LTI
Let's Solve

Celebrating 20 Years

SAP HANA internal training – Session 2

March 2018

A Larsen & Toubro Group Company

Working with nodes: Join, Union, Projection, Aggregation, Rank, Semantics Node

We have 6 different types of nodes in Graphical calculation views, these are

- **Join**: This node is used to join two source objects and pass the result to the next node. The join types can be inner, left outer, right outer and text join.
  - Note: We can only add two source objects to a join node.
- **Union**: This is used to perform union all operation between multiple sources. The source can be n number of objects.
- **Projection**: This is used to select columns, filter the data and create additional columns before we use it in next nodes like a union, aggregation and rank.
  - **Note**: We can only add one source objects in a Projection node.
- **Aggregation**: This is used to perform aggregation on specific columns based on the selected attributes.
- **Rank**: This is the exact replacement for RANK function in SQL. We can define the partition and order by clause based on the requirement. Rank Node also supports generation of Rank Columns based on Partition and Order By criteria.
- **Semantics:** The semantics provide meaning to attributes and measures of an information view.

Working with nodes: Join, Union, Projection, Aggregation, Rank, Semantics Node

Join

- The join nodes help limit the number of records or to combine records from both the data sources, so that they appears as one record in the query results.
- There are different types of joins supported in views:
- **Left Outer Join:** Performs a join starting with the left table and then matching with right table records.
- **Inner Join:** Performs a join if the join-predicate match between both the tables.
- **Right Outer Join:** Performs a join starting with the right table and then matching with left table records.
- **Full Outer Join:** Joins all data from both the tables.
- **Temporal Join:** Allows you to join the master data with the transaction data (fact table) based on the temporal column values from the transaction data and the time validity from the master data.
- **Text join:** Helps obtain language-specific data. It retrieves columns from a text table based on the user's session language.
- **Dynamic joins:** Improve the join execution process and help reduce the number of records that join view node process at run time. Dynamic joins enforces aggregation before executing the join

Working with nodes: Join, Union, Projection, Aggregation, Rank, Semantics Node

Union

- A union node combines multiple data sources, which can have multiple columns.
- You can manage the output of a union node by mapping the source columns to the output columns or by creating a target output column with constant values.
- For a source column that does not have a mapping with any of the output columns, you can create a target output column and map it to the unmapped source columns.
- You can also create a target column with constant values.
- For performance optimization, we can prune the data coming from union nodes.
- For pruning data in union nodes, you need to define the pruning configuration table
- Pruning configuration table holds the filter conditions that limit the result set when you execute a query on the union node.

Variables

Details

- Information views contain variables that are bound to specific attributes in an information view. Based on the attribute data, you can provide values to variables, while executing the calculation view.
- You assign variables to attributes in information views, for example, to filter the results. At runtime, you can provide values to variables by manually entering a value or by selecting them from the value help dialog.

Supported Variable types

Type	Description
Single Value	Use this to filter and view data based on a single attribute value.
Interval	Use this to filter and view a specific set of data. For example, to view the expenditure of a company from March to April.
Range	Use this to filter view data based on the conditions that involve operators such as "="(equal to), ">" (greater than) etc.

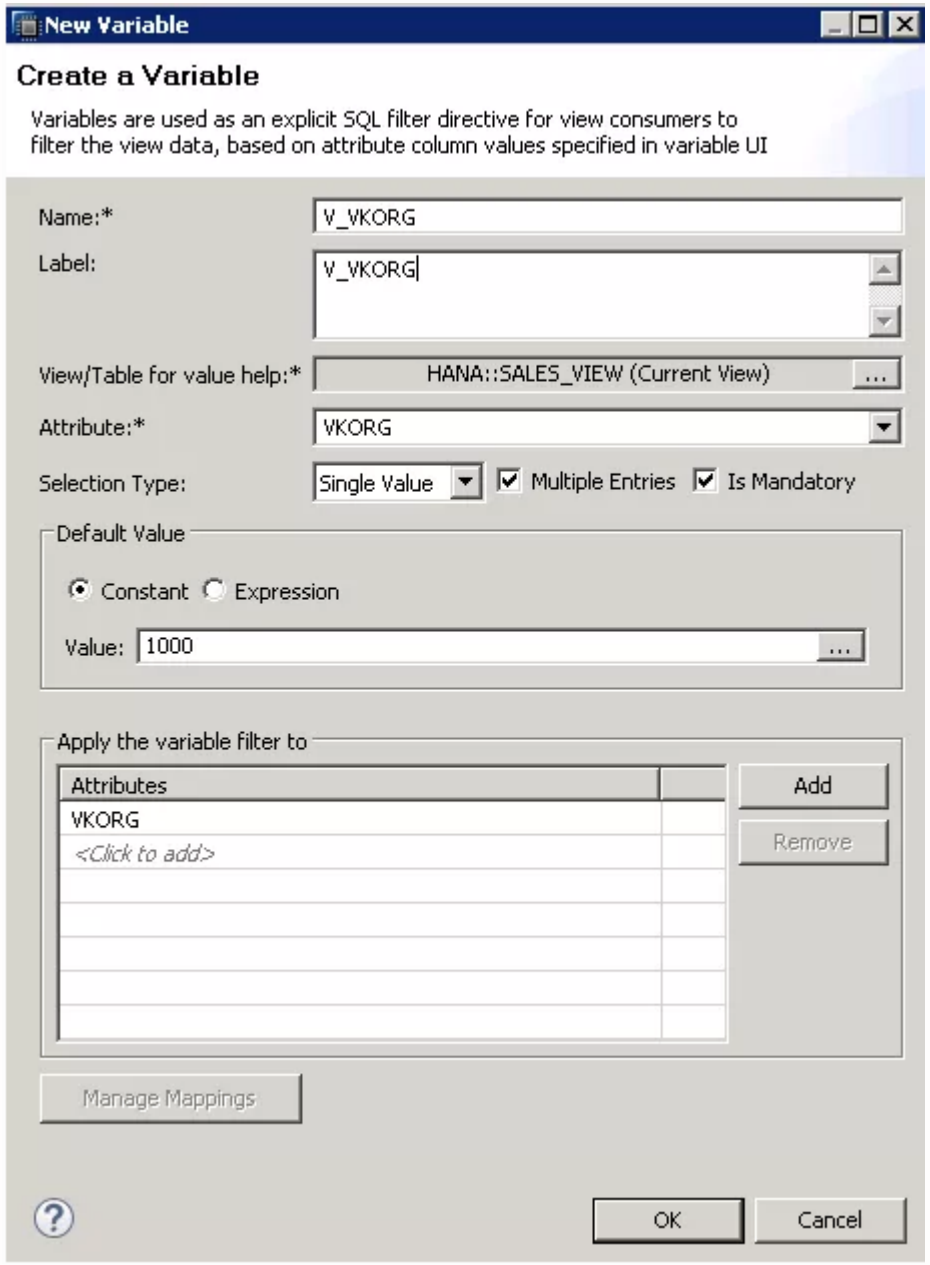
Variable Properties

Property	Description
Attribute	The value of this property specifies the attribute data that modeler uses to provide values in the value help at runtime.
Selection Type	The value of this property specifies the variable type used for creating the variable.
Multiple Entries	The value of this property specifies whether the variable is configured to support multiple values at runtime.

Variables

Example

To create a variable, click on the Semantics node and then select the Parameters/Variables Tab and then click on the green plus button to create a new SAP HANA variable. A pop shall appear where variable shall be defined.



Input Parameters

Details

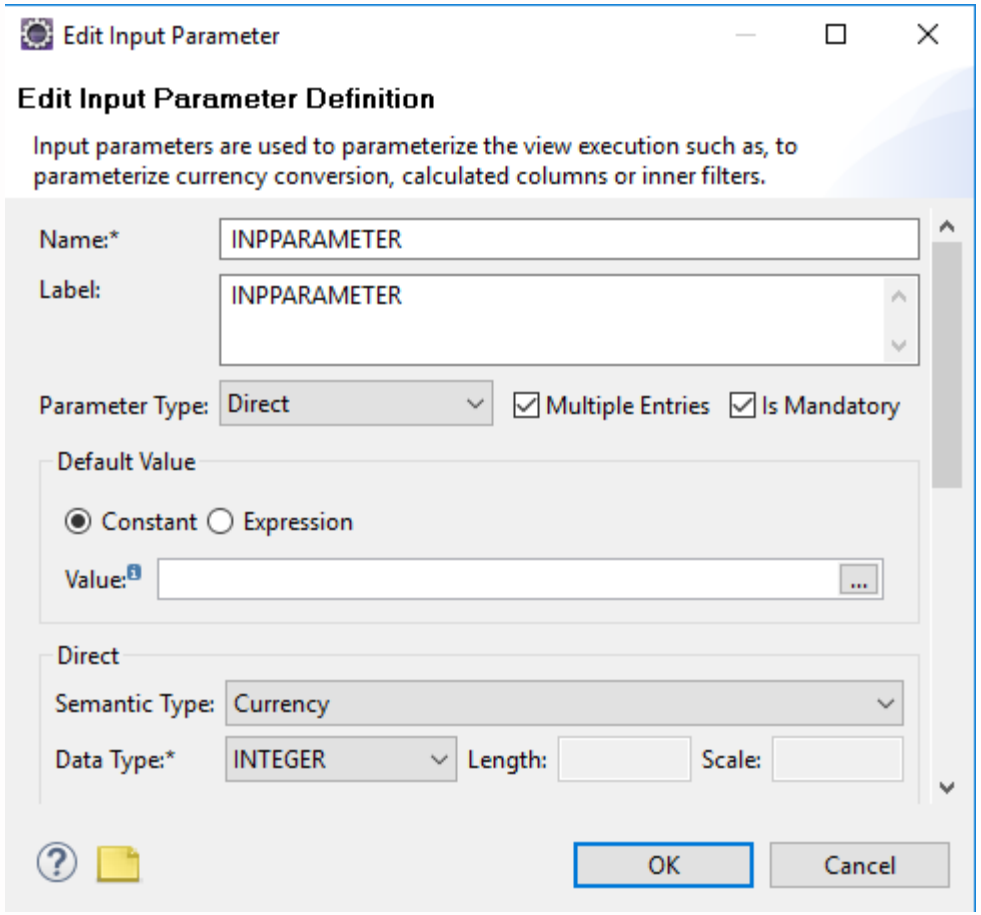
- Input parameters helps you parameterize information views and execute them based on the values you provide to the input parameters at query runtime. The engine considers input parameters as the PLACEHOLDER clause of the SQL statement.
- You create an input parameter at design time (while creating your information views), and provide value to the engine at runtime

Input Parameter types	
Type	**Description**
Column	At runtime, modeler provides a value help with attribute data.
Derived from table	At runtime, modeler uses the value from the table's return column
Direct	Specify the data type and length and scale of the input parameter value that you want to use at runtime.
Static List	At runtime, modeler provides a value help with the static list
Derived from Procedure/Scalar functions	At runtime, modeler uses the value returned from the procedure or scalar function

Input Parameters

Example

To create an input parameter, click on the Semantics node and then select the Parameters/Variables Tab and then click on the dropdown sign near green plus button to create a new SAP HANA input parameter. A pop shall appear where input parameter's properties shall be defined.



Input Parameter vs Variables

Variables	Input Parameter
Variables apply filter after execution of all nodes till the semantics (at the top level)	Input parameters can apply filters at any projection level.
Variables are bound to attributes/specific fields	Input parameter is independent of any field in the view.
Variables have a sole purpose of filtering data	Filtering is only one of the reasons to use an input parameter. They could be used for parameters.
Variables are interpreted as the “Where” clause.	In Graphical calculation views, Input parameters are interpreted as “Having” clause which filters the query after SQL has retrieved, assembled and sorted the results.

Calculated columns

Details

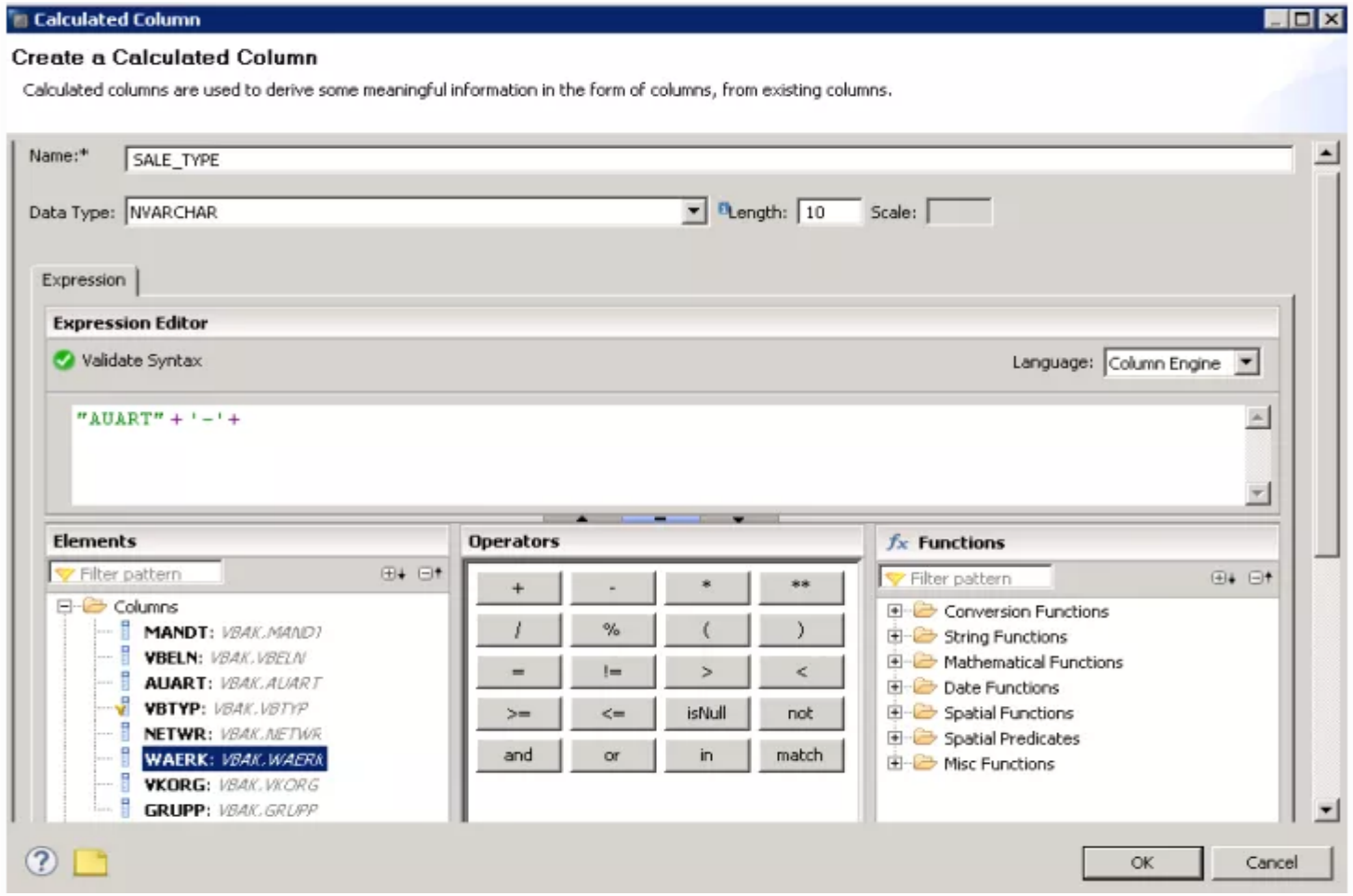
Calculated columns are manually added output columns and calculate its values at runtime based on the result of an expression . Expression may use other column values, functions, input parameters or constants.
Based on the data-type, a calculated column could be a calculated attribute or a calculated measure.

Calculated column properties	
Property	**Description**
Data Type	The value of this property specifies the data type of the calculated attributes or calculated measures.
Semantic Type	The value of this property specifies the semantics assigned to the calculated attributes or calculated measures.
Hidden	The value of this property determines whether the calculated column is hidden in reporting tools.
Drill Down Enablement	The value of this property determines whether the calculated attribute is enabled for drill down in reporting tools. If it is enabled, the value of this property specifies the drill down type.
Display Folder	If the calculated measure is grouped in any of the display folder, the value of this property specifies the display folder that was used to group related measures.

Calculated columns

Example

Calculated attribute



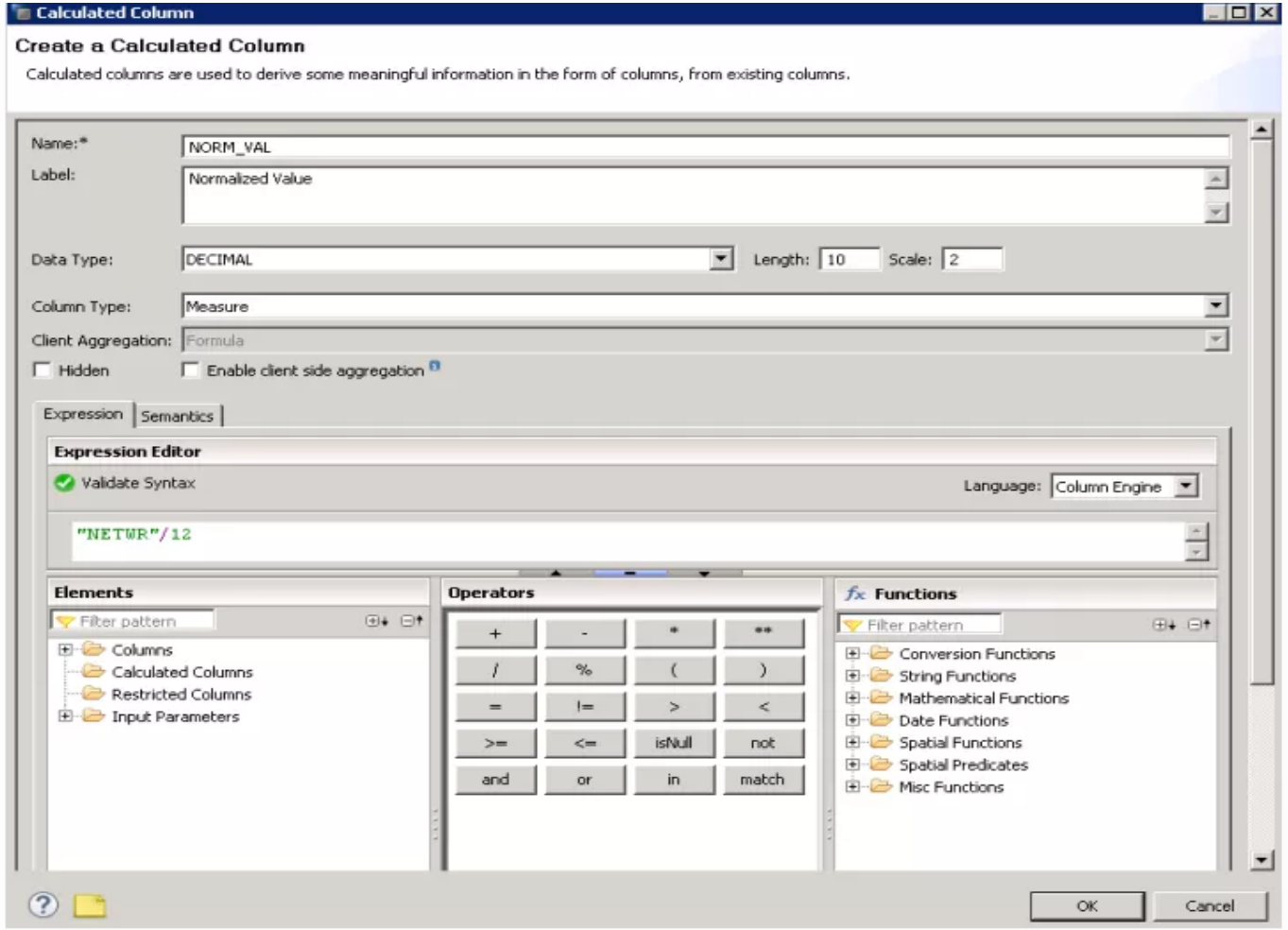
*Another use case could be a conditional expression e.g.

if("PRODUCT_QUANTITY_SOLD" <= 10,'Poor Sales', if("PRODUCT_QUANTITY_SOLD" <30,'Average Sales','Good Sales'))

Calculated columns

Example

Calculated measure



*Another use case could be a percentage calculation expression e.g.

(("PRODUCT_SALES_PRICE" - "PRODUCT_COST_PRICE")/"PRODUCT_COST_PRICE")*100

Restricted columns

Details

Restricted columns are used to restrict values of measures based on attribute restrictions.
Restricted columns could be defined on the measures in semantics node by using any of the below approaches:

- Apply restrictions on attribute values by using values from other attribute columns.
- Apply restriction on attribute values using expressions.

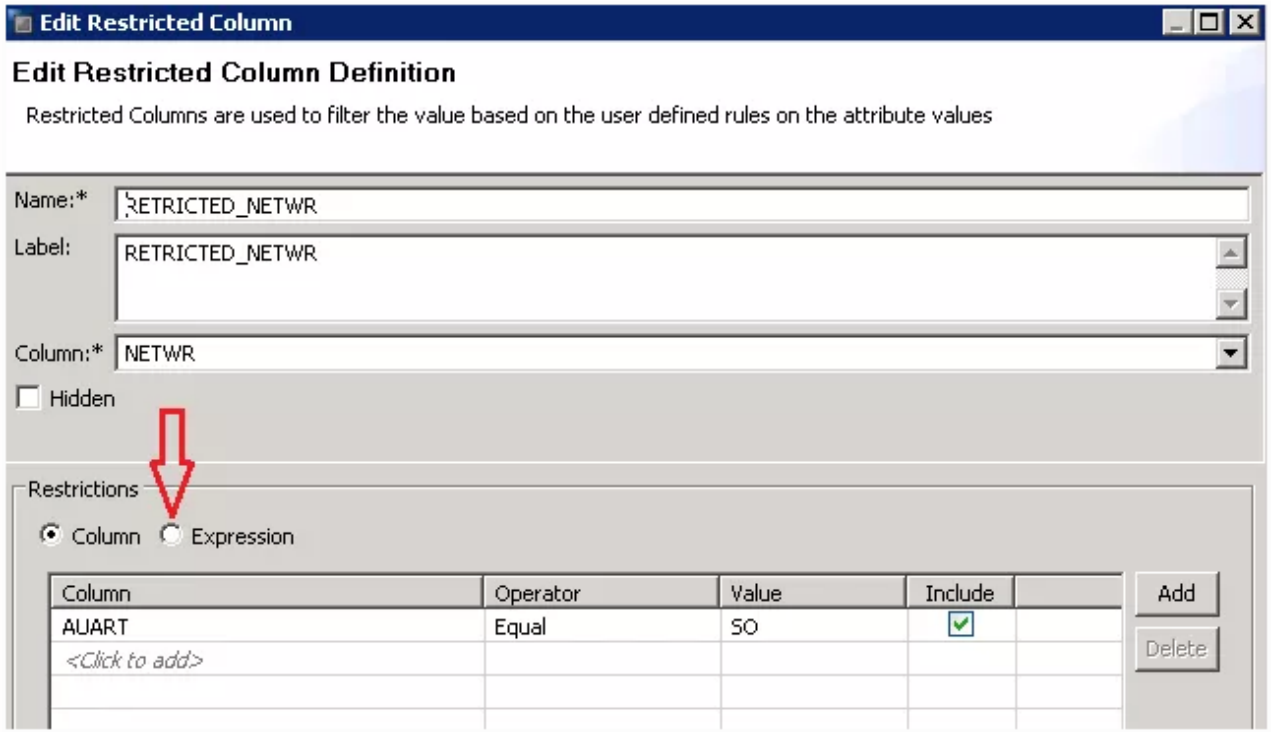
For restricted columns, modeler applies the aggregation type of the base column, and you can create restricted columns in the default aggregation view node or star join node only.

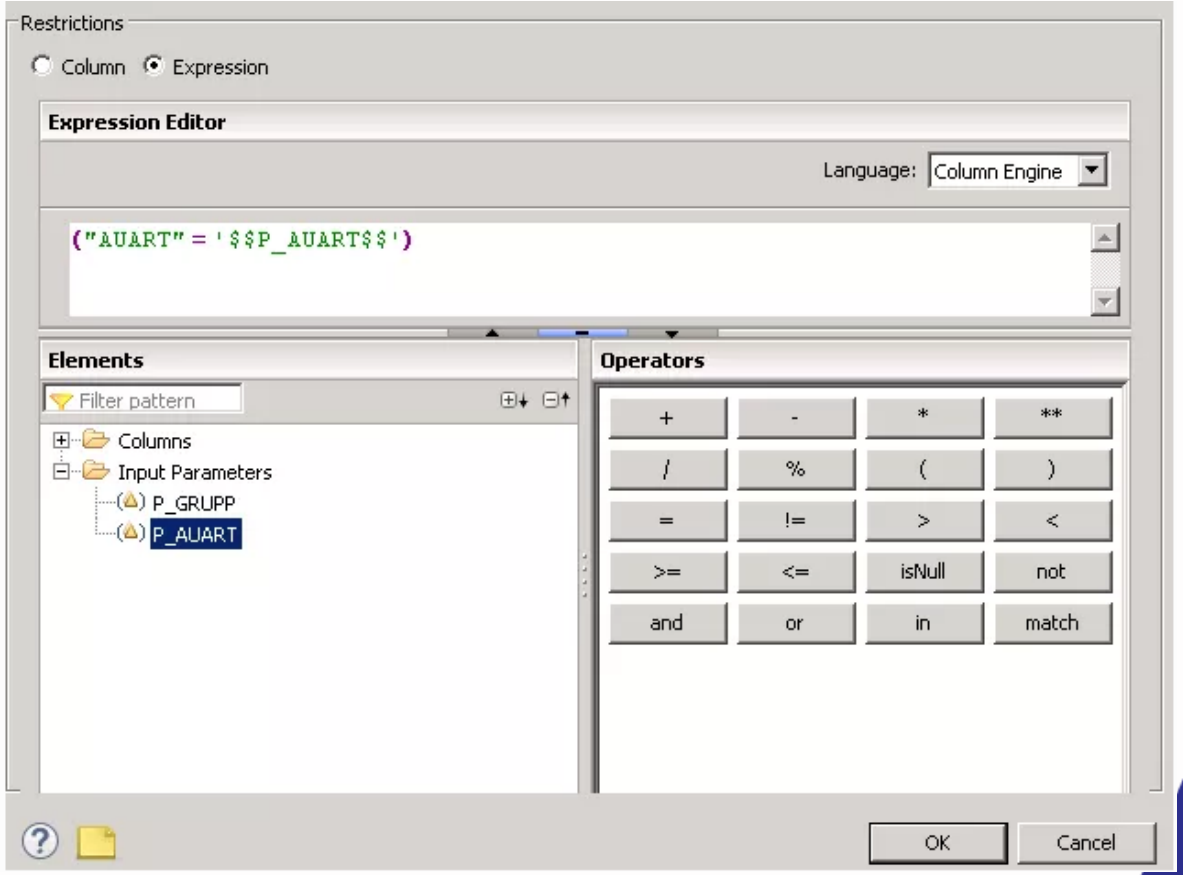
Restricted column properties	
Property	**Description**
Data Type	The value of this property specifies the data type of the restricted column.
Hidden	The value of this property determines whether the restricted column is hidden in reporting tools.
Display Folder	If the restricted measure is grouped in any of the display folder, the value of this property specifies the display folder that was used to group related measures.

Restricted columns

Example

Filter logic for restricted column could be a constant value of a attribute, or a complex expression, and it could use input parameter for dynamic filtering as well.





LTI

Let’s Solve