitung (Virtuell)



Anfrageoptimierung:

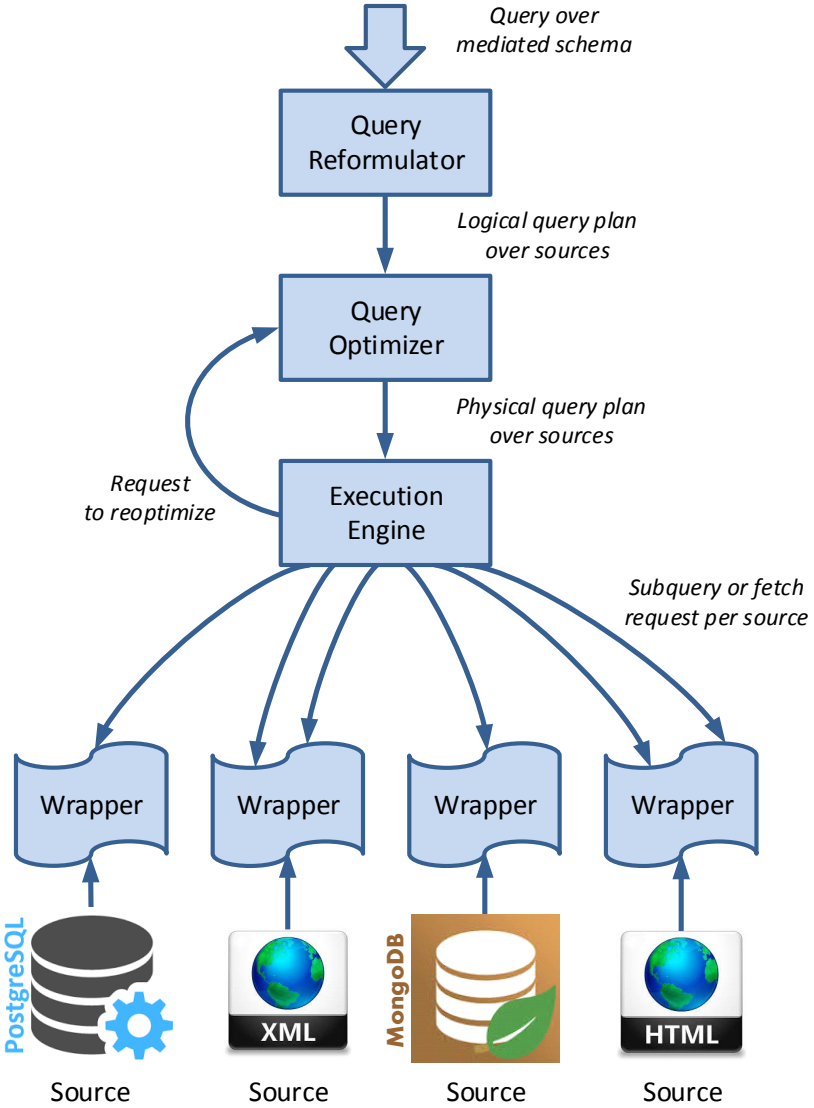
- Ergebnis: Physischer Anfrageplan
- bestimmt exakte Reihenfolge in der die Quellen angefragt werden
- bestimmt wann, wie (z.B. Join, Union) und wo (in Quelle oder im Zielsystem) Quellergebnisse kombiniert werden
- bestimmt wann und wo Selektionen durchgeführt werden
- Verteilung von Ressourcen des Zielsystems (Speicher,Prozessor)
- Schnelligkeit vs. Vollständigkeit

Anfragebearbeitung (Virtuell)



Anfrageausführung:

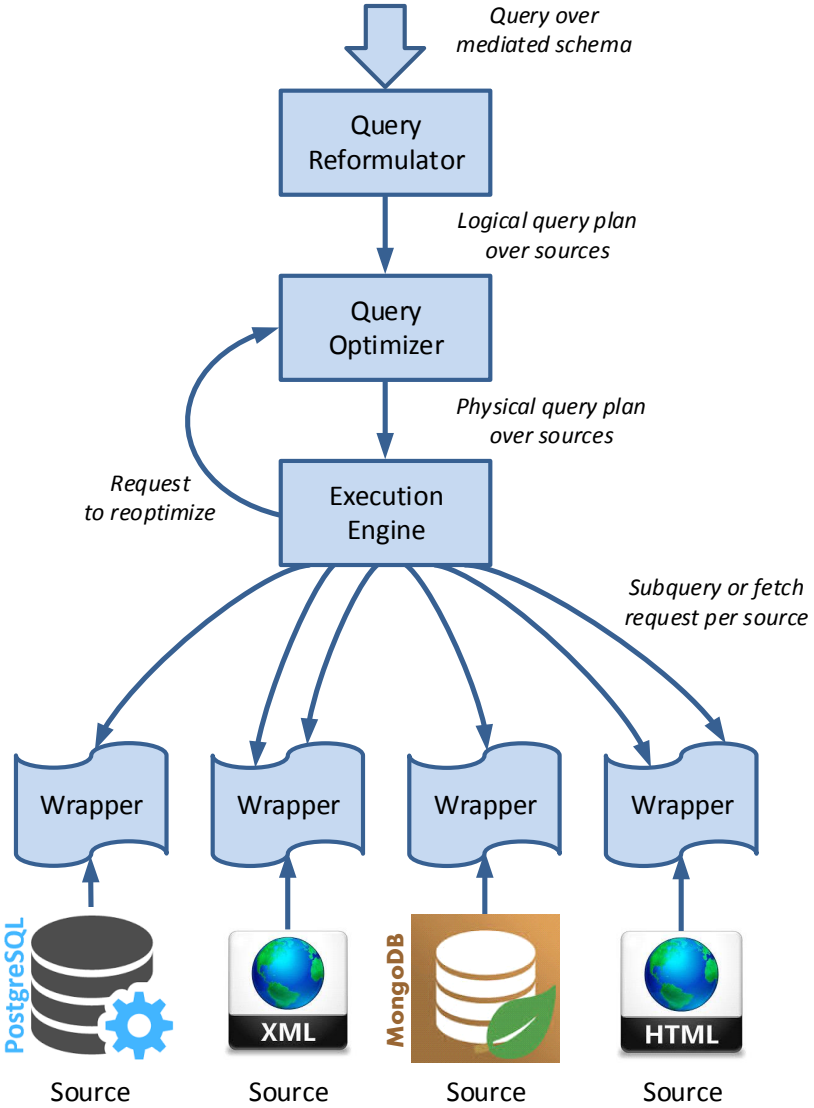
Anfragebearbeitung (Virtuell)



Anfrageausführung:

- Ausführung des Physischen Anfrageplans

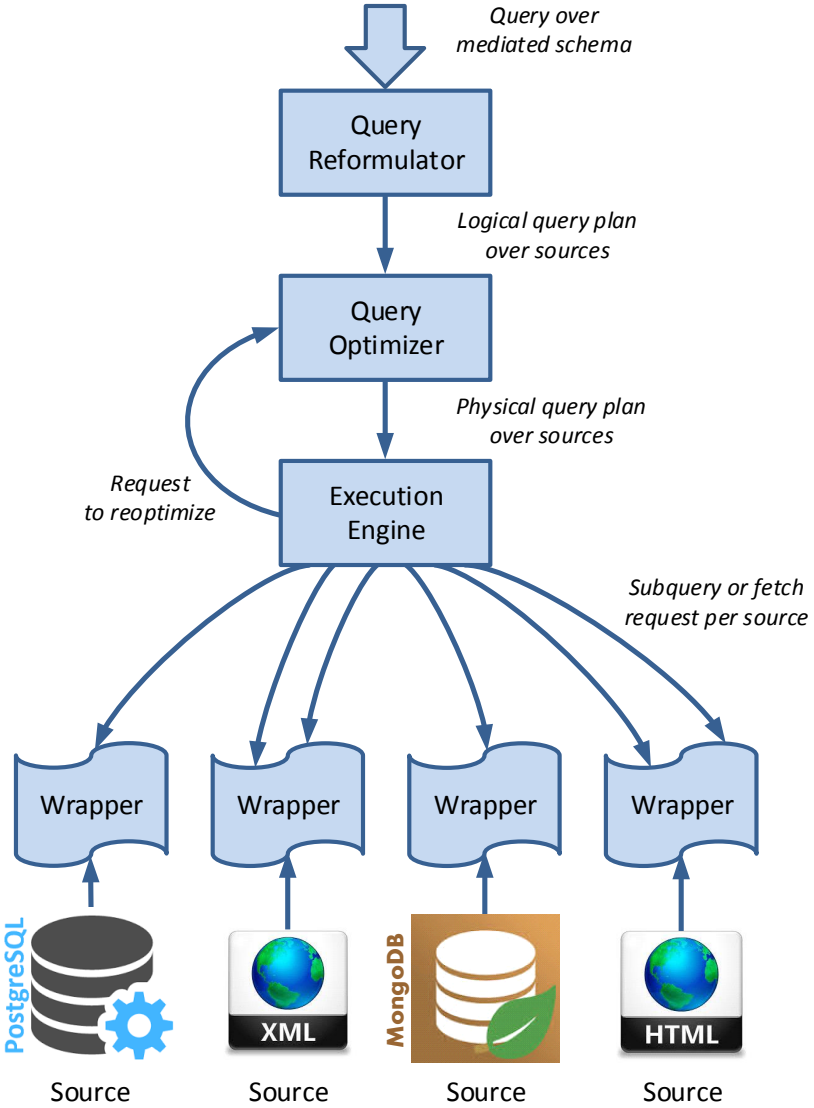
Anfragebearbeitung (Virtuell)



Anfrageausführung:

- Ausführung des Physischen Anfrageplans
- verteilt Teilanfragen an die Wrapper

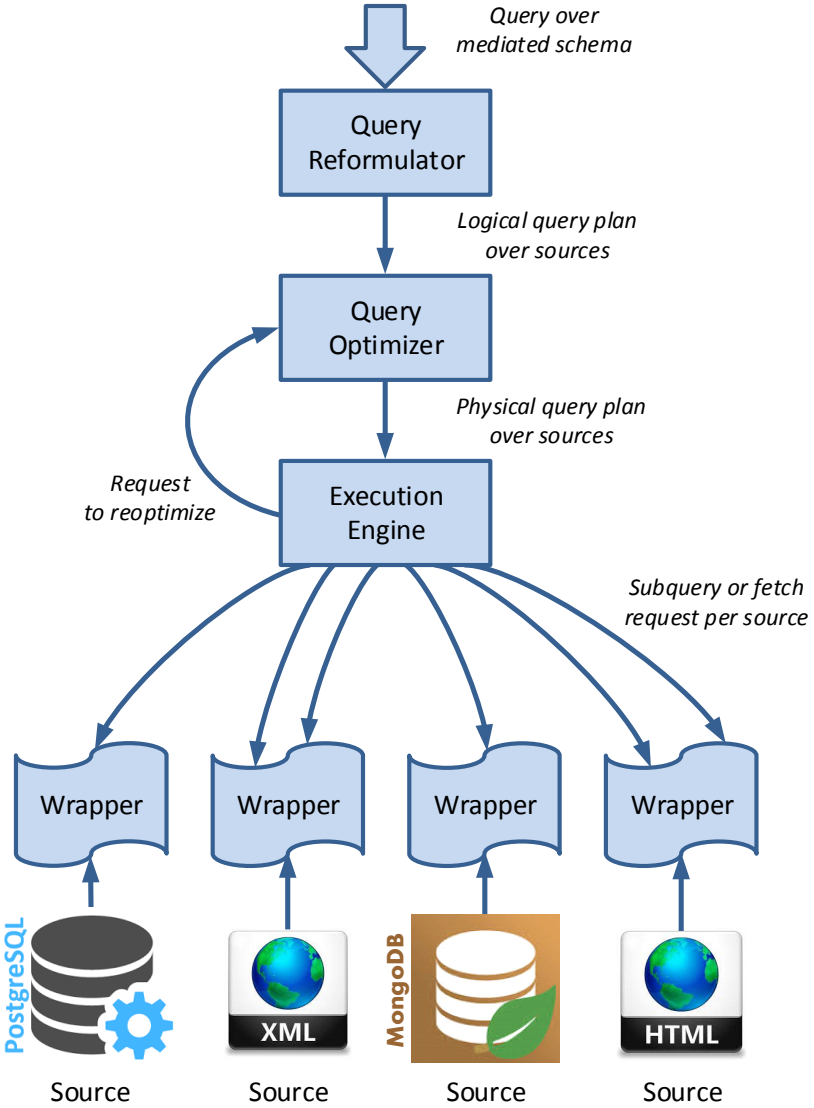
Anfragebearbeitung (Virtuell)



Anfrageausführung:

- Ausführung des Physischen Anfrageplans
- verteilt Teilanfragen an die Wrapper
- kombiniert die Ergebnisse der einzelnen Wrapper

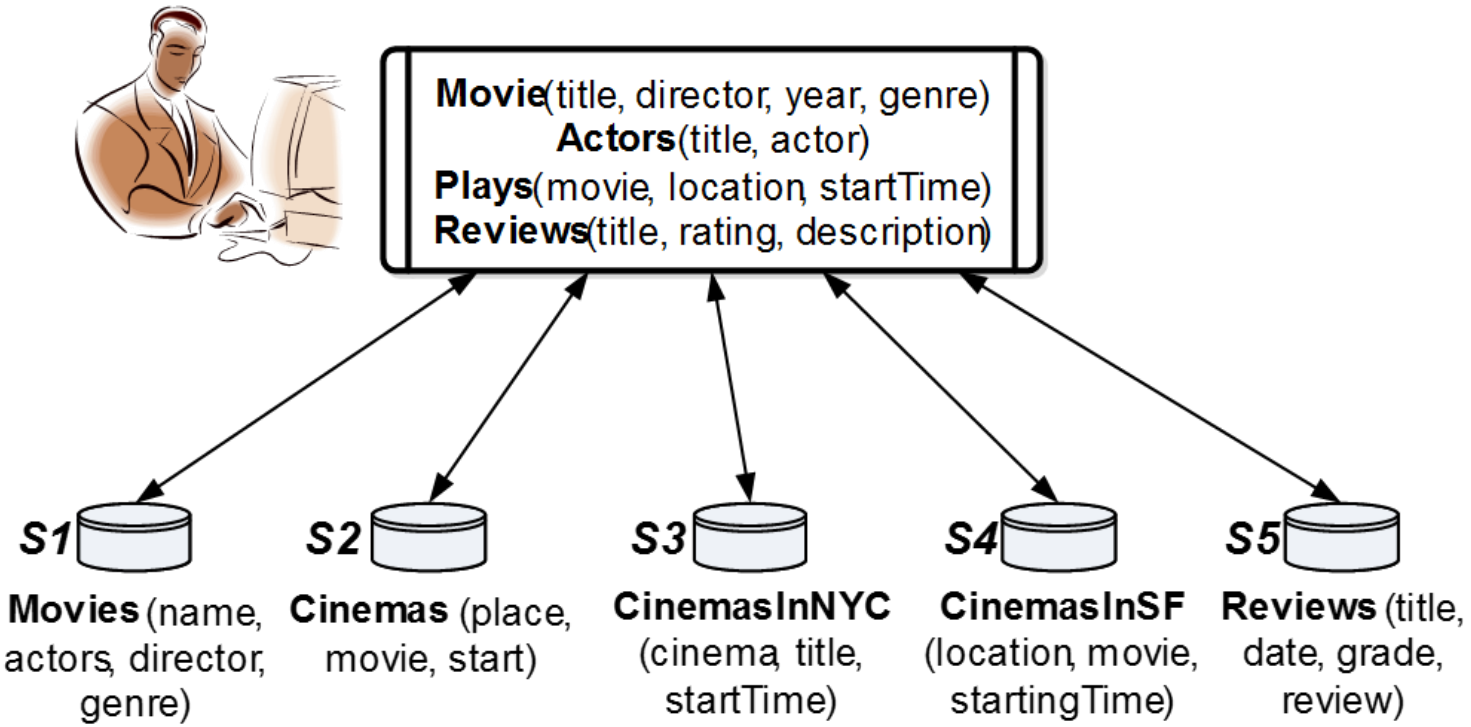
Anfragebearbeitung (Virtuell)



Anfrageausführung:

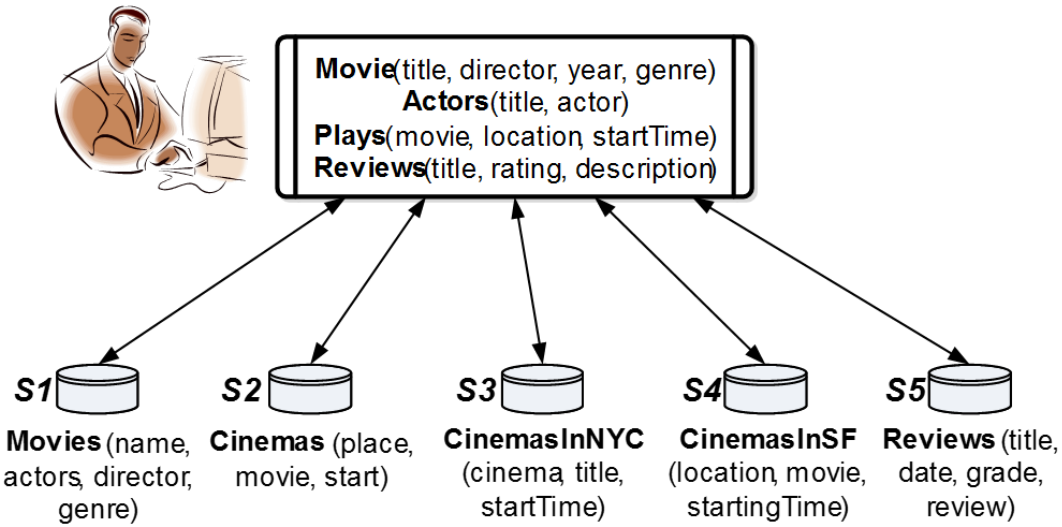
- Ausführung des Physischen Anfrageplans
- verteilt Teilanfragen an die Wrapper
- kombiniert die Ergebnisse der einzelnen Wrapper
- Anfrage beim Optimizers für einen anderen Plan falls Komplikationen auftreten (z.B. Ausfall einer Quelle)

Anfragebearbeitung am Beispiel

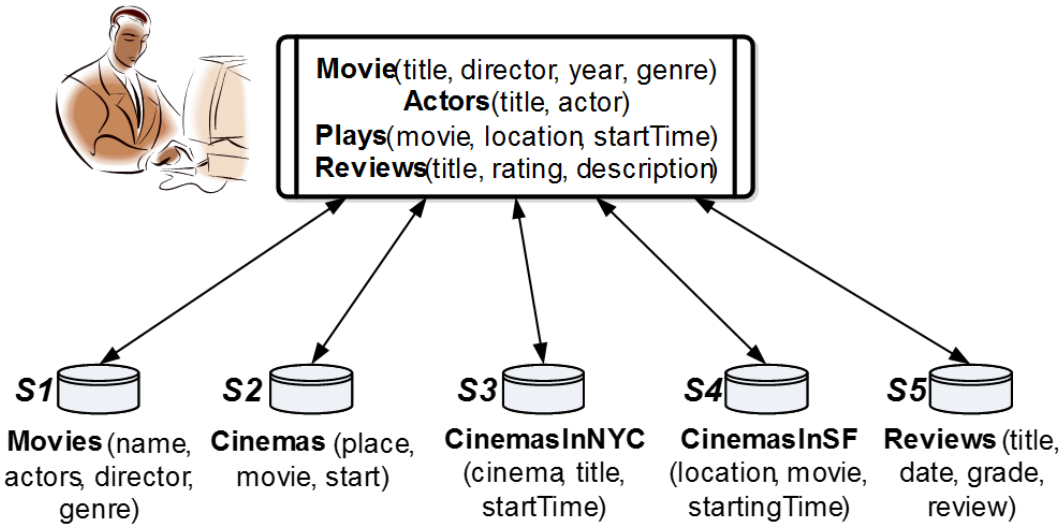


Quelle: Doan, Halevy and Ives. Principles of data Integration (slides), 2012 [DHI12]

Anfragebearbeitung am Beispiel

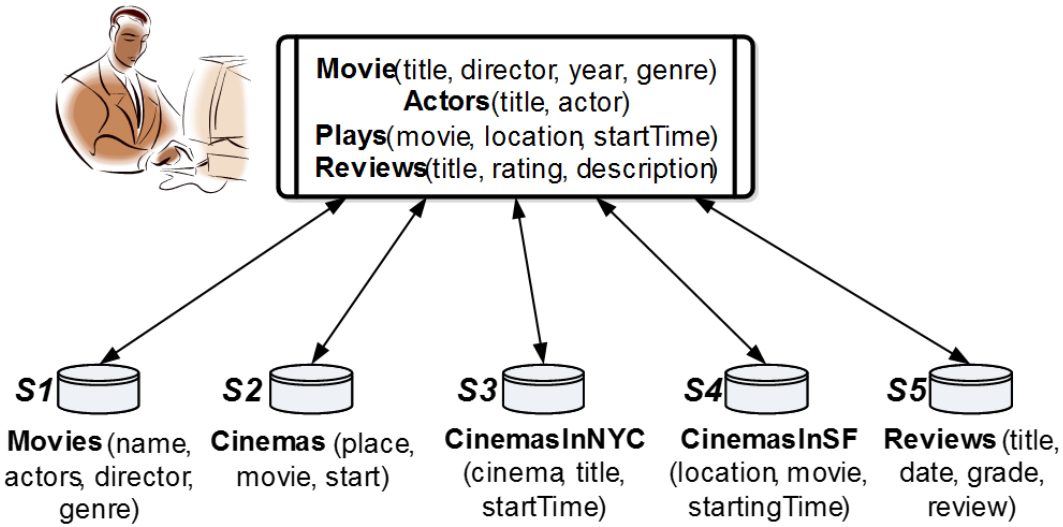


Anfragebearbeitung am Beispiel



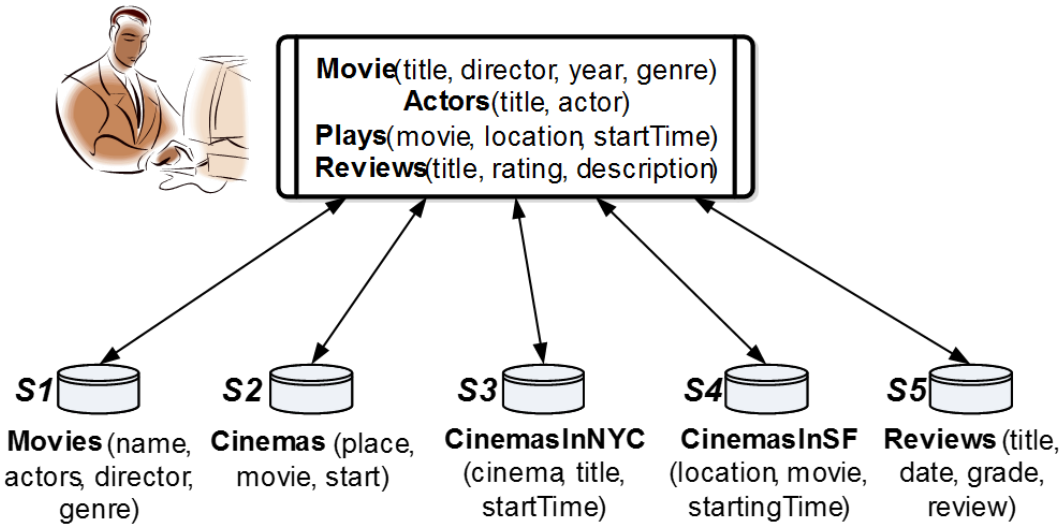
- *S1*: Filme

Anfragebearbeitung am Beispiel



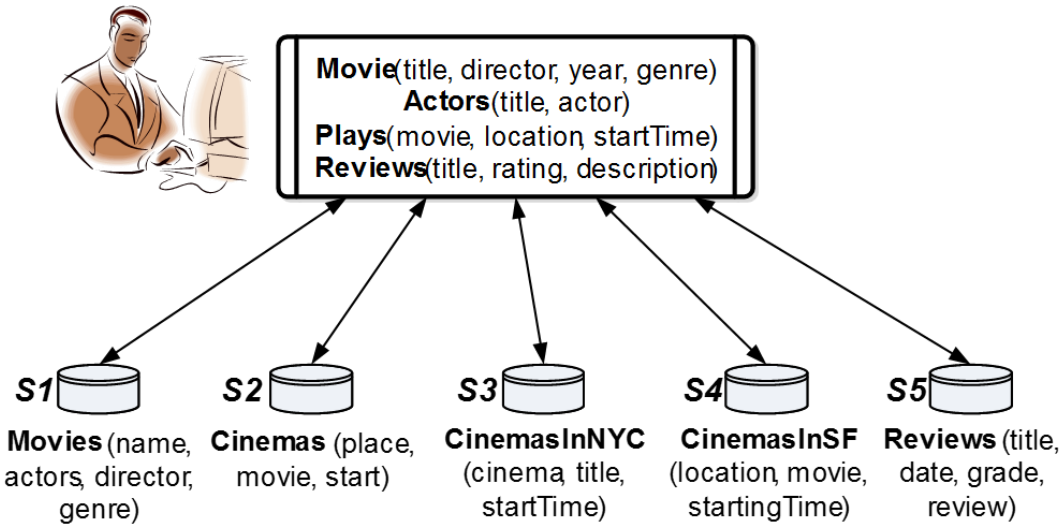
- *S1*: Filme
- *S2*: Filmvorstellungen im ganzen Land (unvollständig)

Anfragebearbeitung am Beispiel



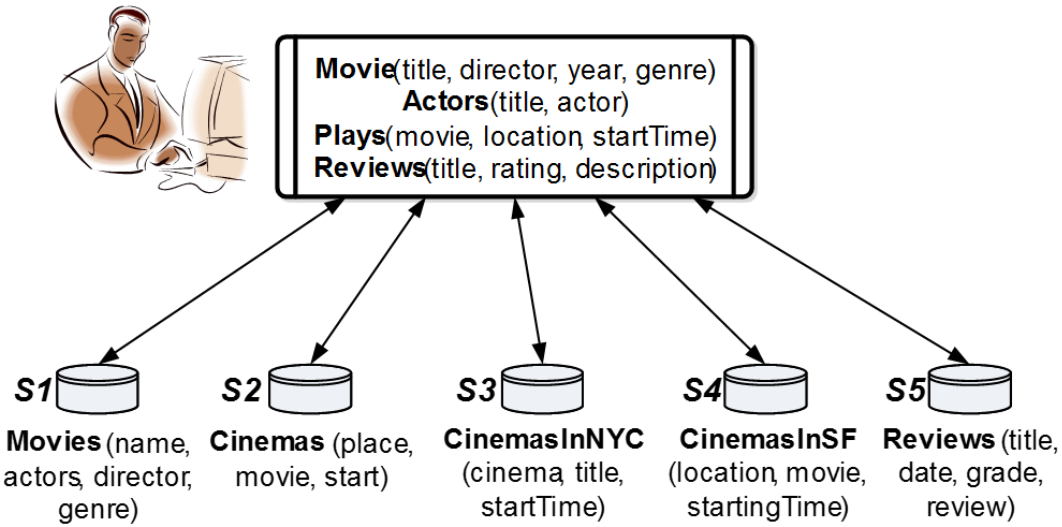
- *S1*: Filme
- *S2*: Filmvorstellungen im ganzen Land (unvollständig)
- *S3*: Filmvorstellungen in New York (vollständig)

Anfragebearbeitung am Beispiel



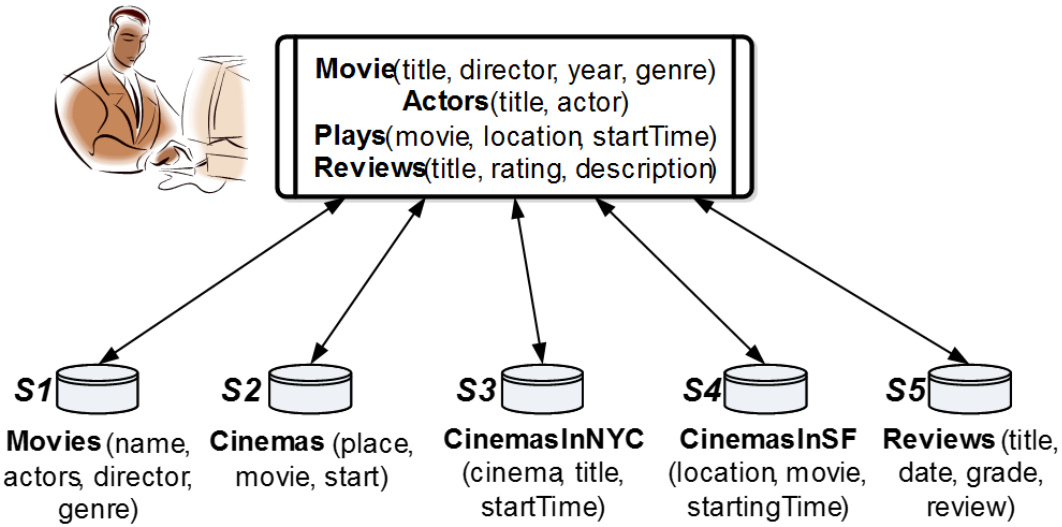
- *S1*: Filme
- *S2*: Filmvorstellungen im ganzen Land (unvollständig)
- *S3*: Filmvorstellungen in New York (vollständig)
- *S4*: Filmvorstellungen in San Francisco

Anfragebearbeitung am Beispiel



- *S1*: Filme
- *S2*: Filmvorstellungen im ganzen Land (unvollständig)
- *S3*: Filmvorstellungen in New York (vollständig)
- *S4*: Filmvorstellungen in San Francisco
- *S5*: Filmreviews

Anfragebearbeitung am Beispiel



- *S1*: Filme
- *S2*: Filmvorstellungen im ganzen Land (unvollständig)
- *S3*: Filmvorstellungen in New York (vollständig)
- *S4*: Filmvorstellungen in San Francisco
- *S5*: Filmreviews
- *S2* - *S4* benötigen einen Filmtitel als Eingabe

Anfragebearbeitung am Beispiel

Filmvorstellungen in New York bei denen der Regisseur 'Woody Allen' heißt:

Movie:	title, director, year, genre
Actors:	title, actor
Plays:	movie, location, startTime
Reviews:	title, rating, description

Anfragebearbeitung am Beispiel

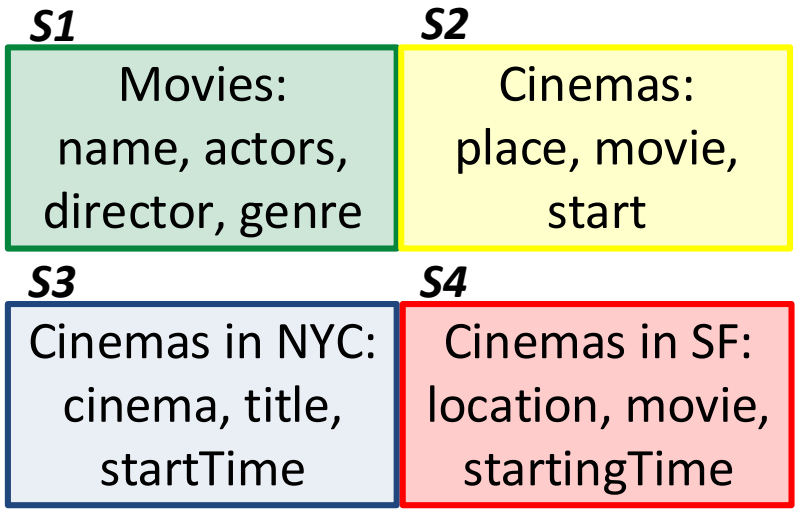
Filmvorstellungen in New York bei denen der Regisseur 'Woody Allen' heißt:

Movie:	title, director, year, genre
Actors:	title, actor
Plays:	movie, location, startTime
Reviews:	title, rating, description

```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

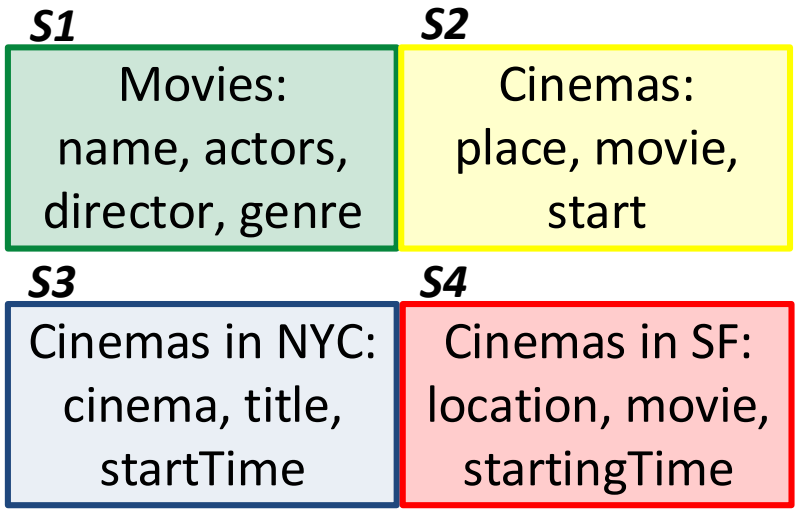
Quelle: Doan, Halevy and Ives. Principles of data Integration (slides), 2012 [DHI12]

Anfragebearbeitung am Beispiel




```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

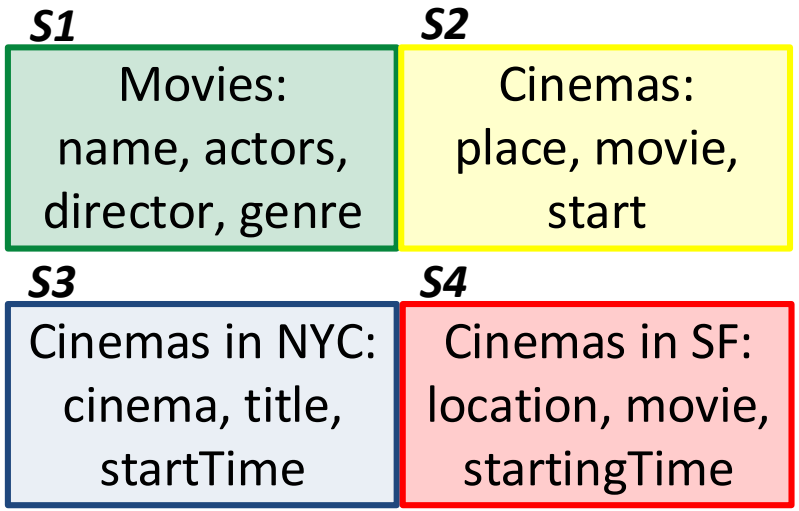
Anfragebearbeitung am Beispiel




```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrageumschreibung:

Anfragebearbeitung am Beispiel

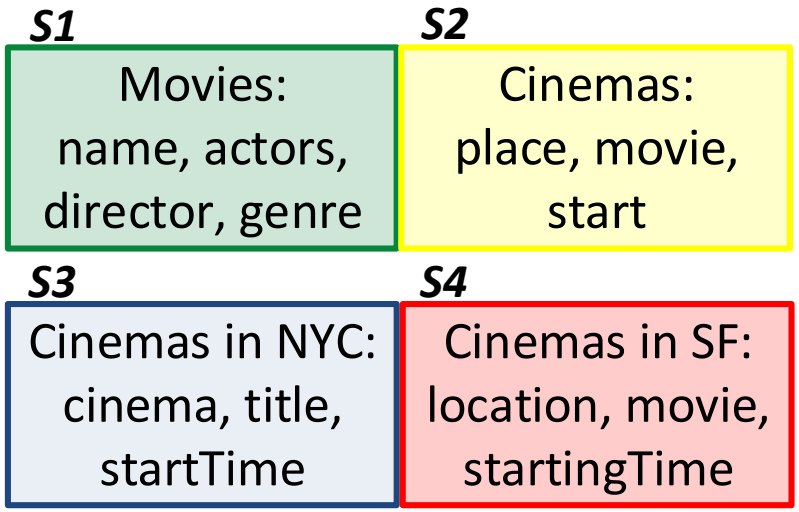



```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrageumschreibung:

- Tuples für *Movie* können Quelle *S1* entnommen werden

Anfragebearbeitung am Beispiel

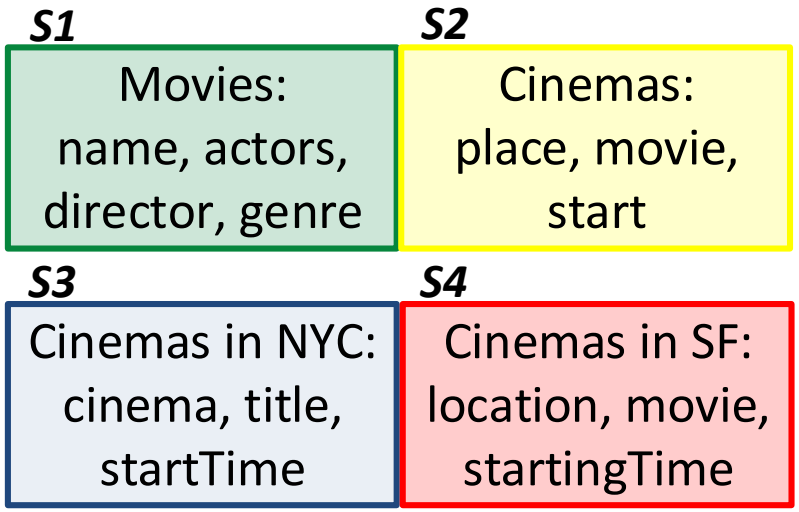



```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrageumschreibung:

- Tuples für *Movie* können Quelle *S1* entnommen werden
- Tuples für *Plays* in New York können den Quellen *S2* und *S3* entnommen werden (*S3* ist vollständig für New York)

Anfragebearbeitung am Beispiel



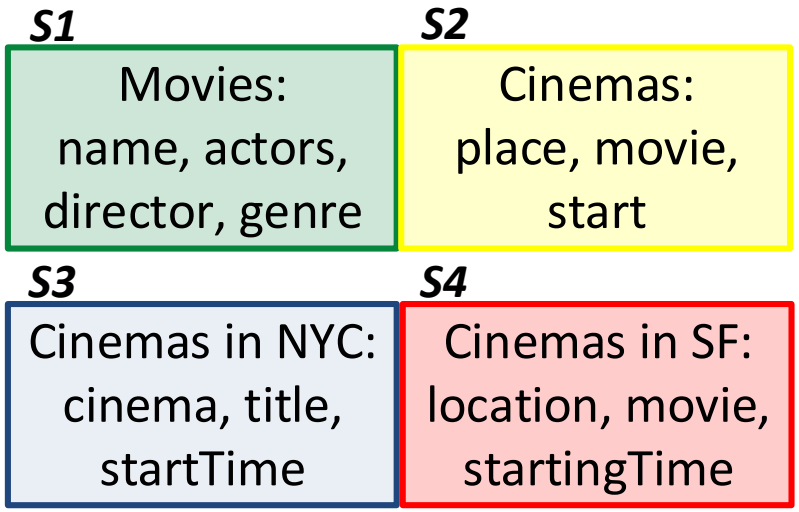

```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrageumschreibung:

- Tuples für *Movie* können Quelle *S1* entnommen werden
- Tuples für *Plays* in New York können den Quellen *S2* und *S3* entnommen werden (*S3* ist vollständig für New York)
- *S2* und *S3* benötigen Filmtitel (nicht Teil der Anfrage)

$\Rightarrow$ *S1* muss zuerst angefragt werden

Anfragebearbeitung am Beispiel




```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

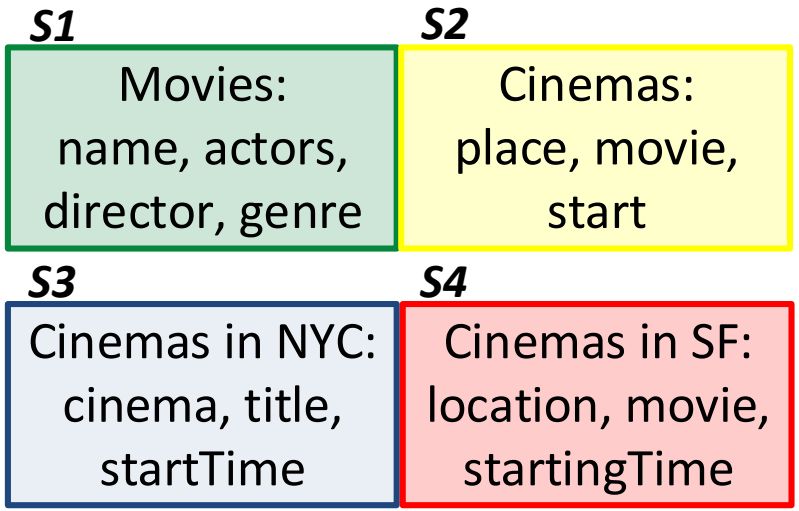
Anfrageumschreibung:

- Tuples für *Movie* können Quelle *S1* entnommen werden
- Tuples für *Plays* in New York können den Quellen *S2* und *S3* entnommen werden (*S3* ist vollständig für New York)
- *S2* und *S3* benötigen Filmtitel (nicht Teil der Anfrage)

$\Rightarrow$ *S1* muss zuerst angefragt werden

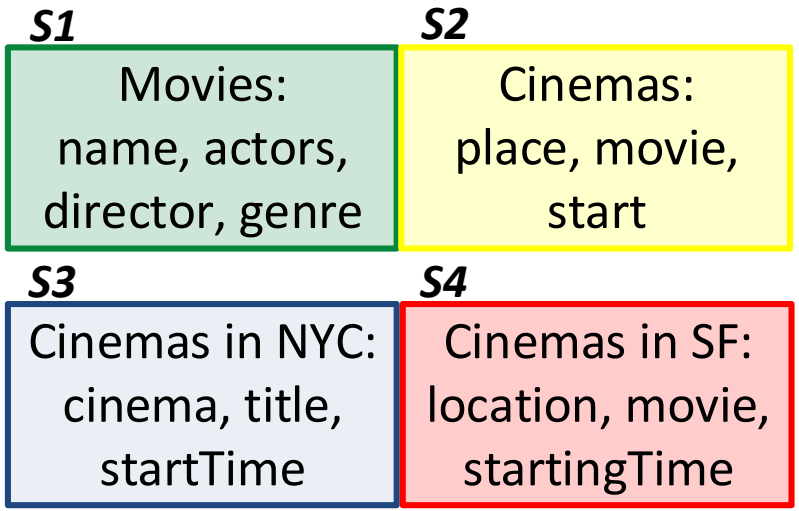
- Zwei Logische Anfragepläne (*S1* und *S2* oder *S1* und *S3*)

Anfragebearbeitung am Beispiel




```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfragebearbeitung am Beispiel

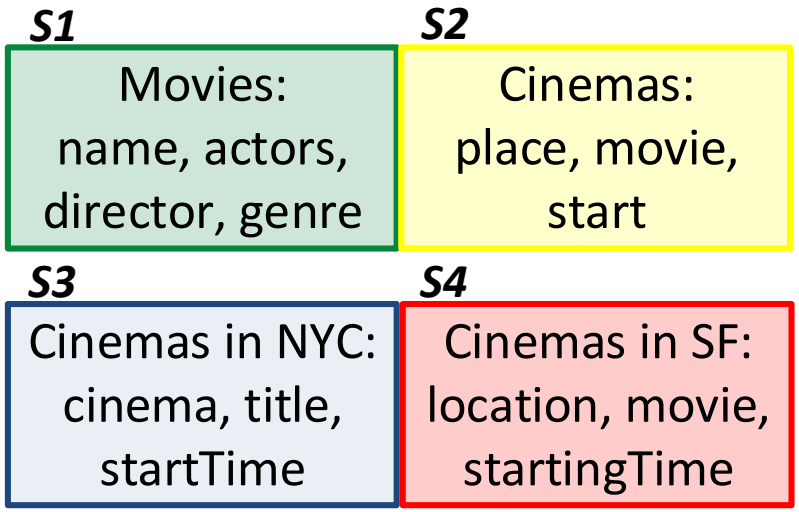



```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrage für Quelle *S1*:

```
SELECT  name AS title
FROM    Movies
WHERE   director = "Woody Allen"
```

Anfragebearbeitung am Beispiel



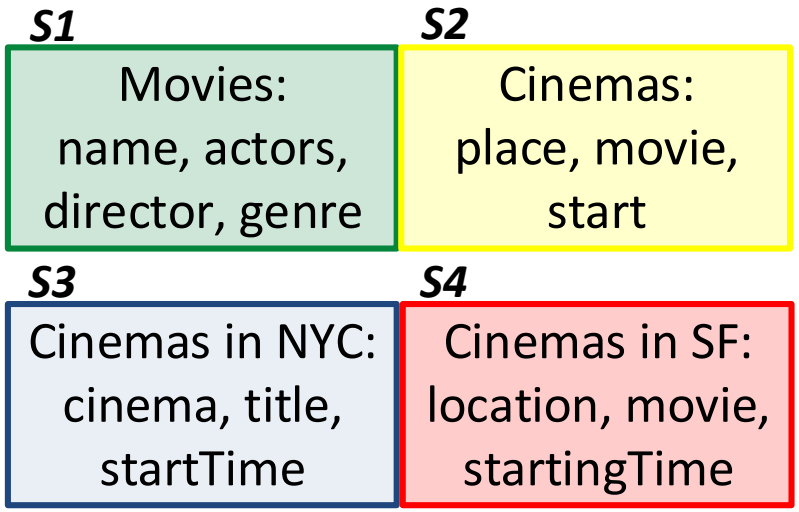

```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrage für Quelle *S1*:

```
SELECT  name AS title
FROM    Movies
WHERE   director = "Woody Allen"
```

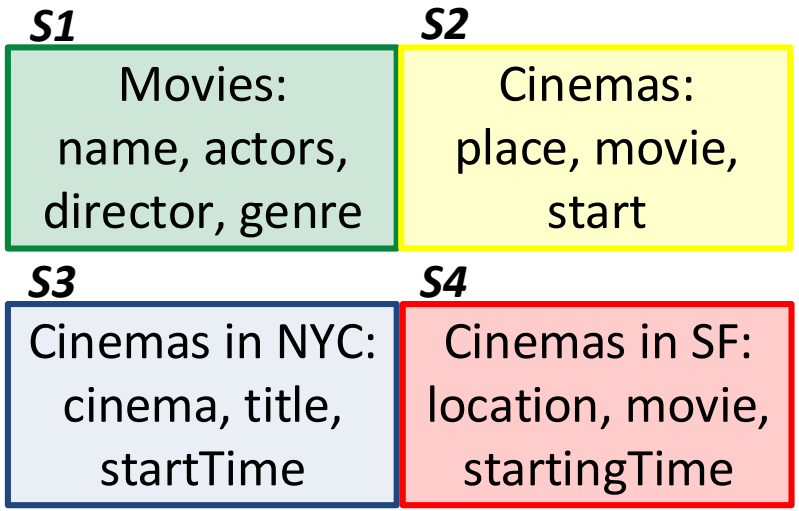
- Selektion auf Regisseur kann direkt in Quelle berechnet werden
- Umbenennung des Attributes *name* in *title*

Anfragebearbeitung am Beispiel




```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfragebearbeitung am Beispiel

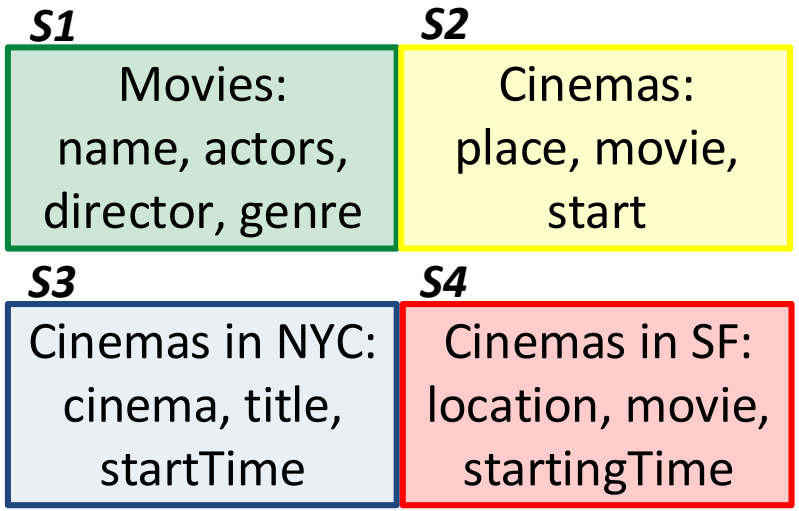



```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrage für Quelle *S2*:

```
SELECT  movie AS title, start AS startTime
FROM    Cinemas
WHERE   place = "New York"
  AND   movie = @argument
```

Anfragebearbeitung am Beispiel



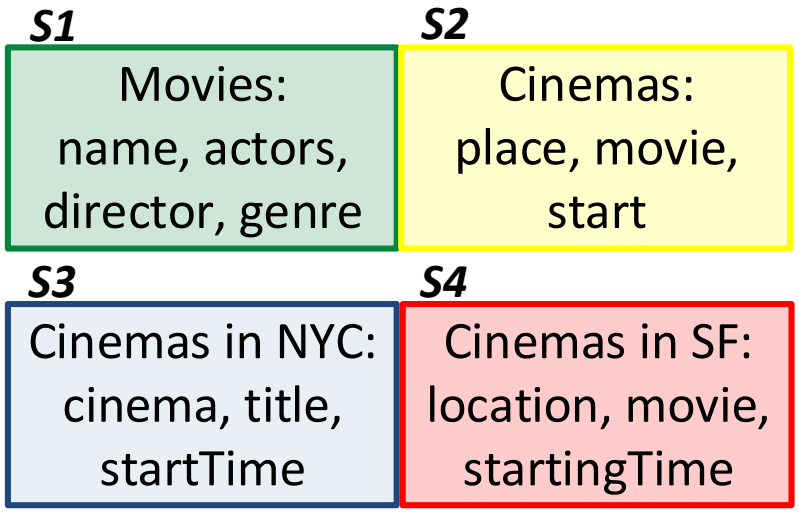

```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrage für Quelle *S2*:

```
SELECT  movie AS title, start AS startTime
FROM    Cinemas
WHERE   place = "New York"
  AND   movie = @argument
```

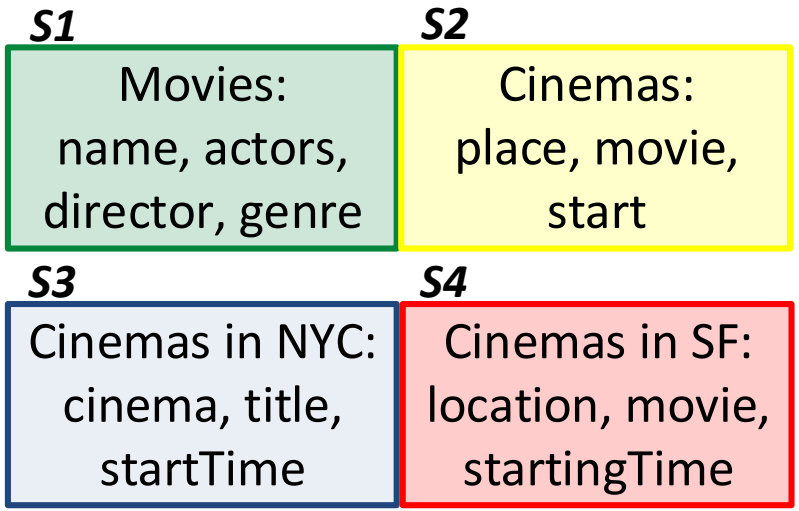
- Selektion auf Ort kann direkt in Quelle berechnet werden
- Umbenennung der Attribute *movie* und *start*

Anfragebearbeitung am Beispiel




```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfragebearbeitung am Beispiel

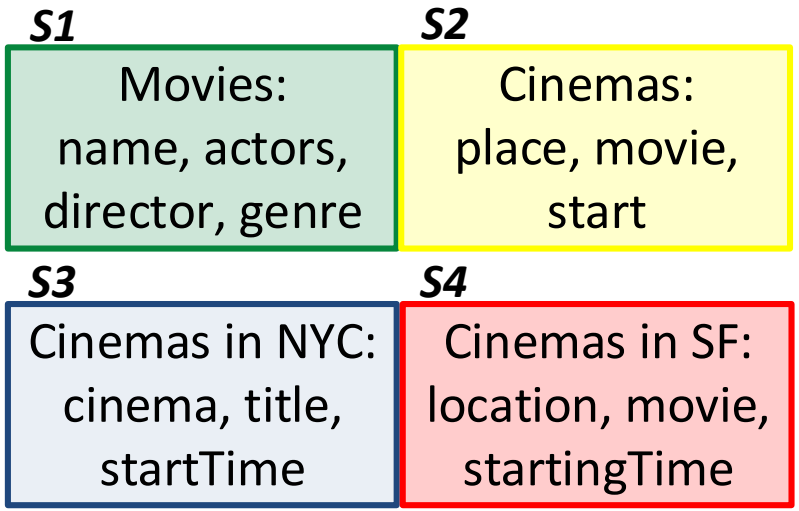



```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrage für Quelle *S3*:

```
SELECT  title, startTime
FROM    CinemasInNYC
WHERE   title = @argument
```

Anfragebearbeitung am Beispiel



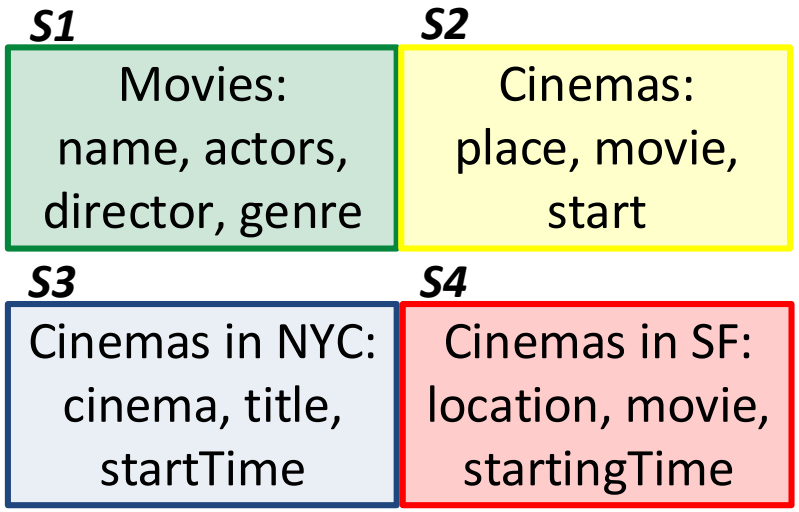

```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrage für Quelle *S3*:

```
SELECT  title, startTime
FROM    CinemasInNYC
WHERE   title = @argument
```

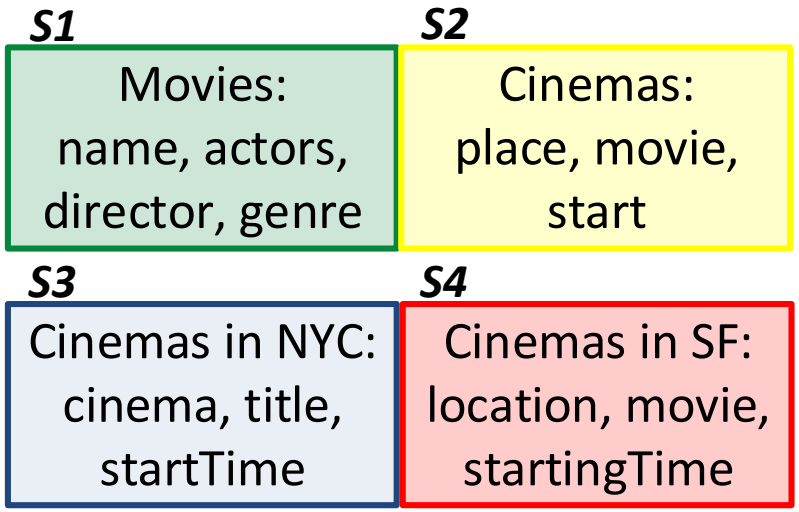
- Selektion auf Ort ist hier nicht notwendig
  (*S3* enthält nur Filme aus New York)
- Umbenennung der Attribute nicht notwendig

Anfragebearbeitung am Beispiel




```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

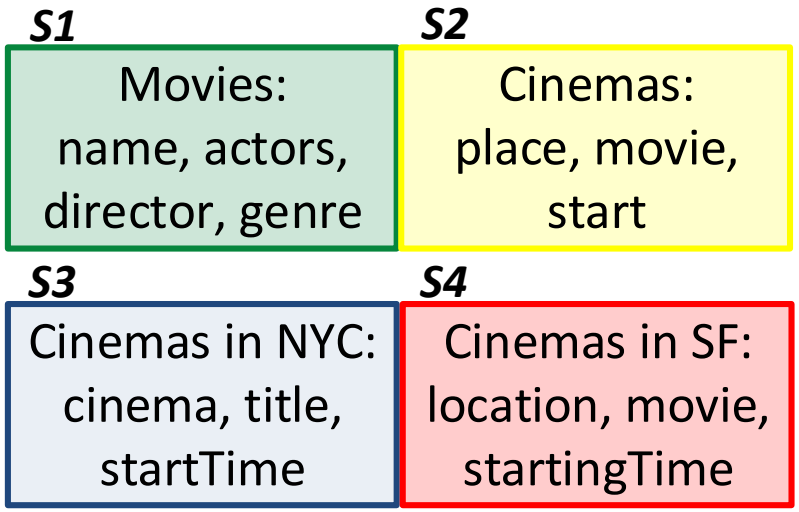
Anfragebearbeitung am Beispiel




```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Auswahl eines oder mehrerer Pläne:

Anfragebearbeitung am Beispiel



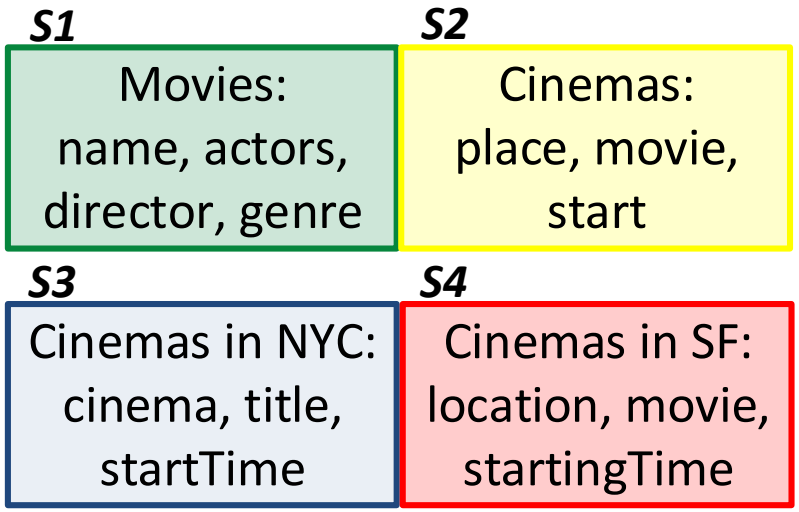

```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Auswahl eines oder mehrerer Pläne:

- *S3* ist vollständig für New York
- *S2* ist evtl. unvollständig für New York

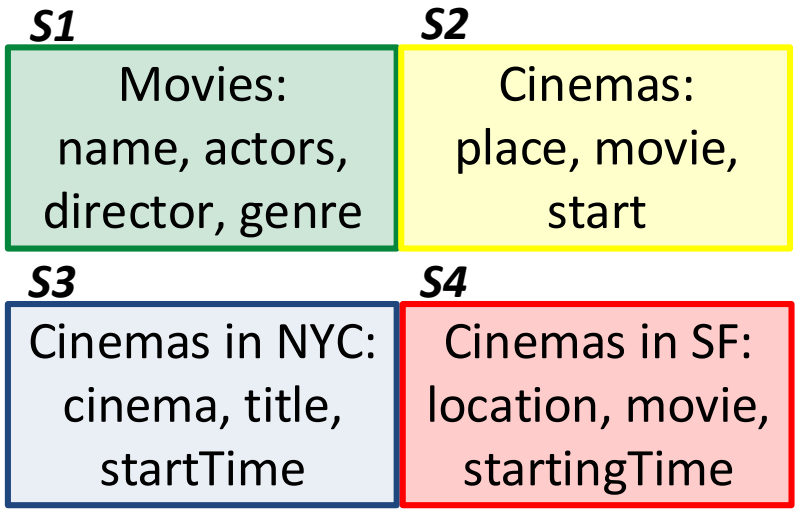
$\Rightarrow$ Wenn nur ein Plan ausgeführt werden soll, dann einer mit *S3*

Anfragebearbeitung am Beispiel




```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

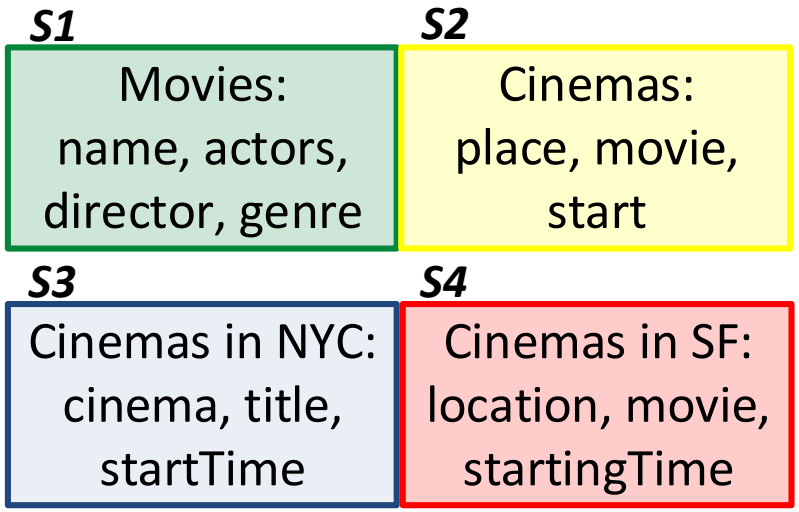
Anfragebearbeitung am Beispiel




```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrageoptimierung des Planes mit *S1* und *S3*:

Anfragebearbeitung am Beispiel

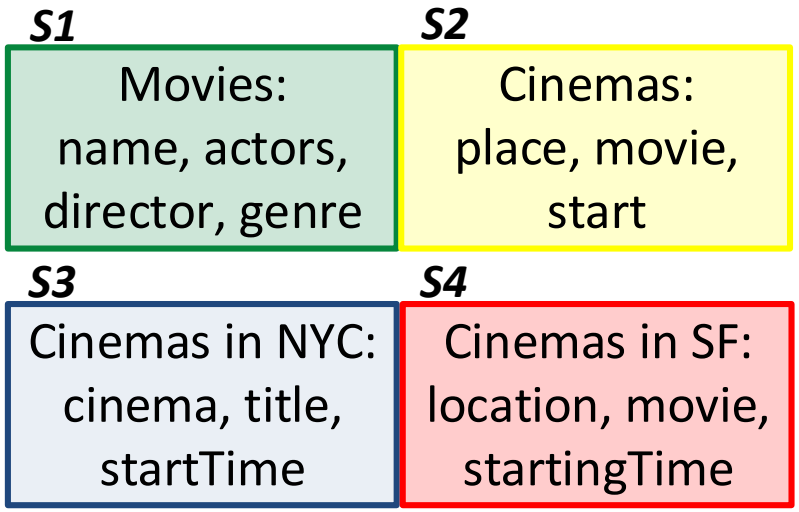



```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrageoptimierung des Planes mit *S1* und *S3*:

- Auswahl eines Algorithmus um *S1* und *S3* zu joinen (streaming Tuples von *S1* zu *S3* oder komplett *S1* vor *S3*)

Anfragebearbeitung am Beispiel

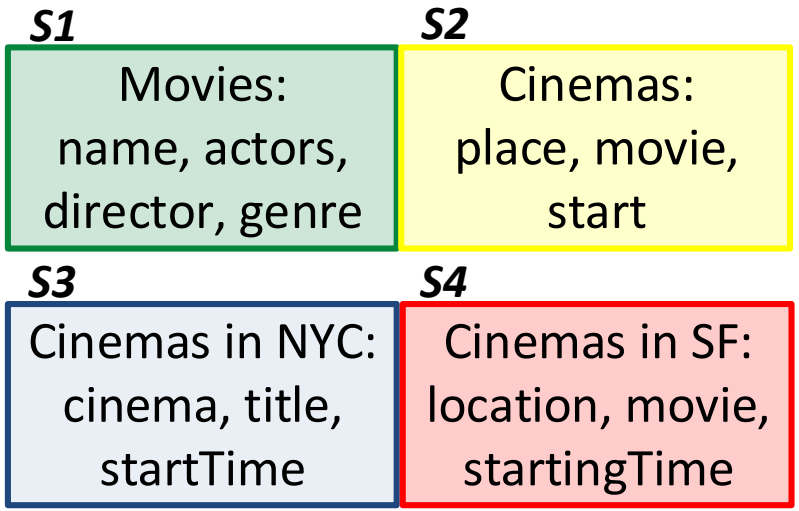



```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrageoptimierung des Planes mit *S1* und *S3*:

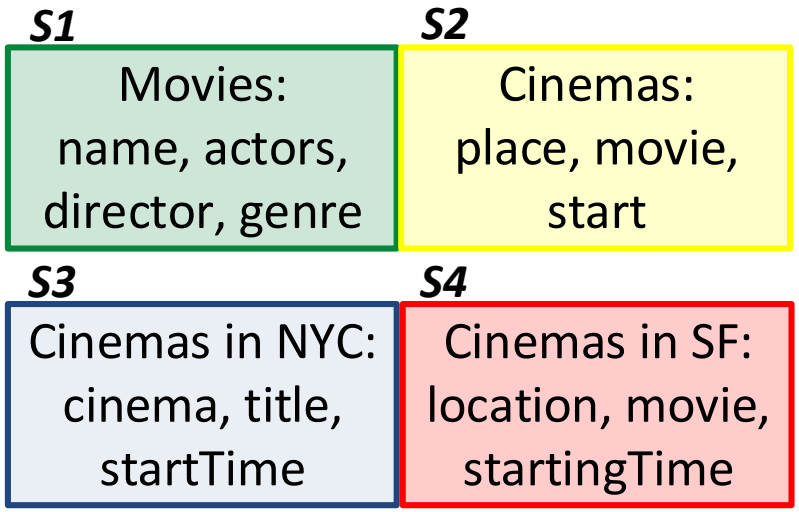
- Auswahl eines Algorithmus um *S1* und *S3* zu joinen (streaming Tuples von *S1* zu *S3* oder komplett *S1* vor *S3*)
- Festlegung wo die Selektion auf den Regisseur durchgeführt wird (in *S1* oder im Zielsystem)

Anfragebearbeitung am Beispiel




```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

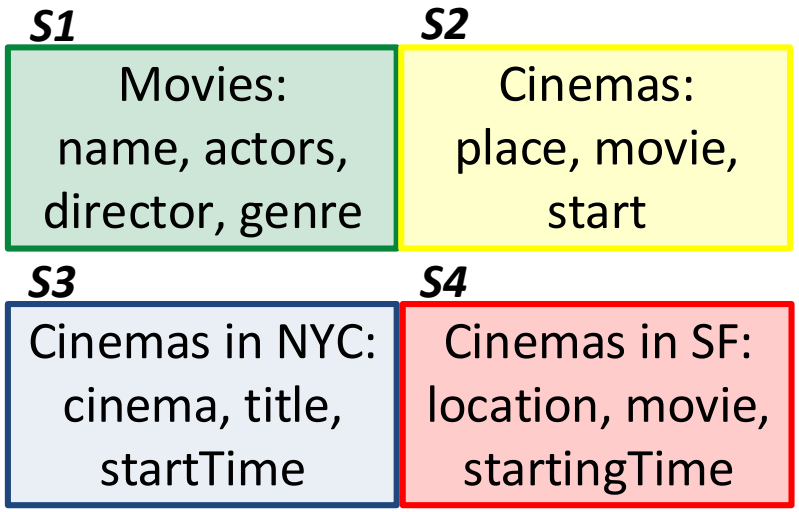
Anfragebearbeitung am Beispiel




```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrageausführung:

Anfragebearbeitung am Beispiel




```
SELECT  title, startTime
FROM    Movie, Plays
WHERE   Movie.title = Plays.movie
  AND   location = "New York"
  AND   director = "Woody Allen"
```

Anfrageausführung:

- Falls *S3* ausfällt oder zu langsam reagiert einen anderen Anfrageplan anfordern (in unserem Fall den mit *S1* und *S2*)

Agenda

Schema Matching, Mapping & Datenintegration

- Schema Matching:
  - Vergleich von Schema-Elementen
  - Zwischen zwei Quellen (Bottom-Up) oder zwischen Quelle und globalem Schema (Top-Down)

Schema Matching, Mapping & Datenintegration

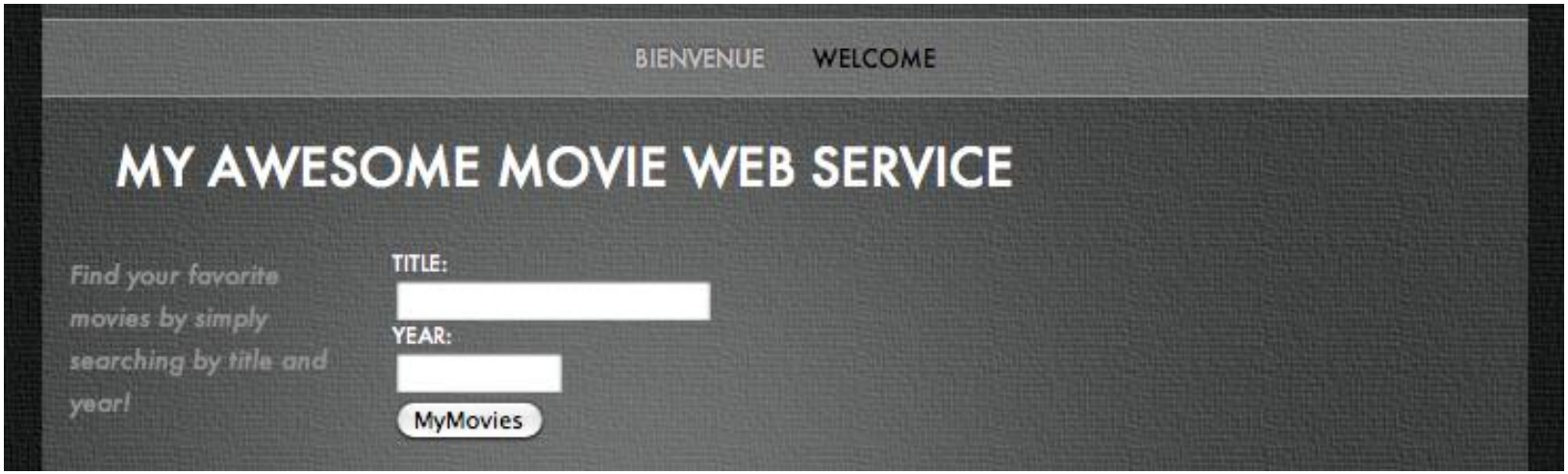
- Schema Matching:
  - Vergleich von Schema-Elementen
  - Zwischen zwei Quellen (Bottom-Up) oder zwischen Quelle und globalem Schema (Top-Down)
- Schema Mapping:
  - Ableiten einer Quellbeschreibung (virtuell) oder Transformationsanfrage (materialisiert) basierend auf den Matchingergebnissen

Schema Matching, Mapping & Datenintegration

- Schema Matching:
  - Vergleich von Schema-Elementen
  - Zwischen zwei Quellen (Bottom-Up) oder zwischen Quelle und globalem Schema (Top-Down)
- Schema Mapping:
  - Ableiten einer Quellbeschreibung (virtuell) oder Transformationsanfrage (materialisiert) basierend auf den Matchingergebnissen
- Datenintegration:
  - Zusammenführen der Ergebnisse der einzelnen quellenspezifischen Teil-/Transformationsanfragen
  - Erkennen von semantischen Redundanzen (Duplikaterkennung)
  - Zusammenführen von Duplikaten (Datenfusion)

Beispiel zu Schema Matching/Datenintegration

- Gegeben: Zwei Webservices *getMov* und *myMov*



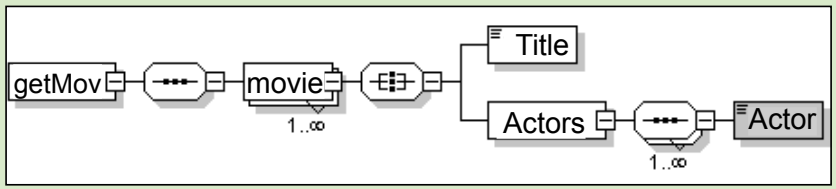
Beispiel: Quellen

Web Service getMov

Web Service getMov

```
<movie>
  <Title> Troy </Title>
  <Actors>
    <Actor> Eric Bana </Actor>
    <Actor> Brad Pitt </Actor>
  </Actors>
</movie>
```

- Operationen:
  - `getMovieByActor(firstName, lastName)`
  - `getMovieByTitle(title)`
- Ausgabestruktur:

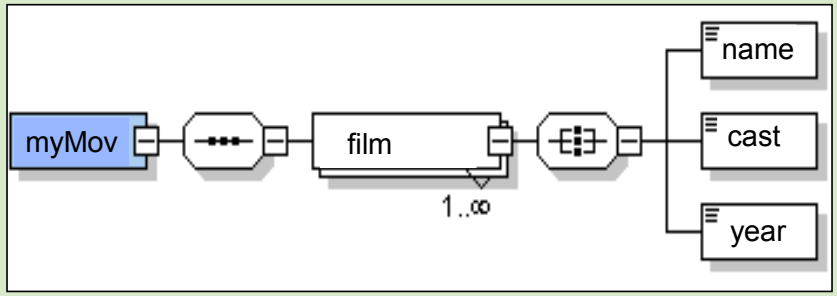


Web Service myMov

Web Service myMov

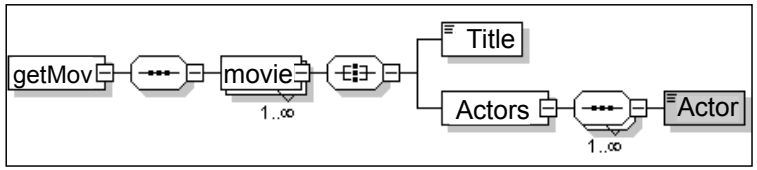
```
<film>
  <name> Troy </name>
  <cast> Pitt & Cox</cast>
  <year> 2003 </year>
</film>
```

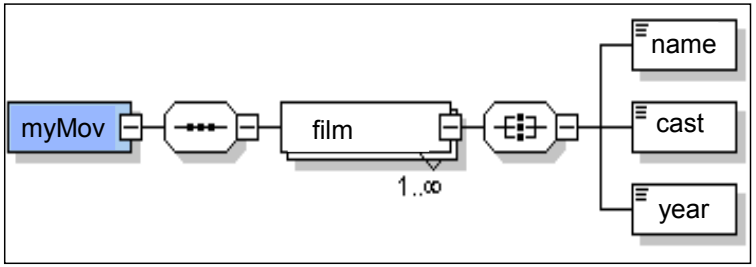
- Operation: `myMovies(Actor, Year)`
- Ausgabestruktur:



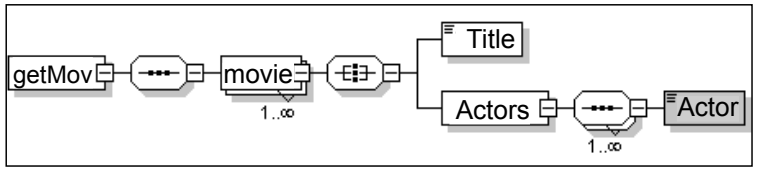
Quelle: Melanie Herschel, Universität Stuttgart

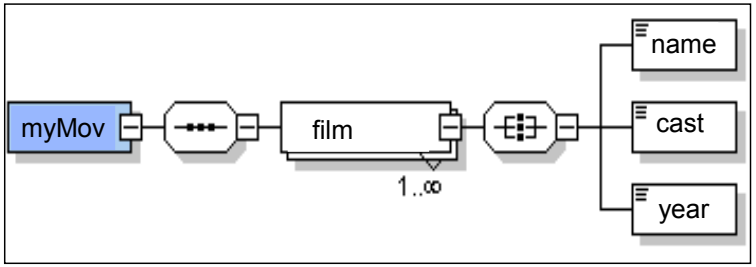
Beispiel: Schema Matching



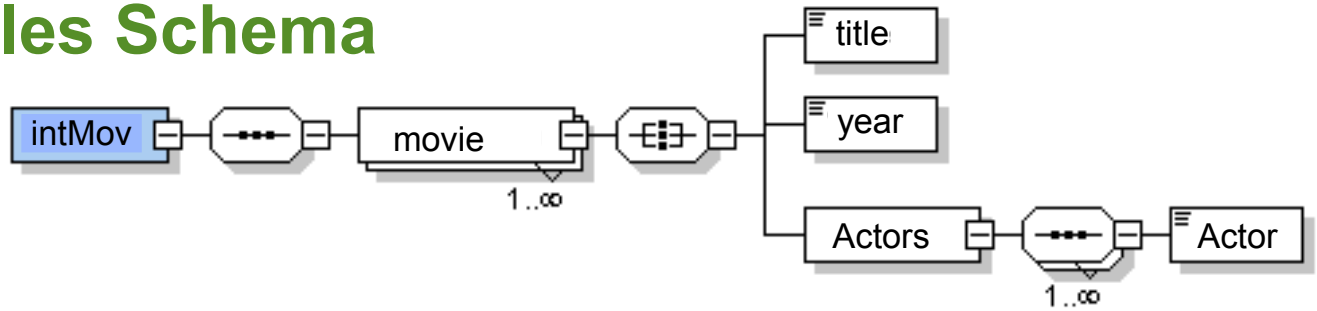


Beispiel: Schema Matching

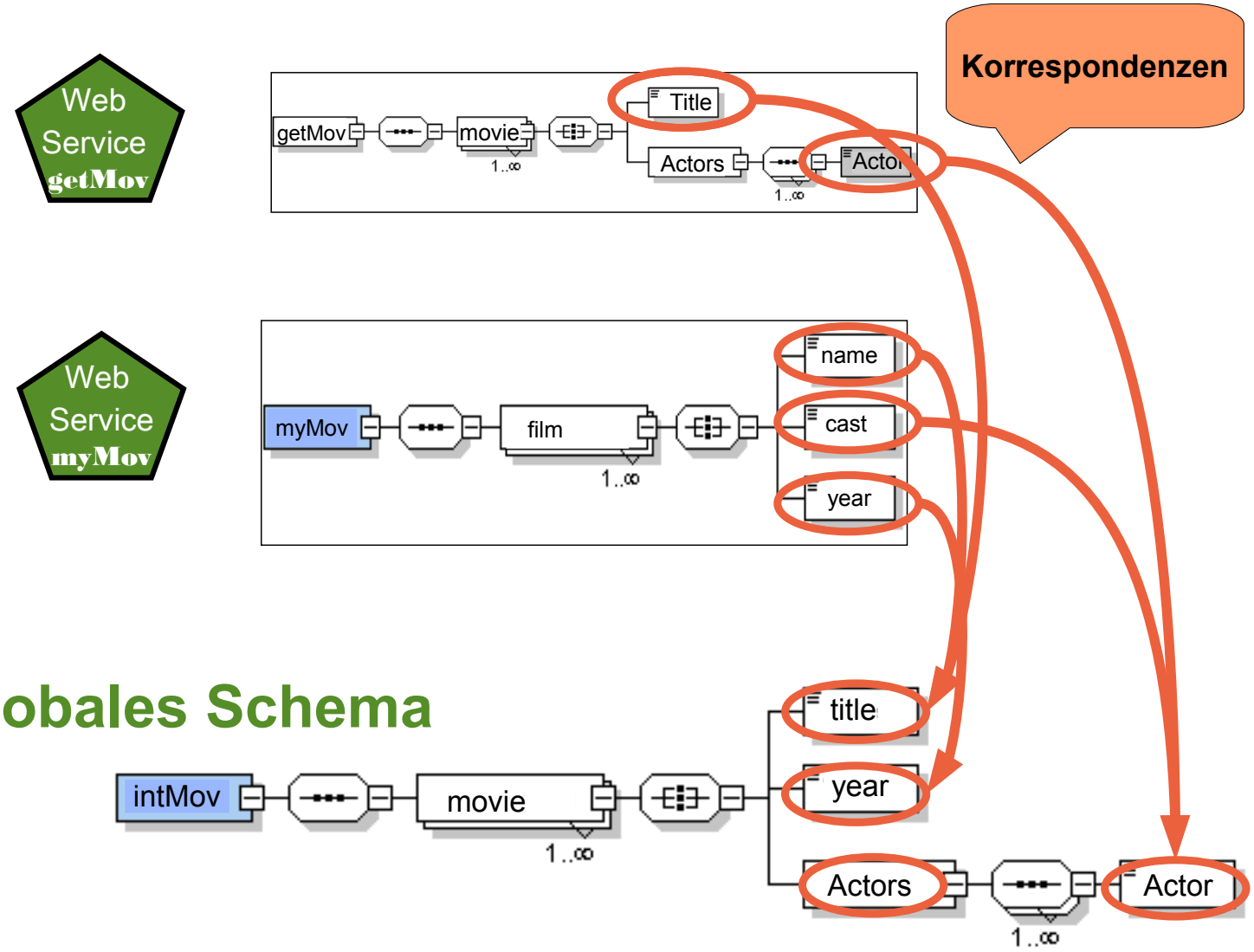




Globales Schema



Beispiel: Schema Matching



Beispiel: Duplikaterkennung

- Handelt es sich bei den Filmen, die von Web Services getMov und myMov zurückgegeben werden, um denselben Film?

Beispiel: Duplikaterkennung

- Handelt es sich bei den Filmen, die von Web Services getMov und myMov zurückgegeben werden, um denselben Film?
- Um dies festzustellen, müssen wir
  (1) semantische Äquivalenzen (Korrespondenzen) beider Strukturen ermitteln und
  (2) die Daten vergleichen.

Beispiel: Duplikaterkennung

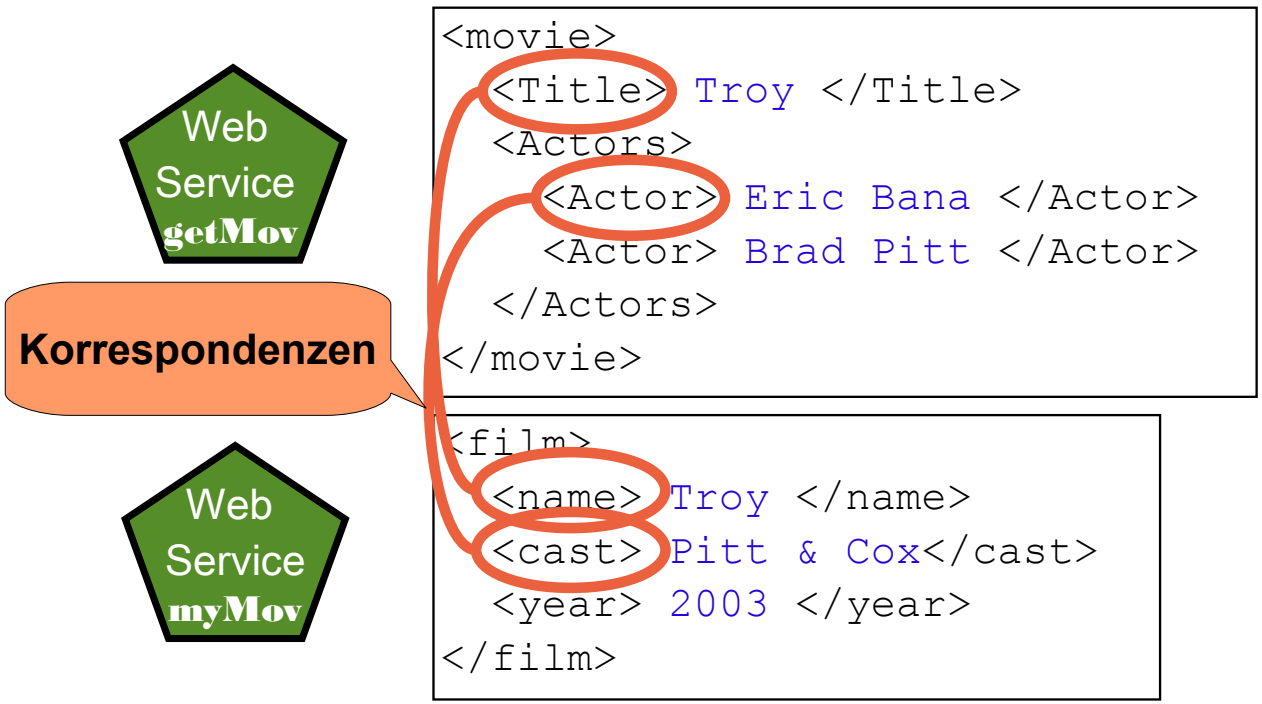
- Handelt es sich bei den Filmen, die von Web Services getMov und myMov zurückgegeben werden, um denselben Film?
- Um dies festzustellen, müssen wir
  (1) semantische Äquivalenzen (Korrespondenzen) beider Strukturen ermitteln und
  (2) die Daten vergleichen.


```
<movie>
  <Title> Troy </Title>
  <Actors>
    <Actor> Eric Bana </Actor>
    <Actor> Brad Pitt </Actor>
  </Actors>
</movie>
```


```
<film>
  <name> Troy </name>
  <cast> Pitt & Cox</cast>
  <year> 2003 </year>
</film>
```

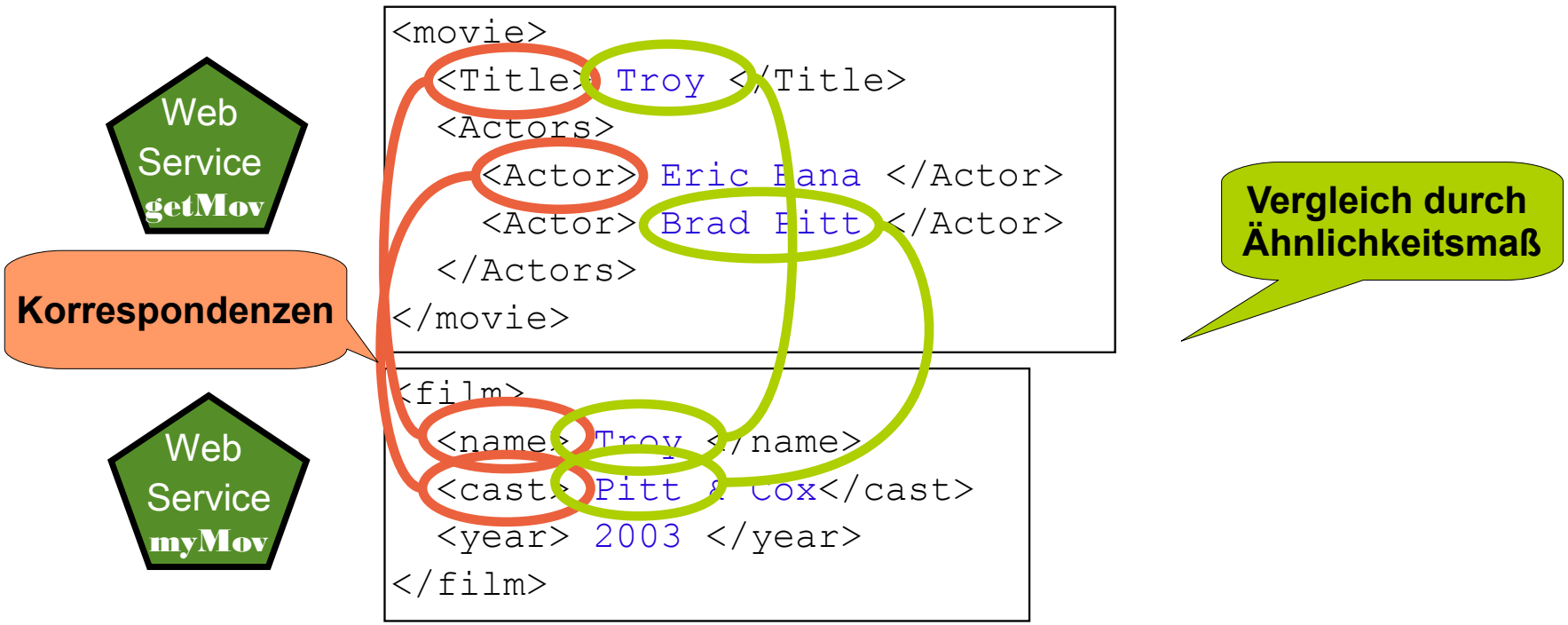
Beispiel: Duplikaterkennung

- Handelt es sich bei den Filmen, die von Web Services getMov und myMov zurückgegeben werden, um denselben Film?
- Um dies festzustellen, müssen wir
  (1) semantische Äquivalenzen (Korrespondenzen) beider Strukturen ermitteln und
  (2) die Daten vergleichen.



Beispiel: Duplikaterkennung

- Handelt es sich bei den Filmen, die von Web Services getMov und myMov zurückgegeben werden, um denselben Film?
- Um dies festzustellen, müssen wir
  (1) semantische Äquivalenzen (Korrespondenzen) beider Strukturen ermitteln und
  (2) die Daten vergleichen.

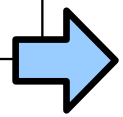


Beispiel: Datenfusion

- Titel stimmt überein ⇒ kein Konflikt
- `Eric Bana`, `Cox & 2003` nur in einer Quelle ⇒ Unsicherheit
- Widersprüchliche Daten ⇒ Konflikt


```
<movie>
  <Title> Troy </Title>
  <Actors>
    <Actor> Eric Bana </Actor>
    <Actor> Brad Pitt </Actor>
  </Actors>
</movie>
```


```
<film>
  <name> Troy </name>
  <cast> Pitt & Cox</cast>
  <year> 2003 </year>
</film>
```



```
<movie>
  <Title> Troy </Title>
  <Actors>
    <Actor> Bana </Actor>
    <Actor> Pitt </Actor>
    <Actor> Cox </Actor>
  </Actors>
  <year> 2003 </year>
</movie>
```

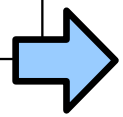
Integriertes Ergebnis

Beispiel: Datenfusion

- Titel stimmt überein ⇒ kein Konflikt
- Eric Bana, Cox & 2003 nur in einer Quelle ⇒ Unsicherheit
- Widersprüchliche Daten ⇒ Konflikt


```
<movie>
  <Title> Troy </Title>
  <Actors>
    <Actor> Eric Bana </Actor>
    <Actor> Brad Pitt </Actor>
  </Actors>
</movie>
```


```
<film>
  <name> Troy </name>
  <cast> Pitt & Cox</cast>
  <year> 2003 </year>
</film>
```

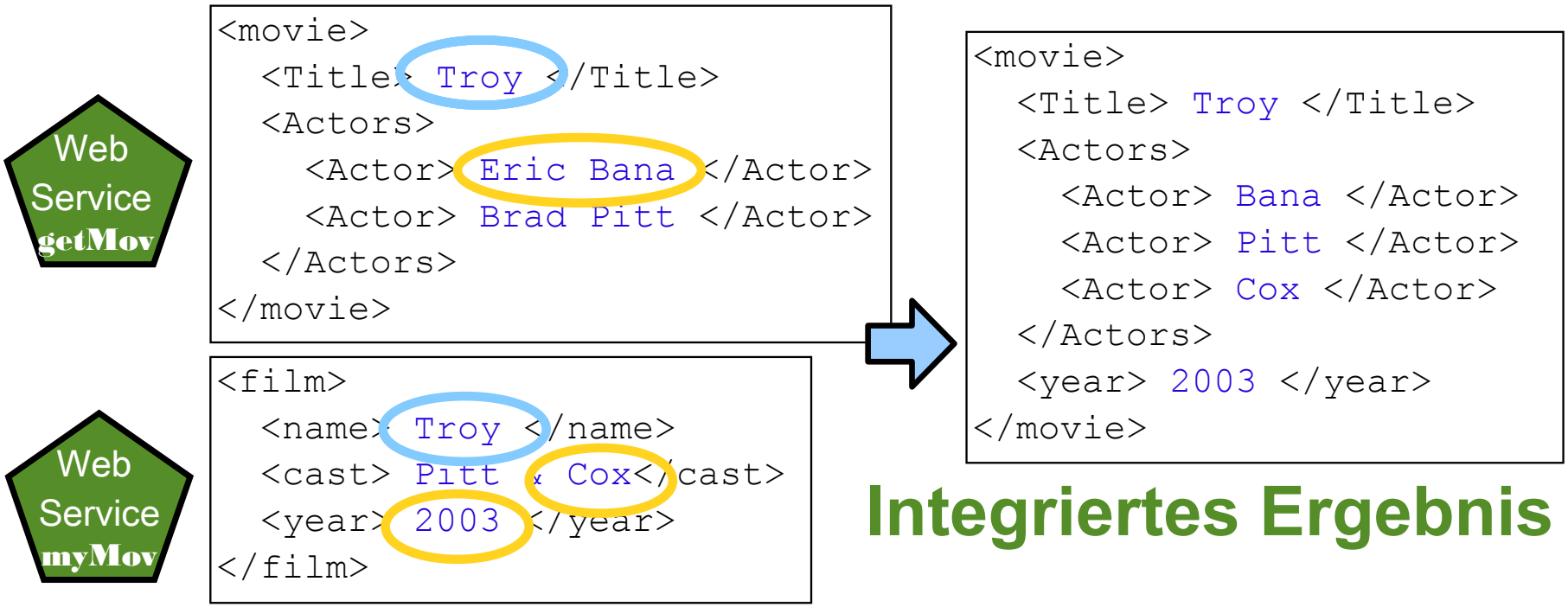


```
<movie>
  <Title> Troy </Title>
  <Actors>
    <Actor> Bana </Actor>
    <Actor> Pitt </Actor>
    <Actor> Cox </Actor>
  </Actors>
  <year> 2003 </year>
</movie>
```

Integriertes Ergebnis

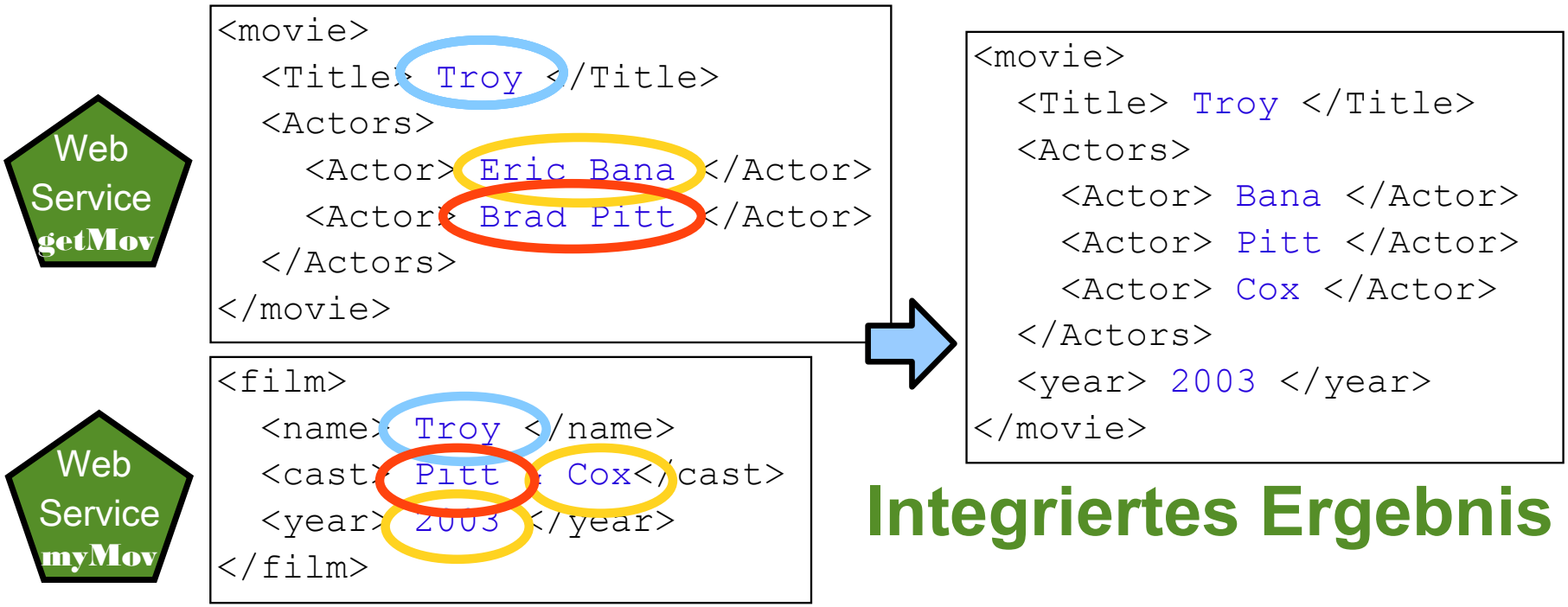
Beispiel: Datenfusion

- Titel stimmt überein ⇒ kein Konflikt
- Eric Bana, Cox & 2003 nur in einer Quelle ⇒ Unsicherheit
- Widersprüchliche Daten ⇒ Konflikt



Beispiel: Datenfusion

- Titel stimmt überein ⇒ kein Konflikt
- `Eric Bana`, `Cox` & 2003 nur in einer Quelle ⇒ Unsicherheit
- Widersprüchliche Daten ⇒ Konflikt



Literatur

[Chr12] Peter Christen.
Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection.
Springer, 2012.

[DHI12] Anhai Doan, Alon Halevy, and Zachary Ives.
Principles of Data Integration.
Morgan Kaufmann, 2012.

[LN06] Ulf Leser and Felix Naumann.
Informationsintegration.
dpunkt.verlag, 2006.
In German.

[NH10] Felix Naumann and Melanie Herschel.
An Introduction to Duplicate Detection.
Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2010.