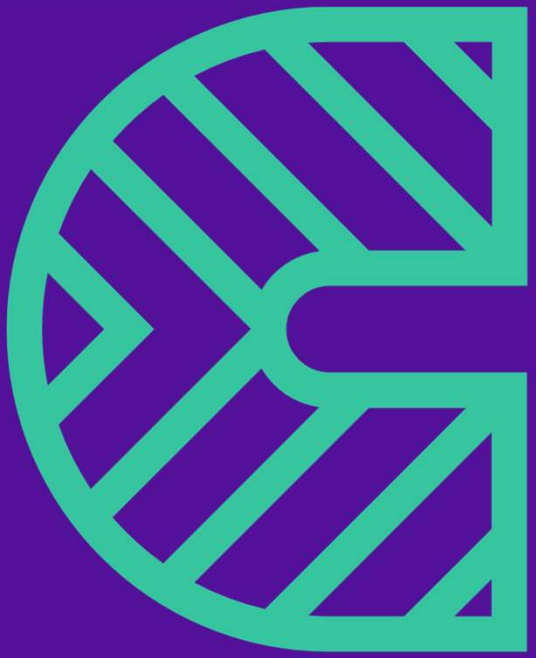
Bitemporale Historie

im DWH ohne Zusatzaufwand implementieren



11.04.2023 | Jörg Stahnke

Agenda

1. **Warum bitemporale Historie?**
2. **Datenbeladung**
3. **Datenabfrage in DataVault**
4. **Datenbereitstellung im Datamart**
5. **Simulations-/Korrekturberechnungen**

Kurzvorstellung DVG (DataVault-Generator)

100% Automatisierung aller Projektschritte

Validierung aller Eingaben

fachliche Analyse

Codeerstellung

Rule Engine mit Fachlogik

Simulationsläufe

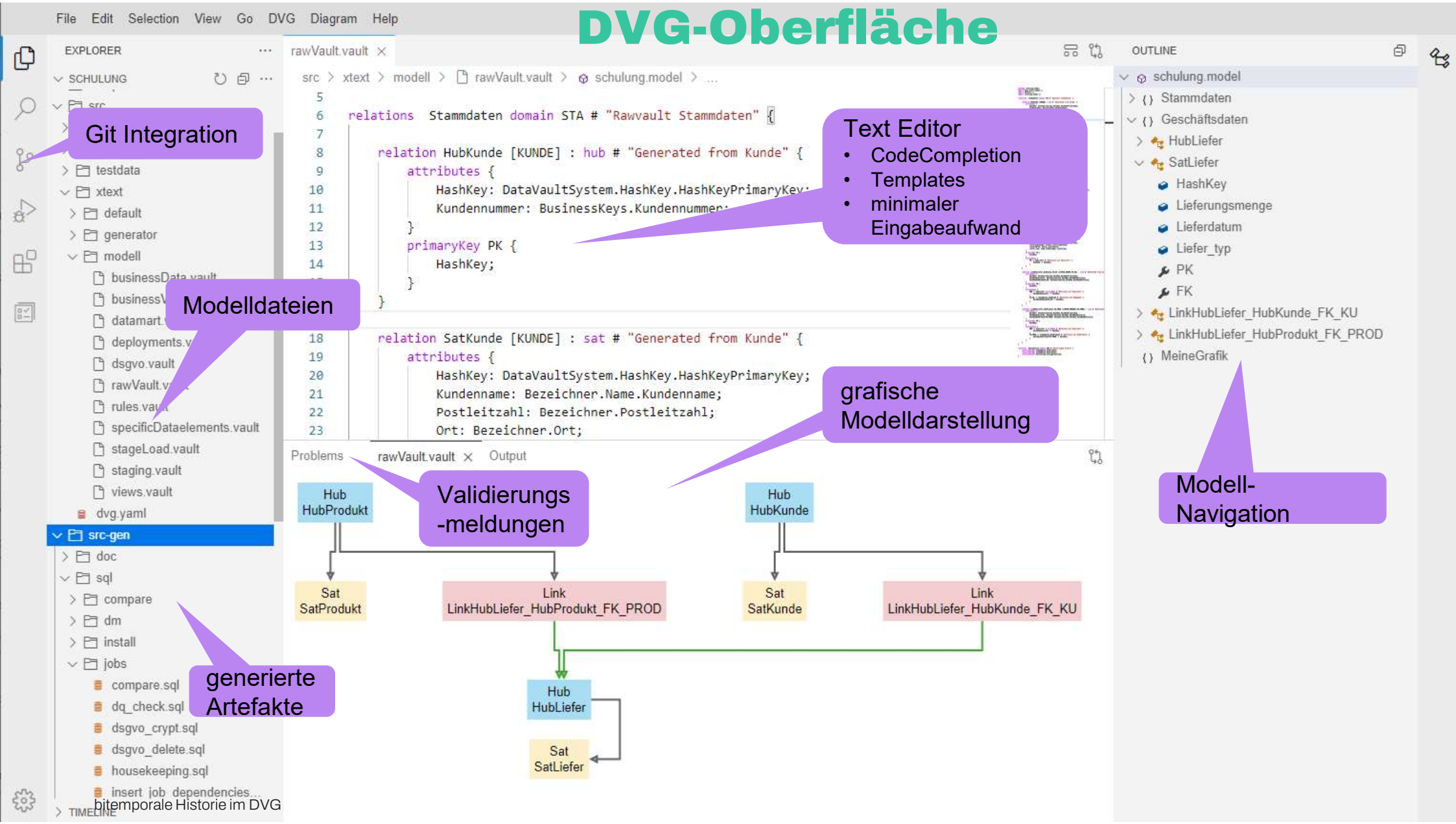
bitemporale Historie

Thema heute

Dokumentation | DSGVO | Templates | Data Quality | Code Completion | Prozess-steuerung

Data-Lineage | Test | Migration | CI/CD Git | Sandboxes | risikolose Produktivnahme

DVG-Oberfläche



bitemporale Historie im DVG

Warum bitemporale Historie?

Warum bitemporale Historie?

Wiederholbarkeit

Ich möchte den Bilanzgewinn der letzten 10 Jahre nach Filialen aufschlüsseln

Typische Beispiele

- Konzernbilanzen
- Meldungen an Aufsichtsbehörden

Aktualität

Für das Meeting mit Kunde XY benötige ich die aktuelle 360 Grad Sicht

Typische Beispiele

- Kundenauskünfte
- Unternehmenssteuerung

Wie gehe ich mit rückwirkenden fachlichen Änderungen um?
z.B.

- Schadensmeldungen bei Versicherungen
- Korrektur von Fehleingaben

bitemporale Historie

technisch

Wann bekomme ich Kenntnis von einer Information?

fachlich

Ab wann gilt diese Information?

rückwirkend / sofort / in der Zukunft?

Warum bitemporale Historie?

Testen / Warten von KI-Modellen

- Was sagt mein Modell mit dem Kenntnisstand von vor 4 Wochen für heute voraus?
- Wie gut stimmt dies mit den realen heutigen Tatsachen überein?

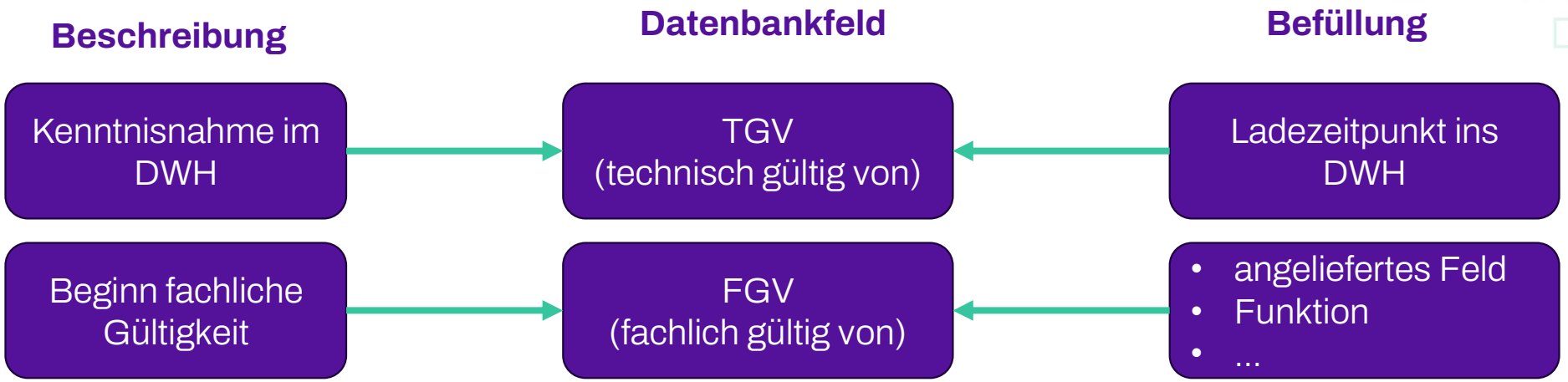
Data Science

» bitemporale Historie hat beide erforderlichen Datenstände gleichzeitig

» Überwachung Modell Shift und Re-Training leicht möglich

Datenbeladung

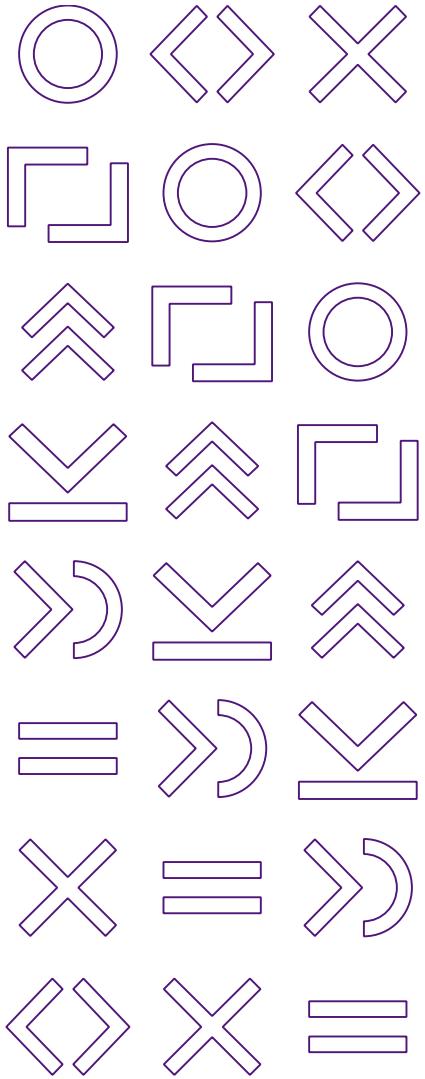
Umsetzung in Datenbank / Beladung



	HK_KUNDE	FGV	TGV	POSTLEITZAHL	ORT	STRASSE	HAUSNUMMER
1	0b48b1841bf2b209986a30f9720aafd1	2021-10-01	2021-11-01 00:00:00.000	22301	Hamburg	Moorfuhrtweg	13
2	2041f5bc5a5df10fe0d1ca3d90177e70	2021-10-01	2021-11-02 00:00:00.000	21635	Jork	Osterjork	14
3	23ff49ed815108d610c1a75d592488dd	2021-10-01	2021-11-01 00:00:00.000	22846	Norderstedt	Rathausalle	45
4	2c729497d91709de6bf1ccd875cf28e5	2021-10-01	2021-11-01 00:00:00.000	21129	Hamburg-Neuenfelde	Fährdeich	7
5	70333d89c64ddd3b7291f4a54aaf9641	2021-10-01	2021-11-01 00:00:00.000	24118	Kiel	Einsteinstraße	5
6	a1bba9704da78d22c74f1c2b0acf4213	2021-10-01	2021-11-01 00:00:00.000	21720	Grünendeich	Kirchenstieg	12
7	be2268c01180c5580c4be9a0c38c3686	2021-10-01	2021-11-01 00:00:00.000	22844	Norderstedt	Alter Kirchenweg	12
8	dce0eb67e20bada2f2a2ac1084c34d7d	2021-10-01	2021-11-01 00:00:00.000	22095	Hamburg	Jungfernstieg	33
9	ecebd75fc367b9aa41be06278689dc63	2021-10-01	2021-11-01 00:00:00.000	21723	Hollern-Twietenfleth	Hollernstr.	34

Ende fachlicher Gültigkeiten

Grundprinzip	Begründung
Insert Only **KEIN** Update oder Delete	Informationen gehen niemals verloren
fachliche Gültigkeit endet mit Beginn des nächsten Satzes	Das Ende einer fachlichen Gültigkeit hängt vom betrachteten technischen Gültigkeitszeitraum ab. Daher ist ein Feld „Gültigkeitsende“ nicht sinnvoll.
Sätze mit höherer technischer Gültigkeit haben Vorrang	Bei Kenntnisnahme neuer Tatsachen können rückwirkende fachliche Änderungen auftreten. (z.B. Korrektur Fehleingaben, neue Gutachten, rückwirkende Meldungen u.a.)



Ende fachlicher Gültigkeiten

Legende

technische Gültigkeit

fachliche Gültigkeit

geltende fachliche Werte

Modellierung im DVG

Beschreibung	DVG-Modell
Kenntnisnahme im DWH	keine Eingabe notwendig
Beginn fachliche Gültigkeit	Konfiguration beim Laden der Staging-Tabellen (Quell-Feld oder Berechnungsfunktion definieren)

Aufwand:
nur eine einzige Eingabe pro Staging-Tabelle

```
sourceSystem OFFICE # "Importierte Daten aus dem Office System" {

    stageRelation OfficeStaging.Kunde # "Anlieferung aller Kunden" {
        validFrom Gueltig_ab;
        validity "Y";
        duplicateOrder "0";
```

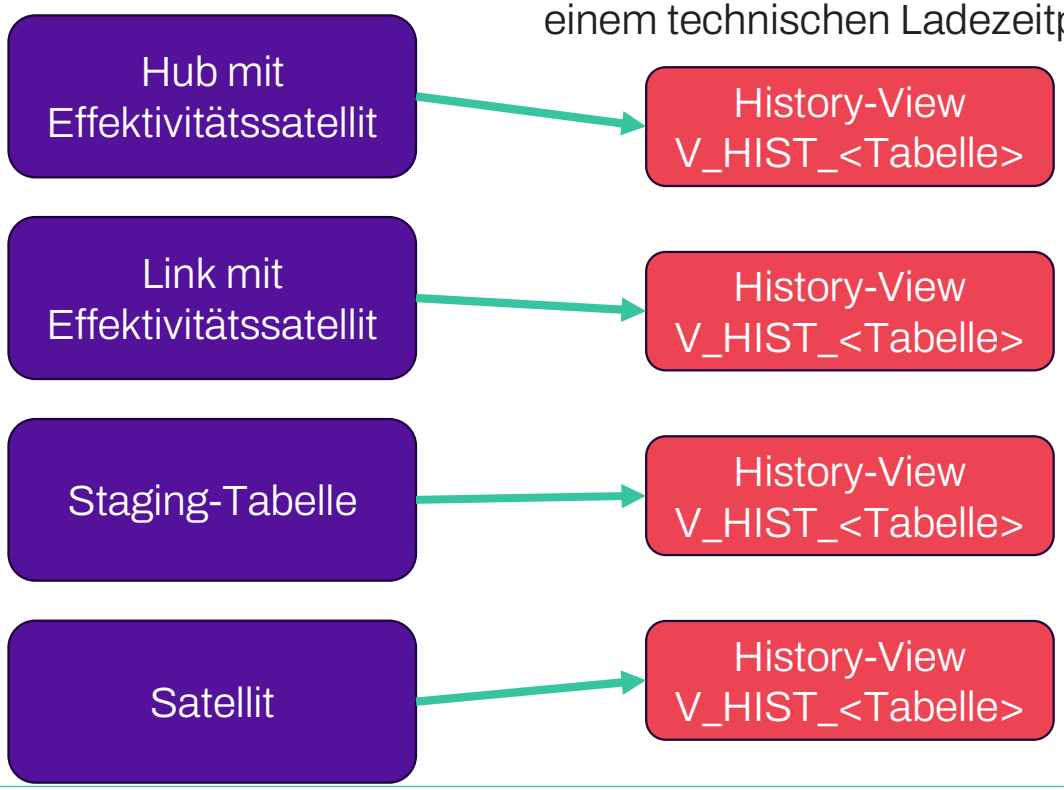
Dieses Feld enthält den Beginn der fachlichen Gültigkeit

Datenabfrage in DataVault

einfache Abfrage einzelner Tabellen

jede Tabelle hat History-View

zeigt eindimensionale fachliche Historie zu einem technischen Ladezeitpunkt an



Datenmodellierer

» setze technischen Ladezeitpunkt für DB-Session

```
call
setActLoadTime('20231101000000.000');
```

» Abfrage History-View

```
select * from V_Hist_TabelleA;

select * from V_Hist_TabelleB;
```

zeigt eindimensionale fachliche Historie zum gewünschten technischen Zeitpunkt

einfache Joins von Tabellen

Modellierung im DVG

- nur Angabe, welche Hubs, Links und Satelliten im Join enthalten sind
- Joins kombinieren die Historie der beteiligten Tabellen oder joinen zu einem fachlichen Zeitpunkt
- unterstützt durch Templates

Umsetzung in der Datenbank

- Generator erzeugt Views in DB
- Kaskade von Views (für jeden Join Schritt)
- bitemporale Historie automatisch berücksichtigt

Verwendung im DVG

- BusinessVault Befüllung: Input-Parameter für Geschäftsregeln
- Data-Quality Checks
- Befüllung Datamart

```
view TM_ViewTF0009_4 [VTF0009_4] simple # "testet gesetztes valid"
    from hub TM_RawVaultDemo.TM_HubMitglied alias MITGLIED # "der Einstiegshub" {
        satellites {
            TM_SatMitgliedsstammdaten alias mitgliedsdaten;
        }
        joins {
            link TM_RawVaultDemo.TM_LinkBuchMitglied alias LBMG # "" {
                joins {
                    hub TM_RawVaultDemo.TM_HubBuch alias BUCH # "" {
                        satellites {
                            TM_SatBuchdaten alias buchdaten;
                        }
                        joins {
                            link TM_RawVaultDemo.TM_LinkBuchStichwortN2M alias LBST # "" {
                                joins {
                                    hub TM_RawVaultDemo.TM_HubStichwort alias STW # "" {
                                        satellites {
                                            TM_SatStichwortdaten alias STW_STICHWORT;
                                        }
                                    }
                                    on valid buchdaten.Erscheinungsdatum
                                }
                            }
                        }
                    }
                }
            }
        }
    }
    where buchdaten.Testfall = 'TF0009'
    attributes {
        MITGLIED.Mitgliedsnummer alias Mitgliedsnummer primaryKey;
        mitgliedsdaten.Nachname alias Nachname any_value;
        BUCH.BuchID alias BuchID any_value;
        buchdaten.Preis alias Preis any_value;
        STW.StichwortID alias StichwortID any_value;
        buchdaten.Erscheinungsdatum alias Erscheinungsdatum any_value;
        STW_STICHWORT.Stichwort alias Stichwort any_value;
    }
    ;
```

Stichworte mit Stand des Erscheinungsdatums des Buches joinen

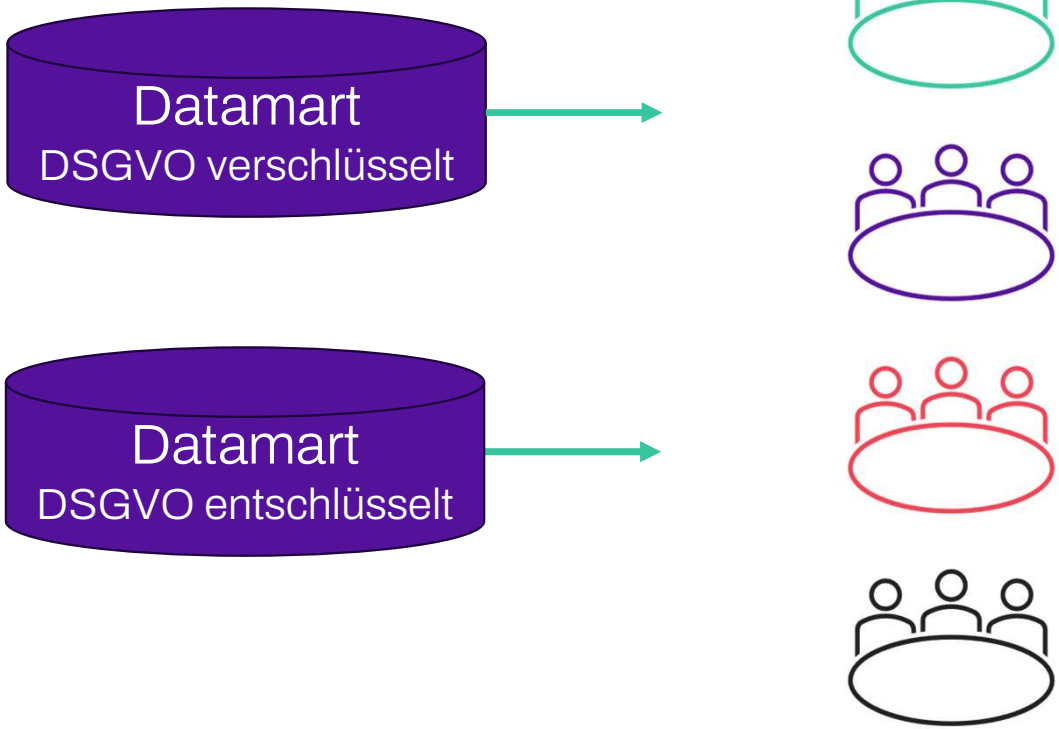
Datenbereitstellung im Datamart

individuelle Datamartzugriffe

nicht jeder will / muss / darf alles sehen

physische Tabellen

logische Zugriffsschicht pro Zielgruppe



Konfigurations-möglichkeiten

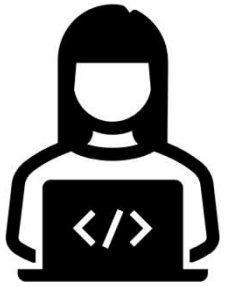
- Welche Datamarts sind sichtbar?
- Welche DSGVO Daten sind entschlüsselt?
- Art der Historie technisch
  - vollständig
  - Zeitraster (z.B. Ultimos)
  - nur aktuellste
- Art der Historie fachlich
  - vollständig
  - Zeitraster (z.B. Ultimos)
  - nur aktuellste
- beliebig kombinierbar

=> reduzierte Komplexität für Standard-User

- Struktur ist immer gleich, nur Inhalte unterscheiden sich

=> Wechsel zwischen Zugriffschichten möglich

individuelle Datamartzugriffe - Konfiguration



Datenmodellierer

- konfiguriert Zugriffsschichten
- bei Bedarf Anpassung / Erweiterung
- minimale Eingaben

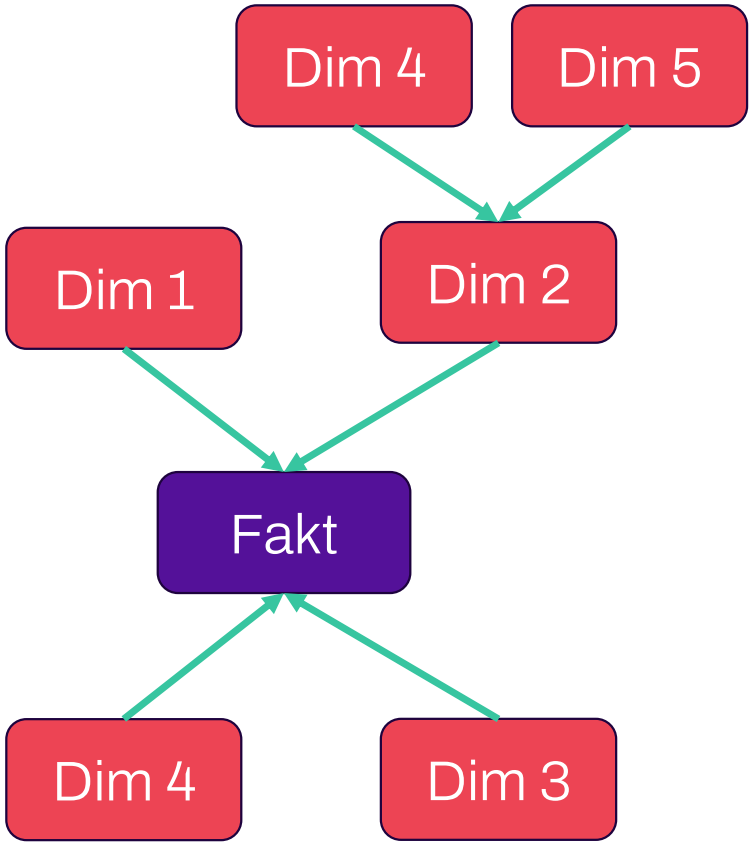
```
datamartAccessModul Demo # "Anlegen von verschiedenen Zugriffsschemata für Datamarts" {
    dbschema dsgvo_all # "alle Datamarts und alle DSGVO Rechte, komplette Historie" {
        dbrole dsgvo_all;
        relationModule datamarts;
        dsgvo Dsgvo.Produktdaten;
        technicalHistory all;
        validHistory all;
    }

    dbschema DemoSingle # "nur ein verschlüsselter Datamart,
                           nur eine technische und fachliche History" {
        dbrole demorolle;
        relation datamarts.DimProdukt;
        technicalHistory latest;
        validHistory latest;
    }

    dbschema DemoUltimo # "alle Datamarts verschlüsselt, nur aktuelle technische History
                           und fachliche Historie mit Ultimowerten" {
        dbrole DM1;
        relationModule datamarts;
        technicalHistory latest;
        validHistory V_ULTIMO;
    }
}
```

einfache Joins im Datamart

sehr einfache Endanwendersicht



Historie

Datamarts haben ein Feld TGA (technisch gültig am)

TODO für Anwender

- Auswahl der gewünschten technischen Historie (TGA=...) [1)]
- Auswahl aller gewünschten fachlichen historischen Daten [1)]
- **nur in der Fakten-Tabelle !!!!**

Joins

- alle Joins erfolgen mit Hilfe von Feldern GLOBAL_MART_ID_<Tabellenname>
- Jede Tabelle hat ihre eigene Global_Mart_ID sowie die Global_Mart_ID‘s übergeordneter Dimensionen
- Die Global_Mart_ID berücksichtigt korrekte Businesskeys / die bitemporale Historie

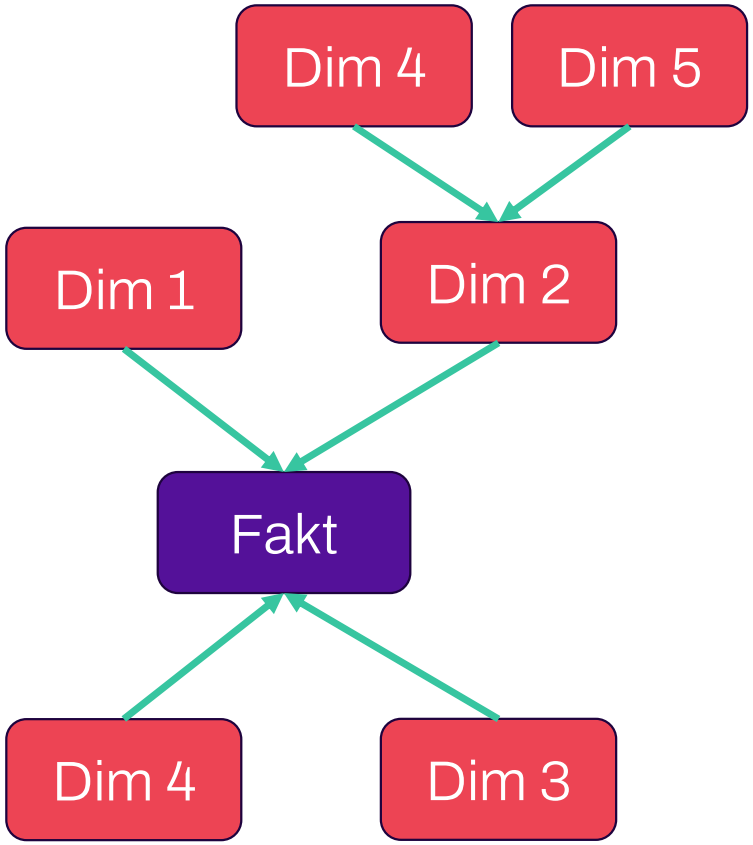
=> Join im BI-Tool für Anwender sehr einfach

- nur ein Feld (wichtig für Power-BI)
- join über gleichnamige Felder (häufig automatischer Standardjoin im BI-Tool)

1) nur, falls in der Access-Schicht mehrere technische / fachliche Historien enthalten sind

einfache Joins im Datamart

wenig Aufwand für Modellierer



- definiert Foreign Keys zwischen Fakten / Dimensionen
- legt in Datamartbefüllung fest, wie Global_Mart_ID berechnet werden
  - Kombination Historie Fakt/Dimension (Default: keine Eingabe)
  - Dimension zu einem bestimmten Zeitpunkt (z.B. Dimension „Versicherungsvertrag“ zum Fakt „Versicherungsschaden“ per Schadeneintrittsdatum)
  - aktuellste Werte der Dimension
  - ...

```
globalMartIdReferences {
    foreignKey DIM1 valid target.Zeitstempel;
    foreignKey DIM2 valid mostRecentValid();
}
```

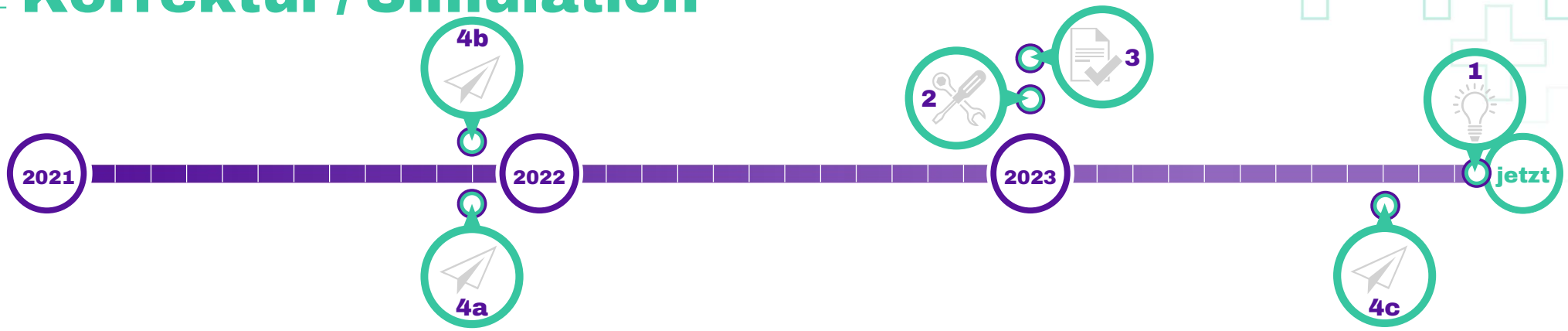
Zeitpunkt

aktuellste Version

Simulationen
Korrekturen

de fakto „Branches“ für Daten

Korrektur / Simulation



1 Standardbeladung

- enthält aktuellste verfügbare Daten
- basiert auf allen vorherigen Standardbeladungen

Ziel: aktuelle Daten bereitstellen

2 erste Korrektur Bilanzstichtag

- ergänzt Korrekturdaten, die nicht in folgende Standardbeladungen einfließen
- basiert auf allen Standardbeladungen bis 31.12.2023

Ziel: qualitativ geprüfte und korrigierte Daten bereitstellen

3 zweite Korrektur Bilanzstichtag

- ergänzt Korrekturdaten, die nicht in folgende Standard-beladungen einfließen
- basiert auf allen Standardbeladungen bis 31.12.2023 sowie erster Korrekturbeladung

4 Simulationen mit geänderter Ruleengine

- ergänzen Simulationsdaten, die in keine andere Beladung einfließen
- basiert auf allen vorherigen Standardbeladungen

Ziel: Test von Änderungen ohne Einfluss auf aktuelle Daten

Ziel: What if Analysen (z.B. Stresstest)

Umsetzung Korrekturen / Simulationen

Nutzung bitemporale Historie

- Einführung eines Feldes TGB (technisch gültig bis)
- Daten sind ab diesem Zeitpunkt technisch ungültig und werden nicht mehr berücksichtigt
- Simulationen / Korrekturen nutzen Millisekunden [1)] des Ladezeitpunktes

Ladeart	technisch gültig von TGV	technisch gültig bis TGB
Standard	Ladezeitpunkt.000	31.12.9999 23:59:59.999
Korrektur	Ladezeitpunkt.<lfdNr>	Ladezeitpunkt + 1 Sekunde
Simulation	Ladezeitpunkt.<lfdNr>	Ladezeitpunkt + <lfdNr> + 1 MillisekundeSekunde

mögliche Abfragen

Technischer Abfragezeitpunkt	Ergebnis
31.12.9999 23:59:59.000	technisch aktuellste Standardbeladung
31.12.2020 00:00:00.000	Standardbeladung per 31.12.2020
17.04.2020 12:30:00.000	Standardbeladung per 17.04.2020 12:30 Uhr
31.12.2020 00:00:00.999	aktuellste Korrekturladung per 31.12.2020
17.04.2020 12:30:00.008	exakt die 8. Korrektur-/ Simulationsbeladung per 17.04.2020 12:30 Uhr

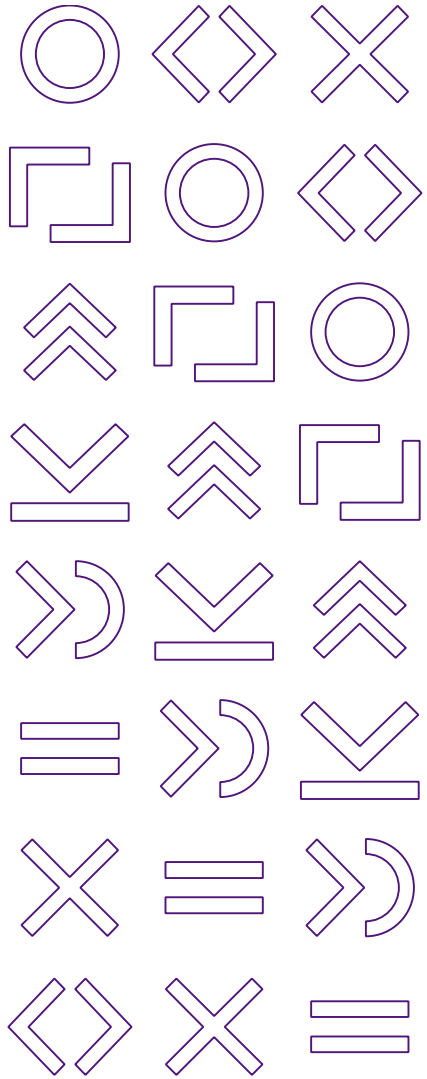
1) daher sind maximal 999 Simulationen / Korrekturen pro Ladezeitpunkt möglich

Anwendungsfälle Simulationen / Korrekturen

- What-If Analysen
- Testen von geänderten Geschäftsregeln
- Testen von Hot-Fixes
- Korrekturen für die Konzernbilanz
- von Aufsichtsbehörden geforderte Stresstests
- Korrektur von fehlerhaften Anlieferungen

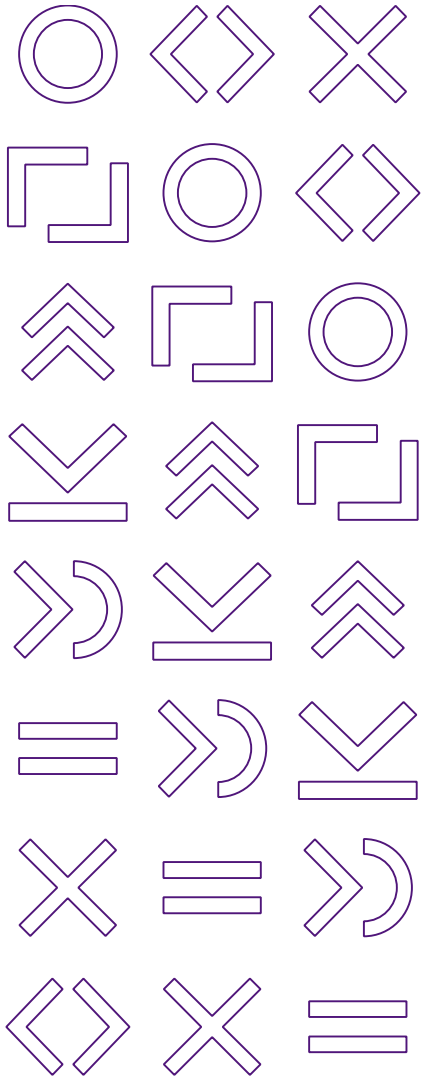
sollte kombiniert werden mit folgenden DVG-Features

- Sandboxing (parallele Installation mehrerer Modelle / Varianten Geschäftsregeln auf gleichen Ausgangsdaten)
- automatisierter Test (detaillierter Vergleich zweier Datenstände)



Fazit

- bitemporale Historie ermöglicht gleichzeitig:
  - unveränderte Wiederholung aller Abfragen
  - rückwirkende fachliche Korrekturen
  - Abfrage fachlich aktueller Daten
- Modellierungsaufwand im DVG ist minimal
- kein Programmierungsaufwand
- geringe Komplexität für Abfragen durch BI-Tools



crossnative. Wir machen digital genial.