Noch mehr Flexibilität- DataVault mit virtuellen Datamarts

Konferenz Data Analytics März 2018

Agenda

1 Data Vault

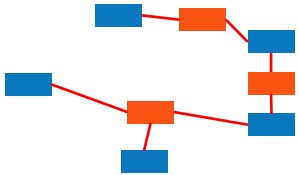
2 UDG - der PPI Generator

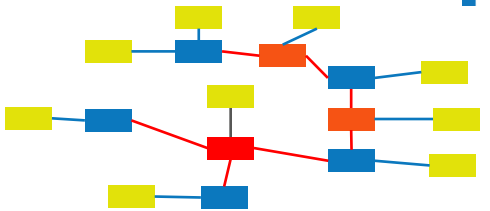
3 Virtuelle Datamarts

4 Das DWH der Zukunft von PPI

01 Data Vault

Kurzvorstellung Data Vault





Hubs
Geschäftsobjekte

Links
Beziehungen von Geschäftsobjekten

Satellites
Eigenschaften von Geschäftsobjekten

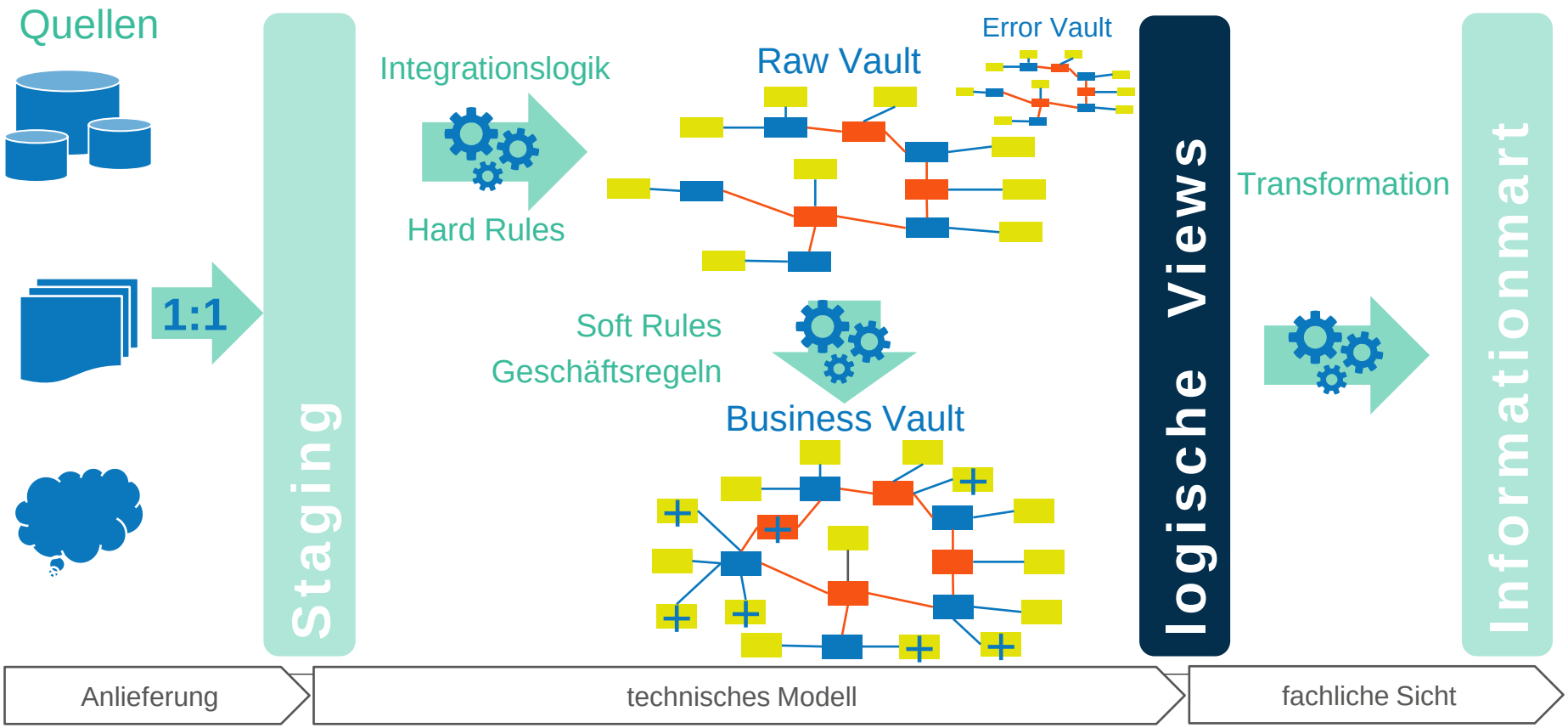
Ziel:

- Entkopplung einzelner Komponenten / Fragestellungen für
  - fachliche Analyse
  - Entwicklung / Test
  - Beladung
- agiles Vorgehen
- Ergänzungen schnell produktiv nehmen (Time to Market)

- erfunden von Dan Linstedt
- zuerst in USA und Niederlande verbreitet
- seit ca. 6 Jahren weltweit genutzt

Data Vault
Typische Architektur

PPI Ergänzung



02 UDG - der PPI Generator

UDG = Universal Datamodel Generator

Data Vault Regel „**Reduzierung der Komplexität**“ gilt auch für die Tools

herkömmliches Tool

- deckt Anforderungen verschiedener Kunden ab
- hohe Komplexität
- erfordert viel Know How
- Gefahr von Fehlbedienungen

Wer nutzt mehr als 5% der Funktionen von Word?

PPI Ansatz

- basiert auf Eclipse
- statt eines Tools wird eine anpassbare API geliefert
- ist vorkonfiguriert
- nutzt eigene Domain Specific Language (DSL)

➡ kann nicht alles, aber genau das, was das Projekt braucht

Umsetzung mit UDG

1.Basiskomponente → 2.Anpassung Projektspezifika → 3.Generatorverwendung im Projekt

Basis Data Vault Implementierung

Standard Funktionalitäten
Code bringt PPI in Projekt ein
voll funktionsfähig

projektspezifische Architektur

initiale Projektphase

fachliche Projektarbeit

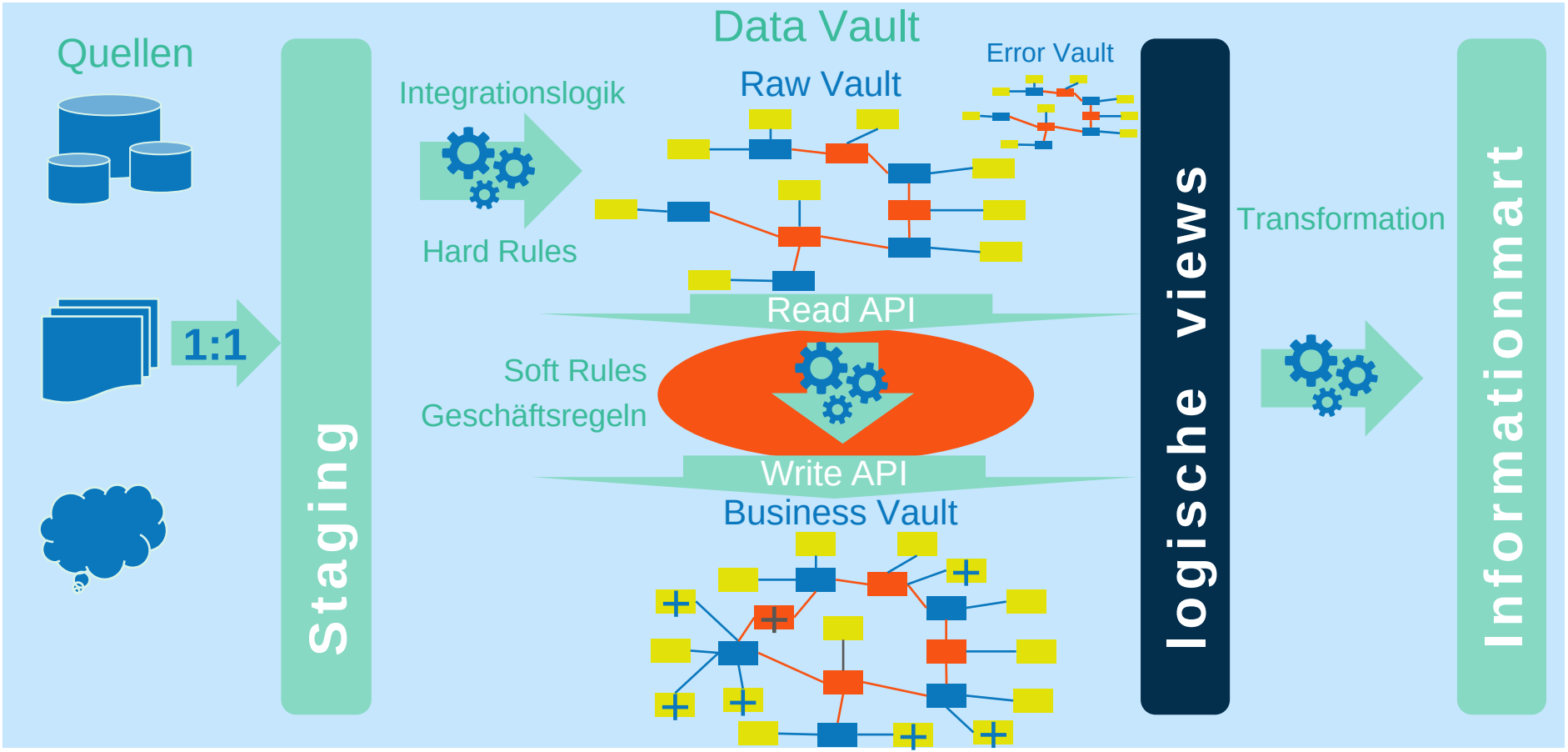
Standardhaus

Haus wohnlich eingerichtet

Haus wird genutzt

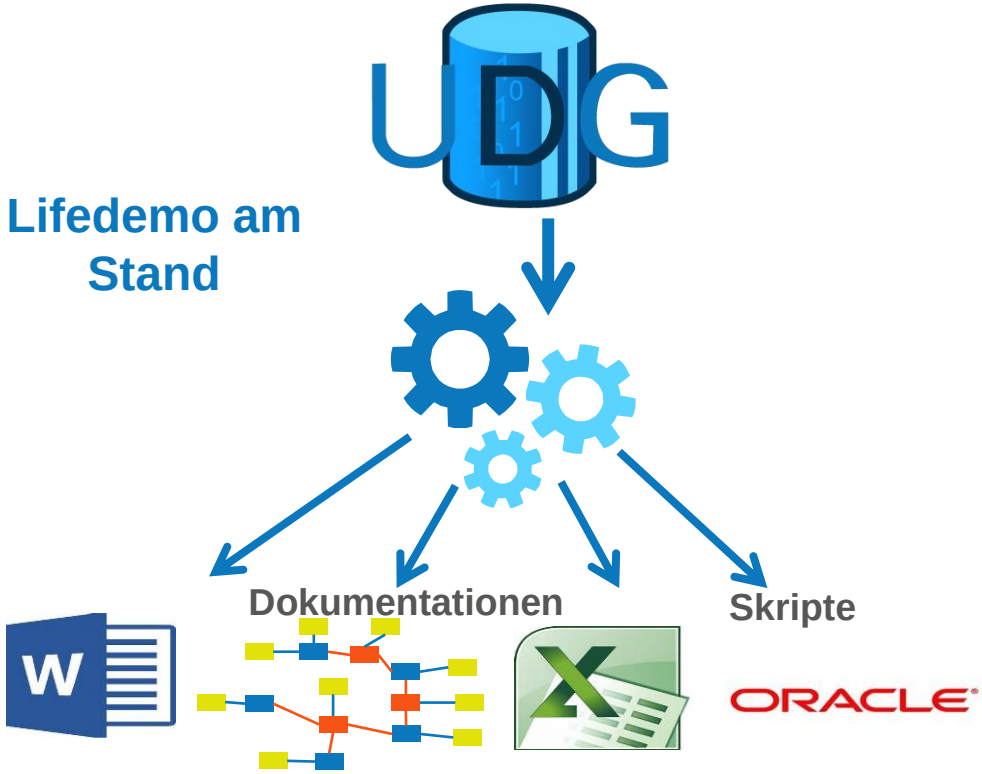
individueller Programmcode | generiert durch UDG

Technische Umsetzung - Generatorabdeckung



Features UDG

- **Validierung**
  - Fehleingaben werden erkannt
- **Code Completion**
  - Unterstützung bei der Eingabe
- **Generierung**
  - Aufruf per Kontextmenü
- **Ergebnisse**
  - Skripte für
    - Anpassung Datenmodell
    - Beladung Datenmodell
    - Ergänzung logisches Modell
  - Steuerung angepasst
  - Dokumentationen aktualisiert
    - Grafiken / Word / Excel



03 Virtuelle Datamarts

Ergänzung DataVault um virtuelle Datamarts

Herausforderung DataVault

- Flexibiltät geht bei der Marterstellung verloren
- abweichende Logiken bei Erstellung verschiedener Datamarts
- zusätzlicher Aufwand Erstellung / Test von Datamarts
- aufwendige Migrationen, da Datamarts hohes Datenvolumen haben

Vorteile Virtualisierung

- logische Views durch UDG generierbar
- alle Views zeigen gleiche Daten an, d.h. Logik ist immer einheitlich
- Aufwand Erstellung / Test extrem reduziert
- keine Datenmigrationen
- Fachbereich hat sofort einfaches leicht verständliches Modell
- Fachbereich greift nicht direkt auf DataVault zu

Aber: Was ist mit der Performance?

Performance virtuelle Datamarts

Caching von Oracle

- „normaler“ Datencache
- Big Table Cache
- InMemory Cache
- Full Database Cache

Parallel Query

ExaData

Optimizer Join Elimination

- View hat viele Felder und Joined viele Tabellen
- bei Auswahl einzelner Felder (typisch für Reports) greift Oracle aber nur auf wenige Tabellen zu
- vollautomatische Entscheidung, auf welche Tabelle bei welchen ausgewählten Feldern zugegriffen werden muss

Oracle liefert Performance!

Opimizer Join Elimination - View mit vielen Feldern und komplexem SQL

V_SCHADEN | COREX01_CORE

Spalten | Daten | Zugriffsberechtigungen | Abhängigkeiten | Details | Trigger | SQL

Aktionen...

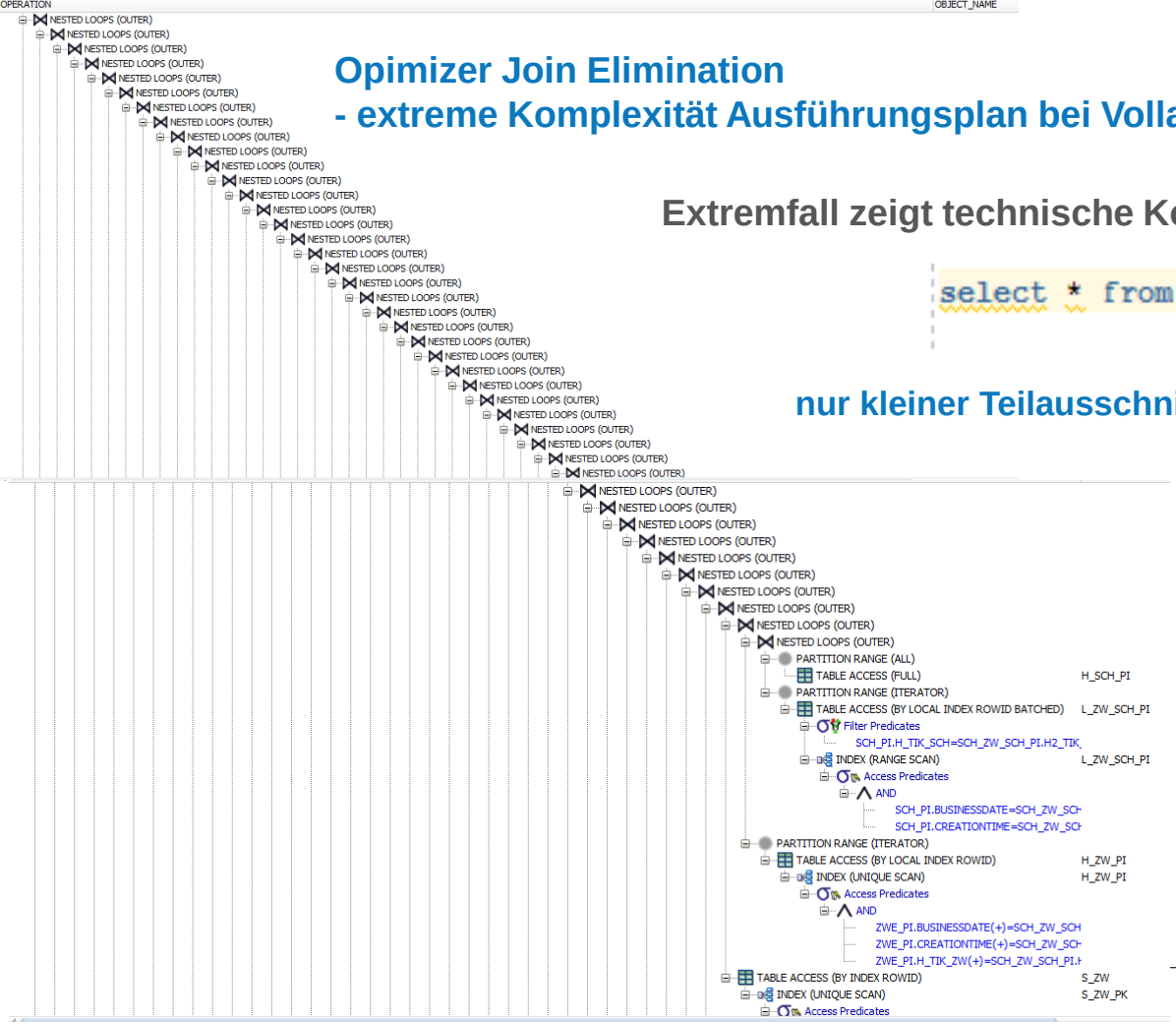
	COLUMN_NAME	DATA_TYPE
1	SCH_BUSINESSDATE	DATE
2	SCH_CREATIONTIME	DATE
3	SCH_PK_HASHKEY	CHAR(32 CHAR)
4	SCH_MANDANT_NUMMER	VARCHAR2(30 CHAR)
5	SCH_SOB	VARCHAR2(30 CHAR)
6	SCH_SUBMANDANT_NUMMER	VARCHAR2(30 CHAR)
7	SCH_ANZ_VERL_AST_Z	NUMBER
8	SCH_ANZ_VERL_PERS_Z	NUMBER(5)
9	SCH_BELAST_VERM_FG	CHAR(1 CHAR)
10	SCH_BETR_FG	CHAR(1 CHAR)
11	SCH_COC_SPERR_DT	DATE
12	SCH_COC_SPERR_FG	CHAR(1 CHAR)
13	SCH_DECK_VORH_FG	CHAR(1 CHAR)
14	SCH_FSRI_AEND_DT	DATE
15	SCH_FREMD_OB	VARCHAR2(20 CHAR)
16	SCH_FREMD_OB_TXT	VARCHAR2(150 CHAR)
17	SCH_FREMD_OB_CD	VARCHAR2(30 CHAR)
18	SCH_FREMD_OB_DE	VARCHAR2(1000 CHAR)
19	SCH_FREMD_OB_EN	VARCHAR2(1000 CHAR)
20	SCH_FUEH_UEB_DRT_FG	CHAR(1 CHAR)
21	SCH_GKA_KD_KENN_NM	VARCHAR2(30 CHAR)
22	SCH_GKA_OB	VARCHAR2(20 CHAR)
23	SCH_GKA_EXT_OB	VARCHAR2(20 CHAR)
24	SCH_GER_SOB	VARCHAR2(20 CHAR)
25	SCH_GS_NM	VARCHAR2(30 CHAR)
26	SCH_ALT_GES_TXT	VARCHAR2(150 CHAR)
27	SCH_SCH_GRP_BEW_FG	CHAR(1 CHAR)

insgesamt 163 Felder

kleiner Ausschnitt des SQL

```
-- Join satellite SatSchaden
LEFT JOIN S_SCH SCH
     ON  SCH_pi.H_TIK_SCH = SCH.H_TIK_SCH
     AND SCH_pi.S_SCH_T = SCH.TECH_GUELTIG_VON
     AND SCH_pi.S_SCH_F = SCH.FACH_GUELTIG_VON
-- Join satellite SatSchaden_KPI
LEFT JOIN S_SCH_KPI SCH_KPI
     ON  SCH_pi.H_TIK_SCH = SCH_KPI.H_TIK_SCH
     AND SCH_pi.S_SCH_KPI_T = SCH_KPI.TECH_GUELTIG_VON
     AND SCH_pi.S_SCH_KPI_F = SCH_KPI.FACH_GUELTIG_VON
-- Join link Link_Schaden_zu_Geschaeftsstelle
LEFT JOIN L_GS_SCH_PI SCH_GS_SCH_pi
     ON   SCH_pi.H_TIK_SCH = SCH_GS_SCH_pi.H2_TIK_SCH
     AND SCH_pi.BUSINESSDATE = SCH_GS_SCH_pi.BUSINESSDATE
     AND SCH_pi.CREATIONTIME = SCH_GS_SCH_pi.CREATIONTIME
  --loese Foreign Key Geschaeftsstelle_1 rekursiv auf
  --Start Rekursionlevel 2 fuer HubGeschaeftsstelle (H_GS)
  --Gueltig Beziehung
  --read the top level relation
  LEFT JOIN H_GS_PI GST_pi
  ON GST_pi.BUSINESSDATE = SCH_GS_SCH_pi.BUSINESSDATE
  AND GST_pi.CREATIONTIME = SCH_GS_SCH_pi.CREATIONTIME
  AND GST_pi.H_TIK_GS = SCH_GS_SCH_pi.H1_TIK_GS
  -- Join Hub /Link
  LEFT JOIN H_GS GST_hub
       ON GST_pi.H_TIK_GS = GST_hub.H_TIK_GS
  -- Join satellite SatGeschaeftsstelle
  LEFT JOIN S_GS GST
       ON  GST_pi.H_TIK_GS = GST.H_TIK_GS
       AND GST_pi.S_GS_T = GST.TECH_GUELTIG_VON
       AND GST_pi.S_GS_F = GST.FACH_GUELTIG_VON
  --Ende Rekursionlevel 2 fuer HubGeschaeftsstelle (H_GS)
-- Join link Link_Schaden_zu_Mandant
LEFT JOIN L_MANDANT_SCH_PI SCH_MANDANT_SCH_pi
     ON   SCH_pi.H_TIK_SCH = SCH_MANDANT_SCH_pi.H2_TIK_SCH
     AND SCH_pi.BUSINESSDATE = SCH_MANDANT_SCH_pi.BUSINESSDATE
     AND SCH_pi.CREATIONTIME = SCH_MANDANT_SCH_pi.CREATIONTIME
```

Opimizer Join Elimination
- extreme Komplexität Ausführungsplan bei Vollabfrage



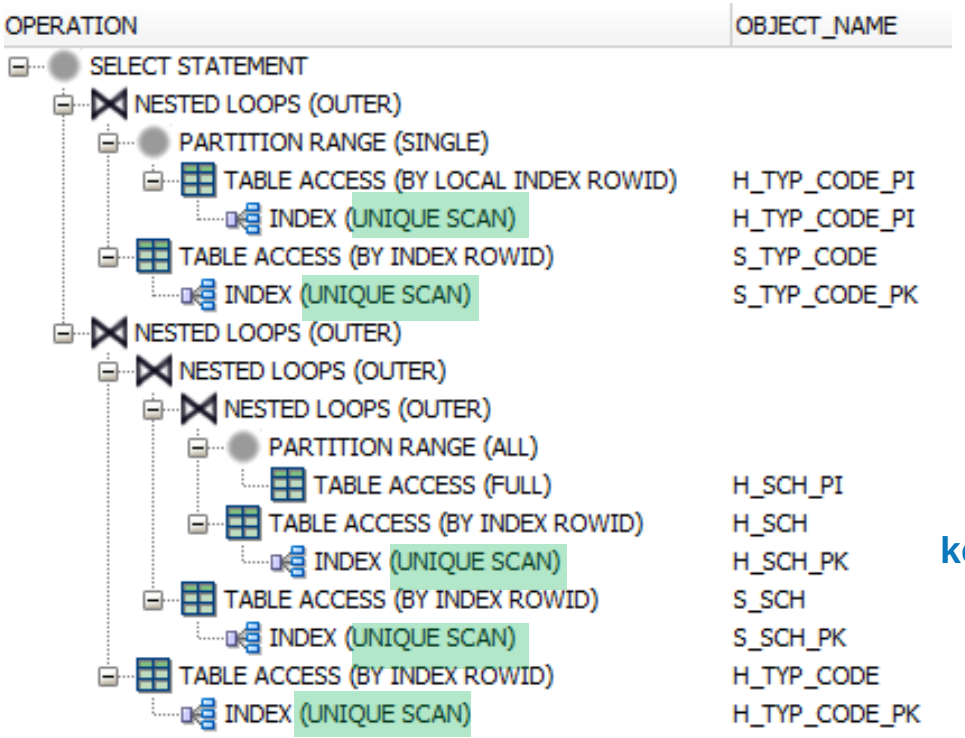
Extremfall zeigt technische Komplexität der View

```
select * from v_schaden;
```

nur kleiner Teilausschnitt des Plan dargestellt

Opimizer Join Elimination - geringe Komplexität Ausführungsplan bei Teilabfrage

dies ist der Standardfall bei praktischer Nutzung mit BI




```
select SCH_BUSINESSDATE,
SCH_MANDANT_NUMMER,
SCH_SOB,
SCH_BETR_FG,
SCH_COC_SPERR_DT,
SCH_COC_SPERR_FG,
SCH_DECK_VORH_FG,
SCH_FSRI_AEND_DT,
SCH_FREMD_OB,
SCH_FREMD_OB_TXT,
SCH_SCH_PRIO_CD,
SCH_SCH_PRIO_DE
 from v_schaden;
```

kompletter Plan dargestellt

- sehr wenige Tabellen
- nur 1x Full Scan
- alle anderen Zugriff optimal per eindeutigem Index

Opimizer Join Elimination - keine Komplexität Ausführungsplan bei Minimalabfrage

Extremfall beweist den vollständigen Verzicht auf alle unnötigen Tabellenzugriffe

```
select SCH_BUSINESSDATE
 from v_schaden;
```

OPERATION	OBJECT_NAME
SELECT STATEMENT	
PARTITION RANGE (ALL)	
INDEX (FULL SCAN)	H_SCH_PI

kompletter Plan dargestellt

- nur ein Indexzugriff

Referenzprojekte

Kunde 1

- Anbindung eines Liefersystems inklusive archivierter historischer Daten als Pilotprojekt
- nach Abschluss Anbindung weiterer Liefersysteme und fachlicher Inhalte jeweils inklusive archivierter historischer Daten

Talanx - oneData

talanx. Versicherungen. Finanzen.

- Aufbau eines einheitlichen DWH für Industrieversicherungen
- Start mit Schäden und Risiken
- Ausbau um Verträge, Partner, Buchungen

04 Das DWH der Zukunft

alles ist flexibel änderbar - nicht nur flexible Ergänzungen

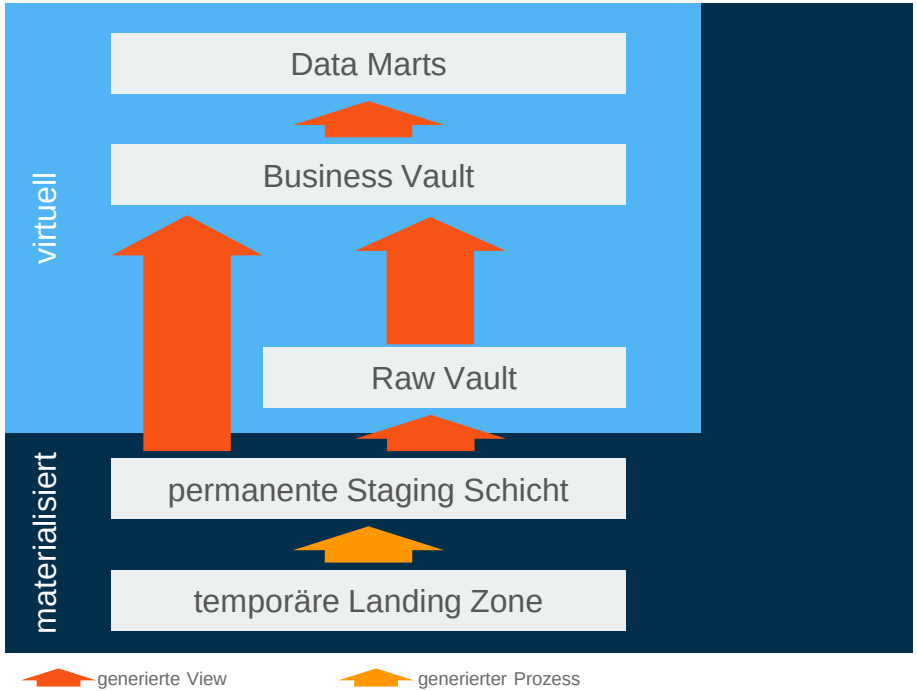
vollständige Virtualisierung

sofortige Produktivnahme von fachlichen Änderungen

stärkere Einbeziehung der Fachabteilung

Skalierbarkeit und Flexibilität – Erfolgsfaktoren für das DWH der Zukunft

Data-Vault-Architektur mit virtualisierten Schichten



Vorteile Data Vault

- entkoppelte Komponenten innerhalb des Datenmodells
- standardisierte und automatisierte Datenbewirtschaftungsprozesse
- flexible Erweiterbarkeit

zusätzliche Vorteile Virtualisierung

- sofortige Wirksamkeit neuer Daten und neuer Regeln
- reduzierter Speicherbedarf
- weniger Datenredundanz
- geringer Migrationsaufwand bei Änderungen
- geringer Aufwand für Regressionstests
- **alles** ist flexibel änderbar

joerg.stahnke@ppi.de

Kennen Sie folgende Situation?

Fachabteilung

Die IT ist viel zu langsam

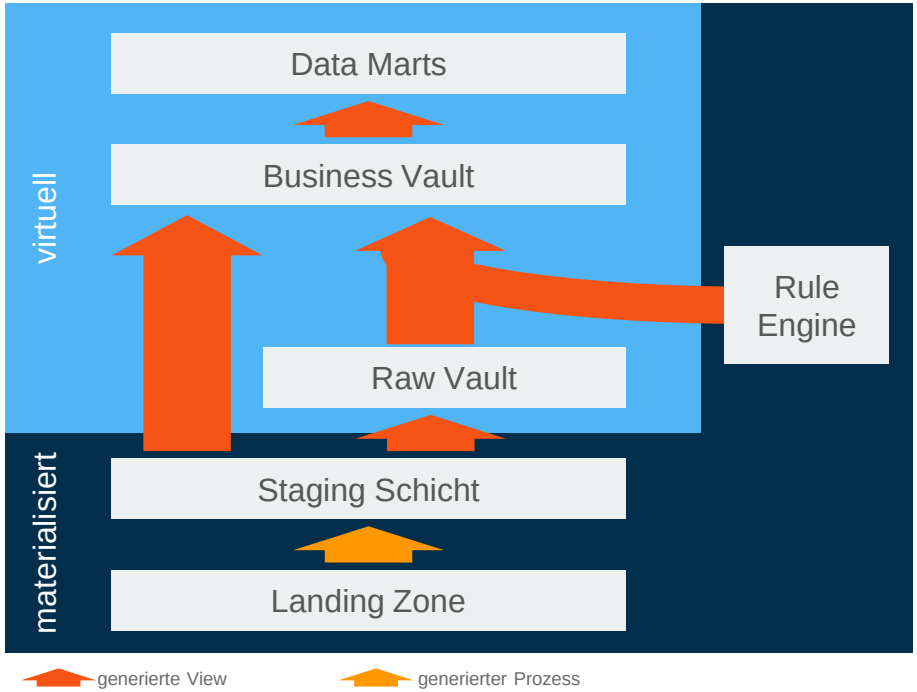
IT- Abteilung

Die Fachabteilung weiß nicht was sie will und ändert ständig die Anforderungen

PPI Lösung: Einsatz einer Rule Engine

Skalierbarkeit und Flexibilität – Erfolgsfaktoren für das DWH der Zukunft

Data-Vault-Architektur mit virtualisierten Schichten und Rule-Engine



Vorteile Rule Engine

- IT nur initial beteiligt
- Fachabteilung kann Logik im laufenden Produktionsbetrieb anpassen
- Anpassungen extrem schnell möglich

Prinzip Rule Engine

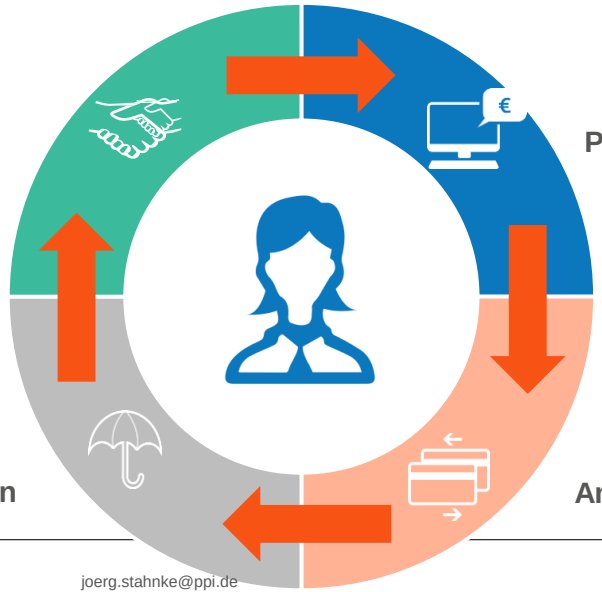
Entwicklung

technische Programmierung Rule Engine

fachliche Festlegung initiale Regeln

Initialer Test

Produktion



Freigabe

4-Augen Prinzip

Produktivnahme

Anpassung Regeln

Simulationsrechnung
Vergleich mit vorherigen Berechnungen

joerg.stahnke@ppi.de

Fazit

Wollen Sie diesen innovativen Weg gehen?

Wir helfen Ihnen gern dabei.

Kontakt

Jörg Stahnke
Principal Software Engineer

T +49 402227433-1329
M +49 160 3841771
joerg.stahnke@ppi.de
www.ppi.de

PPI AG
Moorfurthweg 13
22301 Hamburg