Improve your Reporting Performance in SAP BusinessObjects Planning and Consolidation Version for NetWeaver

Applies to:
SAP BusinessObjects (SBOP) Planning and Consolidation version for NetWeaver 7.x, based on a NetWeaver BW 7.01 platform. For more information, visit the Enterprise Performance Management homepage.

Summary
This white paper covers the main aspects to be considered when creating any EVDRE reports or input templates for SBOP Planning and Consolidation version for NetWeaver (SBOP PC_NW or BPC) to achieve optimal performance. Areas addressed include: EVDRE definition, RSDRI queries, and Tips and Tricks in the specific context of SBOP PC_NW.

Author: Rob Marshall
Company: SAP
Created on: 15 June 2011

Author Bio
Rob Marshall is a certified BPC consultant working in the SAP CSA (formerly RIG) group. Rob has been working with BPC for over seven years starting with OutlookSoft 4.0.
# Table of Contents

SBOP PC_NW Architecture ........................................................................................................... 3
Anatomy of an EVDRE ..................................................................................................................... 3
Types of EVDRE Queries ................................................................................................................. 6
Sparsity Coefficient ......................................................................................................................... 6
UJSTAT ........................................................................................................................................ 6
  RSDRI Events ................................................................................................................................. 7
  MDX Events .................................................................................................................................. 8
Tips for better Performance ............................................................................................................... 9
  Create RSDRI queries ...................................................................................................................... 9
  Limit MDX queries .......................................................................................................................... 9
Create Multiple EVDRE’s ................................................................................................................... 9
  Expansions .................................................................................................................................... 9
  Orphaned Data ............................................................................................................................... 10
Custom Suppression .......................................................................................................................... 11
EVDRE’s by ACCTYPE ....................................................................................................................... 13
Summation of the fly ......................................................................................................................... 13
Summary ......................................................................................................................................... 16
Related Content ............................................................................................................................... 17
Copyright ....................................................................................................................................... 18
SBOP PC_NW Architecture

SAP BusinessObjects Planning and Consolidation, version for NetWeaver is a four layer application composed of a client stack, a .NET Application server, the NetWeaver ABAP Application back-end, and the database server.

This paper focuses on the report performance between the client tier and the database/OLAP engine and how to maximize the performance in getting returned.

When executing EVDRE reports from the users’ computer, the request goes through the .Net Layer to the Shared Query Engine (SQE) on the ABAP application server. The SQE then instructs the query to either retrieve the data from the cube (MDX) or directly from the fact tables (RSDRI).

Anatomy of an EVDRE

EVDRE stands for Everest Dynamic Range Expansion and it is the recommended way to query the BPC database from the MS Excel frontend of BPC.

Within all EVDRE’s, there are three main parts needed to be defined as they control the behavior and formatting of the EVDRE report: the Application, Key ranges, and the Expand ranges (what dimensions members the report is rendering).
- **Application** – used to denote which cube or application the data is currently stored in.
- **Key Ranges** – used to denote where the columns and rows begin and end.
- **Expand Range** – denotes the dimensions and dimension members that will be rendered.

### Key Ranges:

<table>
<thead>
<tr>
<th>RANGE</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>PageKeyRange</td>
<td>Sheet1!$H$3:$H$14</td>
</tr>
<tr>
<td>ColKeyRange</td>
<td>Sheet1!$B$37:$HV$38,Sheet1!$X$37:$Y$38</td>
</tr>
<tr>
<td>RowKeyRange</td>
<td>Sheet1!$S$30:$S$32</td>
</tr>
<tr>
<td>CellKeyRange</td>
<td></td>
</tr>
<tr>
<td>GetOnlyRange</td>
<td></td>
</tr>
<tr>
<td>FormatRange</td>
<td></td>
</tr>
<tr>
<td>OptionRange</td>
<td>Sheet1!$A$25:$B$41</td>
</tr>
<tr>
<td>SortRange</td>
<td></td>
</tr>
</tbody>
</table>

- **PageKeyRange**: This directs the EVDRE to the current view or what intersections the report should return data for.
- **ColKeyRange**: Defines the range of column cells that the report should render data. As you can see above, you can have multiple expansions defined which are separated by a comma. In the above example, columns will be defined from cells HB37 to HV38 as well as HX37 to HY38. You might have noticed the column ranges HB37:HV37 and HX37:HY38 are hidden, this will be explained in the Trips and Tricks section.
- **RowKeyRange**: Defines the range of row cells that the report should render data. As with column expansion, you can also define multiple row expansions.
- **OptionRange**: Defines the options of the report. The most common options are listed in the picture below:
Expand Ranges:

- **ExpandIn** – defines where to expand, either the columns (COL) or row (ROW)
- **Dimension** – defines which dimension is being expanded.
- **MemberSet** – defines the members to be expanded. This can be an MDX command like SELF, BAS, MEMBERS, DEP (dependants), or All. Also, as evidenced in the above first expansion, you can list members, separated by commas if you know exactly which members you want data for. Additionally, in the second expansion, you can utilize pipes “|” which denotes a second expansion. Using pipes within a single EVDRE will often, but not always, have better query performance than using multiple EVDRE’s.
- **BeforeRange** – used to insert formulas in the column or row before the expansion begins.
- **AfterRange** - used to insert formulas in the column or row after the expansion ends. For example, if you would like a summation of the columns, you can enter a formula in a cell that is not within the expansion cells (ColKeyRange), reference those cells with EVRNG(the cells containing the formula), and the report will create an extra column.
- **Suppress** – If you would like to suppress the data (bring back only rows/columns having data), you can enter a “Y” or a particular intersection of data (Sales,2009.JAN).
- **Insert** (Optional) – Setting the parameter to Y for any row or column will allow the user to perform a runtime insertion of suppressed row or column values.
Types of EVDRE Queries

To understand the theory of an EVDRE, you must first understand the different types of queries that can be produced from an EVDRE. There are two types of queries executed from the SQE:

- **RSDRI** – is the function group in SAP NetWeaver BW that deals with access to InfoProviders. In the SQE, an RSDRI query calls the internal SAP NetWeaver BW functions to read data from an InfoProvider and accesses the E and F fact tables. This is the preferred query mode as RSDRI queries are faster than MDX queries. However, RSDRI queries do not bring in aggregated data, just base level information.

- **MDX** – SAP NetWeaver BW exposes OLAP BAPIs to access data from an InfoProvider through MDX. The MDX query method in SAP BPC generates the MDX statements required to retrieve data with the SAP BPC application logic applied. It is always used when there are dimension member formulas, measure formulas, or hierarchy parent values (2009.TOTAL) to be retrieved and are often slower than RSDRI queries.

Prior to release 7, service pack 3, there was an additional type of query, a cell query. This was associated with only MDX type of queries and was often very slow as it would retrieve individual cells. Since this the performance was so slow, all queries are now Axis queries. Axis queries request a contiguous region of data. Due to this, often times the query fetches more data than is needed, however this is still faster than individual cell retrievals.

**Sparsity Coefficient**

Prior to version 7 service pack 3, an administrator could customize when a cell query was used by manipulating the Sparsity Coefficient. Now that cell queries are no longer available, the Sparsity Coefficient is no longer used.

**UJSTAT**

Within the SAP GUI, there is a transaction (UJSTAT) allowing a user to understand what type of query (RSDRI or MDX) is being used as well as how long a query takes to run. In order for UJSTAT to work, you must turn on the BPC_STATISTICS in the web administration portal, specifically in the Set Application Parameters.

**Administration >> Set Application Parameters**

<table>
<thead>
<tr>
<th>Type</th>
<th>KeyID</th>
<th>Value</th>
<th>Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td>(R)</td>
<td>APPROVALORG</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>BPC_STATISTICS</td>
<td>ON</td>
<td></td>
</tr>
</tbody>
</table>

To get to this transaction, login to the SAP GUI and type in UJSTAT

**SAP Easy Access**

On the next screen, enter the appropriate Start Date, Start Time, Appset, and Application. If multiple people are running reports at the same time, it would be wise to also enter the User ID to segment out the queries to just the ones you run.
After an EVDRE has successfully completed in SBOP Planning and Consolidation, go to transaction UJSTAT to see the results.

The screenshot informs the user of several key factors for this query:

- This query is a RSDRI query
- The Runtime, Technical Runtime, and Cumulative Runtime
- The number of records returned (Nr of Recs)

**RSDRI Events**

When a report or input template contains multiple EVDRE’s, it is important to know which one you are debugging. To do this, double click on the Unwrap XML event and look for “Filter.”

```
<Filter><parameter>CV application=M_5 DATASRC="TOTALADJ" ISSUANCE="ISSUE_3_0_0914" MEASURES="PERIODIC" RPTCURRENCY="LC" VEHICLE="TOT_VEHICLES" VEHICLE_REGION="ALL_REGIONS"</Filter>
```
This will tell you what the current view is as well as what data will be returned for the column and row axis.

By utilizing UJSTAT, we learn the below RSDRI query took 7.34 seconds to run and of that, it took 4.34 seconds to download the dimension cache and 2.844 seconds to return 1,450 records from the infoprovder.

### MDX Events

An EVDRE that creates an MDX statement is shown below, designated by the event MDX Query Kernel.

<table>
<thead>
<tr>
<th>Event Type</th>
<th>AppSet ID</th>
<th>App ID</th>
<th>Seq</th>
<th>Start Date</th>
<th>Start Time</th>
<th>Runtime</th>
<th>T_Runtime</th>
<th>C_Runtime</th>
<th>Nr of Recs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared Query Engine Read</td>
<td>ELITE_TRAN_W_N_5</td>
<td>0 10.01.2011 21:45:04</td>
<td>7.34000 7.34000 7.34000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Run Axis Query</td>
<td>ELITE_TRAN_W_N_5</td>
<td>1 10.01.2011 21:44:56</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Check Validation for Axis Query</td>
<td>ELITE_TRAN_W_N_5</td>
<td>2 10.01.2011 21:44:56</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Check &amp; Member Security (Axis)</td>
<td>ELITE_TRAN_W_N_5</td>
<td>3 10.01.2011 21:44:56</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MDX Query</td>
<td>ELITE_TRAN_W_N_5</td>
<td>4 10.01.2011 21:44:56</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Get Dimension and Member Information</td>
<td>ELITE_TRAN_W_N_5</td>
<td>5 10.01.2011 21:44:56</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Build Sub Queries for Multiple Hierarchies</td>
<td>ELITE_TRAN_W_N_5</td>
<td>6 10.01.2011 21:44:56</td>
<td>4.34000 4.34000 4.34000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Miscellaneous Processing Time</td>
<td>ELITE_TRAN_W_N_5</td>
<td>7 10.01.2011 21:45:01</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Run Sub Query for Measures</td>
<td>ELITE_TRAN_W_N_5</td>
<td>8 10.01.2011 21:45:01</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Miscellaneous Processing Time</td>
<td>ELITE_TRAN_W_N_5</td>
<td>9 10.01.2011 21:45:01</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Convert_to_EIS_INPUT</td>
<td>ELITE_TRAN_W_N_5</td>
<td>10 10.01.2011 21:45:01</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Run RSDRI Query</td>
<td>ELITE_TRAN_W_N_5</td>
<td>11 10.01.2011 21:45:01</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSDRI Query</td>
<td>ELITE_TRAN_W_N_5</td>
<td>12 10.01.2011 21:45:01</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Miscellaneous Processing Time</td>
<td>ELITE_TRAN_W_N_5</td>
<td>13 10.01.2011 21:45:01</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSDRI_Informal_Req</td>
<td>ELITE_TRAN_W_N_5</td>
<td>14 10.01.2011 21:45:01</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ProcessAggregate</td>
<td>ELITE_TRAN_W_N_5</td>
<td>15 10.01.2011 21:45:01</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ConvertRSDRIResult</td>
<td>ELITE_TRAN_W_N_5</td>
<td>16 10.01.2011 21:45:01</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Process SUE Batch</td>
<td>ELITE_TRAN_W_N_5</td>
<td>17 10.01.2011 21:45:04</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Process SUE Batch</td>
<td>ELITE_TRAN_W_N_5</td>
<td>18 10.01.2011 21:45:04</td>
<td>0.00000 0.00000 0.00000 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Within the MDX Query Kernel, the event Mddataset_create_object is useful in that it shows the actual MDX created.
By using this event, the user can understand the MDX being created.

Tips for better Performance

By keeping the following tips in mind when creating a report or input template, the performance will be maximized.

Create RSDRI queries

RSDRI queries are the fastest way to get data out of the database, as long as the database is properly tuned and indexed. The easiest way to do this is to bring back base level members or non aggregated data for the column and row expansions. If 2010.TOTAL is specified in the current view and the expansion is SELF, the SQE will create a MDX call to the database. On the other hand, if BAS is specified in the expansion, the SQE will create a RSDRI query.

Limit MDX queries

While MDX queries don’t always perform as well as RSDRI queries, they are not bad if you use them correctly. To do this, limit the amount of columns needed in the MDX statement. Typically, this will create multiple EVDRE’s where you can segregate the base members from the calculated members, thus easing the burden in the number of columns returned or need for the MDX query.

Create Multiple EVDRE’s

It is best practice to divide up one large query into multiple EVDRE’s. Not only will this help the performance of the report, but it will also help the application server deliver the information in a more timely manner since it will not have to store the information. Additionally, the developer will be able to segment RSDRI queries and MDX, so performance will be better.

Expansions

When using multiple EVDRE’s within a report or input template, make sure the report only expands on what it really needs. For instance, if the first EVDRE expands time along the columns and Products along the rows, the second EVDRE doesn’t need to make another database call for the rows.

The first expansion is defined as …
And the second EVDRE expansion is defined without the Row expansion.

### Orphaned Data

When having multiple EVDRE’s, it is possible to have orphaned data when a smaller dataset is returned after a larger dataset is returned. For instance, if the report returns 1,000 rows, and the second run only brings back 750 rows, you will still have the initial 250 rows “left over.”

To guard against this, encapsulate the second or more EVDRE’s within the scope of the initial EVDRE. To demonstrate this, create an EVDRE and change the ColKeyRange to the below:

Now, in the report area where data is returned (where the initial EVDRE was supposed to be – cells J45:U45), use cell referencing to the data area of the first EVDRE.
The last step is to hide the data columns of the initial EVDRE (AJ45:AU45) so the user doesn’t get confused.

An important note to remember when doing this is this only works when the columns are fixed, meaning the report will always have the same number of columns.

Custom Suppression

Within the EVDRE Expand Range, a user can choose to turn on/off suppression. The key here is to figure out what the report should actually be suppressing on, whether it is a certain account, category, entity, or time range. By placing a “Y”, the EVDRE will try to suppress all dimensions specified within the EVDRE which is best practice. When a custom suppression is used, the SQE creates another event within UJSTAT that brings back the same number of rows as the first EVDRE, so in essence the SQE is duplicating the workload.

To demonstrate this, create a simple query with accounts along the rows, time along the columns, and place a “Y” in the Suppress for rows.

In UJSTAT, it creates one event that returns data in approximately 7 seconds.
If we change the suppression from Y to SALES,2009.TOTAL, UJSTAT will show two events have been created and have taken longer than the original query.

With that said, custom suppression can be used, if used correctly. The main thing to remember is to create an RSDRI query for the suppression and stay away from MDX. For example, if the current view contains Sales as the account but we wanted to suppress all data where PART_VOLUME for all of 2009, we could enter (PART_VOLUME, 2009.TOTAL). To take it one step further, if we wanted to suppress rows for multiple years, we could enter (PART_VOLUME, [2009.TOTAL,2010.TOTAL]), but this will turn into MDX query. The following example amplifies this:

We created an ad hoc report with the following with expansion ranges.

By looking in transaction UJSTAT, we will see the run time being excessively long due to the axis query creating an MDX statement to return the data.

By changing the suppression to PART_VOLUME,[BAS(2009.TOTAL,2010.TOTAL)], the statistics in UJSTAT are much better since an RSDRI query is produced.
Also, notice the far right column in the two pictures of UJSTAT, the MDX query returned 409 rows of data while the RSDRI query returned 1,856. Even though the RSDRI query returned more rows, it was almost 400% faster.

**EVDRE’s by ACCTYPE**

When an EVDRE is rendering more than one ACCTYPE (INC, EXP, AST, LEQ), it is best to create multiple EVDRE’s rather than using a pipe ("|") or comma (",") due to the MDX performed by the Measures dimension. For example, we created one EVDRE rendering one EXP account and one AST account on the columns, as depicted below.

By viewing UJSAT, the report renders in just over 27 seconds.

But, if we create multiple EVDRE’s, by account, the report actually returns faster, even though the exact same data is being returned.

**Summation of the fly**

It is not widely known, but you can do summations in the Key Range of an EVDRE. This helps in creating more RSDRI queries than MDX due to the fact you are only bringing back base members instead of aggregated or calculated members. For example, create a sample report out of APSHELL defined with Activity down the rows and Time along the columns; this produces an MDX query because the Account dimension is an aggregated member (summation of three accounts)
By looking in UJSTAT, this will produce an MDX query.

Instead of having the aggregated account in the key range, place the actual accounts as seen below.

You will notice, not only does this produce a RSDRI query, but it is faster and also brings back the same data. This can also work if the requirements for a report are to bring back full year totals (2009.Total). Instead of creating an MDX query by bringing in the aggregated amount, you can create formula's creating the twelve months in the year.
Taking the same report as above and changing the columns to the aggregated year will produce an MDX query.

In UJSTAT….

By adding formula’s (EVPRO’s) and referencing all twelve months in the Key Range, as shown below, this will produce a RSDRI query.

In UJSTAT, a RSDRI query is produced.

A couple of notes on the above report, the formula was created using the function EVPRO and EVCVW to take into account the current action pane changing. The actual formula is as follows:
=EVPRO($H$2,EVCVW($H$2,"TIME"),"YEAR")&".JAN,"&EVPRO($H$2,EVCVW($H$2,"TIME"),"YEAR")&".FEB,"&EVPRO($H$2,EVCVW($H$2,"TIME"),"YEAR")&".MAR,"&EVPRO($H$2,EVCVW($H$2,"TIME"),"YEAR")&".APR,"&EVPRO($H$2,EVCVW($H$2,"TIME"),"YEAR")&".MAY,"&EVPRO($H$2,EVCVW($H$2,"TIME"),"YEAR")&".JUN,"&EVPRO($H$2,EVCVW($H$2,"TIME"),"YEAR")&".JUL,"&EVPRO($H$2,EVCVW($H$2,"TIME"),"YEAR")&".AUG,"&EVPRO($H$2,EVCVW($H$2,"TIME"),"YEAR")&".SEP,"&EVPRO($H$2,EVCVW($H$2,"TIME"),"YEAR")&".OCT,"&EVPRO($H$2,EVCVW($H$2,"TIME"),"YEAR")&".NOV,"&EVPRO($H$2,EVCVW($H$2,"TIME"),"YEAR")&".DEC"

Secondly, notice the highlighted box over the column. It now shows LC because the EVDRE will not be able to interpret the description when multiple members are specified. To get around this, place a base member from one of the other dimensions, in this case RPTCURRENCY, and link to the Time dimension in the description.

Lastly, be careful when summing accounts across time frames as INC/EXP accounts do aggregate, but LEQ/AST accounts do not.

**Summary**

In summary, when the above performance tips are properly utilized, EVDRE performance increases dramatically. In one customer’s instance, these tips provided over 200% better performance than how it was originally designed.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>12:38</td>
<td>39,593</td>
<td>2:00</td>
<td>21,513</td>
</tr>
<tr>
<td>Intersection 2</td>
<td>Timed out at 19:19</td>
<td>344,953</td>
<td>8:45</td>
<td>189,599</td>
</tr>
<tr>
<td>Intersection 3</td>
<td>Timed out at 20:03</td>
<td>344,332</td>
<td>10:30</td>
<td>187,239</td>
</tr>
<tr>
<td>Intersection 4</td>
<td>Timed out at 35:40</td>
<td>857,783</td>
<td>22:39</td>
<td>469,628</td>
</tr>
</tbody>
</table>

As a best practice when developing a report or input template, make sure you completely understand the requirements, where the data is coming from, and how the data needs to be rendered. Then, spend ample time creating spec’s and using UJSTAT to get the most performance out of the report as you can. In the end, the development of these reports/input templates may take longer, but the extra time will be well spent as they will perform much better than having one EVDRE with a lot of MDX.
Related Content
For more information, visit the Enterprise Performance Management homepage.