PinkDB & DataVault

Prinzipien hinter PinkDB und deren Anwendung auf eine DataVault Datenarchitektur

Einleitung

- SmartDB und PinkDB
- DataVault Datenarchitektur mit PinkDB
- Take-Away
  - Modellgetriebene Entwicklung
  - Nutze relationale Stärken
  - Stick to the pattern

Me

Torsten Glunde

Solution Architect DWH and BI

- http://alligator-company.com seit 2002
- Consultant seit 1994
- BI/DWH seit 2002
- DataVault Evangelist seit 2009

https://www.xing.com/profile/Torsten_Glunde/cv

@tglunde

www.linkedin.com/in/torsten-glunde-097aba8

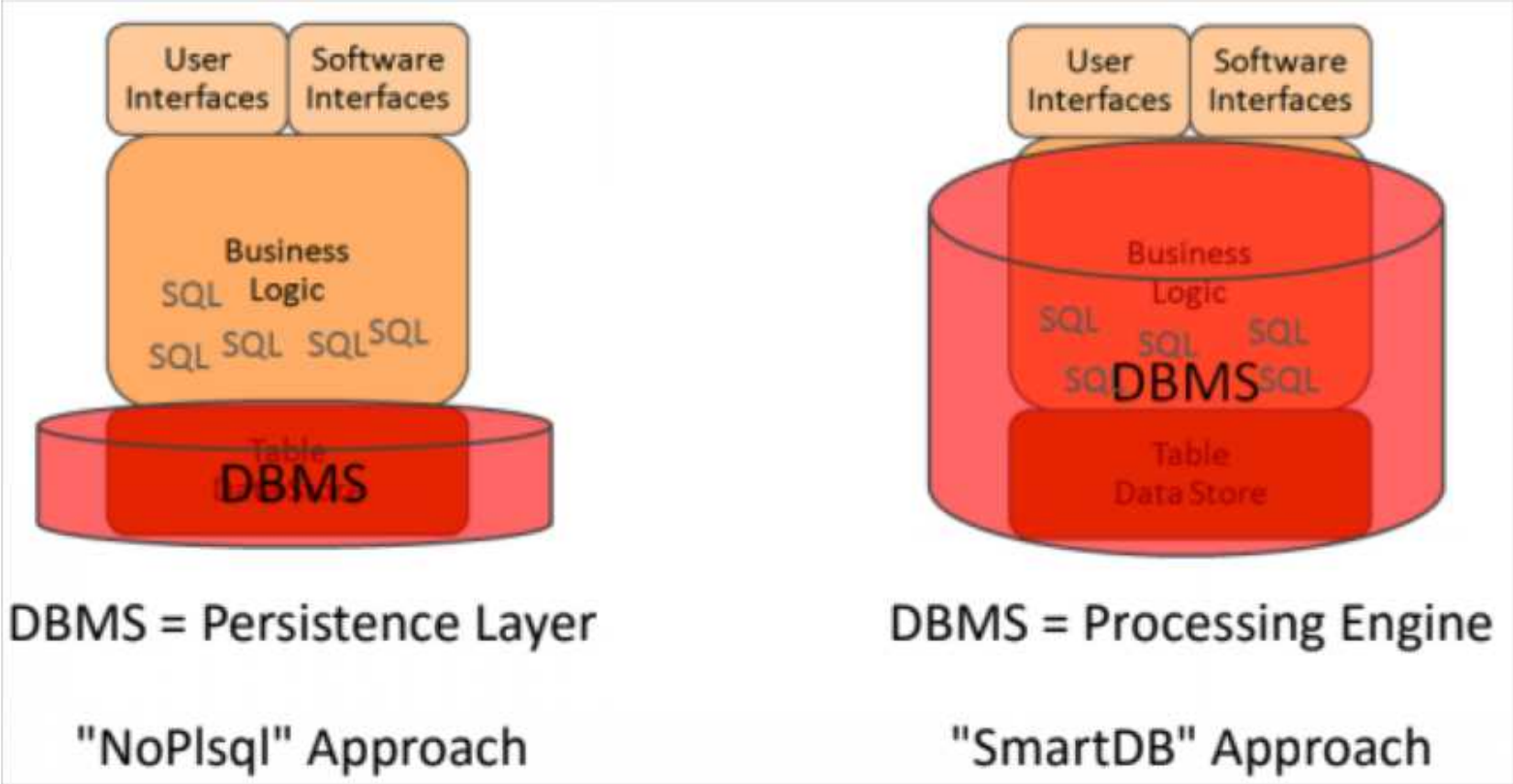
http://www.oraclethoughts.com/

https://www.salvis.com/blog/

https://danischnider.wordpress.com/

https://stevenfeuersteinonplsql.blogspot.com/2018/05/the-smartdb-resource-center.html

SmartDB – OLTP Applikationen



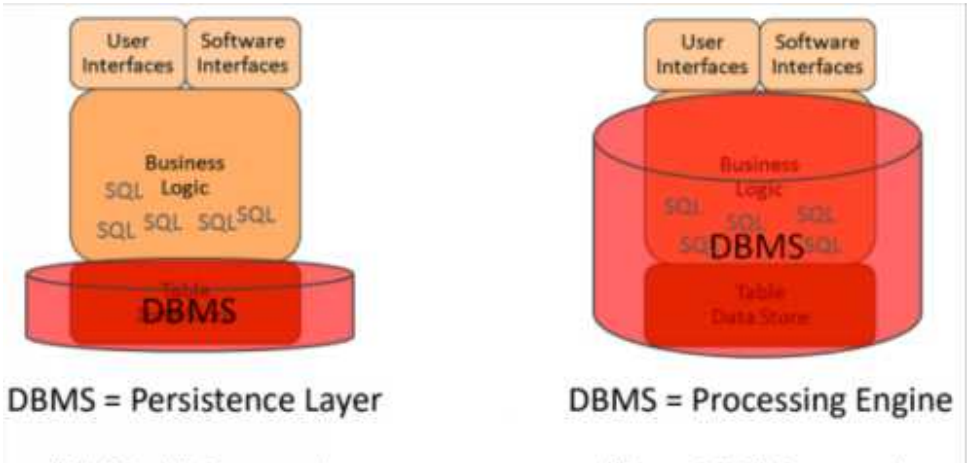
Quelle vom Oktober 2016 – abgerufen am 23.10.2018: https://www.youtube.com/watch?v=8jiJDflpw4Y

Probleme mit Middleware außerhalb der DB

- Stabilität des Technologie Stacks
  C++, C#, Java, JavaScript
  - SQL ist tabiler

- Entwicklungs- und Wartungskosten
  Fehler finden in mehreren Ebenen
  Benötigen wir wirklich OO?
  Wartung mehrerer Systeme

- Performance und Skalierbarkeit
  Experiment im Labor:



	Java/JDBC	PL/SQL row-by-row	PL/SQL set-based
DB-CPU	437	204	7
APP-CPU	217	0 (n/a)	0 (n/a)
Total	654	204	7

Almost 100X
Just think about this…

Quelle vom Oktober 2016 – abgerufen am 23.10.2018: https://www.youtube.com/watch?v=8jiJDfIpw4Y

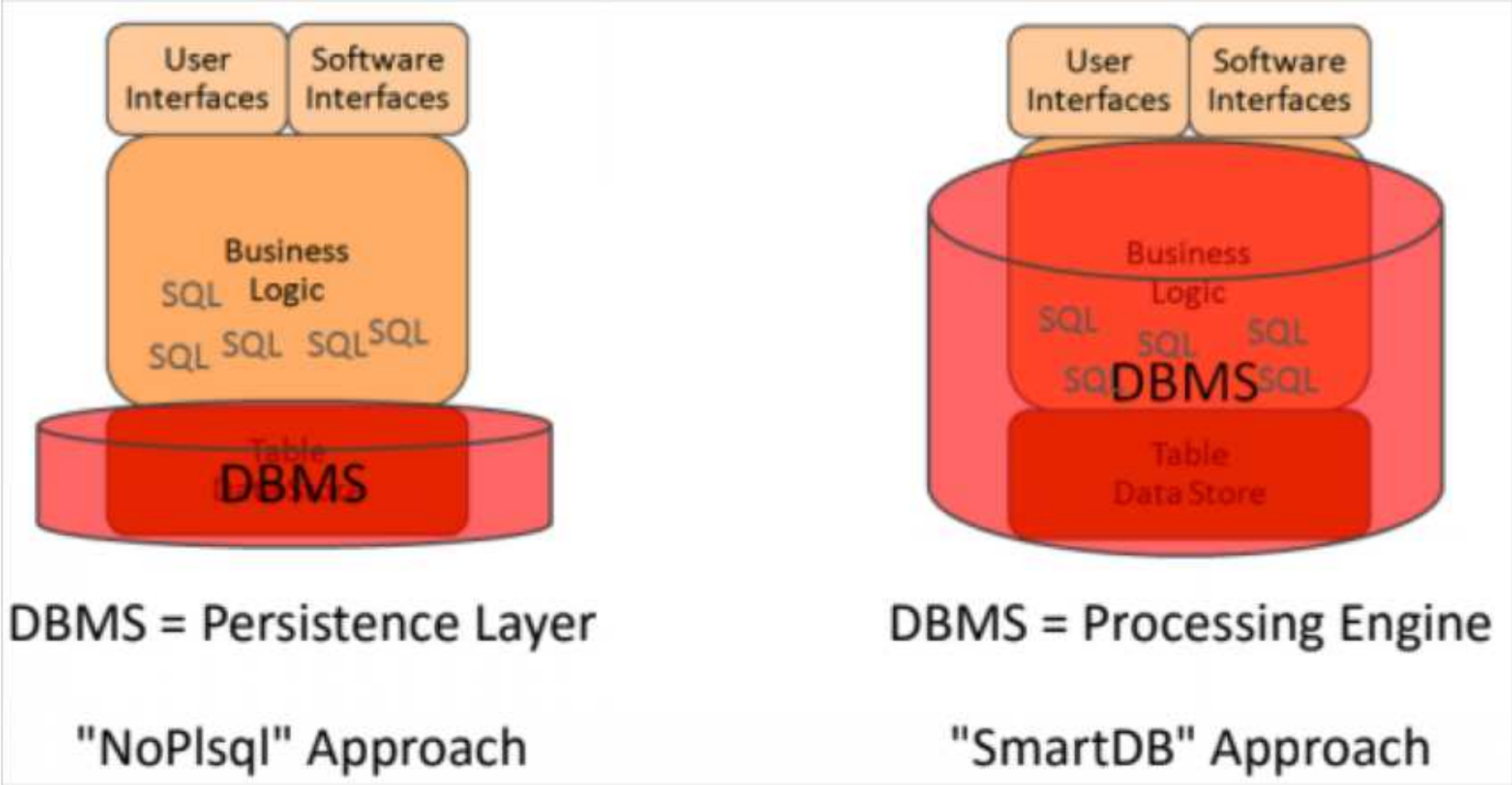
SmartDB

- PL/SQL „hard shell“ API
- API hangeschrieben
- Connect User besitzt keine Datenbankobjekte
- Connect User can execute PL/SQL API units only
- Transaktionskontrolle in der DB
- Set-based SQL

```
begin Pkg.Bulk_Insert(:Rows); end;
```

Quelle vom Oktober 2016 – abgerufen am 23.10.2018: https://www.youtube.com/watch?v=8jiJDflpw4Y

Und Data Warehousing?

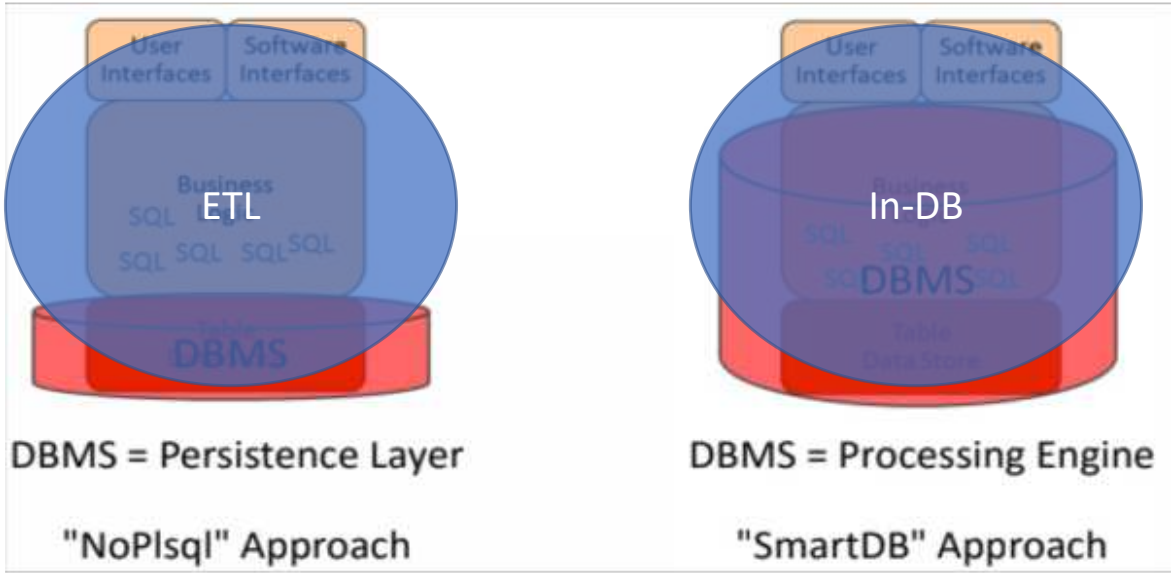


Quelle vom Oktober 2016 – abgerufen am 23.10.2018: https://www.youtube.com/watch?v=8jiJDflpw4Y

Und Data Warehousing?

ETL oder in-DB Verarbeitung

- Vorteile der DB ausnutzen
- ETL/Middleware fällt weg
- Wartung und Betrieb vereinfachen
- Set-based anstatt Row-By-Row

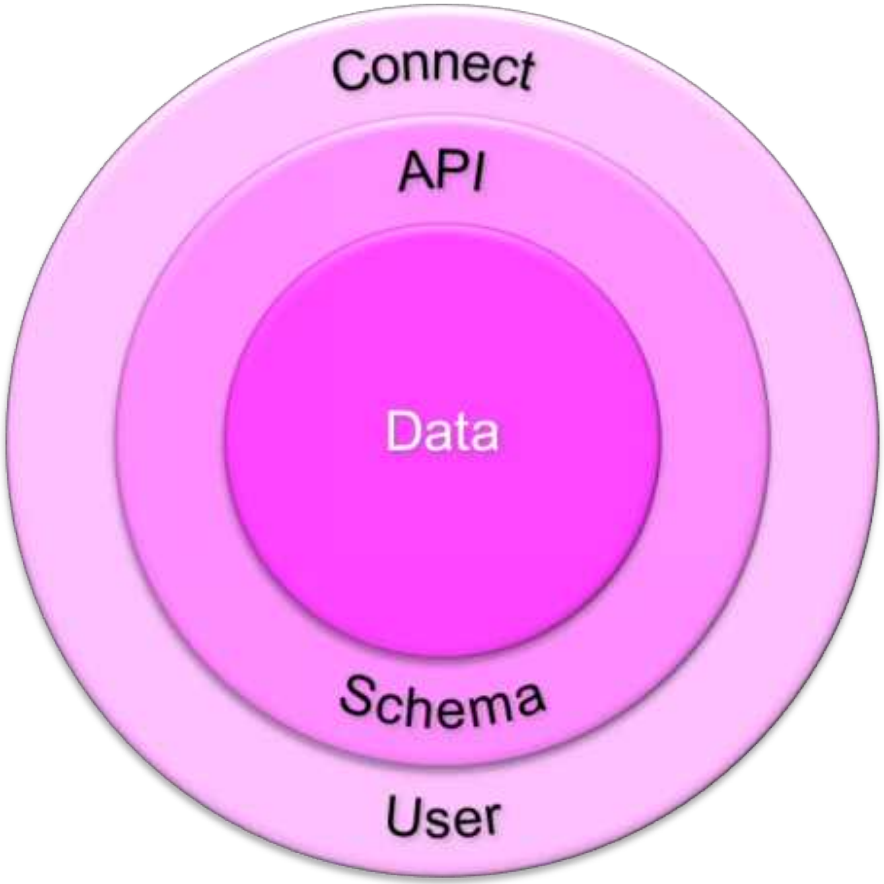


Quelle vom Oktober 2016 – abgerufen am 23.10.2018: https://www.youtube.com/watch?v=8jiJDflpw4Y

PinkDB Paradigm

Prinzipien

- Set-based
- View-API
- Fast

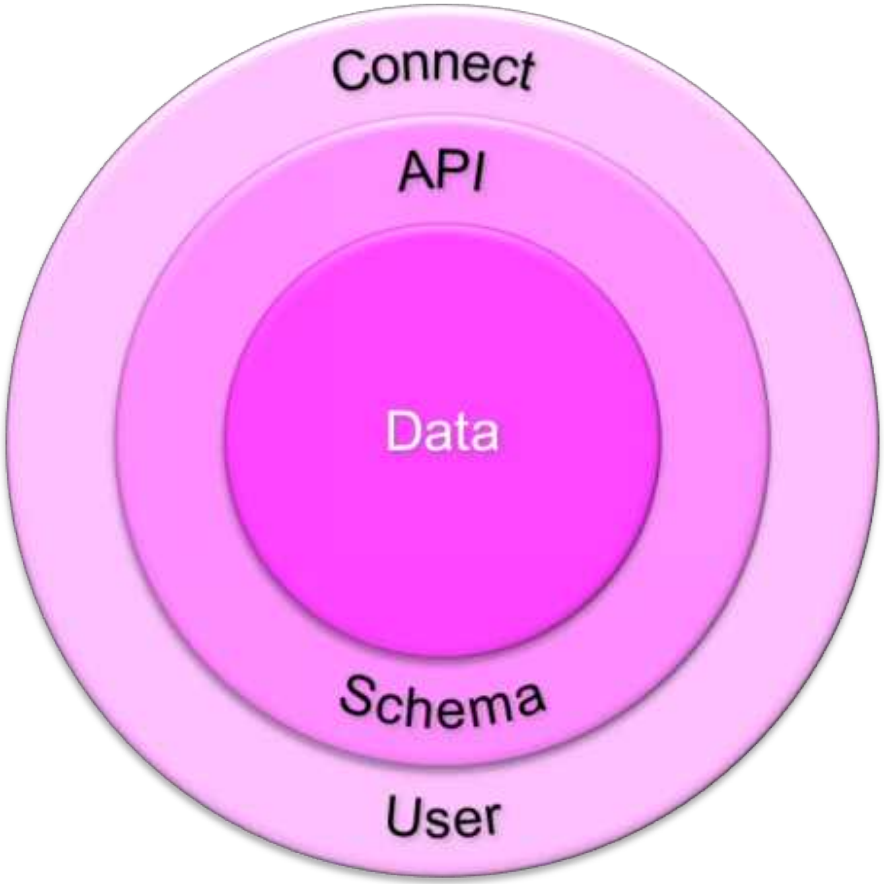


Quelle abgerufen am 23.10.2018: https://www.salvis.com/blog/2018/07/18/the-pink-database-paradigm-pinkdb/

PinkDB Paradigm

Struktur

- Connect User besitzt keine Datenbankobjekte
- API Schema: Datenzugriff, gespeicherte Objekte und Views, **keine Tabellen**
- Data – speichert Informationen in vom Datenmodell definierten Struktur

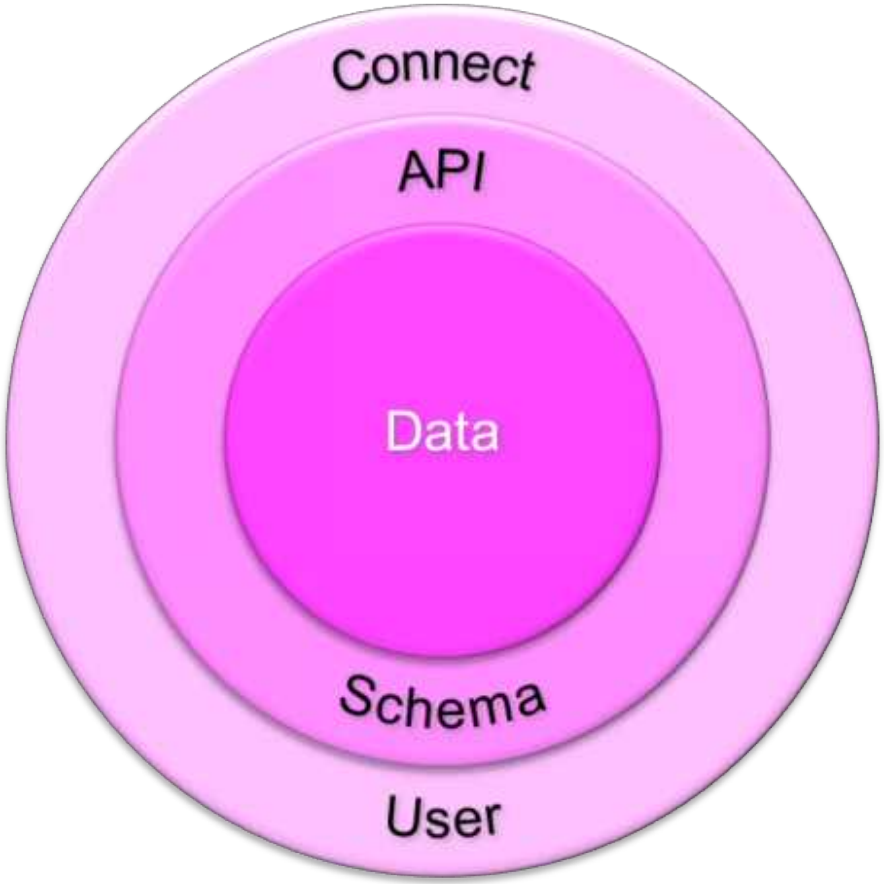


Quelle abgerufen am 23.10.2018: https://www.salvis.com/blog/2018/07/18/the-pink-database-paradigm-pinkdb/

PinkDB Paradigm

Features

- Set-based
  - Gute Performance
  - Ausnahme: Aktualisierung wenigen Sätzen von UI
- Datenbank unabhängig
- Schnell
- Ausnahmen von der Regel müssen dokumentiert werden

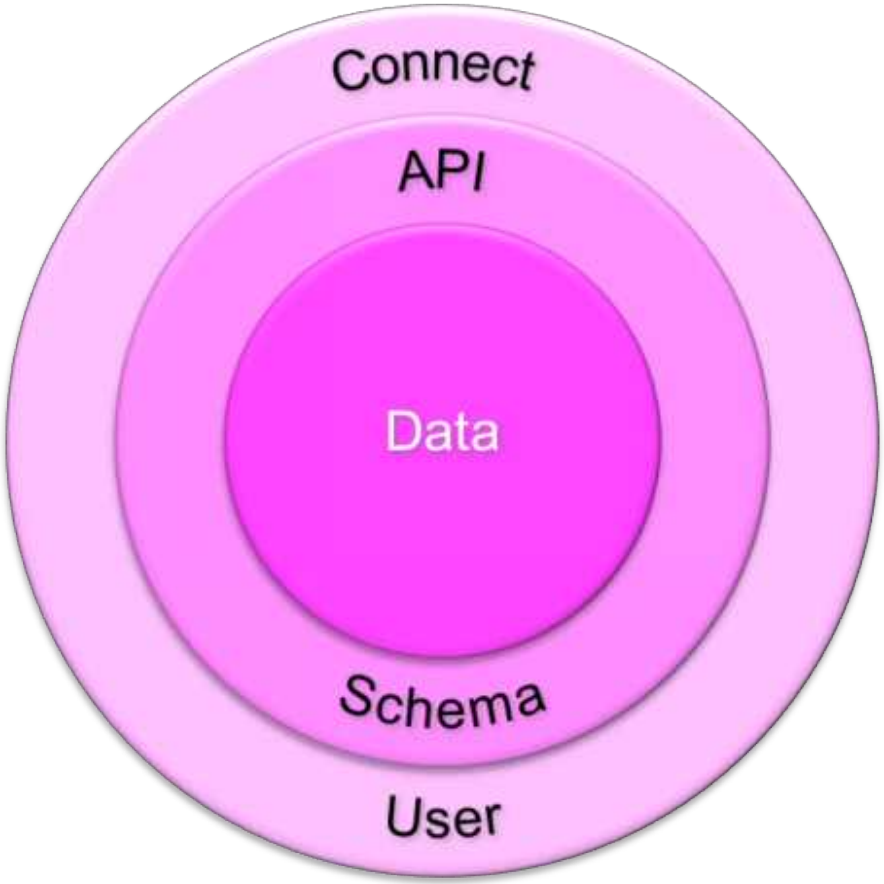


Quelle abgerufen am 23.10.2018: https://www.salvis.com/blog/2018/07/18/the-pink-database-paradigm-pinkdb/

PinkDB Paradigm

Unterschiede zu SmartDB

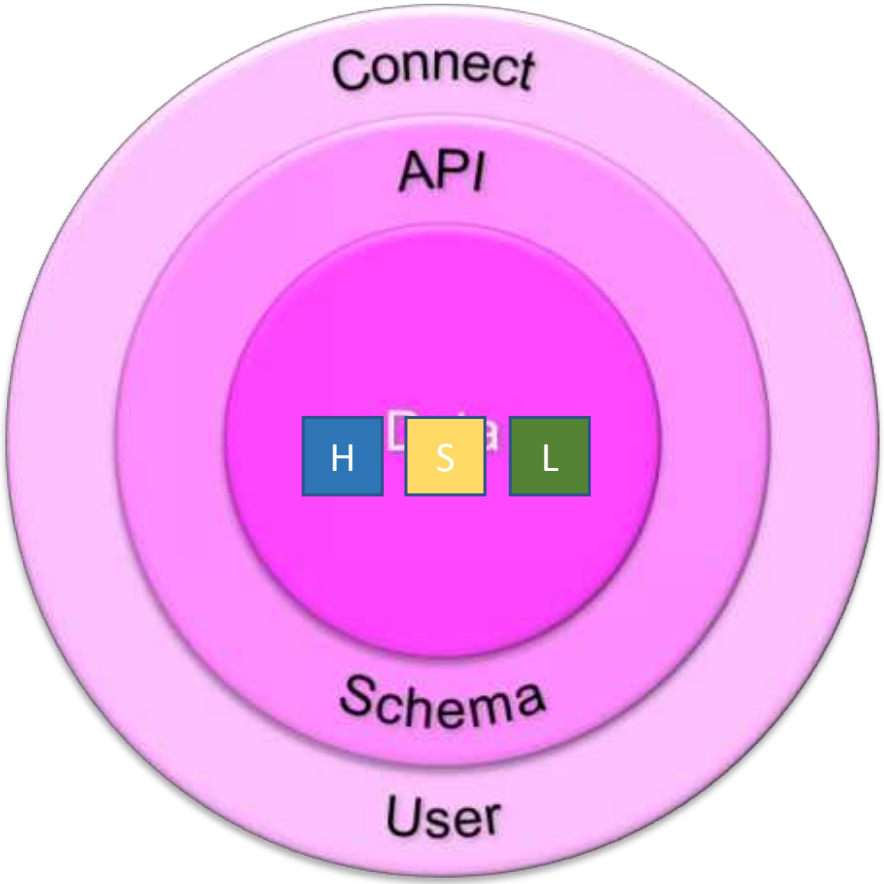
- Datenbank unabhängig
- Views erlaubt
- **Generatoren erlaubt**
- Data – speichert Informationen in vom Datenmodell definierten Struktur



Quelle abgerufen am 23.10.2018: https://www.salvis.com/blog/2018/07/18/the-pink-database-paradigm-pinkdb/

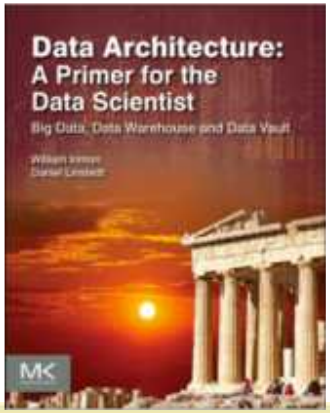
DataVault Implementierung

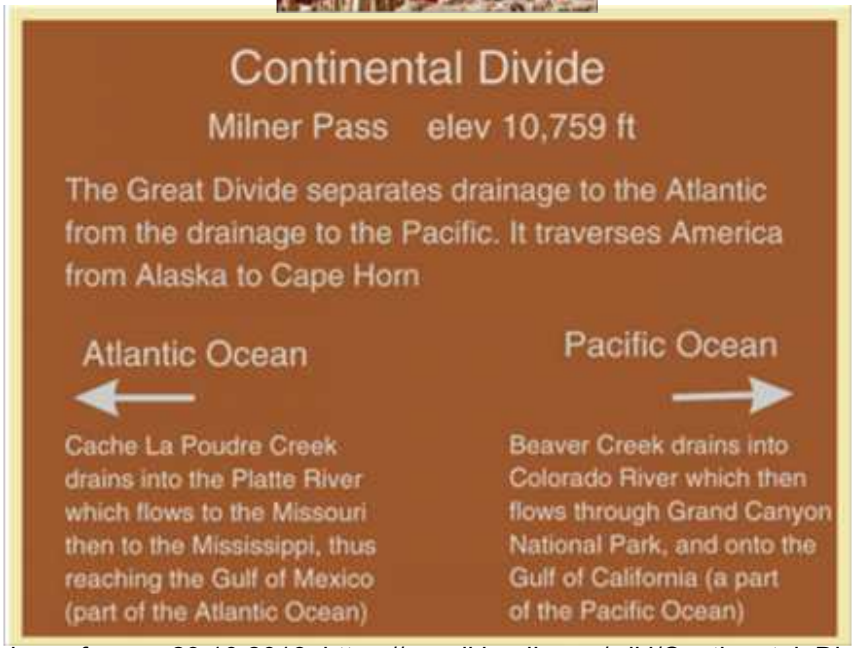
- Lademuster in DataVault sind Set-based
- Automat generiert DataVault Datenschema
- Schema API?
- Soft Rules?



Quelle abgerufen am 23.10.2018: https://www.salvis.com/blog/2018/07/18/the-pink-database-paradigm-pinkdb/

The „Great Divide“ in DataVault?



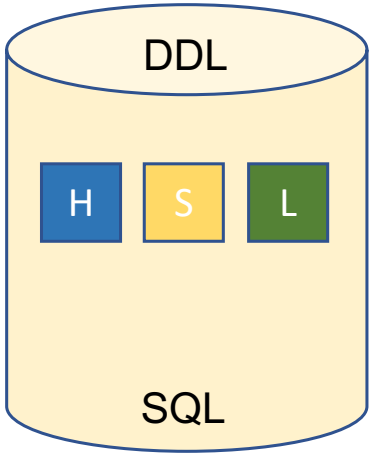


Quelle abgerufen am 23.10.2018: https://en.wikipedia.org/wiki/Continental_Divide_of_the_Americas

Anforderung

Fachseite

Technik



Anforderung

Fachseite

Technik

Physisches Modell

Mapping
Source

Generator

DDL

H S L

SQL

Speichern
Historisieren
Automatisieren
Nachvollziehbarkeit

Anforderung

Fachseite

Technik

Physisches Modell

Mapping
Source

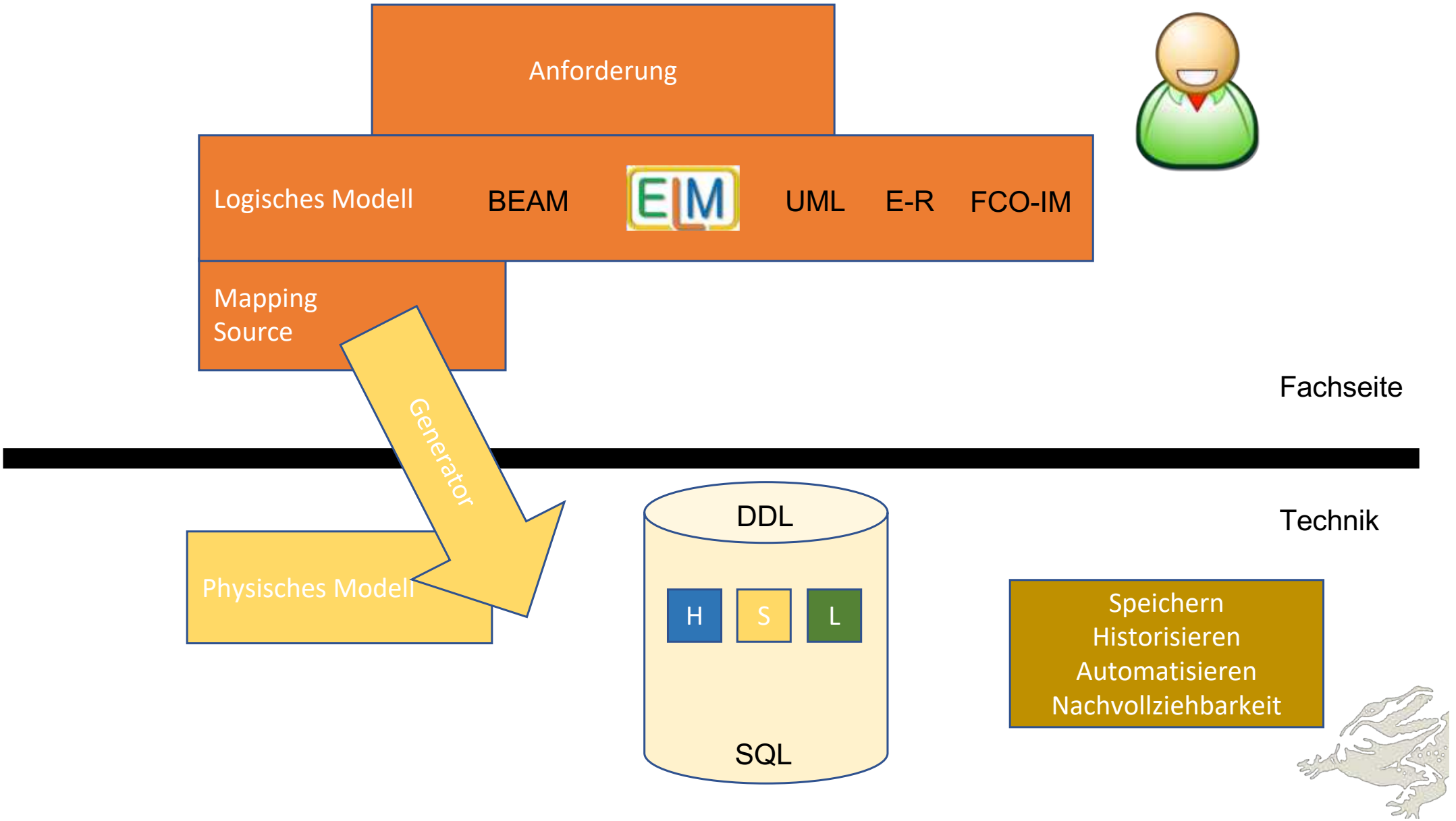
Generator

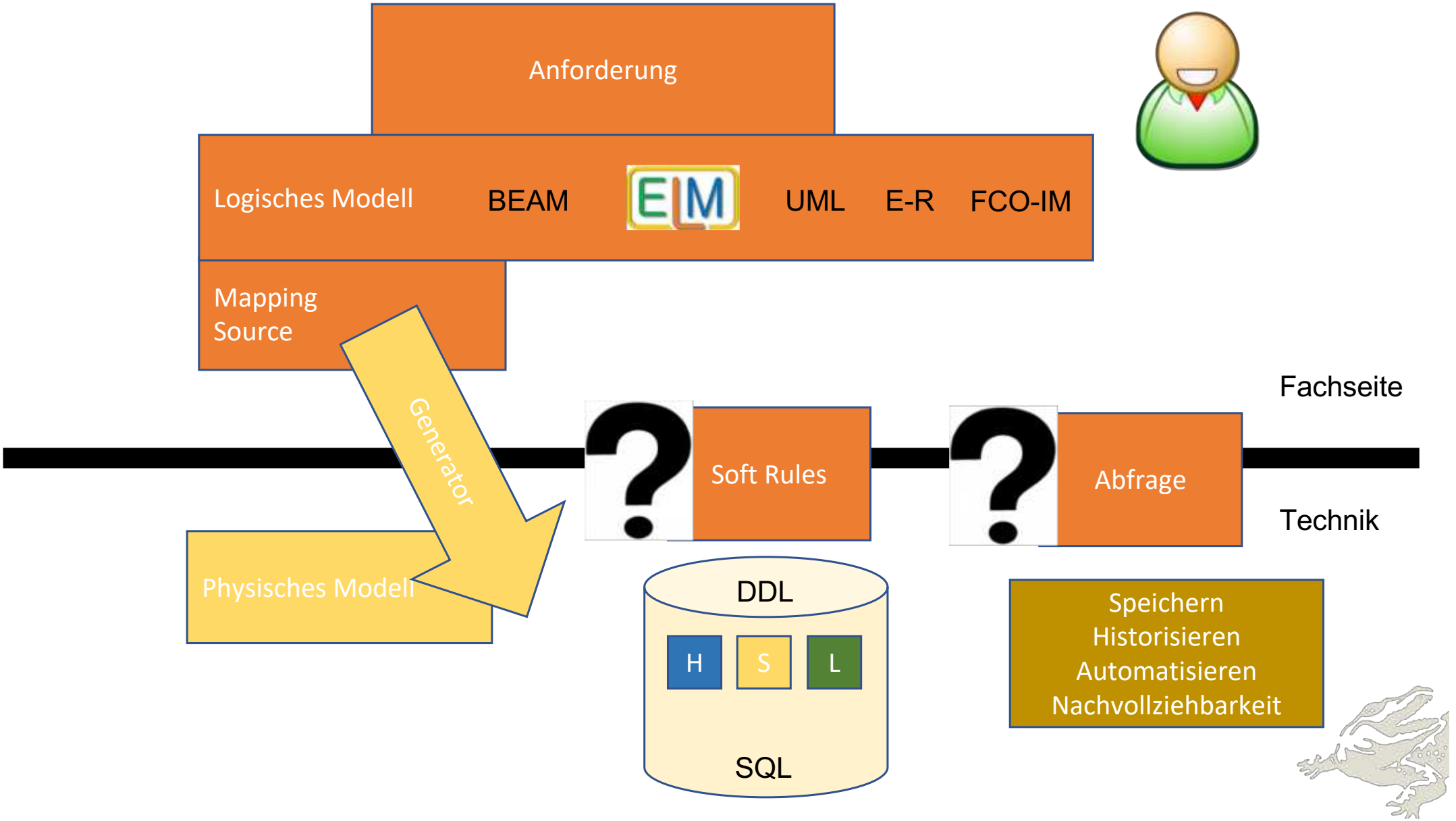
DDL

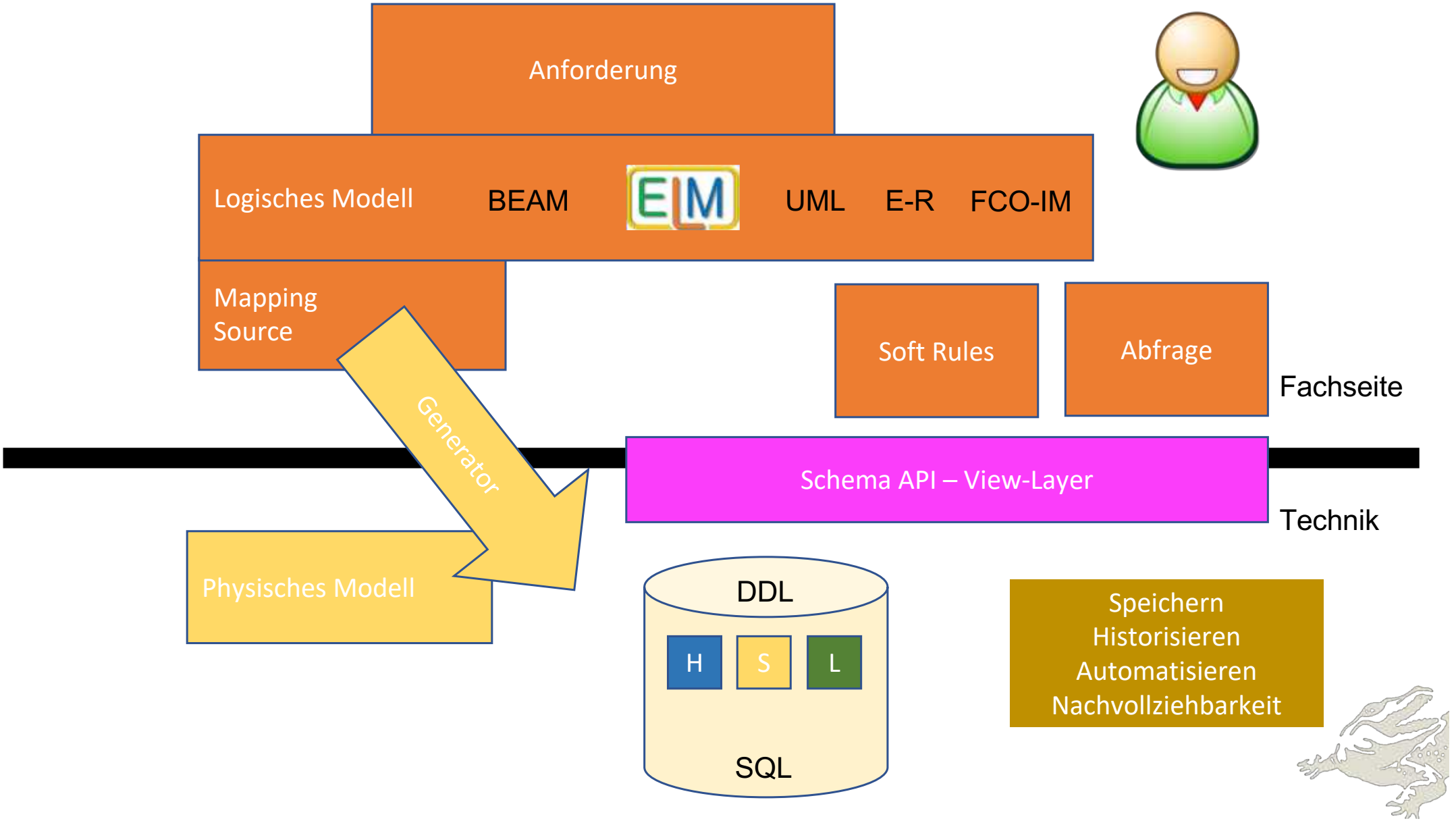
H S L

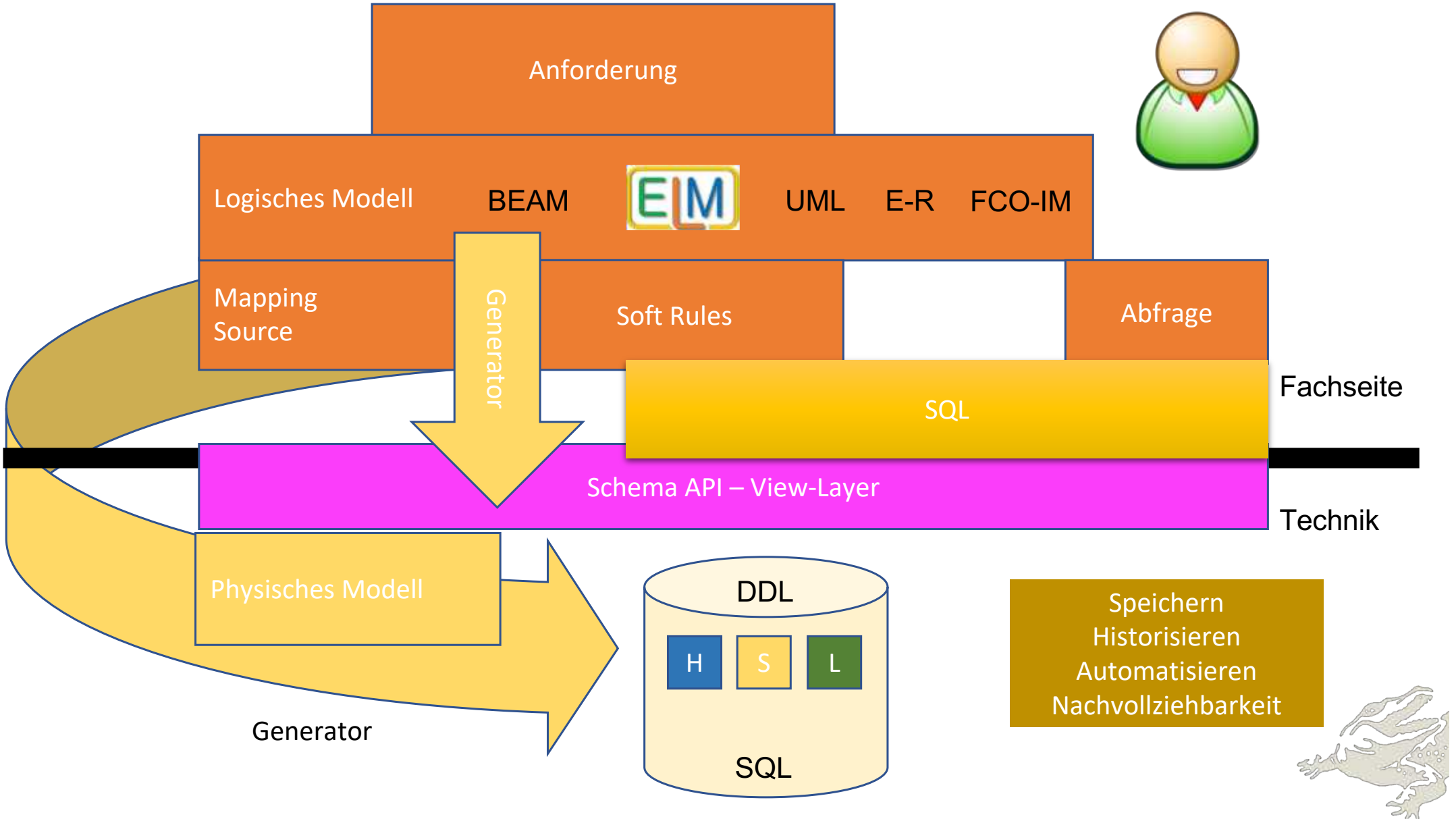
SQL

Speichern
Historisieren
Automatisieren
Nachvollziehbarkeit


Anforderung
Logisches Modell
BEAM
E|M
UML
E-R
FCO-IM
Mapping
Source
Generator
Fachseite
Physisches Modell
DDL
H
S
L
SQL
Technik
Speichern
Historisieren
Automatisieren
Nachvollziehbarkeit

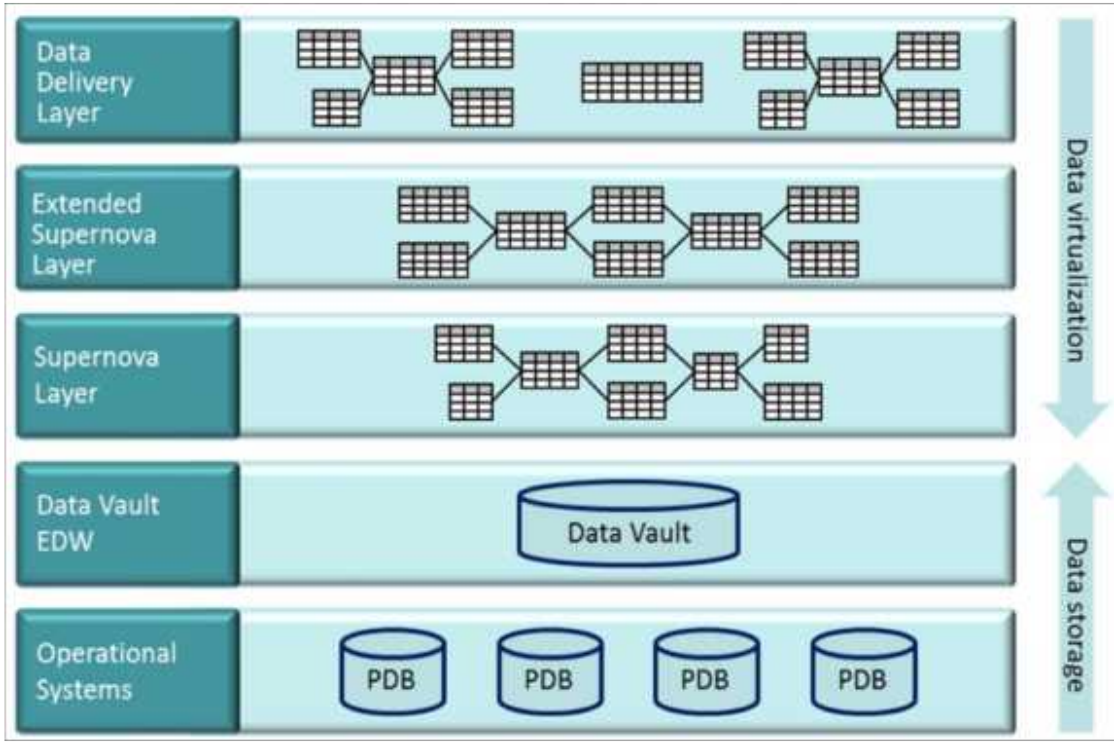

Anforderung
Logisches Modell
BEAM
E|M
UML
E-R
FCO-IM
Mapping
Source
Generator
Physisches Modell
Soft Rules
Abfrage
Fachseite
Technik
DDL
H
S
L
SQL
Speichern
Historisieren
Automatisieren
Nachvollziehbarkeit


Anforderung
Logisches Modell
BEAM
E|M
UML
E-R
FCO-IM
Mapping
Source
Soft Rules
Abfrage
Fachseite
Generator
Schema API – View-Layer
Technik
Physisches Modell
DDL
H
S
L
SQL
Speichern
Historisieren
Automatisieren
Nachvollziehbarkeit


Anforderung
Logisches Modell
BEAM
E|M
UML
E-R
FCO-IM
Mapping Source
Generator
Soft Rules
Abfrage
SQL
Fachseite
Schema API – View-Layer
Technik
Physisches Modell
DDL
H
S
L
SQL
Generator
Speichern
Historisieren
Automatisieren
Nachvollziehbarkeit

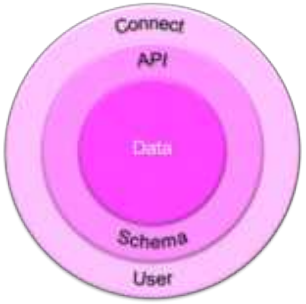
Schema API – View Layer

- Einheitlicher Zugriff auf DataVault
- Information Hiding
- Zugriffsschicht, Soft Rules und Nutzerzugriffe basieren auf dem selben, logischen Modell
- Optmimierung der SQL an zentraler Stelle
- Generierbar aus dem Modell



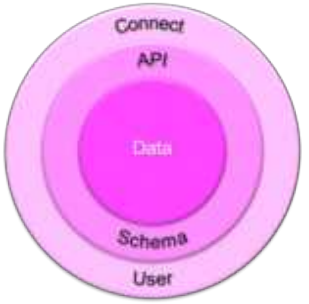
Source: Rick van der Lans sponsored by Cisco Centennium: http://www.cisco.com/c/dam/en_us/services/enterprise-it-services/data-virtualization/documents/whitepaper-cisco-datavaul.pdf

PinkDB und DataVault



- Model Driven
  - Strukturierte Modellierung ermöglicht Automation
  - Generierung von Data Vault
  - Generierung der Zugriffsschicht
- Set Based
  - Lademuster
  - Zugriffs-View in API Schema
  - Information Hiding
- Vereinfachung der Infrastruktur
  - Einfache Lineage über SQL / View ermöglicht GPDR
  - Reduktion der Wartungskosten
  - Einfache Fehlersuche, Einfaches Skillset
  - SQL / relationales Modell als stabile Schnittstelle
- Algorithmen nah an den Daten

PinkDB und DataVault



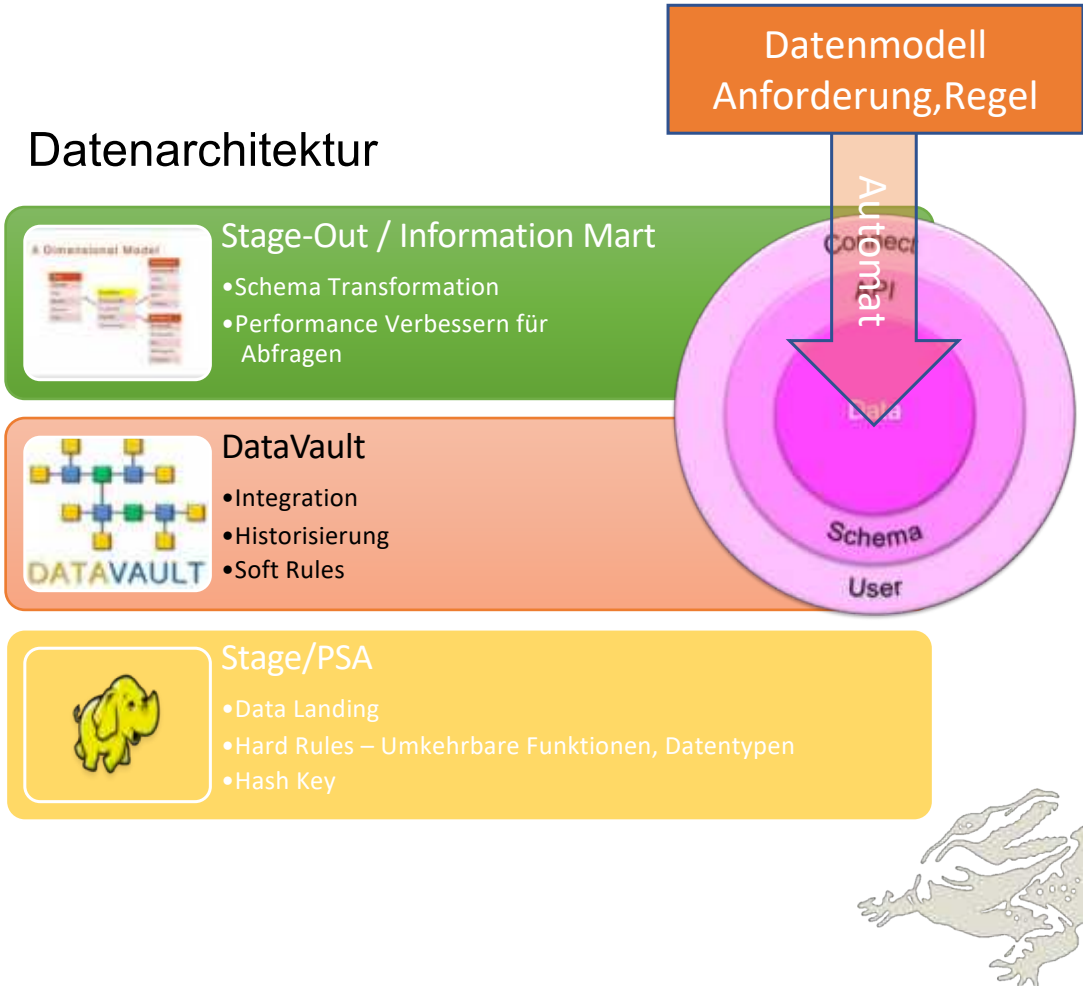
- Was ist mit Fehlerbehandlung und Hard Rules?
  - Fehlertolerante SQL (Functions ON-Error, Default Werte)
  - DQ-Regeln und DQ-Bewertung Downstream, nachgelagert, Error Mart
  - Landing in Hadoop / mittels Kafka und Flink
- Prozesssteuerung
  - DataVault ermöglicht parallele Beladung und damit einfache Steuerung
  - SoftRules müssen Abhängigkeiten berücksichtigen
- Fehlende Flow-Control in SQL
  - Einfache Muster erfordern keine komplexe Flow-Control
- ETL Werkzeuge benötigen keine Datenbankkenntnisse
  - Datenbankkenntnisse sind essentiell

Architekturvorteile von Data Vault

Versprechen von DataVault

- Flexibilität für Änderungen
  - Inkrementell
- Effiziente Historisierung
- Effiziente Lademuster
  - Automatisierbar
  - Insert Only
  - Hohe Parallelität
- Nachvollziehbarkeit

Datenarchitektur



Take-Away

Don‘ts

- ETL – im Sinne von satzweiser Verarbeitung ist nicht zukunftsfähig
- ETL – im Sinne von Middleware außerhalb der Datenhaltung
- Fehlerbehandlung, Datenqualität in Muster integrieren
    - Separation of Concerns wahren
- Werkzeuge als Strategie zu behandeln
    - sollten austauschbar sein
- Bestehende ETL Frameworks wiederverwenden

Dos

- Modellgetriebene Entwicklung
- Datenarchitektur und deren Nachhaltigkeit ist strategisch
    - Regelwerk aufstellen, dokumentieren und nachhalten
- Bleib bei den Mustern, Set-Based
- DYMD - Dont Move Your Data, Vorsicht mit ETL
- Automation ist der Schlüssel
    - Vorsicht bei der Werkzeugauswahl

PinkDB & DataVault

Vielen Dank für die Aufmerksamkeit