Virtual Key Figures and Characteristics

- Content of Note 1717880 Virtual Key Figures and Characteristics
- Documentation
- Technical Overview
  - Development in ABAP
  - OLAP ‘Read Requests’
  - F4 Help · ‘Select Filter Value’
  - List of common problems/symptoms
  - Analysis
- Remarks/Topics

Content of Note 1717880 Virtual Key Figures and Characteristics

Documentation

There is a sample code available you can use as an template for your BADI Implementation, please see class CL_EXM.IM.RSR.OLAP_BADI or follow the following path: SE19 -> Classic BADI RSR.OLAP_BADI -> Menu -> Goto -> SampleCode Please see also Using the Cache Despite Virtual Characteristics and Key Figures.

Note 1488910 introduces the new kernel-based BADI RSROA.OLAP_BADI which reduces the runtime for the BADI considerably. This reduction in runtime is only achieved if you create a new implementation of the BADI RSROA.OLAP_BADI, the classic BADI (and corresponding implementations)remains unchanged. Please see the sample code of the class CL_EXM_RSROA.OLAP_BADI. There you will notice that new methods have been introduced as e.g. COMPUTE_TABLE. This method is called once per PartProvider and DataManager-call (data package -> structure C.T_DATA) and gives you the possibility to process the entire data package. If you don't need this flexibility you can use the method COMPUTE_SINGLE which is called for each data record of C.T_DATA (as for the classic BADI). In the following discussion we refer only to the classic BADI since all points are valid for the new BADI as well. The formal differences are addressed above.

The following notes are of general interest:

- 1488910 Kernel-based BAdI to improve runtime
- 891029 general information/hints, e.g. to navigation attributes
- 1088469 the new mode CHNG_W_SEL was introduced with this note
- 1574673 enhancement for RSROA.OLAP_BADI - method FILTER_F4_MODE_QUERY called by ‘Select Filter Values’ (relational browse)
- 1143411 different data when data is fetched from BWA and database
- 822302 OLAP CACHE for remote providers
- 2137702 Query-Handler: Invalid cursor
- 2158666 Query-Handler: Invalid cursor (BW 7.3x)
- 2118240 Virtual keyfigures do not work as expected with selection of structure elements (non-cumulative key figures)
- 2201882 No Query cache generated when using Virtual Characteristic/Key Figure
- 2122239 Virtual Key Figures display unexpected values

Technical Overview

Development in ABAP

There are three methods where you can deposit coding: DEFINE: This method is only called during generation of the query; you need to specify the characteristics/key figures (use the technical names, e.g. chanm='0MATERIAL' or kyfnm='0AMOUNT') you would like to use in the exit. Moreover please choose one of the following ‘modes’(RRKE_C_MODE) for the characteristics:

- no_selection: The values of this characteristic can be changed in the query. If you define filter values in the query regarding this infoobject, these selections are ignored when the data manager retrieves the data from the provider (performance) but 'applied' after the user exit was processed.
- read: You can use the values of this infoobject in the COMPUTE method, but you cannot change them!
- space: Is interpreted as 'READ'
- chng_w_sel: As with mode NO_SELECTION you can change the characteristic in the calculation of the exit (method COMPUTE), but the global filters of the query are transfered to the database/BIA.(please review note 1088469)
- filter: You cannot change characteristic values in COMPUTE_TABLE method, but you can delete complete records from the data table (filter the data). In addition method IF_EX_RSROA.OLAP_BADI-FILTER_F4_MODE_QUERY is called where you can adjust the C.TS.KEYS table according to the filtering in COMPUTE_TABLE method. If the virtual characteristic with mode "FILTER" is drilled in the query, the values for the F4 help may already be available in the OLAP cache. In this case the BADI is no longer called. The BADI is also in general not called on the variable screen.

COMPUTE: Here the changes and calculations are carried out; this method is called once per data record (C.S_DATA) which may lead to performance problems(If RSROA.OLAP_BADI is used, also method COMPUTE_TABLE is available - see above). The following parameters can be used in the method: i_s_rkb1d -> e.g. technical name of provider and query i_partcube -> partprovider of data package i_ncum -> indicates if non-cumulative key figures are appear in the data package
**INITIALIZE**: In general there is no need to add something to the provided coding (see sample code) which basically only determines the position (corresponding instance attributes) of the key figures/characteristics in the structure C_S_DATA (containing the data records you can change in COMPUTE). This method is called only once during query execution (navigation step).

**Instance Attributes**: You need to add public instance attributes for each Infoobject you want to use in the method COMPUTE: Characteristic Variable: P_CHA_<Technical Name of Characteristic> Key Figure Variable: P_KYF_<Technical Name of Key Figure> Level: Instance Attribute Visibility: Public Associated Type: I In INITIALIZE these attributes are filled with the position of the InfoObject in the structure C_S_DATA. In the COMPUTE method these attributes are then used to address the corresponding values of C_S_DATA.

**II.B Point of Calculation**

The COMPUTE method is called for each data package within each read request (see below), basically right after the data retrieval from the database/BWA. No OLAP aggregations or Query functions have been applied up to this point. Take a look at the picture Processing Layers, the processing block ‘Checking Selections’ contains the method CUMPUTE. This implies a problem which is specific to virtual key figures: in case the ‘calculation of the formula’ and the ‘aggregation of data records’ are not **commutative** (commutative meaning that changing the order of the operations does not change the result) the query result in general depends on the detail level read from the provider. Even the usage of aggregates, compression and BWA may make a difference (see note 1143411 ‘BWA and Virtual key Figures’ and note 379832 ‘Calculation before Aggregation’). A very simple example would be the formula \( F = KF + 1 \) where \( KF \) is a basis key figure. If we assume that the cube contains only 2 data records, one in the \( F \) and one in the \( E \) table and that both have the same combination of keys, the virtual key figure returns different results before and after compression (or if a compressed aggregate is used):

**BEFORE COMPRESSION: 2 data packages**

\[
\begin{array}{ccc}
\text{material} & \text{product group} & \text{KF} \\
M1 & G1 & 2 \\
\end{array}
\]

F table

\[
\begin{array}{ccc}
\text{material} & \text{product group} & \text{KF} \\
M1 & G1 & 2 \\
\end{array}
\]

E table

Query result: 6

**AFTER COMPRESSION: 1 data package**

\[
\begin{array}{ccc}
\text{material} & \text{product group} & \text{KF} \\
M1 & G1 & 4 \\
\end{array}
\]

E table

Query result: 5

Please note that the difference is caused by the fact that the data is retrieved seperately (in different data packages) from the fact tables and the BADI is called before further aggregation.

As described in note 1143411, such virtual non-commutative (aggregation and formula calculation do not commute) key figures in general also cause inconsistent query results during query navigation, even when the cube is compressed and the BWA (note 1143411 is relevant as well when BW is based on Hana since HANA uses the same Analytics API than BWA) is not used! The following approaches may help to get the desired query output (see also the simple example mentioned in note 2122239):

- You can specify a level of detail in the user exit that is the minimum level for the reading of the data.
- You can deactivate the BWA and assure that the affected cube is fully compressed for the queries that use virtual key figures.
- You can use formulas with exception aggregation instead of virtual key figures.
- You can switch off the usage of the local (and global) OLAP cache. This is only possible as of BW74 release. It implies that the query needs to access the data base/BWA during every navigation step which in turn means that the exit is processed during every navigation step.

**OLAP ‘Read Requests’**

More information to ‘Read Requests’ and ‘Data Packages’:
- The OLAP/Analytic Engine can split the read access into groups which are called read requests. They e.g. retrieve the data from partproviders of the same type (see PartProvider Grouping) but also OLAP features like constant selection, exception aggregation and non-cumulative key figures lead to separate read requests.

- PartProvider Grouping: different options can be set in transaction RSRT - button Properties (see also Specifying the InfoProvider Grouping for Delta Caching). 'No Grouping' means that there is only one read request.

- Query with non-cumulative key figures: in this case there is always a read request for the non-cumulative and one for the cumulative key figures (flag ncm of the COMPUTE method). See also [III.7]

- Within these read requests, data is fetched from the data manager in data packages (separately from each infoprocessor, fact tables and if the result set is larger than 20000 records).

- For each data package, the processing block 'Checking Selections' (which contains the method CUMPUTE, see the picture Processing Layers) is executed within a loop. The result set is written into the internal table SP. In case the query uses the OLAP cache (local or global), the content of this table SP is saved/used as the corresponding cache object (see picture Processing Layers).

- The method INITIALIZE is only called once for each read request.

F4 Help - 'Select Filter Value'

As explained in note 1574673, if a query has virtual characteristics then the function 'Select Filter value' (when used for these virtual characteristics) is executed with the F4 mode M (master data - see note 1565809). In case mode Q is required, please follow the instructions of note 1574673. Please note that this has no effect on the selection of an F4 help on a variable screen.

List of common problems/symptoms

1. Exit (methods as DEFINE) isn't called at all: please activate the class and the BADI again.

2. Virtual characteristics are not filled: please check the chosen mode (RRKE_C_MODE) for the characteristics. If you use the mode 'read', changes of the characteristic values in COMPUTE are not saved by the system.

3. Don't forget to define the corresponding instance attributes with correct types

4. SQL Error INVALID_CURSOR: if there is the statement COMMIT (ABAP command commit work, or e.g. implicit commit work by RFC call) somewhere in the coding of the method COMPUTE, it may lead to this error message (if more than one package was read from an open SQL cursor). The corrections of the notes 2137702 'Query-Handler: Invalid cursor' and 2158666 'Query-Handler: Invalid cursor (BW 7.3x)' may solve such issues.

5. DBMAN114 'You are executing a nested query': in this case you probably run a query (e.g. by using RSDR_INFOPROV.READ) within the exit (CUMPUTE) which is not allowed. A possible workaround is to run the query in its own role area by using an RFC call (e.g. RSDR_INFOPROV.READ_RFC with DESTINATION 'NONE')

6. Performance problems: the mode READ may adds additional Infoobjects to the query (not visible drilldown characteristics) and leads to an increased amount of data which has to be fetched from the provider.

7. Issues with non-cumulative key figures: the OLAP processor processes the cumulative data separately from the non-cumulative data (two read requests, please see above). This means in particular that the virtual-non-cumulative key figures cannot be changed in the context of the cumulative data. So the OLAP engine ignores cumulative key figures in context of non-cumulative key figures and vice versa. See also 2118240.

8. OLAP features like constant selection and exception aggregation lead to separate read requests (similar to the case of non-cumulative key figures). This can cause issues since e.g. key figures used in the BADI may be processed/retrieved in different read requests.

9. Virtual key figures seem multiplied by a factor/ virtual key figures return different results when reading from BWA or a multiprovider with different part provider types: check your results with query property 'No grouping' (RSRT - Properties).

10. If the provider is an Infoset you need to use the technical names which are internally used by the OLAP engine. They are made up of the technical name of the Infoset and the field alias, e.g. ISET___F1 (more information to Infosets can be found under the link https://wiki.sdn.sap.com/wiki/display/BI/OT-ISET).

11. BWonHANA (similar issue may occur when a query uses BWA)

   1. Wrong data: please note that the calculation of virtual key figures (see ILB 'point of calculation') may depend on the detailed level read from the provider. Then the result of a query that reads the data via the HANA Analytics API could be different from the result the query returns when SQL is used. Review also note 1143411 where this problem is discussed in detail.

   2. In case you add data records in the exit, you need to take into account the following point: if the Analytics API is used, the internal field _FEMS of the internal table C_S_DATA (which contains the data records) plays an important role (e.g. see the blog BW-on-HANA and the _FEMS) since it helps to assign the record to the relevant structure elements. This field is for internal use only, so please just fill it with zero.

   3. As explained in point 8, OLAP features like exception aggregation lead to separate read requests. In case a pushdown of this feature is carried out, the OLAP BADI is not called for the corresponding read request (this is a technical restriction) which leads to incorrect values for the virtual key figures. In such a case, you need to switch off the pushdown by changing TREXOPS to 3, see Query Property: Operations in HANA BWA (TREXOPS).

12. Usage of Hierarchies

   1. In general it is not recommended to work with hierarchies (node filters) in the BADI since the internal handling is rather complex and it is not possible to rely on a certain level of nodes which is transferred to the OLAP engine. See points below.

   2. Hierarchy in the drilldown and expanded to level n: in this case the system requests all relevant (from level n) leaves and nodes from the database and then calculates the values of all nodes on the higher levels. In case it is not possible to get nodes from the Infoprocessor, the OLAP Engine receives all leaves (which meet the conditions of the node filters) and then has to carry out all necessary aggregations by itself. See Example II.

   3. Hierarchy Node filter: In case the exit uses a characteristic with an hierarchy node filter, you need to take into account that normally the DataManager returns the corresponding nodes (negative SIDs). In addition, it is not possible to rely on a certain level of nodes which is transferred to the OLAP engine. It even may happen that the system needs to request the hierarchy on leaf level. See Example I.

   4. In case a characteristic with the active hierarchy or node restriction is used in the exit (with any mode (RRKE_C_MODE), see also Resolve of Hierarchy Node Restrictions), the corresponding filters are always resolved.

Analysis
BADI RSR_OLAP_BADI
In case of problems please set breakpoints in all three methods and check whether the coding is processed (called) as expected. When
COMPUTE is called it might be important to know how many read requests exist for this query and which one is currently processed. In order to
find this out you can use the standard breakpoint 'OLAP Processor/DataManager'(note 1591837) in RSRT and check the internal table
c_tax_read_request. The following fields may contain interesting information:

- READ_ID: identifies the read request
- NCUM: if true, this read request contains non-cumulative key figures
- SFC: list of all requested characteristics
- SFK: list of all requested key figures
- SELDR: all filter restrictions

The method COMPUTE has 3 importing parameters you can use in your coding and while debugging:

- l_s_rkb1d -> e.g. technical name of query
- i_partcube -> partprovider of data package
- i_ncum -> equal X when C_S_DATA contains ncum key figures

Please note that the method INITIALIZE is once called for each read request.

BADI RSROA_OLAP_BADI
Similar to analysis of BADI RSR_OLAP_BADI. Difference: Method COMPUTE is available in two versions: COMPUTE_TABLE for processing the
whole data package and COMPUTE_SINGLE for processing record by record.

Remarks/Topics

- As of BW74 the definition of the table type RSDD_TH_SFK has change which may lead to syntax error in your coding, please see Table
  Type RRKG_TH_SFK for further details.
- Problem of non-commutability between aggregation and formula calculation: Non-commutative Virtual Key Figures
- BADI interface is enhanced from 740SP8; Note 2118240 - Virtual keyfigures do not work as expected with selection of structure elements
- Example I: Simple Example where a query is restricted to filters combined with a logical OR