SAP HANA 1.0
Modeling Fundamentals and best practices for optimal data models

Customer Solution Adoption (CSA)
Customer Solution Adoption Mission

Our people make innovation adoption a beautiful experience for SAP customers

Our Critical Success Factors

Readiness for take-off with our customers

Referenceability by taking end-to-end accountability for early adopters

Repeatability for rapid and reliable deployments by SAP & our ecosystem
Disclaimer

This presentation outlines our general product direction and should not be relied on in making a purchase decision. This presentation is not subject to your license agreement or any other agreement with SAP. SAP has no obligation to pursue any course of business outlined in this presentation or to develop or release any functionality mentioned in this presentation. This presentation and SAP's strategy and possible future developments are subject to change and may be changed by SAP at any time for any reason without notice. This document is provided without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement. SAP assumes no responsibility for errors or omissions in this document, except if such damages were caused by SAP intentionally or grossly negligent.
Agenda

Modeling Fundamentals & Best Practices

- Analytical & Calculation View Fundamentals
- Design time vs. Runtime objects
- General Modeling Principles
- Restricted Measures vs Logical Partitioning
- Calculated Attributes vs Calculated Columns
- Calculation Before Aggregation
- SQL vs Calculation Engine Functions
SAP HANA Modeling Fundamentals

- **Views**
  - Attribute Views (Dimensions, Time, Hierarchy, Derived)
  - Analytical Views (Facts/Star Schemas)
  - Calculation Views (Composite views, Modeled or Script)
  - Transportable design time artifacts stored in the repository
  - Database objects (Column store views) are generated from these development artifacts

- **Language**
  - Main procedural language of the SAP HANA database
  - Push data intensive operations into the database
  - Utilized in Calculation Views and Procedures
  - Read-Only procedures (Calculation views, highly optimized, potential of parallelism)
  - Read-Write procedures (cursors, loops, conditions, transactions)
SAP HANA Modeling Best Practices

Calculation View

Analytical View

Attribute View

Column Table
Agenda

• Modeling Fundamentals & Best Practices

Analytical & Calculation View Fundamentals

• Design time vs. Runtime objects
• General Modeling Principles
• Restricted Measures vs Logical Partitioning
• Calculated Attributes vs Calculated Columns
• Calculation Before Aggregation
• SQL vs Calculation Engine Functions
SAP HANA Analytical View

- Star schema consist of one fact table containing the key figures
- Dimensions describe the key figures and enrich the data
- Slicing and dicing is a feature whereby users can take out (slicing) a specific set of data and view (dicing) the slices from different viewpoints
- Cardinality in star schemas is generally N:1 fact to dimension
- Joins in star schemas are generally Left Outer Joins
- Analytical models are highly optimized for aggregating mass data
SAP HANA Calculation View

- Several options available
  - Use the Graphical Modeler
  - Write SQL Script and use CE Functions
  - Write SQL Script and using SQL
- Suggested option = Graphical Modeler
  - No SQL or SQL Script knowledge required
  - Built-in Union Constant support
- Graphical & SQL Script + CE Functions
  - Result in similar performance gains (e.g. Field pruning, Parallelization, join ommision)
- Standard SQL
  - Does not provide field pruning and can be less optimized. Useful for POCs and rapid prototyping
Agenda

- Modeling Fundamentals & Best Practices
- Analytical & Calculation View Fundamentals
- Design time vs. Runtime objects
  - General Modeling Principles
  - Restricted Measures vs Logical Partitioning
  - Calculated Attributes vs Calculated Columns
  - Calculation Before Aggregation
  - SQL vs Calculation Engine Functions
SAP HANA Design Time vs. Runtime Time Objects

- Activation of models creates an executable, optimized Column (database) view of the model
- Front end’s queries column views and as a result a specific execution plan is instantiated based on fields requested
- Un-used fields are pruned; un-used tables joins are omitted; filters are pushed down
- Attributes are retrieved and Measures are calculated in parallel

```
SELECT PERIO, VKORG, SUM(CM1)
```
Agenda

- Modeling Fundamentals & Best Practices
- Analytical & Calculation View Fundamentals
- Design time vs. Runtime objects

General Modeling Principles

- Restricted Measures vs Logical Partitioning
- Calculated Attributes vs Calculated Columns
- Calculation Before Aggregation
- SQL vs Calculation Engine Functions
SAP HANA General Modeling Principles

- Avoid transfer data of large resultsets between the HANA DB and client application.
- Do calculation after aggregation.
- Avoid Complex expressions (IF, CASE, ...)
- Reduce data transfer between views.
- Aggregate data records (e.g., using GROUP BY, reducing columns).
- Join on Key Columns or Indexed Columns.
- Avoid calculations before aggregation on line item level.
- Filter data amount as early as possible in the lower layers (CONSTRAINTS, WHERE Clause, Analytical Privileges.)
SAP HANA Basic Modeling Principle (Aggregate, filter, pushdown)
Agenda

• Modeling Fundamentals & Best Practices
• Analytical & Calculation View Fundamentals
• Design time vs. Runtime objects
• General Modeling Principles

Restricted Measures vs Logical Partitioning
• Calculated Attributes vs Calculated Columns
• Calculation Before Aggregation
• SQL vs Calculation Engine Functions
Restricted Measures vs Logical Partitioning

Analytical View
• Define 3 Restricted Measures - once for each PLANT (A, B & C) within a single Analytical View

Multiple Analytical Views + 1 Calculation View
• Create 3 Analytical views – one for each PLANT
• Define a design time filter (PLANT) on each Analytical View
• Create a Calculation view that reads each Analytical view
• Combine the results using Union Constants
Agenda

- Modeling Fundamentals & Best Practices
- Analytical & Calculation View Fundamentals
- Design time vs. Runtime objects
- General Modeling Principles
- Restricted Measures vs Logical Partitioning

**Calculated Attributes vs Calculated Columns**

- Calculation Before Aggregation
- SQL vs Calculation Engine Functions
Calculated Attributes vs Calculated Columns

**Analytical View**
- Define 4 Calculated Attributes
  - Current Year \( \text{midstr(now(),1,4)} \)
  - Previous Year \( \text{"current_year"}-1 \)
  - Earliest Year \( \text{"previous_year"}-1 \)
  - Current Key \( \text{if("cur"="year",1,if("prev"="year",2,if("earl"="year",3),-1))} \)

**Calculation View**
- Re-use the same Analytical View 3x
- Define a Current Year Key filter in each Projection Node
- Combine results using Union Constants

**Analytical View + Calculation View**
- Create a Analytical View without any Calculated Attributes, instead define Calculated Columns in a Calculation View
- Create a Calc. View reading from a single Analytical View
- Add one Projection Node above the Analytical View & add Calculated Columns
- Stack 3 Projection nodes and Filter by Current Year Key
- Combine results using Union Constants
Agenda

- Modeling Fundamentals & Best Practices
- Analytical & Calculation View Fundamentals
- Design time vs. Runtime objects
- General Modeling Principles
- Restricted Measures vs Logical Partitioning
- Calculated Attributes vs Calculated Columns

**Calculation Before Aggregation**

- SQL vs Calculation Engine Functions
Calculate Before Aggregation

Analytical View
- Define Calculated Measure threshold (i.e. NETWR > 99) using the Expression Editor
- Check Calculate Before Aggregation
- The result will either be 1 or 0 per row
- Use regular SUM aggregation to calculate count based on threshold calculation

Use with caution, row based processing significantly slower than set based processing
Agenda

- Modeling Fundamentals & Best Practices
- Analytical & Calculation View Fundamentals
- Design time vs. Runtime objects
- General Modeling Principles
- Restricted Measures vs Logical Partitioning
- Calculated Attributes vs Calculated Columns
- Calculation Before Aggregation

SQL vs Calculation Engine Functions
Calculation View – SQL vs CE Functions

Calculation Engine (CE) Functions
- Preferred over SQL
- Improved performance, can be optimized by the engine (i.e. field pruning & parallelized)

\[
\text{BEGIN} \\
\text{var\_out = SELECT} \\
\text{DATE\_SAP, VBELN, MATNR,} \\
\text{SUM("NETWR") AS NETWR,} \\
\text{SUM("KWMENG") AS KWMENG} \\
\text{FROM "SYS\_BIC"."demo/EXAMPLE\_2\_SET\_AN"} \\
\text{GROUP BY DATE\_SAP,VBELN, MATNR;} \\
\text{END}
\]

Even though only MATNR & NETWR are requested by the front end tool all fields will be retrieved from the table including all calculations will be done irrespective.

\[
\text{BEGIN} \\
\text{VAR\_OLAP = CE\_OLAP\_VIEW("SYS\_BIC"."demo/EXAMPLE\_2\_SET\_AN");} \\
\text{var\_out = CE\_PROJECTION (}:\text{VAR\_OLAP,} \\
\text{[ "DATE\_SAP", "VBELN", "MATNR", "NETWR", "KWMENG" ]} \\
\text{);} \\
\text{END}
\]

Only MATNR & NETWR will be retrieved from the database. Only NETWR will be calculated. Columns requested by the front end tool can be parallelized.
Thank You!

SAP Labs, LLC
SAP Customer Solution Adoption