Twine & Co. auf Data Vault – Umsetzung & Performance

27. Mai 2021

Vorstellung

- Günther Herzog
  - DWH Consultant seit 2005
  - Guenther.Herzog@dwh-consult.de

- Sebastian Flucke
  - BI Consultant seit 1995
  - SFlucke@online.de

Inhalt

- Fachliche Aufgabenstellungen bzgl. temporaler Join(t)s
  - Satelliten/Faktentabelle mit Statusänderungen
  - Dimensionsanbindung SCD2
  - Anbindung Satellit an Hub/Link in DataVault
  - Temporale Kennzahlenberechnung
- Zugabe: Feiertagsthematik
  - Vollständig aufgefüllte Zeitdimension
  - Anzahl Arbeits-/Feiertage zwischen zwei beliebigen Zeitpunkten

Temporale Fragestellungen

- **As Was** - Zustand zum Entstehungszeitpunkt
- **As Is** - Aktueller Zustand
- **As Of** - Zustand zu einem beliebigen Zeitpunkt

- Ermitteln der verschiedenen Zeit-Aspekte
  - As Was: Persistieren beim Insert
  - As Is: Persistieren beim täglichen Aufbereiten (truncate-Insert/Update)
  - As Of: Abfrage mit zeitabhängigem Join (Between/CrossApply/Twine/...)

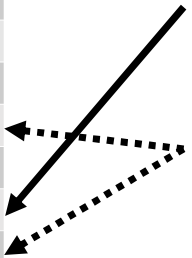
- Performanceoptimierung

Architektonische Anwendungsfälle

- Fakten mit zeitlicher Gültigkeit
- Dimensionsanbindung mit zeitlicher Gültigkeit/ SCD2
  - veränderliche Hierarchien
  - veränderliche Attribute
- Data Vault: Hub/ Link mit zeitabhängigem Satelliten

Satellit/Faktentabelle, die Statusänderungen repräsentiert I

ORDERS_STATUS – FAKT/Satellit		*(optional)*	
ORDER_ID	**TimeStamp_FROM**	***TimeStamp_To***	**ORDER_STATE**
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-03 09:51:31.000	*2011-01-05 05:50:46.000*	0
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-05 05:50:47.000	*2011-01-05 19:43:55.000*	8
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-05 19:43:56.000	*2011-01-26 03:01:47.000*	8
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-26 03:01:48.000	*2011-01-26 03:31:03.000*	7
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-26 03:31:04.000	*9999-12-31 00:00:00.000*	10
F81FBA73ED2600D5544BB6F275B0E4A7	2008-09-12 14:24:12.000	*2011-06-14 03:23:32.000*	8
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 03:23:33.000	*2011-06-14 09:55:33.000*	15
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 09:55:34.000	*2011-06-14 12:40:21.000*	11
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 12:40:22.000	*2011-06-14 13:13:21.000*	12
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 13:13:22.000	*9999-12-31 00:00:00.000*	0



müsste bei jeder Beladung mindestens teilweise neu befüllt werden:
- für alle neuen Datensätze
- für alle alten Datensätze, die einen Nachfolger bekommen

as-of
Darstellung in der Gültigkeit zu einem bestimmten Zeitpunkt

- wie viele Orders heute im Status 10 -> 1 Stück
  (= as is, wegen „heute“)
- wie viele Orders am 2011-01-10 im Status 8 -> 2 Stück

Satellite/Faktentabelle, die Statusänderungen repräsentiert II

ORDERS - DIMENSION	
ID	**START_DATE**
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-03 10:25:11.000
F81FBA73ED2600D5544BB6F275B0E4A7	2008-09-12 14:24:12.000

ORDERS_STATUS - FAKT		*(optional)*	
ORDER_ID	**TimeStamp_FROM**	***TimeStamp_To***	**ORDER_STATE**
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-03 09:51:31.000	*2011-01-05 05:50:46.000*	0
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-05 05:50:47.000	*2011-01-05 19:43:55.000*	8
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-05 19:43:56.000	*2011-01-26 03:01:47.000*	8
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-26 03:01:48.000	*2011-01-26 03:31:03.000*	7
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-26 03:31:04.000	*9999-12-31 00:00:00.000*	10
F81FBA73ED2600D5544BB6F275B0E4A7	2008-09-12 14:24:12.000	*2011-06-14 03:23:32.000*	8
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 03:23:33.000	*2011-06-14 09:55:33.000*	15
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 09:55:34.000	*2011-06-14 12:40:21.000*	11
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 12:40:22.000	*2011-06-14 13:13:21.000*	12
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 13:13:22.000	*9999-12-31 00:00:00.000*	0

as-of
Darstellung in der Gültigkeit zu einem bestimmten Zeitpunkt

- wie viele Orders heute im Status 10 -> 1 Stück
  (= as is, wegen „heute“)
- wie viele Orders am 2011-01-10 im Status 8 -> 2 Stück

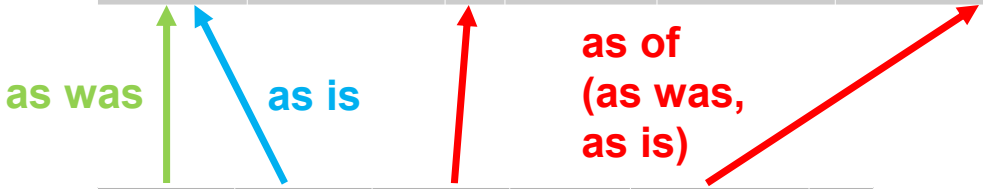
as-was

- Status zum Start-Datum -> JOIN zur Dimension
- Status 100 Tage nach Start-Datum

müsste bei jeder Beladung mindestens teilweise neu befüllt werden:
- für alle neuen Datensätze
- für alle alten Datensätze, die einen Nachfolger bekommen

Dimensionanbindung 1.a (veränderliche Gebietshierarchie)

ProduktDIM						
HistoryID	**Bezeichnung**	**ID**	**Bezirk**	**Region**	**von**	**bis**
11	Vertreter 1	1	DEF	Ost	13.04.2019	30.11.2020
22	Vertreter 1	1	AGX	Ost	01.12.2020	
33	Vertreter 2	2	AGX	Ost	01.03.2019	31.12.2020
44	Vertreter 2	2	AGX	Nord	01.01.2021	



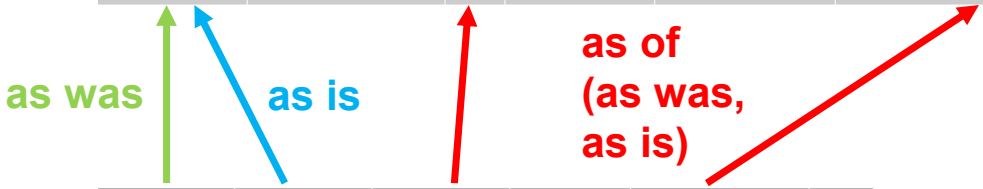
Auftrag FAKT					
HistoryID	**AsIsID**	**Produkt**	**Menge**	**Datum**	
11	22	1	2	05.11.2020	
44	44	2	6	23.04.2021	

(wird bei jeder Be-ladung neu befüllt)

- as-is
  Darstellung in der aktuellen Gültigkeit
- as-was
  Darstellung in der Gültigkeit zum Berichtszeitpunkt
- as-of
  Darstellung in der Gültigkeit zu einem bestimmten Zeitpunkt

Dimensionanbindung 1.b (veränderliche Produkthierarchie)

ProduktDIM						
HistoryID	**Bezeichnung**	**ID**	**Produkt-gruppe**	**Produkt-Hauptgruppe**	**von**	**bis**
11	Kaffee 250g	1	Powder	Food	13.04.2019	30.11.2020
22	Kaffee 250g	1	Powder	Beverages	01.12.2020	
33	Milch	2	Powder	Beverages	01.03.2019	31.12.2020
44	Milch	2	Liquid	Beverages	01.01.2021	



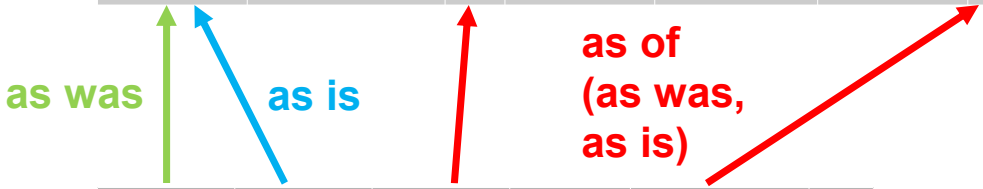
Auftrag FAKT					
HistoryID	**AsIsID**	**Produkt**	**Menge**	**Datum**	
11	22	1	2	05.11.2020	
44	44	2	6	23.04.2021	

(wird bei jeder Be-ladung neu befüllt)

- as-is
  Darstellung in der aktuellen Gültigkeit
- as-was
  Darstellung in der Gültigkeit zum Berichtszeitpunkt
- as-of
  Darstellung in der Gültigkeit zu einem bestimmten Zeitpunkt

Dimensionanbindung 2 (veränderliche Attribute)

ProduktDIM						
HistoryID	**Bezeichnung**	**ID**	**Messgröße**	**Basismenge**	**von**	**bis**
11	Kaffee 250g	1	Stück	1	13.04.2019	30.11.2020
22	Il Magnifico 250g	1	Stück	1	01.12.2020	
33	Milch	2	Liter	1	01.03.2019	31.12.2020
44	Milch	2	Liter	0,5	01.01.2021	



Auftrag FAKT					
HistoryID	**AsIsID**	**Produkt**	**Menge**	**Datum**	
11	22	1	2	05.11.2020	
44	44	2	6	23.04.2021	

(wird bei jeder Be-ladung neu befüllt)

- as-is
  Darstellung in der aktuellen Gültigkeit
- as-was
  Darstellung in der Gültigkeit zum Berichtszeitpunkt
- as-of
  Darstellung in der Gültigkeit zu einem bestimmten Zeitpunkt
- ggf. Einfluß auf Kennzahlen-Berechnungen

Gemeinsamkeit

- temporaler Join (verschiedene Anwendungsfälle)
- technische Umsetzungsmöglichkeiten
  - Holzhammer:
    Datenvervielfältigung für jeden erdenklichen Zeitpunkt
  - Between-Join
  - Beetwen-Join mit To-Date durch LEAD() erzeugt
  - CrossApply-Cheat
  - Twining-Trick (Lars Rönnbäck)
  - Twining ggf. optimiert

Holzhammer: Datenvervielfältigung für jeden erdenklichen Zeitpunkt

- in den meisten Fällen weder sinnvoll noch machbar…
  - wegen Datenmenge (Speicherplatz)
  - und Aufbereitungszeit (RAM/ Prozessor)
- …aber es gibt sinnvolle Ausnahmen (z.B. Kalender-Dimension)

Between-Join

```
SELECT *
INTO #temp_From_To_Between
FROM [DDVUG].[AA_ORDERS_STATE_FROM_TO] xOrderState
INNER JOIN [DDVUG].[AA_ORDERS_FROM_TO] xOrder
      ON xOrderState.ORDER_ID = xOrder.ID
      AND DATEADD( DAY, 100, xOrder.START_DATE )
            BETWEEN xOrderState.TimeStamp_FROM
            AND xOrderState.TimeStamp_TO
```

Between-Join mit LEAD()

```
; WITH
__prep AS
(
SELECTxOrderState.ORDER_ID
    ,xOrderState.TimeStamp_FROM
    ,LEAD( xOrderState.TimeStamp_FROM, 1 )
       OVER( PARTITION BY xOrderState.ORDER_ID
             ORDER BY xOrderState.TimeStamp_FROM ) AS TimeStamp_TO
    ,xOrderState.ORDER_STATE
FROM[DDVUG].[MM_ORDERS_STATE_10k] xOrderState
)
SELECT *
INTO #temp_From_To_Between
FROM __prep xOrderState
INNER JOIN [DDVUG].[MM_ORDERS_10k] xOrder
       ON xOrderState.ORDER_ID = xOrder.ID
              AND DATEADD( DAY, 100, xOrder.START_DATE )
                     BETWEEN xOrderState.TimeStamp_FROM
                     AND xOrderState.TimeStamp_TO
```

CrossApply-Cheat

```
SELECT *
INTO #temp_From_To_CrossApply
FROM [DDVUG].[II_ORDERS_FROM_TO_10k] xOrder
CROSS APPLY
(
SELECT TOP( 1 ) xOrderState.TimeStamp_FROM
FROM [DDVUG].[II_ORDERS_STATE_FROM_TO_10k] xOrderState
WHERE DATEADD( DAY, 100, xOrder.START_DATE ) <
xOrderState.TimeStamp_TO
AND xOrderState.ORDER_ID = xOrder.ID
ORDER BY DATEADD( DAY, 100, xOrder.START_DATE ) DESC
) a
```

Twining-Trick Step 1 (Lars Rönnbäck)

http://www.anchormodeling.com/♫-lets-twine-again-♫

Twining-Trick Step 2 (Lars Rönnbäck)

ORDERS_STATUS - FAKT			
ORDER_ID	**TimePoint (=FROM)**		**TimeLine**
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-03 09:51:31.000		A
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-05 05:50:47.000		A
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-26 03:01:48.000		A
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-26 03:31:04.000		A
F81FBA73ED2600D5544BB6F275B0E4A7	2008-09-12 14:24:12.000		A
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 03:23:33.000		A
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 09:55:34.000		A
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 13:13:22.000		A

http://www.anchormodeling.com/♫-lets-twine-again-♫

Twining-Trick Step 3 (Lars Rönnbäck)

ORDERS_STATUS - FAKT			
ORDER_ID	**TimePoint (=FROM)**		**TimeLine**
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-03 09:51:31.000		A
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-05 05:50:47.000		A
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-26 03:01:48.000		A
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-26 03:31:04.000		A
F81FBA73ED2600D5544BB6F275B0E4A7	2008-09-12 14:24:12.000		A
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 03:23:33.000		A
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 09:55:34.000		A
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 13:13:22.000		A

ORDER_ID	**Abfrage-Zeitpunkt**		**TimeLine**
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-05 19:43:56.000		B
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 12:40:22.000		B

http://www.anchormodeling.com/♫-lets-twine-again-♫

Twining-Trick Step 4 (Lars Rönnbäck)

```
SELECT
    ORDER_ID AS Id
    , TimeStamp_FROM AS TimePoint
    , 'A' AS TimeLine
    FROM [DDVUG].[MM_ORDERS_STATE_10k]
UNION ALL
SELECT
    ID AS Id
    , DATEADD( DAY, 100, START_DATE ) AS TimePoint
    , 'B' AS TimeLine
    FROM [DDVUG].[MM_ORDERS_10k]
```

ORDERS_STATUS - FAKT			
ORDER_ID	**TimePoint (=FROM)**		**TimeLine**
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-03 09:51:31.000		A
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-05 05:50:47.000		A
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-05 19:43:56.000		B
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-26 03:01:48.000		A
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-26 03:31:04.000		A
F81FBA73ED2600D5544BB6F275B0E4A7	2008-09-12 14:24:12.000		A
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 03:23:33.000		A
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 09:55:34.000		A
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 12:40:22.000		B
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 13:13:22.000		A

http://www.anchormodeling.com/♫-lets-twine-again-♫

Twining-Trick Step 5 (Lars Rönnbäck)

```
WITH
__Twine AS
(
SELECT
    ORDER_ID AS Id
    , TimeStamp_FROM AS TimePoint
    , 'A' AS TimeLine
    FROM [DDVUG].[MM_ORDERS_STATE_10k]
UNION ALL
SELECT
    ID AS Id
    , DATEADD( DAY, 100, START_DATE ) AS TimePoint
    , 'B' AS TimeLine
    FROM [DDVUG].[MM_ORDERS_10k]
)
,
__InEffect AS
(
SELECT
    Id
   ,TimeLine
   ,TimePoint
   , MAX( CASE WHEN __Twine.TimeLine = 'A' THEN __Twine.TimePoint END ) OVER( PARTITION BY Id ORDER BY TimePoint ) AS ValidSince
FROM __Twine
)
SELECT Id
      ,TimeLine
      ,TimePoint
      ,ValidSince
INTO #result
FROM __InEffect
```

ORDERS_STATUS - FAKT			
ORDER_ID	**TimePoint (=FROM)**	***ValidSince***	**TimeLine**
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-03 09:51:31.000	2011-01-03 09:51:31.000	A
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-05 05:50:47.000	2011-01-05 05:50:47.000	A
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-05 19:43:56.000	2011-01-05 05:50:47.000	B
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-26 03:01:48.000	2011-01-26 03:01:48.000	A
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-26 03:31:04.000	2011-01-26 03:31:04.000	A
F81FBA73ED2600D5544BB6F275B0E4A7	2008-09-12 14:24:12.000	2008-09-12 14:24:12.000	A
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 03:23:33.000	2011-06-14 03:23:33.000	A
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 09:55:34.000	2011-06-14 09:55:34.000	A
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 12:40:22.000	2011-06-14 09:55:34.000	B
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 13:13:22.000	2011-06-14 13:13:22.000	A

http://www.anchormodeling.com/♫-lets-twine-again-♫

Twining-Trick Step 6 (Lars Rönnbäck)

ORDERS_STATUS - FAKT			
ORDER_ID	**TimePoint (=FROM)**	***ValidSince***	**TimeLine**
D6EFFF283CAF55B2EAF7366F736CC588	2011-01-05 19:43:56.000	2011-01-05 05:50:47.000	B
F81FBA73ED2600D5544BB6F275B0E4A7	2011-06-14 12:40:22.000	2011-06-14 09:55:34.000	B

```
WITH
__Twine AS
(
SELECT
    ORDER_ID AS Id
    , TimeStamp_FROM AS TimePoint
    , 'A' AS TimeLine
    FROM [DDVUG].[MM_ORDERS_STATE_10k]
UNION ALL
SELECT
    ID AS Id
    , DATEADD( DAY, 100, START_DATE ) AS TimePoint
    , 'B' AS TimeLine
    FROM [DDVUG].[MM_ORDERS_10k]
)
,
__InEffect AS
(
SELECT
    Id
   ,TimeLine
   ,TimePoint
   , MAX( CASE WHEN __Twine.TimeLine = 'A' THEN __Twine.TimePoint END ) OVER( PARTITION BY Id ORDER BY TimePoint ) AS ValidSince
FROM __Twine
)
SELECT Id
      ,TimeLine
      ,TimePoint
      ,ValidSince
INTO #result
FROM __InEffect
WHERE TimeLine = 'B'
```

http://www.anchormodeling.com/♫-lets-twine-again-♫

Vergleichende Betrachtung

Menge	Variante	ValidTo persist.	Laufzeit (s)	CPU (s)	log. Reads	Bemerkungen
5 / 210 Mio	between	x	43,8	134,0	~1.745.000	
5 / 210 Mio	lead/ between		63,3	774,9	~1.510.000	
5 / 210 Mio	twine		95,0	752,6	~1.520.000	
5 / 210 Mio	cross apply	x	187,7	155,4	~43.130.000	aufwendige WorkTable
10 / 421 Tsd	between	x	2,7	1,2	3.460	
10 / 421 Tsd	lead/ between		2,7	3,3	3.036	
10 / 421 Tsd	twine		1,0	1,6	3.036	
10 / 421 Tsd	cross apply	x	3,1	2,3	1.291.000	meiste reads auf einer WorkTable

Einflußfaktoren

- Datenmengen
- Ressourcen-Ausstattung des SQL-Servers
- SQL-Server-Version
- Indizies (inkl. deren Kosten)
  - nicht alle Indizies haben nachvollziehbare/ gewünschte Wirkung
- ToDate persistieren - oder eben nicht!?
-

Einflußfaktoren

- Datenmengen
- Ressourcen-Ausstattung des SQL-Servers
- SQL-Server-Version
- Indizies (inkl. deren Kosten)
  - nicht alle Indizies haben nachvollziehbare/ gewünschte Wirkung
- ToDate persistieren - oder eben nicht!?

- also wie immer: **It depends ;-)))**

Resümee

- beste Laufzeit im Test-Szenario bei großen Datenmengen:
  - BETWEEN mit persistiertem ToDate
- ohne persistiertes ToDate
  - immer noch BETWEEN...
  - ...aber deutlich mehr CPU-Last
- CROSS APPLY
  - deutlich mehr I/O
-

Resümee

- beste Laufzeit im Test-Szenario bei großen Datenmengen:
  - BETWEEN mit persistiertem ToDate
- ohne persistiertes ToDate
  - immer noch BETWEEN...
  - ...aber deutlich mehr CPU-Last
- CROSS APPLY
  - deutlich mehr I/O

- also auch hier wie immer: **It depends ;-)))**

Feiertagsthematiken

- Aufgabenstellung
  - Berechnung der Anzahl Arbeitstage/ Feiertage zwischen zwei beliebigen Zeitpunkten
- Beispiele
  - wieviele Arbeitstage liegen zwischen dem 1. März 2007 und dem 28. Februar 2008?
  - wieviele arbeitsfreie Tage gibt es im Jahr 2016?

Feiertagsthematik - Lösung

- erweiterte Zeitdimension
  - Kennzeichnung der Tages-Typen
  - Feiertage, Samstage, Sonntage
    - Berechnung der schwimmenden Feiertage
- kumulierte Anzahl diverser Typen von Tagen
- Berechnung zur Ermittlung der relevanten Differenz
  - Ermitteln der Anfangs- und Endwerte (genau zwei Seeks)
  - schlaues Verformeln der Anfangs- und Endwerte

Noch Fragen???

- Fachliche Aufgabenstellungen bzgl. temporaler Join(t)s
  - Satelliten/Faktentabelle mit Statusänderungen
  - Dimensionsanbindung SCD2
  - Anbindung Satellit an Hub/Link in DataVault
  - Temporale Kennzahlenberechnung
- Zugabe: Feiertagsthematik
  - Vollständig aufgefüllte Zeitdimension
  - Anzahl Arbeits-/Feiertage zwischen zwei beliebigen Zeitpunkten

Danke für Eure Aufmerksamkeit!

27. Mai 2021