



# Datenintegration & Datenherkunft

## Architekturen

---

Wintersemester 2010/11

Melanie Herschel  
[melanie.herschel@uni-tuebingen.de](mailto:melanie.herschel@uni-tuebingen.de)

Lehrstuhl für Datenbanksysteme, Universität Tübingen

1

## Kapitel 4

### Architekturen

---

- ➔ • Überblick der besprochenen Architekturen
- Föderierte DB Architekturen
  - 3 Schichten Architektur
  - 4 Schichten Architektur
  - Multidatenbankarchitektur
  - 5 Schichten Architektur
- Mediator-Wrapper Architektur
  - Konfigurationen
  - Mediatoren
  - Wrapper



# Betrachtete Datenbanksysteme

---

- **Monolithische Datenbanken**
  - Laufen auf einzelner Rechner
  - ANSI/SPARC 3-Schichten Architektur
- **Verteilte Datenbanken**
  - Verschiedene Rechner
  - Kaum Heterogenität
  - 4-Schichten Architektur
- **Multidatenbanken**
  - Mehrere heterogene Datenquellen
  - Integration auf Anfrageebene
  - Multidatenbankarchitektur (4 Schichten)

# Betrachtete Datenbanksysteme

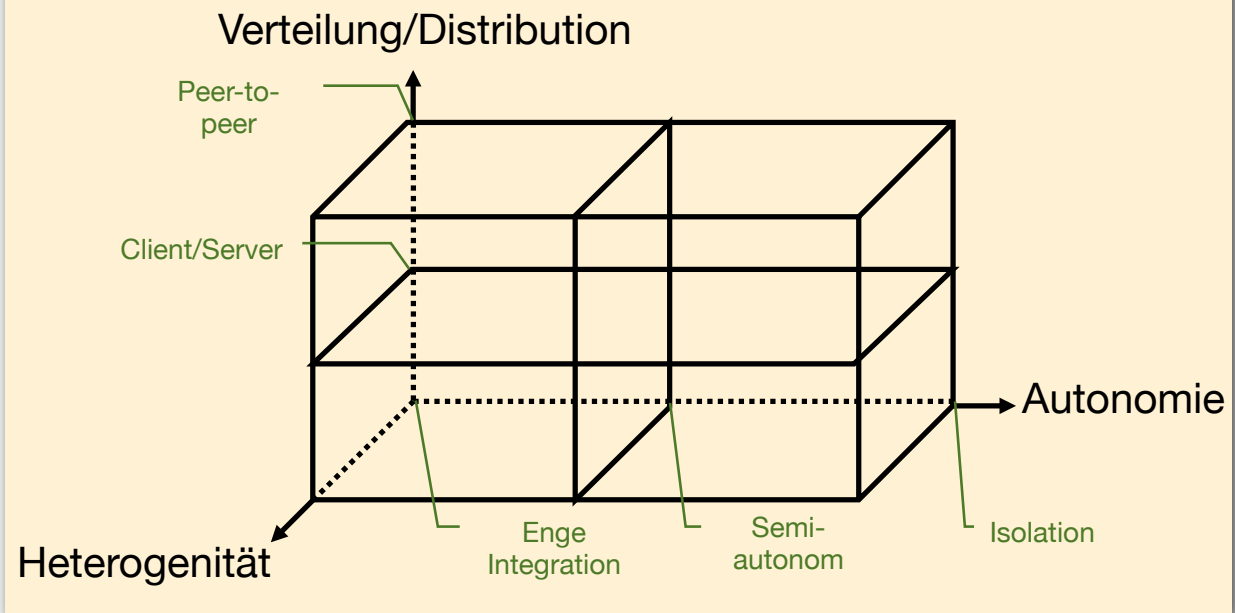
---

- **Föderierte Datenbanken**
  - Mehrere heterogene Datenquellen
  - Integration auf Schemaebene
  - 5-Schichten Architektur
- **Mediatorbasierte Datenbanken**
  - Verallgemeinerung der anderen Ansätze
  - Mediator-Wrapper Architektur
- **Peer-Daten-Management-Systeme**
  - Auflösung der Unterscheidung zwischen Datenquelle und integriertem System
  - PDMS Architektur

# Verteilung

Klassifikation verteilter DBMS nach [ÖV99]

## Platzierung der verschiedenen Datenbanksysteme auf Klassifikation



Datenintegration & Datenherkunft | WS2010/11 | Melanie Herschel | Universität Tübingen

5

## Kapitel 4

### Architekturen

- Überblick der besprochenen Architekturen
- ➔ Föderierte DB Architekturen
  - 3 Schichten Architektur
  - 4 Schichten Architektur
  - Multidatenbankarchitektur
  - 5 Schichten Architektur
- Mediator-Wrapper Architektur
  - Konfigurationen
  - Mediatoren
  - Wrapper



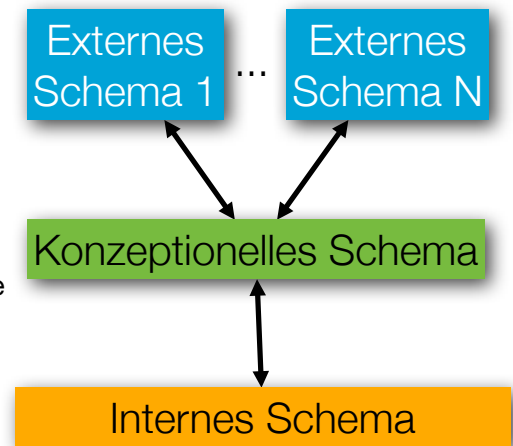
6

# 3-Schichten Architektur

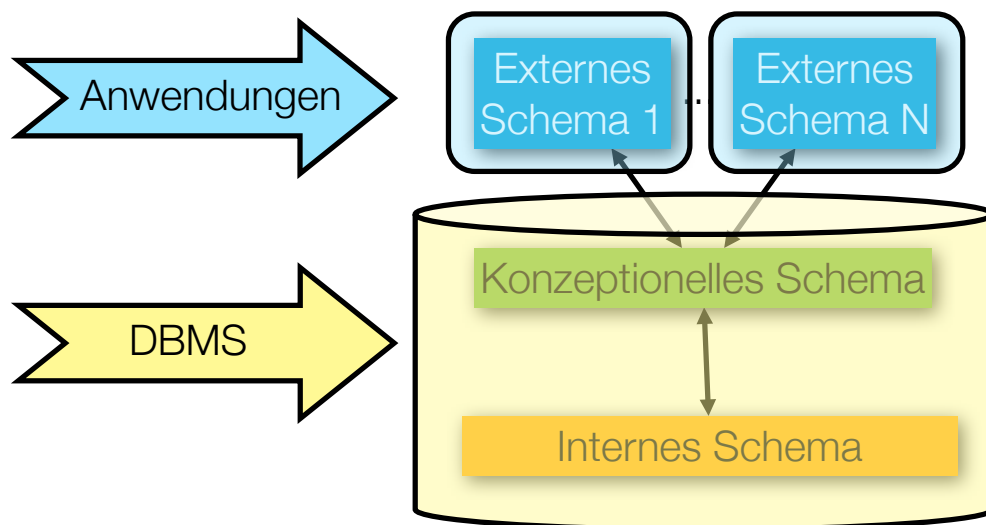
## ANSI/SPARC

### 3-Schichten Architektur für **zentralisierte** DBMS

- Interne (physische) Sicht
  - ▶ Speichermedium (Tape, Festplatte)
  - ▶ Speicherort (Zylinder, Block)
- Konzeptionelle (logische) Sicht
  - ▶ Unabhängig von physischer Sicht
  - ▶ Definiert durch Datenmodell
  - ▶ Stabiler Bezugspunkt für interne und externe Sichten
- Externe (logische) Sicht
  - ▶ Erlauben Anwendungsprogrammen den Zugriff
  - ▶ Nur auf die relevanten Daten
  - ▶ Enthält Aggregationen und Transformationen

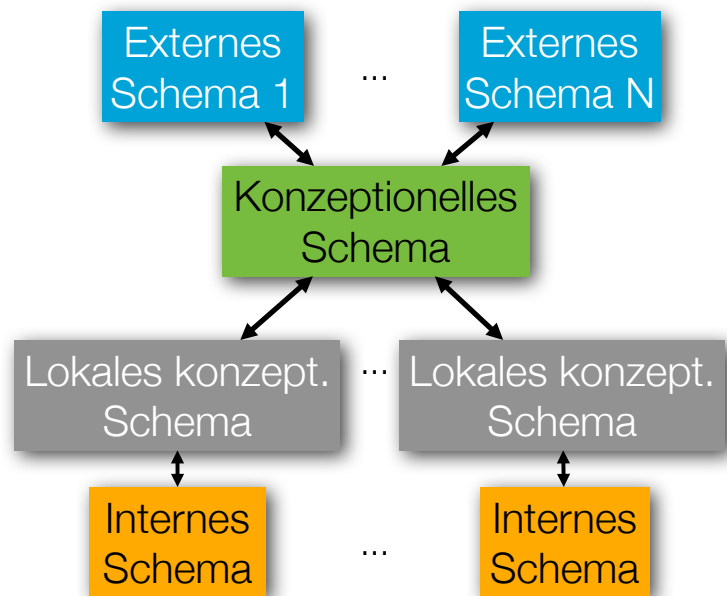


# 3-Schichten Architektur

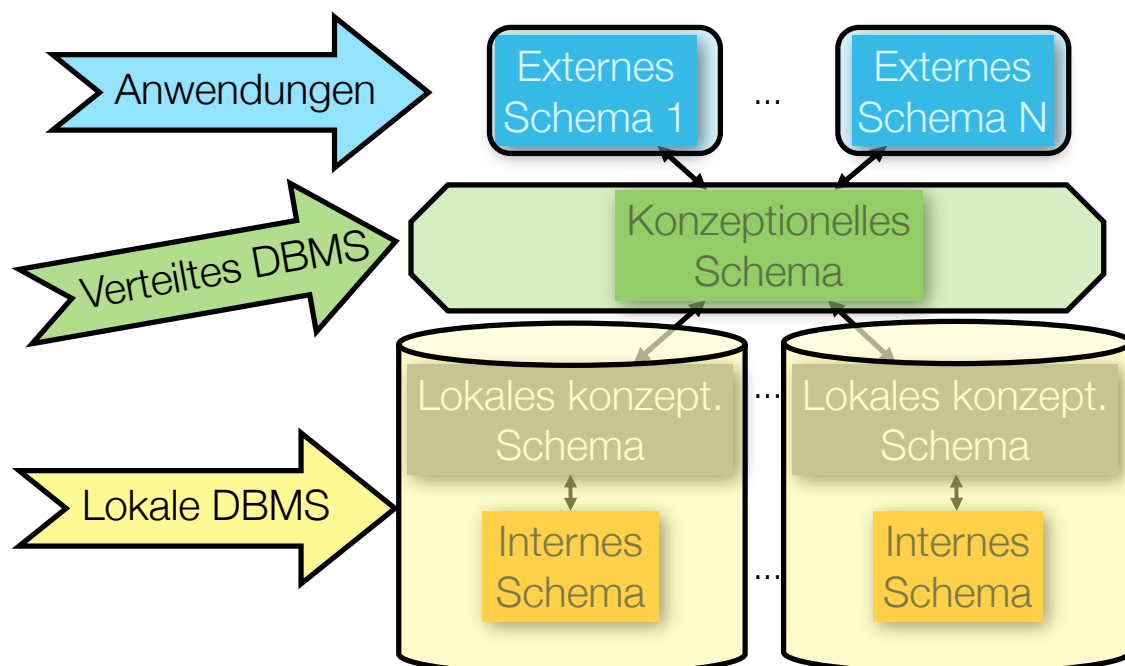


## 4-Schichten Architektur

- Für **verteilte** DBMS
- Neu: Trennung **lokales** vs. **globales konzeptionelles Schema**.
- Globales konzeptionelles Schema ist **integriert** aus den lokalen konzeptionellen Schemas.
- Lokales und globales konzept. Schema können gleich sein.

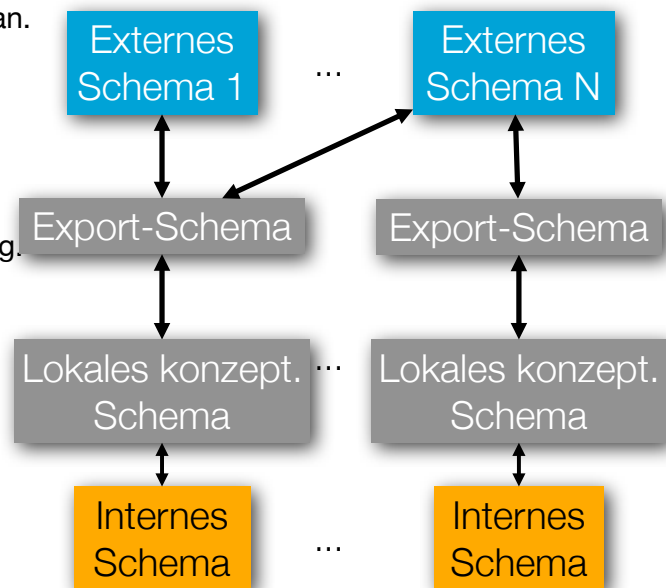


## 4-Schichten Architektur

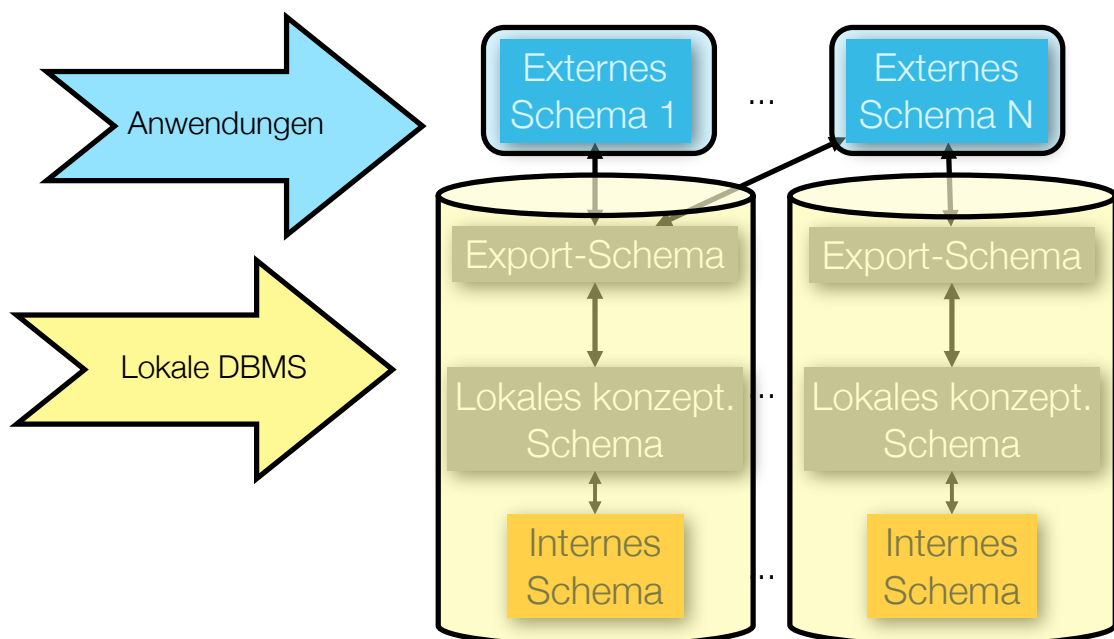


## Multidatenbankarchitektur (4 Schichten) [LMR90]

- Jede Quelle bietet ein Exportschema an.
  - Lokales und Export-Schema können gleich sein.
- Integration ist Teil externen Schemata.
- Multi-DBMS stellt den Anwendungen Multidatenbanksprache zur Verfügung.
- Externe Schemata können mit der Multidatenbanksprache definiert werden.
- Voraussetzung
  - Nutzer/Anwendungen kennen die jeweiligen Schemata
  - Multidatenbanksprache
- Lose Kopplung

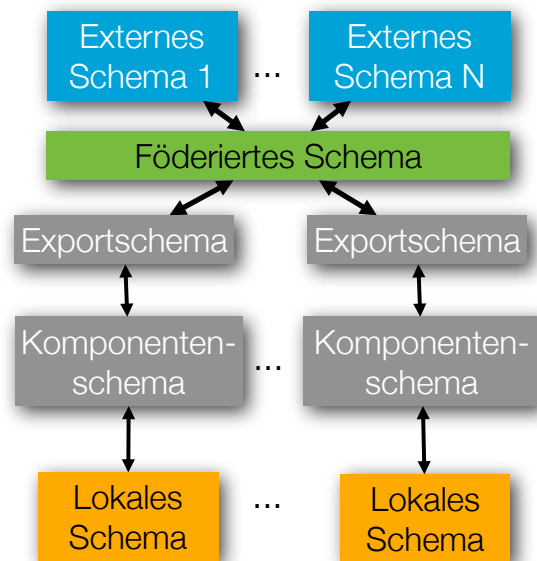


## 4-Schichten Architektur



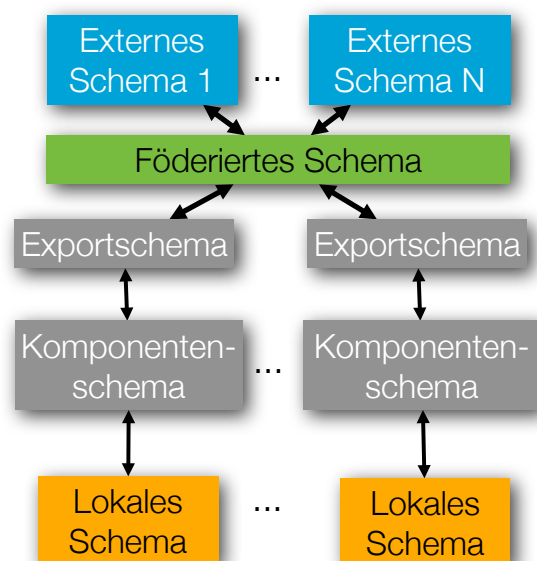
## 5-Schichten Architektur [SL90]

- Neu:
  - ▶ Interne Schemata werden nicht mehr betrachtet.
  - ▶ Exportschemas
  - ▶ Integriertes, föderiertes Schema
- Terminologie
  - ▶ Lokales Schema = lokales konzept. Schema
  - ▶ Föderiertes Schema = globales konzept. Schema



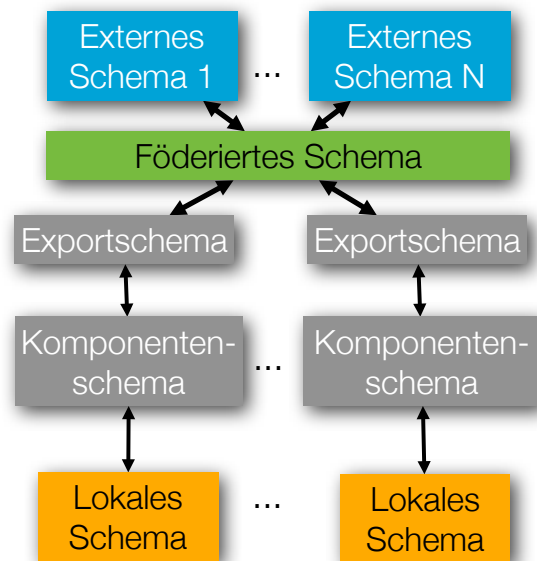
## 5-Schichten Architektur [SL90]

- Lokale Schemas
  - ▶ Konzeptionell
- Komponentenschemas
  - ▶ Kanonisches Datenmodell
  - ▶ Übergang durch Mappings.
  - ▶ Überwindet Datenmodellheterogenität.
- Exportschemas
  - ▶ Teilmenge des Komponentenschemas
  - ▶ Verwaltet Zugangsberechtigungen



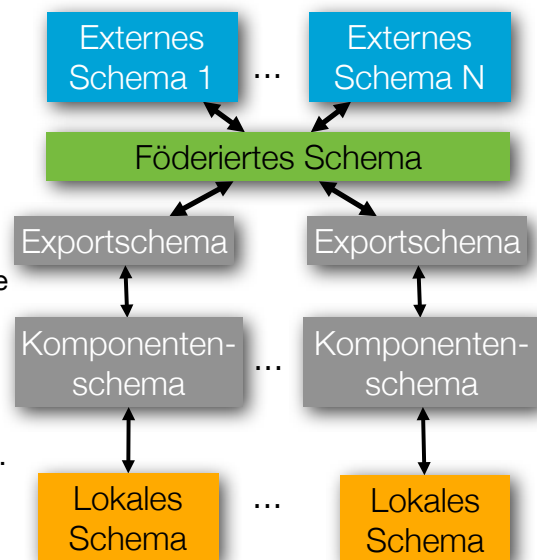
## 5-Schichten Architektur [SL90]

- **Föderiertes Schema**
  - ▶ Integriert aus den Exportschemas
  - ▶ Kennt Datenverteilung
  - ▶ Andere Namen:
    - Import Schema
    - Globales Schema
    - Enterprise Schema
    - Unified Schema
    - Mediator Schema
- **Externes Schema**
  - ▶ Föderiertes Schema kann sehr groß sein → Vereinfachung im Exportschema
  - ▶ „Schema Evolution“ leichter
  - ▶ Zusätzliche Integritätsbedingungen
  - ▶ Zugangskontrollen

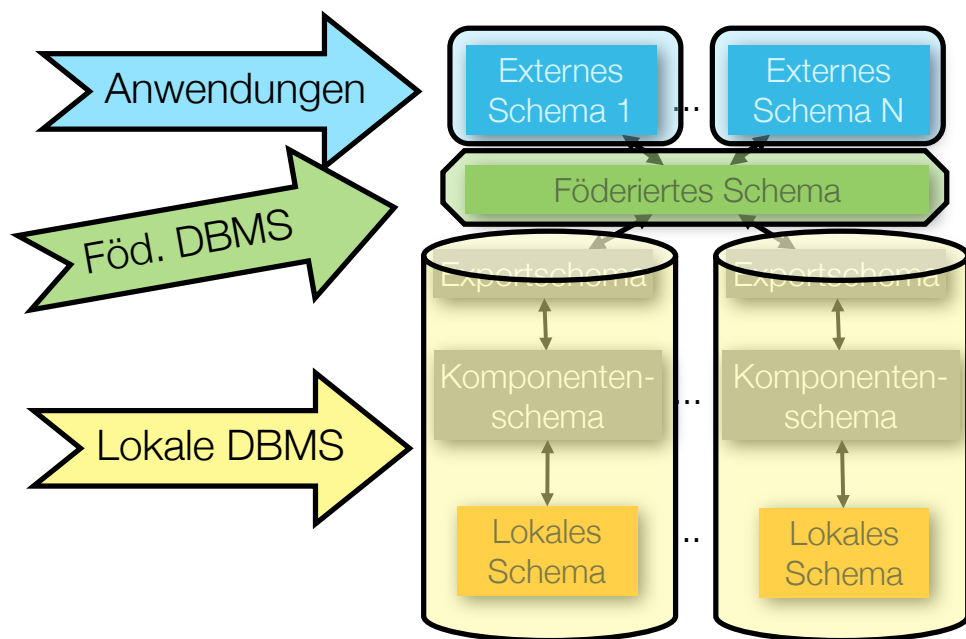


## 5-Schichten Architektur [SL90]

- **Mischformen**
  - ▶ Einige Schichten nicht immer nötig.
    - Z.B. wenn lokales und Komponentenschema gleich sind.
    - Z.B. wenn komplettes Komponentenschema exportiert werden soll.
  - ▶ Ein Komponentenschema kann mehrere Exportschemas haben.
  - ▶ Große FDBS können mehrere föderierte Schemas haben.
- **Föderation!**
  - ▶ Nur semi-autonom
  - ▶ Lokale DBMS müssen bereits kanonisches Datenmodell unterstützen.



## 5-Schichten Architektur [SL90]



## Vergleich der Architekturen [Con97]

Import / Export	4-Schichten	5-Schichten
Lose Kopplung	Lose Kopplung	Enge Kopplung
Integration durch Nutzer und globalen Admin	Integration durch Nutzer	Integration durch globalen Administrator
Zugriff über lokales System	Zugriff über globale Schnittstellen	Zugriff durch globales System
Keine globale DBMS Funktionalität	DBMS-Funktionalität nur durch Multidatenbanksprachen	DBMS-Funktionalität im globalen System

# Kapitel 4

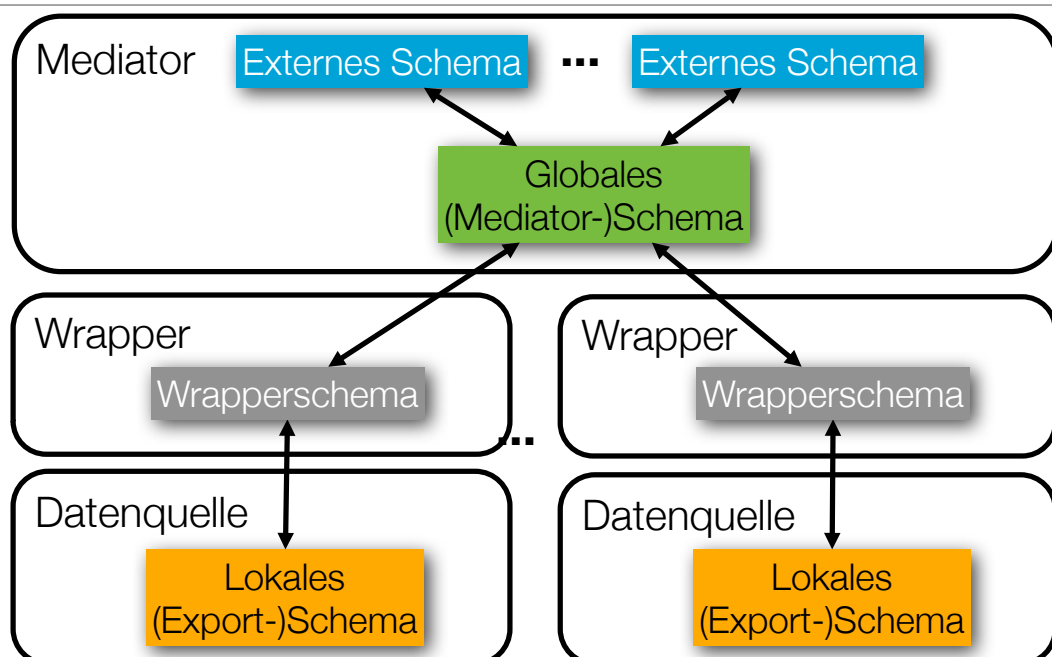
## Architekturen

- Überblick der besprochenen Architekturen
- Föderierte DB Architekturen
  - 3 Schichten Architektur
  - 4 Schichten Architektur
  - Multidatenbankarchitektur
  - ➔ • 5 Schichten Architektur
- Mediator-Wrapper Architektur
  - Konfigurationen
  - Mediatoren
  - Wrapper



19

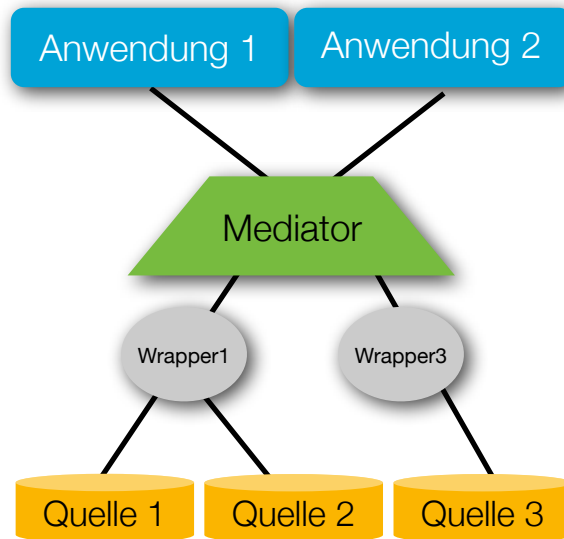
## Die Architektur



Jede Komponente kann (muss aber nicht) ein **autonomes System** sein. 20

# Mediator-Wrapper Architektur

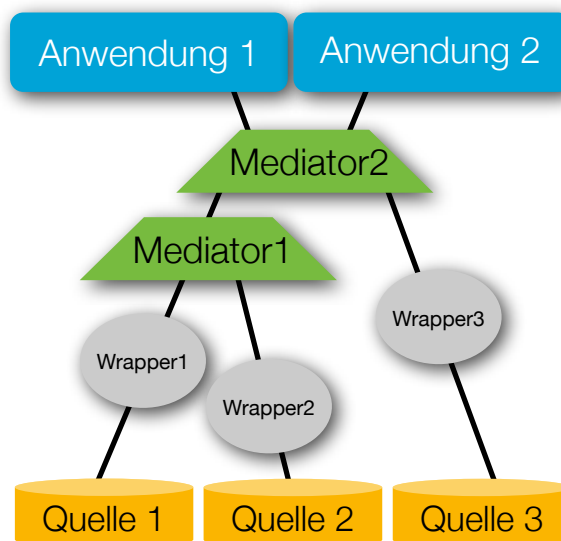
## Wrapper für mehrere Quellen



Quelle 1 und Quelle 2 unterscheiden sich nur leicht, z.B. zwei Oracle Datenbanken mit identischen Schemas.

# Mediator-Wrapper Architektur

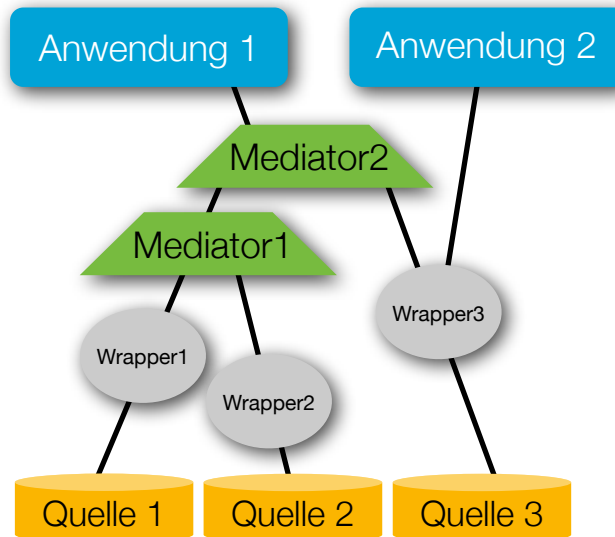
## Schachtelung von Mediatoren



Mediatoren dienen als Quellen für andere Mediatoren. Stufenweise Added-Value.

# Mediator-Wrapper Architektur

## Zugriff von Anwendungen auf Wrapper



Anwendungen können auch direkt mit Quellen kommunizieren.

etc....

## Mediatoren

### Definition

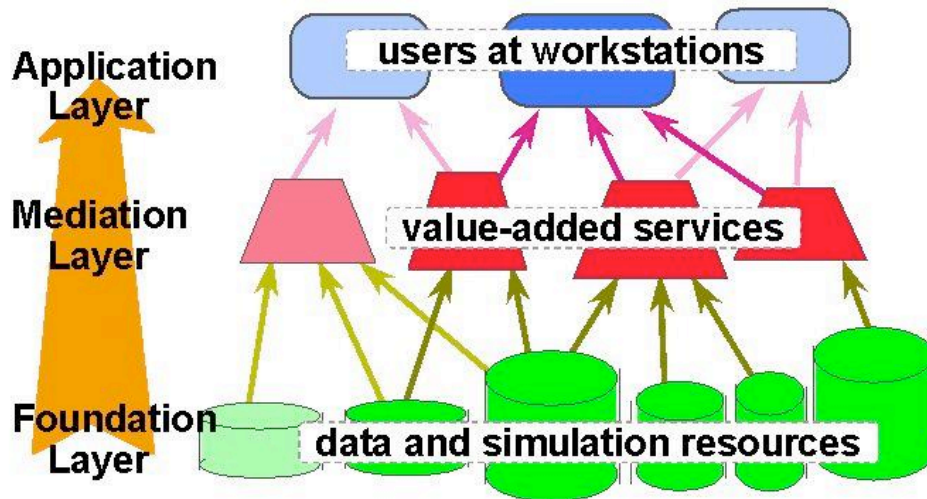


#### Mediator

- „A mediator is a software module that exploits encoded knowledge about certain sets or subsets of data to create information for a higher layer of applications“  
*Wiederhold '92 [Wie92]*
- Ein **Mediator** ist eine **Softwarekomponente**, die Wissen über bestimmte Daten benutzt, um **Informationen** für höherwertige Anwendungen zu **erzeugen**.

# Mediatoren

Daten werden zu Informationen



Gio Wiederhold 1999 15

# Mediatoren

Aufgaben

Zentrale Aufgabe eines Mediators ist, Daten einen Mehrwert zu geben. Dazu gehören Aufgaben wie

- Informationsintegration zur Ergänzung der Daten durch zusätzliche Quellen
- Suche und Auswahl relevanter Datenquellen
- Datentransformationen zur Konsistenzerhaltung
- Bereitstellung von Metadaten zur Weiterverarbeitung
- Integration von Daten mittels gemeinsamer Schlüssel
- Abstraktion bzw. Aggregation zum besseren Verständnis
- Ordnen von Ergebnissen in einem Ranking
- Filter zur Zugangskontrolle
- Semantisch sinnvolle Ausweitungen von Anfragen
- ...

# Mediatoren

## Expertenwissen

---

- Um die Aufgaben zu erfüllen wird in der Regel ein **Domänenexperte** benötigt.
- Das integrierte Schema und die Abbildung der Quellschemata auf das integrierte Schema stellen das Expertenwissen dar.
- Bei der Erweiterung von Anfragen muss ebenfalls auf Expertenwissen zurückgegriffen werden, das idealerweise in einem Thesaurus oder einer Ontologie abgelegt ist.

## Keep it Simple

---

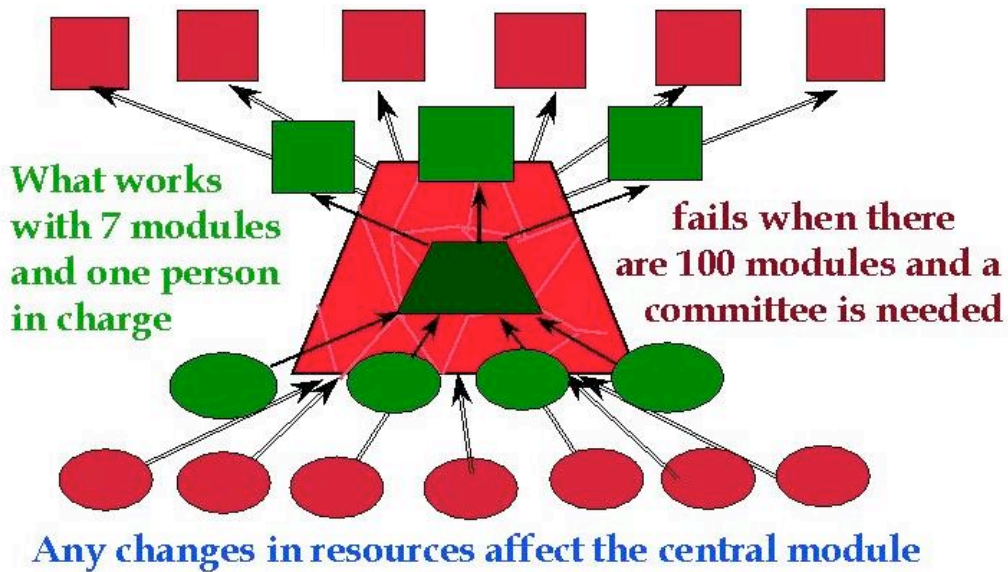
*„[A mediator] should be small and simple, so that it can be maintained by one expert or, at most, a small and coherent group of experts.“ Wiederhold '92*

Ein Mediator sollte **klein und einfach** genug sein, um durch einen einzigen oder höchstens eine kleine Gruppe von Experten gewartet werden zu können.

D.h.: **Einfaches** föderiertes **Schema**, **begrenzte Domäne**, einfache **Schnittstellen**

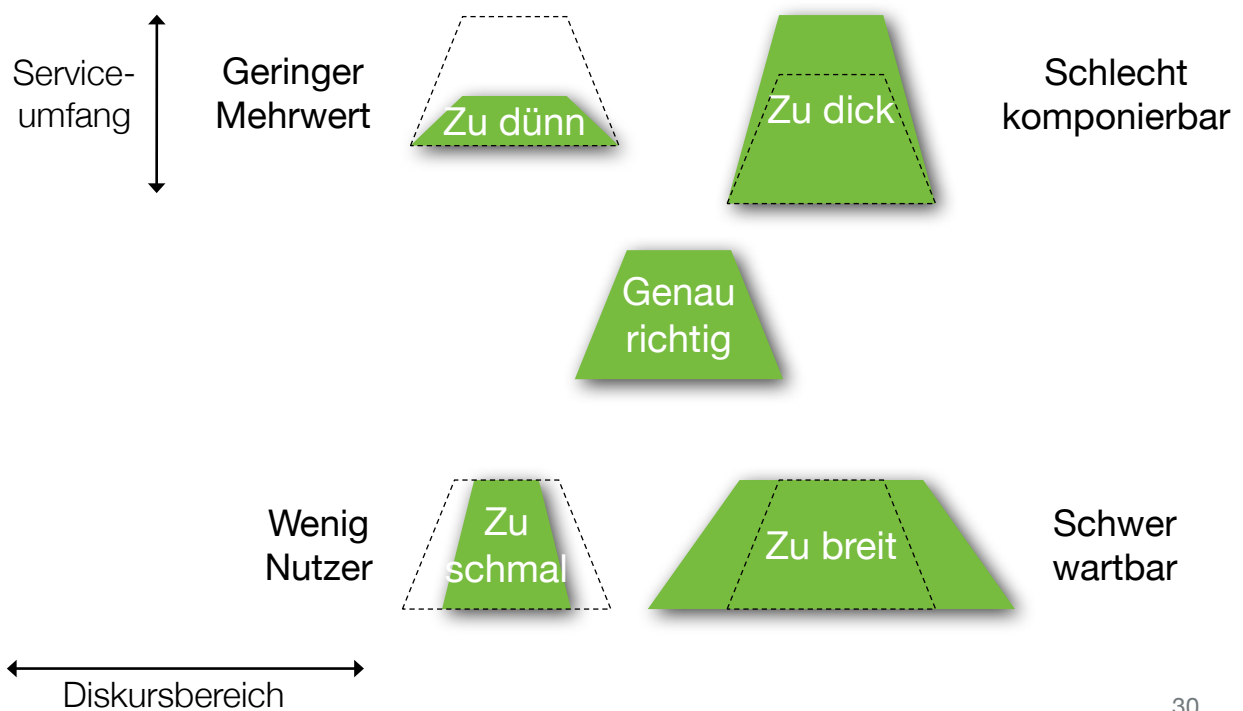
Erfahrung: Suchmaschinen ändern wöchentlich ihre Schnittstelle

# Keep it Simple



Gio Wiederhold 1999 39

# Dicke und Dünne Mediatoren



# Wrapper

## Definition

---

### Wrapper

- Wrapper sind **Softwarekomponenten**, die die **Kommunikation** und den Datenfluss zwischen **Mediatoren** und **Datenquellen** herstellen.
- Wrapper sind jeweils spezialisiert auf eine Ausprägung **autonomer, heterogener** Quellen.
- Wrapper vermitteln zwischen Mediator und Quelle.

# Wrapper

## Aufgaben

---

- Lösen Schnittstellenheterogenität
  - ▶ technisch
  - ▶ SQL, HTML Formulare, http, CORBA, ...
  - ▶ Mächtigkeit der Anfragesprache
- Lösen Datenmodellheterogenität
- Lösen schematische Heterogenität
  - ▶ Liefern globales Schema
- Reduzieren Anzahl der Datenmodelle (mit denen das IIS umgehen muss)
- Reduzieren Anzahl der Schemata
- Unterstützen globale Optimierung
  - ▶ Kostenmodell
  - ▶ Anfragefähigkeiten

# Wrapper

## Anforderungen

---

- Sollten schnell implementiert werden können
  - ▶ (< 1 Woche)
- Sollten wiederverwendbar sein
- Lokale Wartung (bei föderierten Systemen)
  
- An den Wrappern scheitern viele Projekte!
  - ▶ Deshalb Forschung zur schnellen oder sogar automatischen Wrappergenerierung.
  - ▶ Wrapperbibliotheken

# Garlic Wrapper Generierung

---

## Praktische Anforderungen aus [TS97]

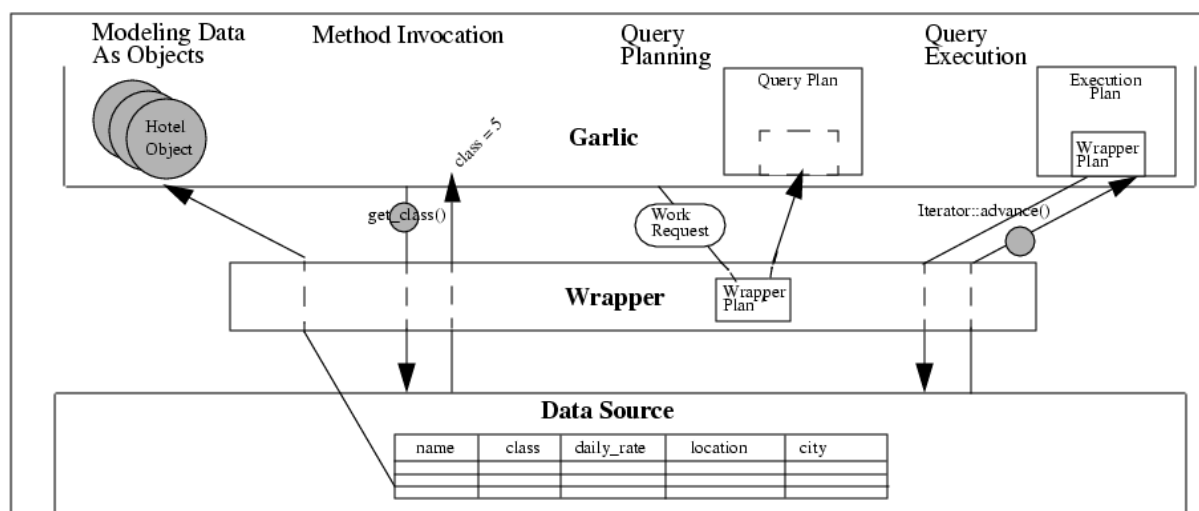
- Start-up Kosten gering (Stunden)
- Erweiterbarkeit
  - ▶ Einfacher Start
  - ▶ Später Fähigkeiten der Quellen hinzufügen
- Flexibilität
  - ▶ Möglichst breites Spektrum an Quellen abdecken
  - ▶ Neue Quellen stören Architektur nicht.
- Optimierung
  - ▶ Nicht durch Autoren sondern durch Garlic

# Garlic Wrapper Generierung

## Vier Grund-Services

1. Modellierung und Zugriff auf die Daten in der Quelle
2. Aufruf von Methoden in der Quelle
3. Mithilfe bei der Anfrageplanung
4. Anfrageausführung

# Garlic Wrapper Generierung



Quelle: [TS97]

# Garlic Wrapper Generierung

---

## Modellierung und Zugriff auf die Daten

- Garlic nutzt OO Modell
- Wrapper stellt Daten als Objekte mit Interface (globales Schema) und Implementierung (lokales Schema) dar.
- Stellt Identität von Objekten her.

# Garlic Wrapper Generierung

---

## Aufruf von Methoden in der Quelle

- Implizit immer: `Get_attr()` für jedes Attribut
- Implizit immer: `Set_attr()` für jedes nicht-read-only Attribut
- Um auch Quellen abzudecken, die nur über Methoden zu erreichen sind.
- Um besondere Fähigkeiten von Quellen auszuschöpfen.
- Beispiel: `display_Image(ImageID)`

# Garlic Wrapper Generierung

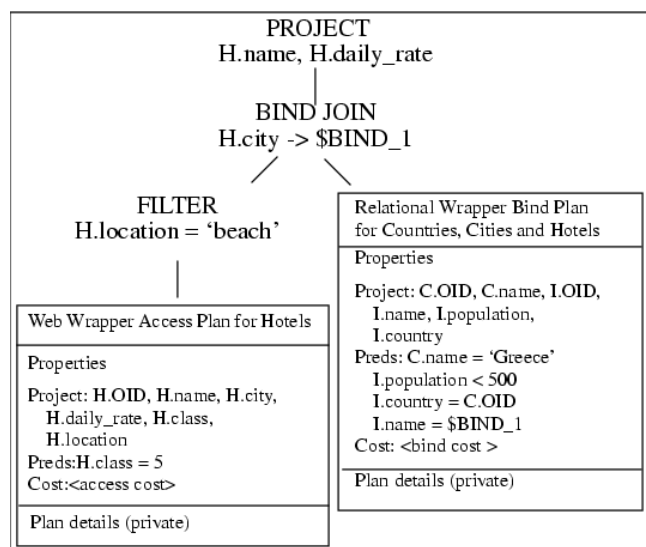
Mithilfe bei der Anfrageplanung (*query planning*)

- Garlics Anfrageplanung betrachtet alternative Pläne und sucht den besten heraus.
  - ▶ Kostenbasiert
- Mediator verschickt „Teilaufgaben“ an Wrapper.
  - ▶ Wrapper kann Teile davon ablehnen (je nach Fähigkeiten der Quelle).
  - ▶ Mediator gleicht aus.
    - Preprocessing
    - Postprocessing
- Wrapper liefert null oder mehr Teilpläne zurück.
- Teilpläne werden in Gesamtplan eingebaut.

# Garlic Wrapper Generierung

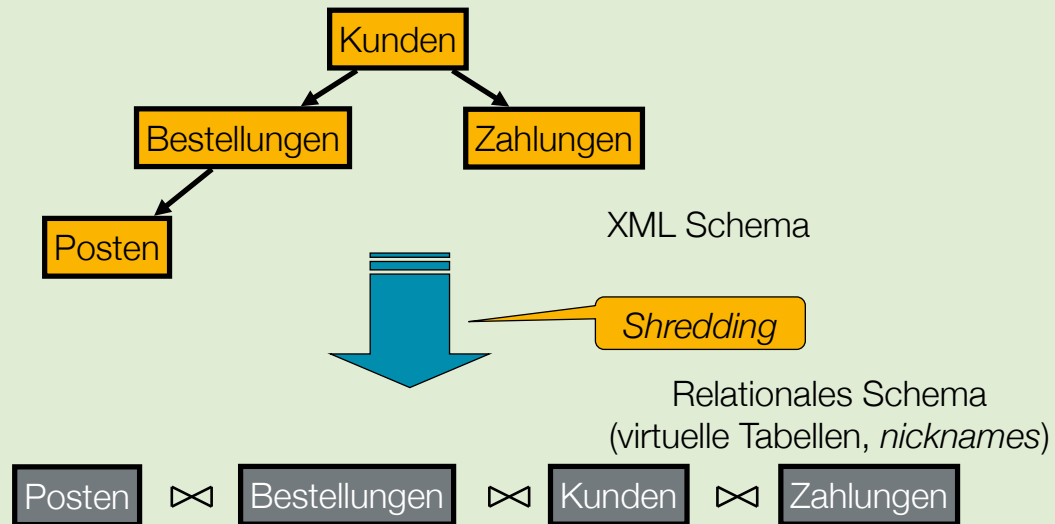
Anfrageausführung (*query execution*)

- Mediator führt einen Operatorbaum aus.
- Wrapper-Teilpläne sind Blätter in dem Baum.
- Pläne werden in Iteratoren umgewandelt.
- Pipelining



# Garlic Wrapper Generierung

## XML Wrapper für DB2



# Garlic Wrapper Generierung

## XML Wrapper für DB2

```
CREATE NICKNAME kunden_NN(  
name          VARCHAR(48) OPTIONS(XPATH './name/text()'),  
adresse       VARCHAR(48) OPTIONS(XPATH './address/text()'),  
kunden_NN_ID VARCHAR(48) OPTIONS(PRIMARY_KEY 'YES')  
FOR SERVER xml_server  
  OPTIONS(XPATH '//customer', FILE_PATH 'customers.xml');
```

```
CREATE NICKNAME order_NN(  
amount        DOUBLE   OPTIONS(XPATH './amount/text()'),  
date          VARCHAR(48) OPTIONS(XPATH './date/text()'),  
order_NN_ID   VARCHAR(48) OPTIONS(PRIMARY_KEY 'YES'),  
customer_NN_FID VARCHAR(48) OPTIONS(FOREIGN_KEY 'CUSTOMER_NN')  
FOR SERVER xml_server OPTIONS(XPATH './order');
```

```
CREATE NICKNAME item_NN(  
name VARCHAR(48) OPTIONS(XPATH './name/text()'),  
quant INTEGER   OPTIONS(XPATH './quant/text()'),  
order_NN_FID   VARCHAR(48) OPTIONS(FOREIGN_KEY 'ORDER_NN')  
FOR SERVER xml_server OPTIONS(XPATH './item');
```

```
CREATE NICKNAME payment_NN(  
amount        INTEGER   OPTIONS(XPATH './amount/text()'),  
date          VARCHAR(48) OPTIONS(XPATH './date/text()'),  
customer_NN_FID VARCHAR(48) OPTIONS(FOREIGN_KEY 'CUSTOMER_NN')  
FOR SERVER xml_server OPTIONS(XPATH './payment');
```

# Zusammenfassung

---

- Föderierte Datenbankarchitekturen
  - Import/Export Architektur
  - 4 Schichten Architektur
  - 5 Schichten Architektur
- Mediator-Wrapper Architektur
  - Mediator
  - Wrapper
  - Wrappergenerierung mit Garlic

