How-to Guide
SAP NetWeaver ‘04

How to...
Create Efficient MultiProvider Queries

Version History:
3.00 – July 2006

Applicable Releases:
SAP NetWeaver '04
1 Business Scenario

In a scenario where queries are executed on a MultiProvider, the system usually will perform a data selection on all InfoProviders that are part of the MultiProvider. Now if the MultiProvider is logically partitioned, for example by region as shown in the figure below, and the user selects data from one InfoProvider only, say “Europe”, the system will run a sub-query on all three regional InfoProviders nevertheless. Therefore, more database accesses are required compared to a query built directly on the European InfoProvider.

Query run time characteristics

- MultiProvider is used between a Query and the InfoProvider
- MultiProvider including many (>10) InfoProviders
- All InfoProviders of an application have the same structure (InfoObjects)
- Each InfoProvider contains homogeneous data slices divided by more than one characteristic (for example year, country or time zone etc.)
- Query runtime usually need data only from 1 or 2 InfoProviders
- High Query runtime due to the parallel execution of the sub-queries (MultiProvider Split)
There are three solutions to gain efficiency:

1. **Set a characteristic as a constant value of InfoProvider**
2. **Include characteristic InfoProvider (0INFOPROV) in query filter conditions**
3. **Restrict the technical InfoObject 0INFOPROV automatically**
2 Introduction

2.1 Benefits
This document describes a solution that provides efficiency gains on one hand while on the other hand avoiding the disadvantages described in the first chapter. The solution will keep the required database selections on the MultiProvider to a minimum. One query will be sufficient for all selection scenarios whether the user selects a single region, several regions, or all regions. At the same time the system will determine dynamically, on which InfoProviders a sub-query needs to be run.

An additional benefit is that the logical partitioning can easily be changed. The solutions provide a flexible way of mapping what content is stored in which InfoProvider and also allows adding or removing InfoProviders to the MultiProvider without the need of maintaining queries.

2.2 Solution Overview
Apart from setting a characteristic to a fixed value in the InfoProvider, there are two solutions provided in this how-to guide:
   1. Restricting the InfoProvider automatically
   2. Restricting the InfoProvider characteristic using a customer exit variable

The first solution can be used if the InfoProvider(s) can be automatically determined by checking the selected entries of the partition criteria set in the global filter of a query against the entries in the dimension table in the InfoProvider(s). Therefore, this solution only works on InfoCubes, but not on InfoSets or ODS Objects. This solution should be used if the data in the InfoProviders is homogeneous. Please note that solution #1 is available as of a certain support package (see step-by-step instructions).

In the second solution, the virtual InfoProvider characteristic is used to develop efficient queries on a MultiProvider. However, instead of restricting the InfoProvider characteristic to a fixed InfoProvider, a customer exit variable is used. Based on the user input, the customer exit is used to dynamically determine, which InfoProviders actually need to be queried. If the user does not select any data from some of the InfoProviders, the system will ignore these InfoProviders, which leads to the desired efficiency gain. Solution #1 needs to be implemented if the data slices in the InfoProviders are not homogeneous and the InfoProviders can not be determined by the global restriction of the query.
2.3 Example and Testing

In this paper we will use a MultiProvider that is logically partitioned by region based on the country characteristic.

We use one query on the MultiProvider. At execution time, the user enters which country or countries should be selected. Therefore, we do not know upfront whether Europe, North America, or Asia data or a mix of them will be selected.

1. The figure on the right shows the query definition. Country is restricted to a user input variable.

2. Execute the query in transaction RSRT using “Execute + Debug” you can use the MultiProvider Explain to analyze which InfoProviders will be accessed.
3. Select the “MultiProvider Explain” option.

4. Enter various selections for country.
   For example, select countries from North America and Asia.

5. The messages will show that both InfoProviders are accessed no matter what countries are selected.
   In the example, you can see that the Europe InfoProvider is read although no European countries were selected.
3 The Step by Step for Solution #1

The main facts about this solution are listed in the following slide:

**Query Design**

- **Restrict the technical InfoObject 0INFOPROV automatically**
  - Maintain customizing table "RRKMULTIPROVHINT" with partitioning criteria (InfoObject and MultiProvider)
  - During the query generation and query run time the not-needed PartProviders are automatically filtered
  - The InfoProvider filter is global and not per restricted key figure
    - Low maintenance effort (only InfoObject and MultiProvider)
    - QueryDesigner user does not need information about the underlying data model
    - Inhomogeneous InfoProviders are not supported (filter is global and not for each calculated key figure)
    - Available as of BW 3.0 SP30 (For BW 3.0 SP28 OSS Note 913975 is a prerequisite for note 911939). Also, Note 950258 needs to be applied.
    - ODS Objects and InfoSets are not supported
    - No Line-Item Dimension is supported for partitioned InfoObject Criteria

**Technical Details:**

The RRKMULTIPROVHINT can be used instead of or in conjunction with using the constants on the individual InfoCubes. The constants will direct the run schedule to the correct InfoCube, but if you want, you can maintain things centrally in this table if the constants aren't set.

In the RRKMULTIPROVHINT table, you can specify the MultiProvider and a characteristic for partitioning and if a query has restrictions on this characteristic, the OLAP processor will check which InfoCubes can return data for the query. The data manager can then completely ignore the remaining InfoCubes. This is handled by the OLAP processor.

The system is doing a quick read of the dimension table of the InfoCube to check if the selected value (or multiple values!) are included and it's "worth" reading the full InfoCube.

**Pre-requisites:**

- MultiProvider must contain only InfoCubes.

- For SAP NetWeaver '04, ensure you are at the correct support package or apply SAP notes 911939, 913975, and 950258.
1. Maintain table “RRKMULTIPROVHINT” with the partitioning criteria Country (InfoObject and MultiProvider name).

See Appendix for details.

2. Execute the Query

Please check notes 911939, 913975, and 950258.
4 The Step by Step for Solution #2

1. Create a new query (or start with a copy of the original query). Right click on the “InfoProvider” characteristic and create a new variable. Select “Customer Exit” for the processing type.

2. In the detailed settings, select “Multiple Single Values” and “Optional”. The variable should not be ready for input or changeable in the query navigation.
3. Finish and save the variable.

4. Finally, add the variable to the filter of the query.
5. Implement the Customer Exit for the InfoProvider Variable. Enter the coding (see also Appendix). Save and activate the coding.

6. Execute the new query using RSRT as described above.

For example, if only data about Germany is selected, only one InfoProvider will be accessed and the other ones will be excluded.

The same query can be used to select US data or a mix of other countries.
5 Comments for Solution #2

- The solution can be implemented for BW release 3.0 or greater.
- The content mapping of countries i.e. regions to InfoProviders is based on a custom table. The table and ABAP coding is built to handle single values as well as ranges.
- In general, the mapping also allows for great flexibility in managing the content. For example, data can be moved between InfoProviders or archived, new InfoProviders can be added over time or removed if necessary. All you need to do is maintain the mapping.
- Although the logical partitioning in the example is based on one characteristic (country), the solution will work also if the InfoProviders are partitioned using several characteristics.
- The example uses homogenous InfoCubes. However, the solution will work the same way when heterogeneous data models and InfoProviders (InfoCubes, ODS Objects, InfoSets, etc.) are combined into a MultiProvider.
- If you need help in implementing the customer exit, please check the online help or see the “How to… Verify Variable Input” document for BW release 2.x.

Note: For completeness, it should be mentioned that there are other ways of making efficient usage of MultiProviders. The three options are shown in the following picture. However, none of them is as flexible as the solution described in this how-to paper.

Other options to make sure that a MultiProvider only retrieves data out of relevant InfoProviders at runtime:

- By using constants in the design of InfoProviders (to be specified at implementation time)
- By using different technical key figures in your InfoProviders if data content is distinct.
- By using characteristic 0INFOPROV when designing a query on the MultiProvider. Example: 0INFOPROV = 'ZEUSALES'
- By using characteristic 0INFOPROV when designing a query on the MultiProvider. Example: 0INFOPROV = 'ZEUSALES'
- By using characteristic 0INFOPROV when designing a query on the MultiProvider. Example: 0INFOPROV = 'ZEUSALES'
- By using characteristic 0INFOPROV when designing a query on the MultiProvider. Example: 0INFOPROV = 'ZEUSALES'
6 Appendix for Solution #1

6.1 Maintaining a table

1. Go to SE16, type in table "RRKMULTIPROVHINT" and choose New.

Data Browser: Initial Screen

Table Name: RRKMULTIPROVHINT

2. Make the entry and choose Save

Table RRKMULTIPROVHINT Insert

| MULTIPROV | GUTC_C10  |
| CHANM     | GCALMONTH |
| POSIT     |           |
7 Appendix for Solution #2

7.1 Custom Mapping Table

```
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Delivery and Maintenance</th>
<th>Fields</th>
<th>Entry help/check</th>
<th>Currency/Quantity Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

```
TRANSP_TABLE: YBW_MAPPING
SHORTTEXT: BW Eficient MultiProviders

DATA:
ls_var TYPE rrs0_s_var_range,
ls_range TYPE rsr_s_rangesid,
l_contained TYPE c,
ls_mapping TYPE ybw_mapping,
l_t_mapping TYPE TABLE OF ybw_mapping.

* Called after variable popup
IF i_step = 2 AND i_vnam = 'INFOPROV'.
```

7.2 Sample Data

```
<table>
<thead>
<tr>
<th>MANDT</th>
<th>LOW</th>
<th>HIGH</th>
<th>INFOProv</th>
</tr>
</thead>
<tbody>
<tr>
<td>800</td>
<td>CA</td>
<td>TMB_AMER</td>
<td></td>
</tr>
<tr>
<td>800</td>
<td>DE</td>
<td>TMB_EMEA</td>
<td></td>
</tr>
<tr>
<td>800</td>
<td>FR</td>
<td>TMB_EMEA</td>
<td></td>
</tr>
<tr>
<td>800</td>
<td>GB</td>
<td>TMB_EMEA</td>
<td></td>
</tr>
<tr>
<td>800</td>
<td>JP</td>
<td>TMB_AsIA</td>
<td></td>
</tr>
<tr>
<td>800</td>
<td>US</td>
<td>TMB_AMER</td>
<td></td>
</tr>
</tbody>
</table>
```

7.3 Customer Exit

```
*----------------------------------------------------------------------*
*   INCLUDE ZXRSRU01                                                   *
*----------------------------------------------------------------------*

*** Enhancement: MultiProvider using InfoProvider Variable
include YBW_INFOPROVIDER_VARIABLE.

*&---------------------------------------------------------------------*
*& Include           YBW_INFOPROVIDER_VARIABLE                        *
*&---------------------------------------------------------------------*

DATA:
ls_var TYPE rrs0_s_var_range,
ls_range TYPE rsr_s_rangesid,
l_contained TYPE c,
ls_mapping TYPE ybw_mapping,
l_t_mapping TYPE TABLE OF ybw_mapping.

* Called after variable popup
IF i_step = 2 AND i_vnam = 'INFOPROV'.

```
* Read mapping table
  REFRESH lt_mapping.
  SELECT * FROM ybw_mapping INTO TABLE lt_mapping.

* Process all selection for country variable
  REFRESH e_t_range.
  LOOP AT i_t_var_range INTO ls_var WHERE vnam = 'S_COUNT'.

  * Process all mapping rules
    LOOP AT lt_mapping INTO ls_mapping.

    * Always fill LOW and HIGH, Otherwise logic below will not work
      IF ls_mapping-high IS INITIAL.
      ls_mapping-high = ls_mapping-low.
      ENDIF.

    * Check if selection is contained in the defined InfoProvider
      CLEAR l_contained.
      CASE ls_var-opt.
      WHEN 'EQ'.
      IF ls_var-low BETWEEN ls_mapping-low AND ls_mapping-high.
      l_contained = 'X'.
      ENDIF.
      WHEN 'BT'.
      IF ls_var-low <= ls_mapping-high AND
      ls_var-high => ls_mapping-low.
      l_contained = 'X'.
      ENDIF.
      ENDCASE.

    * Add InfoProvider to return table
      * Note: Use COLLECT to avoid duplicates
      IF l_contained = 'X'.
      CLEAR ls_range.
      ls_range-sign = 'I'.
      ls_range-opt  = 'EQ'.
      ls_range-low  = ls_mapping-infoprov.
      COLLECT ls_range INTO e_t_range.
      ENDIF.

  ENDLOOP. " lt_mapping

ENDLOOP. " i_t_var_range
EXIT.
ENDIF.